INTRODUCCIÓN A LA COMPLEJIDAD COMPUTACIONAL

Gonzalo Zigarán

Facultad de Matemática, Astronomía, Física y Computación - U.N.C.

Seminario de Alumnes

21 de Octubre de 2022

Teoría de la Computación

El trabajo de A. Turing[1] se considera
como el precursor de la Teoría de la Computación (ToC).

- Máquina de Turing (Alan Turing)
- Cálculo Lambda (Alonzo Church)
- Funciones recursivas (Kurt Gödel)
- Paradigma imperativo (John von Neumann)

[1] A.M Turing; "On computable numbers; with an application to the Entscheidungsproblem"; 1936.

Teoría de la Computación

El trabajo de A. Turing[1] se considera
como el precursor de la Teoría de la Computación (ToC).

- Máquina de Turing (Alan Turing)
- Cálculo Lambda (Alonzo Church)
- Funciones recursivas (Kurt Gödel)
- Paradigma imperativo (John von Neumann)

Tesis de Church-Turing

[1] A.M Turing; "On computable numbers; with an application to the Entscheidungsproblem"; 1936.

El trabajo de A. Turing[1] se considera
como el precursor de la Teoría de la Computación (ToC).

- Máquina de Turing (Alan Turing)
- Cálculo Lambda (Alonzo Church)
- Funciones recursivas (Kurt Gödel)
- Paradigma imperativo (John von Neumann)

Tesis de Church-Turing

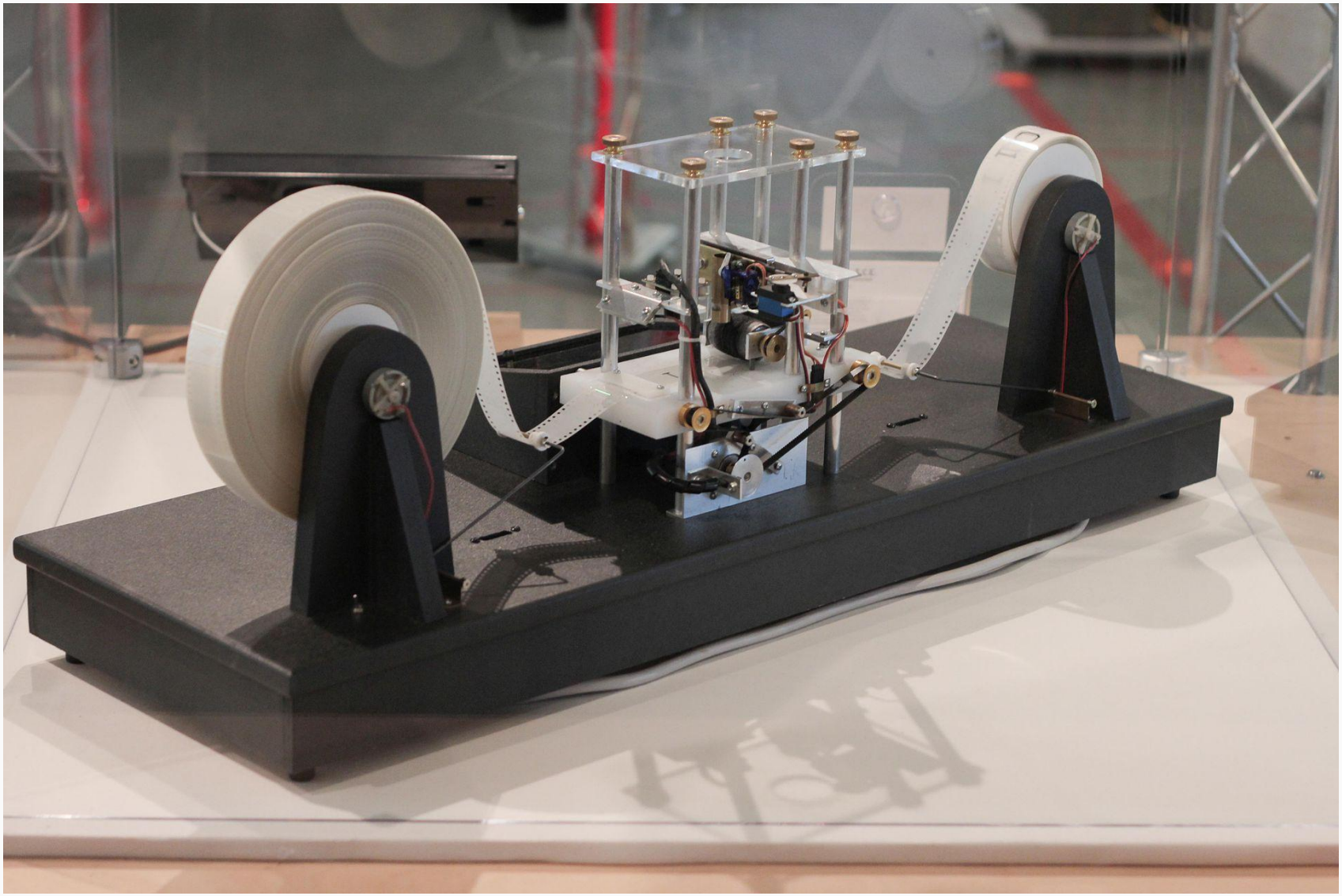
Observación

Lo que motivó fuertemente el desarrollo que tuvo la ToC en estos primeros años, a partir del contexto histórico, fue la criptografía. De alguna manera es una de las áreas que invita al estudio abstracto y formal de la ToC, y a su vez, naturalmente tiene gran influencia en el desarrollo de la Complejidad Computacional.

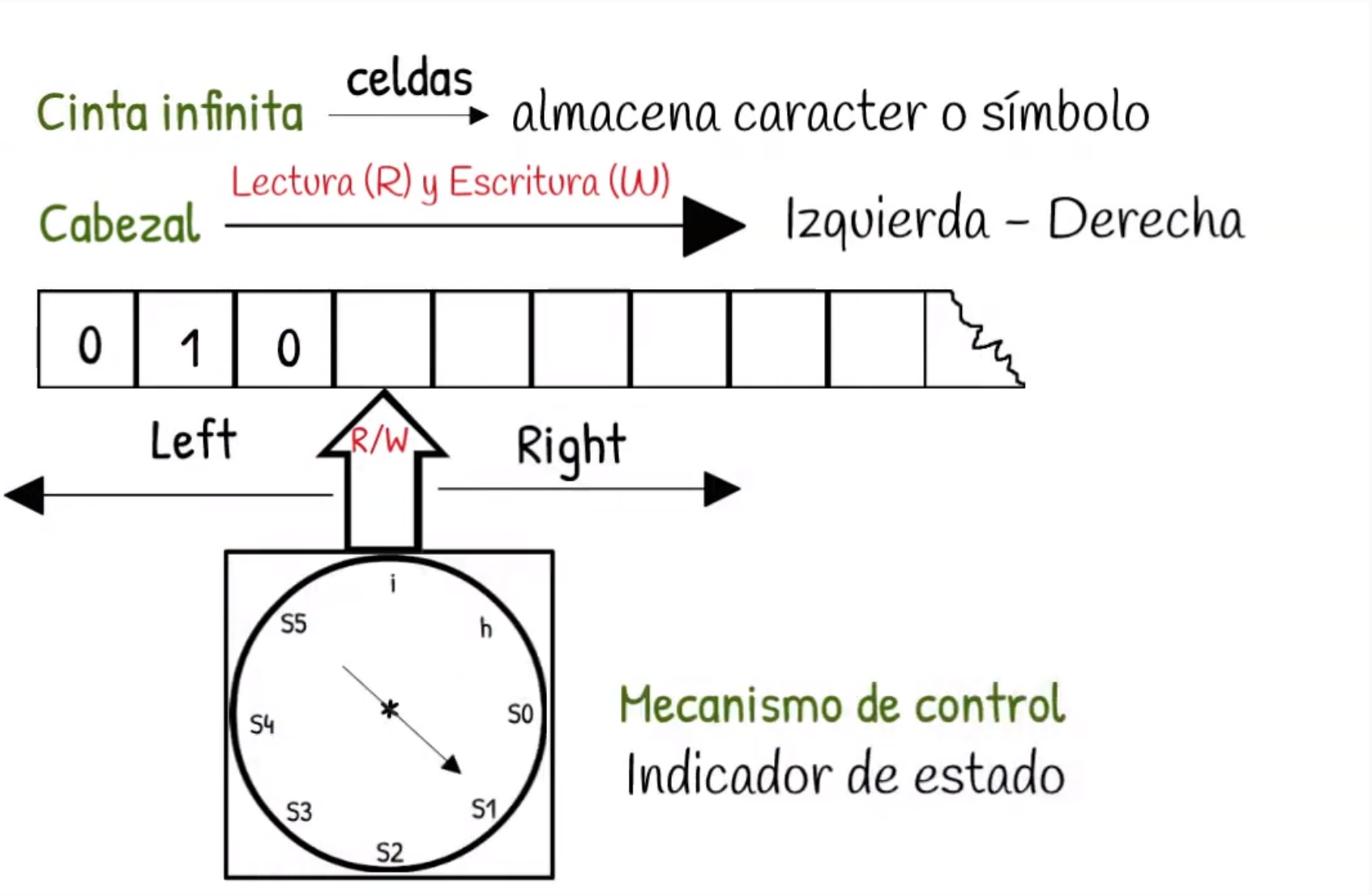
[1] A.M Turing; "On computable numbers; with an application to the Entscheidungsproblem"; 1936.

Máquina de Turing

Máquina de Turing



Máquina de Turing



Ejemplo

- $Q = \{q0, q1\}$
- $\Gamma = \{B, 0, 1\}$
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$
  - $\delta(q0, 0) = (q0, 1, R)$
  - $\delta(q0, 1) = (q0, 1, R)$
  - $\delta(q0, B) = (q1, B, L)$

Complejidad Computacional

”Determinar la cantidad mínima de recursos necesarios para resolver problemas computacionales, con modelos computacionales.”

Complejidad Computacional

”Determinar la cantidad mínima de recursos necesarios para resolver problemas computacionales, con modelos computacionales.”

Complejidad Computacional

”Determinar la cantidad mínima de recursos necesarios para resolver problemas computacionales, con modelos computacionales.”

Cuando pensamos en la complejidad de un problema, se suele pensar en la complejidad del peor caso. ¿Por qué?

Complejidad Computacional

”Determinar la cantidad mínima de recursos necesarios para resolver problemas computacionales, con modelos computacionales.”

Cuando pensamos en la complejidad de un problema, se suele pensar en la complejidad del peor caso. ¿Por qué?

Complejidad de un problema vs. Complejidad de un algoritmo

¿Cúal es la complejidad de ordenar los elementos de una lista?

Ejemplo

1. ¿Qué ecuaciones diofánticas de la forma $Ax^2 + By + C = 0$ tienen una solución entera positiva?
2. ¿Qué nudos en variedades tridimensionales unen una superficie de género $\leq g$?
3. ¿Qué mapas planares son 3-coloreables?

Ejemplo

1. ¿Qué ecuaciones diofánticas de la forma $Ax^2 + By + C = 0$ tienen una solución entera positiva?
2. ¿Qué nudos en variedades tridimensionales unen una superficie de género $\leq g$?
3. ¿Qué mapas planares son 3-coloreables?

Teorema

Los problemas 1., 2. y 3. son equivalentes.

Representación de los problemas

Cada problema hay que decidir cómo representar una entrada (por ej. en 1. una ecuación dada) para decidir luego si el problema se resuelve para esa entrada o no.

Posibles entradas para los ejemplos:

1. (A, B, C)
2. (M, K, G)
3. (V, E)

Representación de los problemas

Cada problema hay que decidir cómo representar una entrada (por ej. en 1. una ecuación dada) para decidir luego si el problema se resuelve para esa entrada o no.

Posibles entradas para los ejemplos:

1. (A, B, C)
2. (M, K, G)
3. (V, E)

Denotamos con I_n a todas las secuencias binarias de largo n, es decir $\mathrm{I}_n := \{0,1\}^n$. Luego,

$$\mathrm{I} := \bigcup \mathrm{I}_n$$

a todas las secuencias binarias de cualquier largo.

Todas las entradas las pensamos dentro de I.

Representación de los problemas

Para cada problema tenemos una función que convierte la entrada en una secuencia binaria. Por ejemplo para 1., tendríamos

$$\begin{aligned} h : \quad \mathbb{Z}^3 \quad &\to \mathrm{I} \\ (A, B, C) \quad &\mapsto x \end{aligned}$$

donde $x \in \mathrm{I}$ es una secuencia binaria.

Representación de los problemas

Para cada problema tenemos una función que convierte la entrada en una secuencia binaria. Por ejemplo para 1., tendríamos

$$\begin{aligned} h : \quad \mathbb{Z}^3 \quad &\to \mathrm{I} \\ (A, B, C) \quad &\mapsto x \end{aligned}$$

donde $x \in \mathrm{I}$ es una secuencia binaria.

Luego, una solución al problema de decisión sería una función computable $f : \mathrm{I} \to \{0, 1\}$ tal que,

$f \circ h(A, B, C) = 1$ si y solo si la ecuación $Ax^2 + By + C = 0$ tiene una solución entera positiva.

Representación de los problemas

Para cada problema tenemos una función que convierte la entrada en una secuencia binaria. Por ejemplo para 1., tendríamos

$$\begin{aligned} h : \quad \mathbb{Z}^3 \quad &\to \mathrm{I} \\ (A, B, C) \quad &\mapsto x \end{aligned}$$

donde $x \in \mathrm{I}$ es una secuencia binaria.

Luego, una solución al problema de decisión sería una función computable $f : \mathrm{I} \to \{0, 1\}$ tal que,

$f \circ h(A, B, C) = 1$ si y solo si la ecuación $Ax^2 + By + C = 0$ tiene una solución entera positiva.

Clase $\mathcal{P}$

Definición

Una función $f : \mathrm{I} \to \mathrm{I}$ está en la clase $\mathcal{P}$ si existe una maquina de Turing que compute f y constantes positivas A y c tal que para todo $n \geq 0$ y todo $x \in \mathrm{I}_n$, la maquina de Turing computa $f(x)$ en a lo sumo An^c pasos.

Clase $\mathcal{P}$

Definición

Una función $f : \mathrm{I} \to \mathrm{I}$ está en la clase $\mathcal{P}$ si existe una maquina de Turing que compute f y constantes positivas A y c tal que para todo $n \geq 0$ y todo $x \in \mathrm{I}_n$, la maquina de Turing computa f(x) en a lo sumo An^c pasos.

Definición '

Una función $f : \mathrm{I} \to \mathrm{I}$ está en la clase $\mathcal{P}$ si existe un algoritmo que compute f y constantes positivas A y c tal que para todo $n \geq 0$ y todo $x \in \mathrm{I}_n$, el algoritmo computa f(x) en a lo sumo An^c pasos.

Clase $\mathcal{P}$

Definición

Una función f : $\mathrm{I} \to \mathrm{I}$ está en la clase $\mathcal{P}$ si existe una maquina de Turing que compute f y constantes positivas A y c tal que para todo n ≥ 0 y todo x $\in \mathrm{I}_n$, la maquina de Turing computa f(x) en a lo sumo An^c pasos.

Definición '

Una función f : $\mathrm{I} \to \mathrm{I}$ está en la clase $\mathcal{P}$ si existe un algoritmo que compute f y constantes positivas A y c tal que para todo n ≥ 0 y todo x $\in \mathrm{I}_n$, el algoritmo computa f(x) en a lo sumo An^c pasos.

Definición ''

Una función f : $\mathrm{I} \to \{0, 1\}$ está en la clase $\mathcal{P}$ si existe un algoritmo que compute f y constantes positivas A y c tal que para todo n ≥ 0 y todo x $\in \mathrm{I}_n$, el algoritmo computa f(x) en a lo sumo An^c pasos.

Clase $\mathcal{P}$

Definición

Una función $f : \mathrm{I} \to \mathrm{I}$ está en la clase $\mathcal{P}$ si existe una maquina de Turing que compute f y constantes positivas A y c tal que para todo $n \geq 0$ y todo $x \in \mathrm{I}_n$, la maquina de Turing computa f(x) en a lo sumo An^c pasos.

Definición '

Una función $f : \mathrm{I} \to \mathrm{I}$ está en la clase $\mathcal{P}$ si existe un algoritmo que compute f y constantes positivas A y c tal que para todo $n \geq 0$ y todo $x \in \mathrm{I}_n$, el algoritmo computa f(x) en a lo sumo An^c pasos.

Definición ''

Una función $f : \mathrm{I} \to \{0, 1\}$ está en la clase $\mathcal{P}$ si existe un algoritmo que compute f y constantes positivas A y c tal que para todo $n \geq 0$ y todo $x \in \mathrm{I}_n$, el algoritmo computa f(x) en a lo sumo An^c pasos.

Definición '''

Un conjunto C está en la clase $\mathcal{P}$ si existe un algoritmo y constantes positivas A y c tal que para todo $n \geq 0$ y todo $x \in \mathrm{I}_n$, el algoritmo decide si x está en C en a lo sumo An^c pasos.

Ejemplos

- **Perfect matching.** Decidir si en un grafo se pueden emparejar los vértices, de modo que haya una arista entre cada par.
- **Primality testing.** Determinar si un número es primo.
- **Planarity testing.** Decidir si un grafo es plano.
- **Linear programming.** Decidir si un conjunto de desigualdades lineales son satisfacibles al mismo tiempo.
- **Hyperbolic word problem.** Dado un grupo hiperbólico y una palabra w de generadores, decidir si w representa el elemento identidad.
- **Graph Connectivity.** Decidir si todo par de vértices de un grafo tiene un camino.
- **Satisfiability.** Dada una fórmula de primer orden, decidir si existen valores que la hacen verdadera.

Clase $\mathcal{NP}$

¿Cuál es la complejidad de decidir si una proposición es un teorema?

¿Cuál es la complejidad de, dada una demostración de un teorema, decidir si es correcta?

Clase $\mathcal{NP}$

¿Cuál es la complejidad de decidir si una proposición es un teorema?

¿Cuál es la complejidad de, dada una demostración de un teorema, decidir si es correcta?

Definición

Un conjunto C está en la clase $\mathcal{NP}$ si existe una función $V_C \in \mathcal{P}$ y una constante k tal que:

- Si $x \in C$, entonces $\exists y$ con $|y| \leq k \cdot |x|^k$ y $V_C(x, y) = 1$;
- Si $x \notin C$, entonces $\forall y$ tenemos que $V_C(x, y) = 0$.

Clase $\mathcal{NP}$

¿Cuál es la complejidad de decidir si una proposición es un teorema?

¿Cuál es la complejidad de, dada una demostración de un teorema, decidir si es correcta?

Definición

Un conjunto C está en la clase $\mathcal{NP}$ si existe una función $V_C \in \mathcal{P}$ y una constante k tal que:

- Si $x \in C$, entonces $\exists y$ con $|y| \leq k \cdot |x|^k$ y $V_C(x, y) = 1$;
- Si $x \notin C$, entonces $\forall y$ tenemos que $V_C(x, y) = 0$.

Si pensamos a $\mathcal{P}$ como una clase de conjuntos, es claro que $\mathcal{P} \subseteq \mathcal{NP}$. (Tomando y la secuencia vacía)

$$¿\mathcal{P} = \mathcal{NP}?$$

Ejemplos

- **Hamiltonian cycles in graphs.** Decidir si un grafo tiene un ciclo hamiltoneano.
- **Factoring integers.** Dada una tupla de enteros (x, a, b), decidir si x tiene un factor primo en el intervalo $[a, b]$
- **Integer programming.** Decidir si un conjunto de desigualdades lineales son satisfacibles al mismo tiempo por soluciones enteras.
- **Matrix group membership.** Dados 3 matrices invertibles del mismo tamaño, decidir si la primera está en el subgrupo generado por las otras 2.
- **Graph isomorphism.** Decidir si 2 grafos son isomorfos.

Clase co$\mathcal{NP}$

Definición

Un conjunto C está en la clase co$\mathcal{NP}$ si y solo si su complemento $\bar{C} = I \setminus C$ está en $\mathcal{NP}$.

Clase co$\mathcal{NP}$

Definición

Un conjunto C está en la clase co$\mathcal{NP}$ si y solo si su complemento $\bar{C} = I \setminus C$ está en $\mathcal{NP}$.

Por ejemplo, decidir si dos grafos son isomorfos está en $\mathcal{NP}$, entonces decidir si dos grafos no son isomorfos está en co$\mathcal{NP}$.

Clase co$\mathcal{NP}$

Definición

Un conjunto C está en la clase co$\mathcal{NP}$ si y solo si su complemento $\bar{C} = I \setminus C$ está en $\mathcal{NP}$.

Por ejemplo, decidir si dos grafos son isomorfos está en $\mathcal{NP}$, entonces decidir si dos grafos no son isomorfos está en co$\mathcal{NP}$.

$$¿\mathcal{NP} = \text{co}\mathcal{NP}?$$

Clase co$\mathcal{NP}$

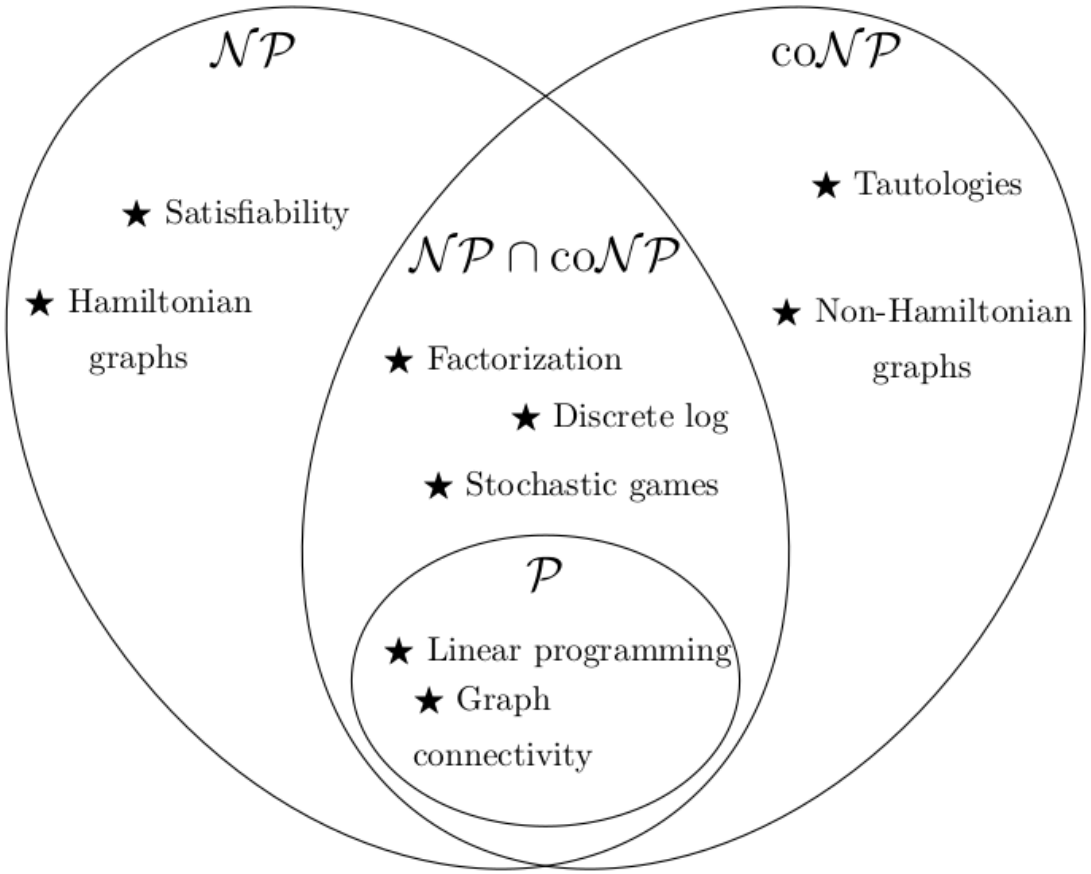
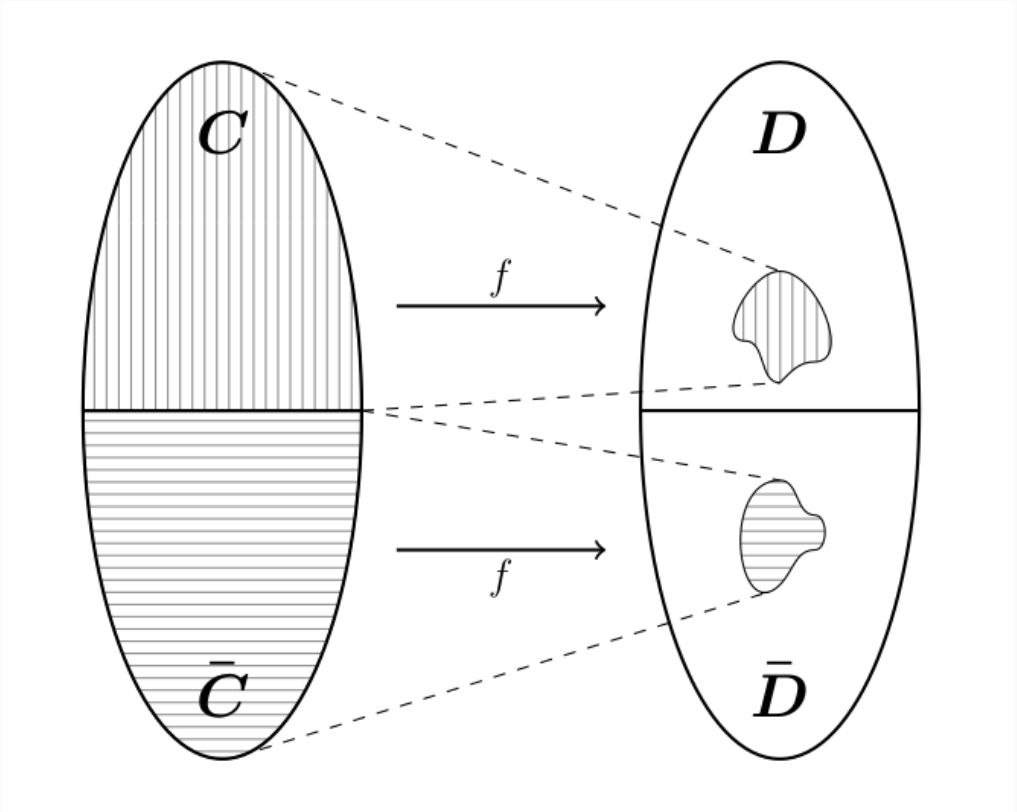


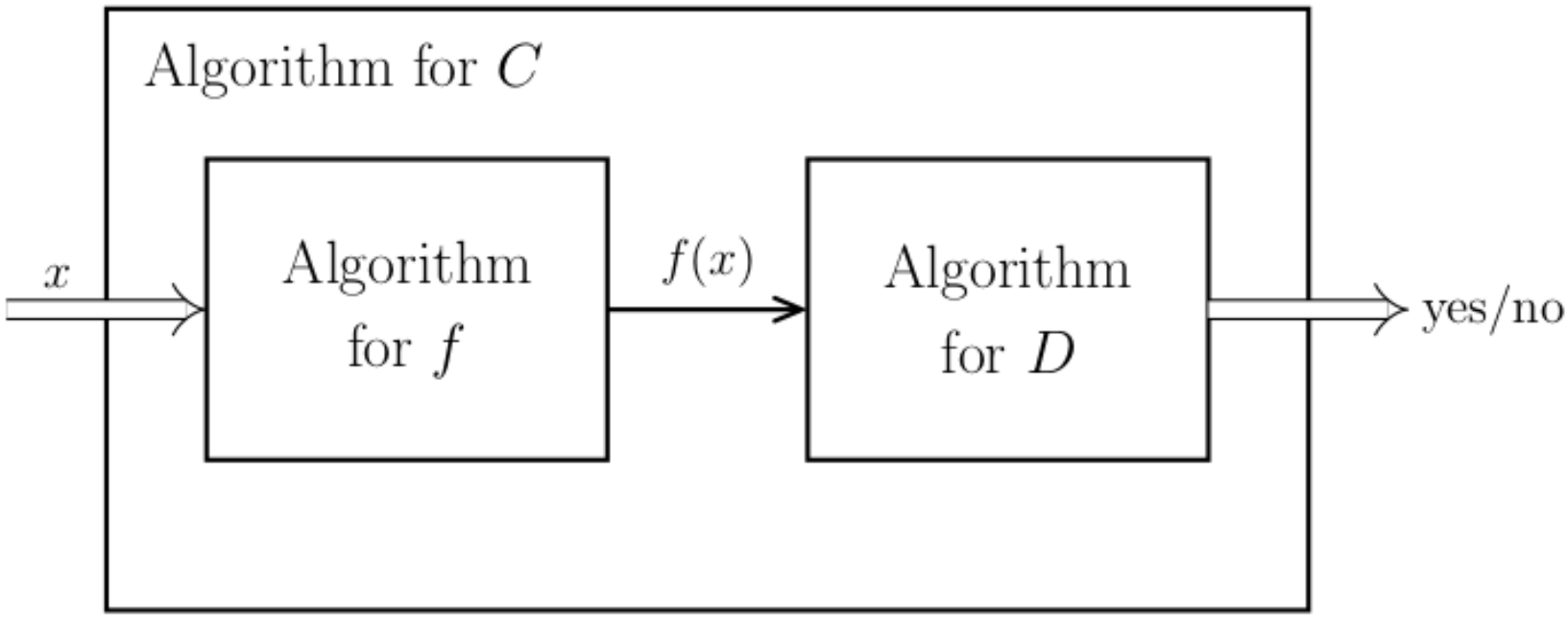
Figure 6. $\mathcal{P}$, $\mathcal{NP}$, and co$\mathcal{NP}$.

Definición

Sean $C, D \subset \mathrm{I}$ dos problemas de clasificación. $f : \mathrm{I} \to \mathrm{I}$ es una reducción eficiente de C a D si $f \in \mathcal{P}$ y para todo $x \in \mathrm{I}$ vale que $x \in C$ si y solo si $f(x) \in D$. Se denota con $C \leq D$ si existe una reducción eficiente de C a D.



Reducciones eficientes



Ejemplo

1. ¿Qué ecuaciones diofánticas de la forma $Ax^2 + By + C = 0$ tienen una solución entera positiva?
2. ¿Qué nudos en variedades tridimensionales unen una superficie de género $\leq g$?
3. ¿Qué mapas planares son 3-coloreables?

Teorema

Los problemas 1., 2. y 3. son equivalentes.

Es decir,

- $1. \leq 2.$
- $2. \leq 3.$
- $3. \leq 1.$

¡Muchas gracias!