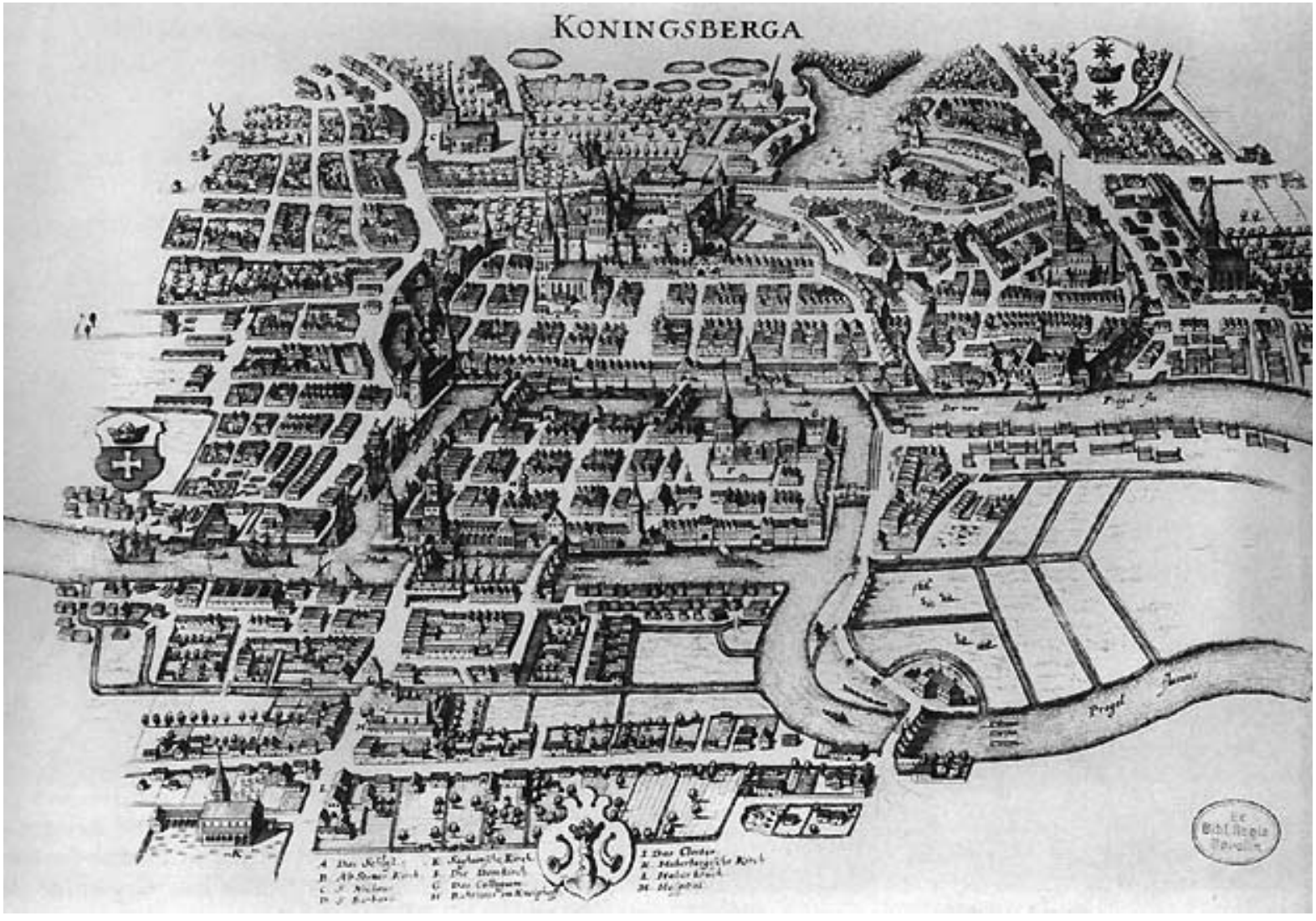
Complejidad computacional

Exploratorio Computación

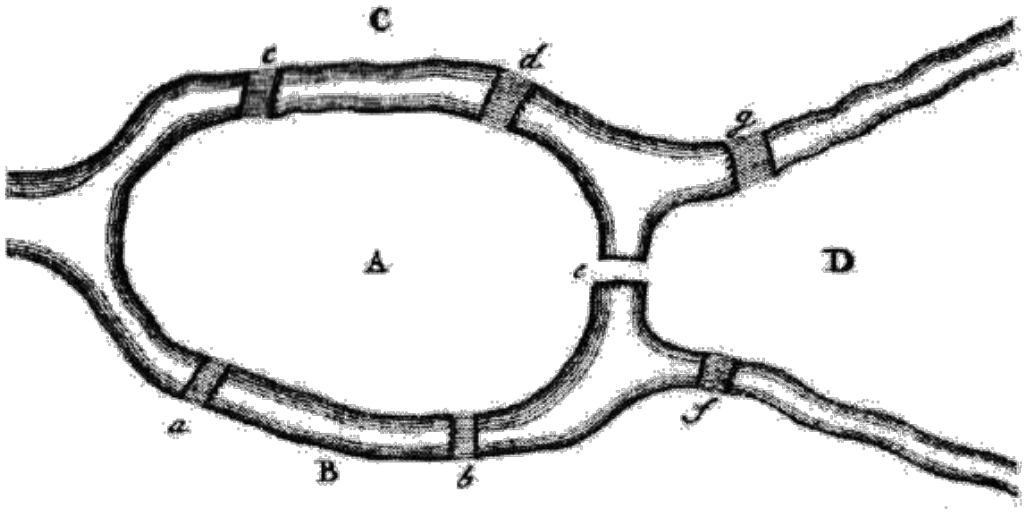
Denis Parra

Invitado: Cristian Riveros

Los siete puentes de Köningsberg (Siglo XVIII)



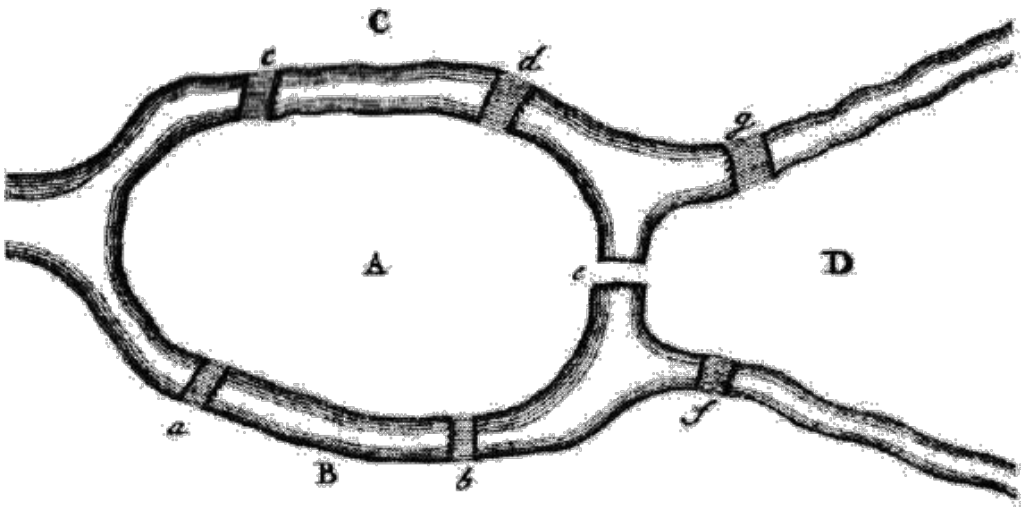
Los siete puentes de Königsberg (Siglo XVIII)

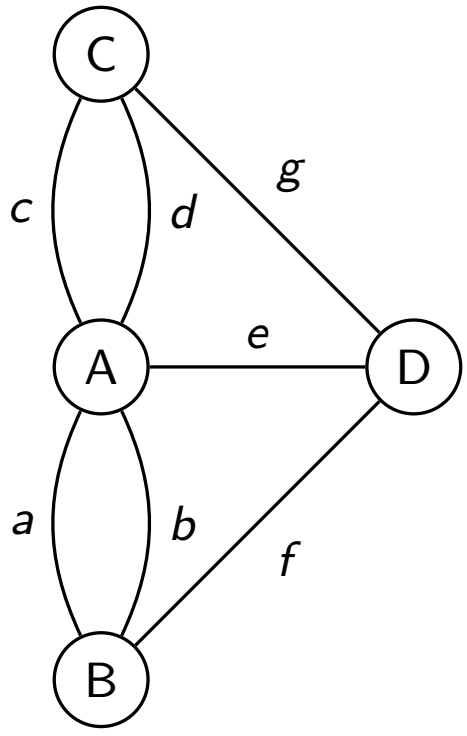


Leonard Euler

¿existe una caminata por la ciudad que recorra cada puente exactamente una vez?

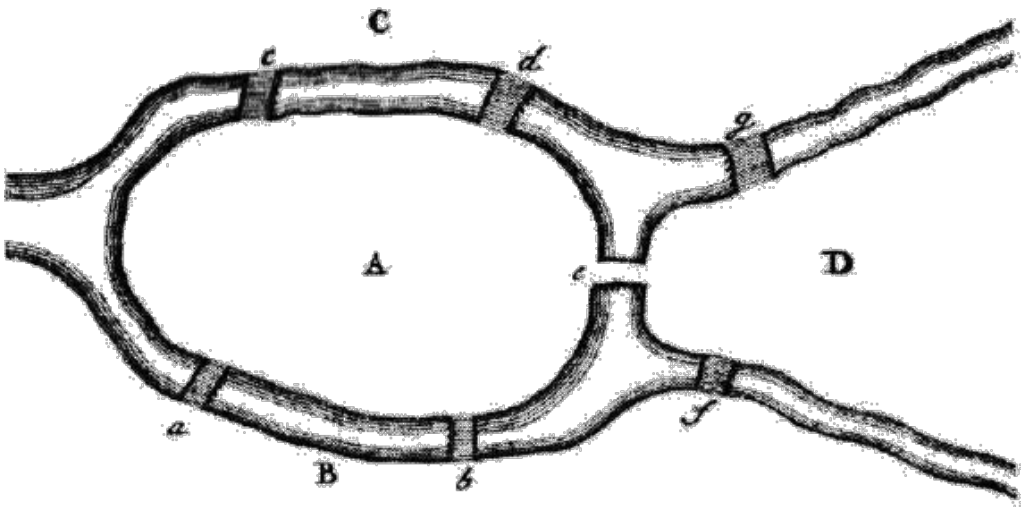
Los siete puentes de Königsberg (Siglo XVIII)

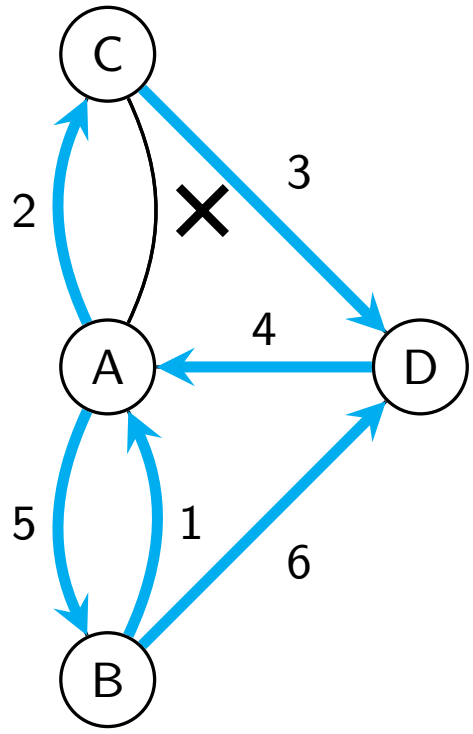




¿existe una caminata por la ciudad que recorra cada puente exactamente una vez?

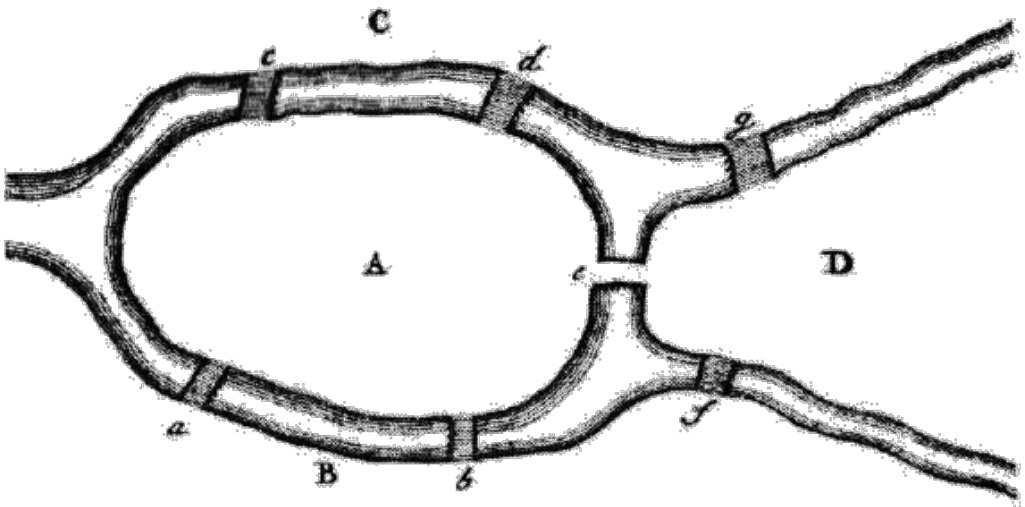
Los siete puentes de Königsberg (Siglo XVIII)

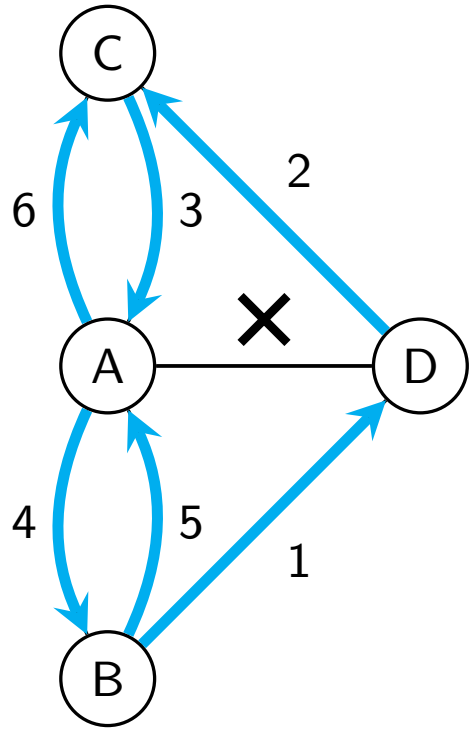




¿existe una caminata por la ciudad que recorra cada puente exactamente una vez?

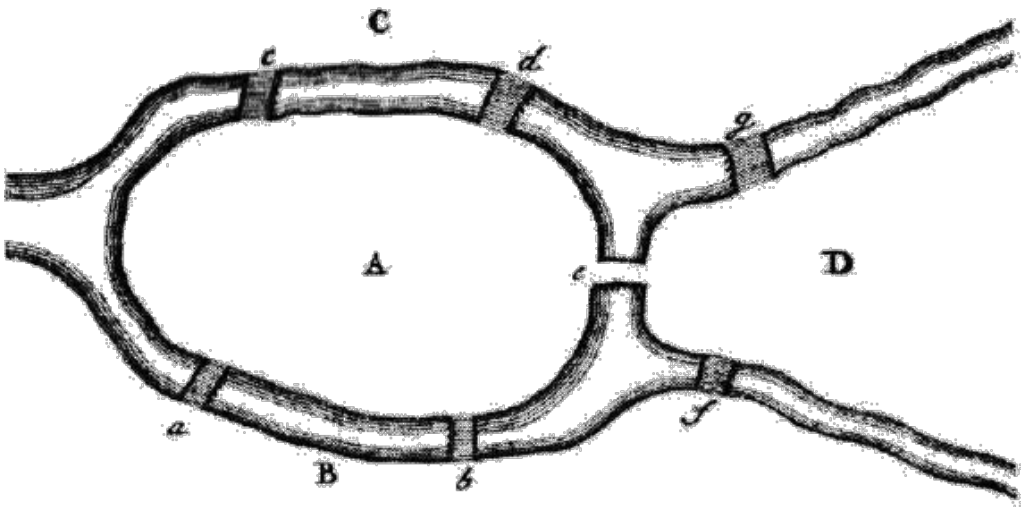
Los siete puentes de Königsberg (Siglo XVIII)

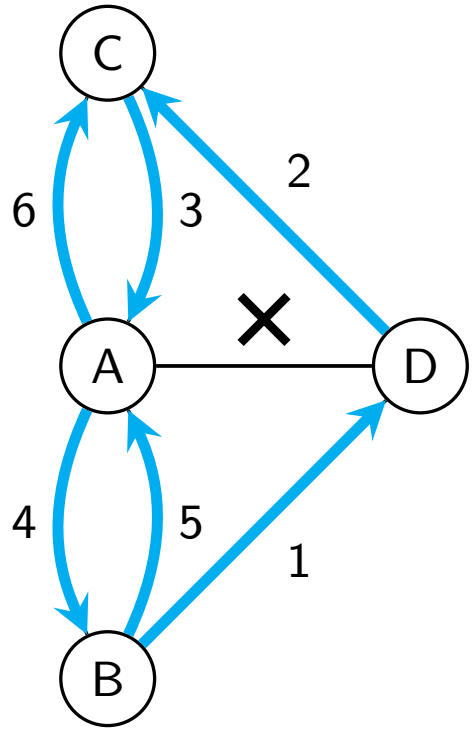




¿existe una caminata por la ciudad que recorra cada puente exactamente una vez?

Los siete puentes de Königsberg (Siglo XVIII)





Definición

Decimos que un grafo tiene un **camino euleriano**
si existe una camino que pasa por cada conección (i.e. arista) una sola vez.

¿cómo verificamos si nuestro grafo es euleriano?

Primera estrategia

Probar todos los caminos y verificar si es un camino euleriano.

¿cuántos caminos tendremos que verificar?

¿cómo verificamos si nuestro grafo es euleriano?

Primera estrategia

Probar todos los caminos y verificar si es un camino euleriano.

- Para los 7 puentes de Köningsberg:

$$\text{cantidad de caminos} \approx 2^7 = 128$$

- Para los 420 puentes de Venecia nos tardaríamos:

$$\text{cantidad de caminos} \approx 2^{420} \geq 10^{100}$$

¿cuántos nos demoraremos en revisar
10^{100} caminos en el último computador?

¿cómo verificamos si nuestro grafo es euleriano?

Primera estrategia

Probar todos los caminos y verificar si es un camino euleriano.

Para ver cuanto nos demoramos en revisar 10^{100} caminos:

$$\begin{aligned} \#\text{ de átomos en el universo} &\approx 10^{80} \text{ átomos} \\ \text{edad del universo} &\approx 10^{20} \text{ milisegundos} \end{aligned}$$

- Si colocamos un microcomputador en cada **átomo**,
- cada uno demora un milisegundo en verificar un **camino**, y
- los ejecutamos durante la edad del **universo** . . .

. . . no alcanzaremos a verificar todos los caminos!

. . . ¿otra estrategia?

¿cómo verificamos si nuestro grafo es euleriano?

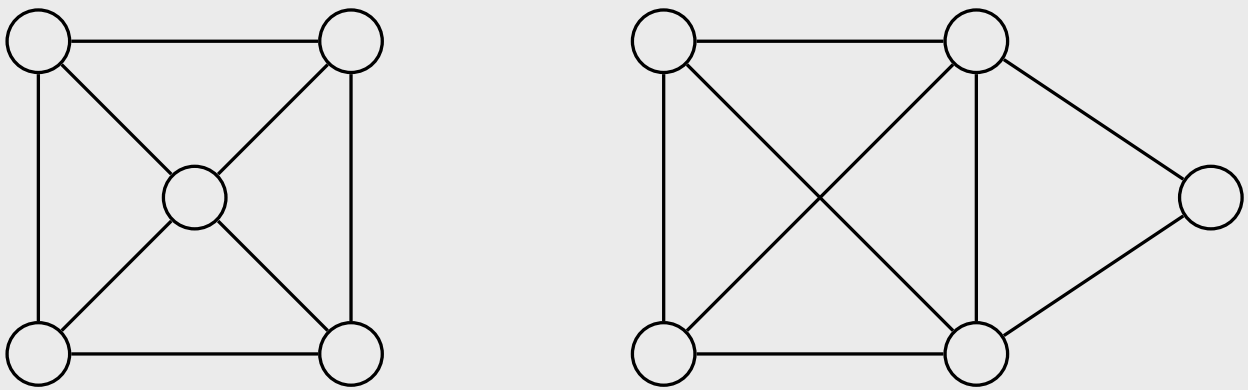
Segunda estrategia (por Euler)

Los puentes deben llegar de a pares a las islas.

Teorema

Un grafo (conexo) contiene un camino euleriano si, y solo si, a cada nodo llegan una cantidad par de aristas con la posible exclusión de dos nodos.

Ejemplo



¿cuál de los dos grafos es euleriano?

¿cómo verificamos si nuestro grafo es euleriano?

Segunda estrategia (por Euler)

Los puentes deben llegar de a pares a las islas.

Teorema

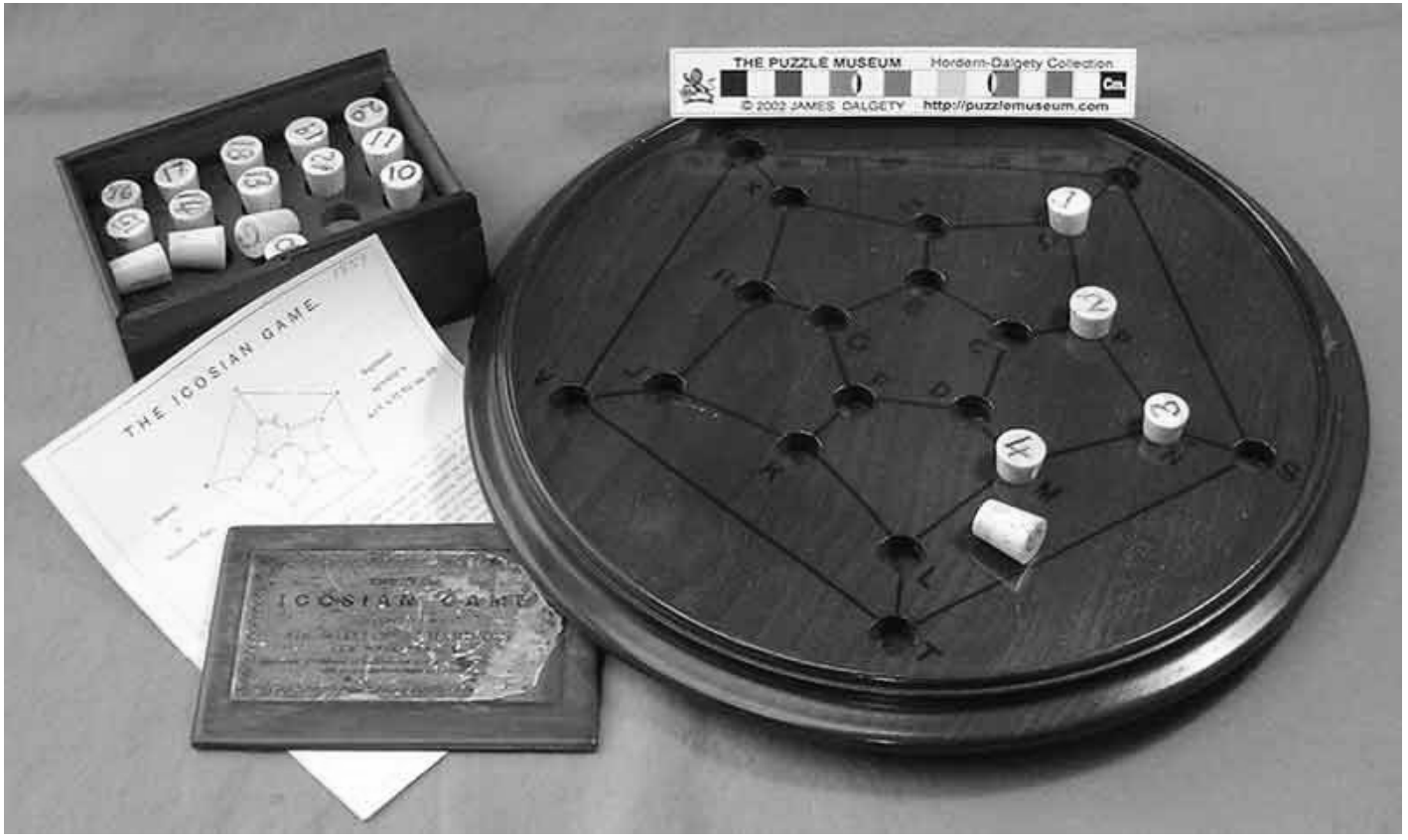
Un grafo (conexo) contiene un camino euleriano si, y solo si, a cada nodo llegan una cantidad par de aristas con la posible exclusión de dos nodos.

Para verificar si un grafo contiene un camino euleriano:

- Recorremos cada nodo del grafo.
- Si todos los nodos tienen una cantidad par de aristas con la posible exclusión de dos nodos aceptamos.
- Si no se cumple la propiedad rechazamos.

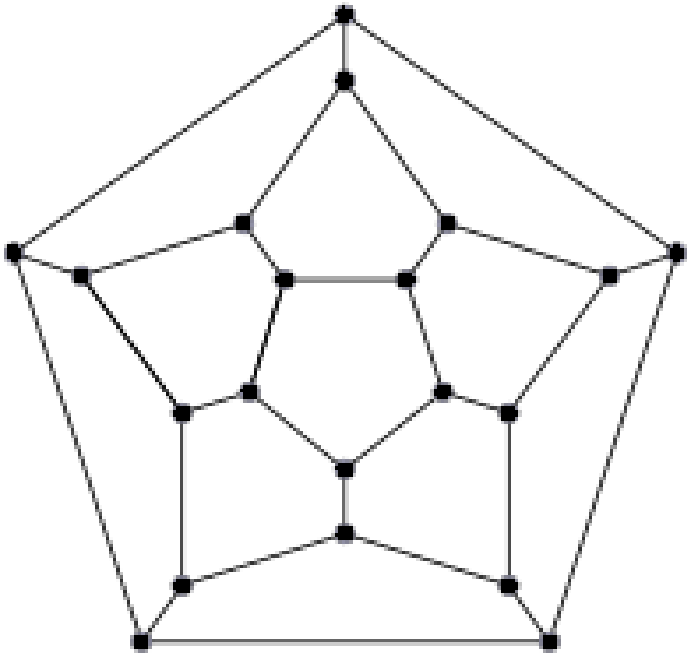
Esto lo podemos hacer en tiempo lineal en la cantidad de aristas!

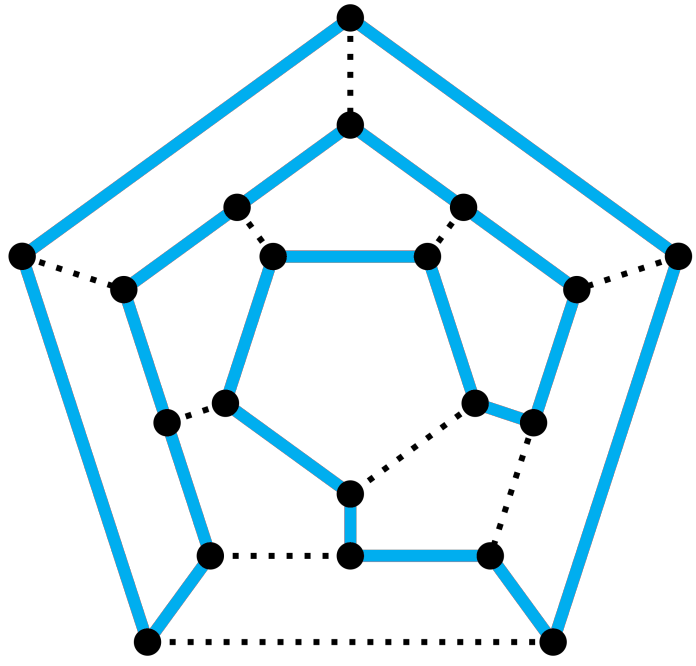
Juegos de Hamilton



Inventados por William Hamilton (1857)

Juegos de Hamilton





Definición

Decimos que un grafo tiene un **camino hamiltoneano**
si existe una camino que pasa por cada **nodo** una sola vez.

¿cómo verificamos si nuestro grafo es **hamiltoniano**?

Primera estrategia

Probar todos los caminos y verificar si algun camino es hamiltoniano.

- De nuevo: una cantidad **exponencial** de caminos que verificar!

¿podemos proponer una estrategia más eficiente?

... esta es la mejor estrategia
que se conoce hasta el momento.

¿és el problema de camino hamiltoneano **difícil**?

¿cuál es un problema difícil?

1. ¿cuál es un problema fácil?
2. ¿cómo identificar un problema difícil?
3. ¿cómo comparar la dificultad entre problemas?
4. ¿cómo (tratar) de solucionar un problema difícil?

Outline

¿qué es un problema?

¿cuál es la complejidad de un problema?

¿cuáles son los problemas difíciles?

¿cuáles son los problemas más difíciles?

Outline

¿qué es un problema?

¿cuál es la complejidad de un problema?

¿cuáles son los problemas difíciles?

¿cuáles son los problemas más difíciles?

¿qué es un problema computacional?

Ejemplos

Camino Euleriano

Input: Un grafo $\mathcal{G}$

Pregunta: ¿tiene el grafo $\mathcal{G}$ un camino euleriano?

Camino Hamiltoniano

Input: Un grafo $\mathcal{G}$

Pregunta: ¿tiene el grafo $\mathcal{G}$ un camino hamiltoniano?

Dado un input a estos problemas la respuesta es SI o NO.

¿qué es un problema computacional?

Definición

Un **problema de decisión** esta compuesto por:

1. Un conjunto de **inputs** (llamados instancias).
    - Números, grafos, palabras, funciones, etc . . .
2. Una **pregunta** sobre los inputs que se responde con **SI** o **NO**

¿qué es un problema computacional?

Mas ejemplos

Números Primos

Input: Un número N

Pregunta: ¿és N primo?

Minimización de funciones

Input: Un función $f : \mathbb{N} \to \mathbb{N}$ y un número c

Pregunta: ¿es el mínimo de f mayor que c?

¿cuáles son las soluciones a nuestros problemas?

¿cuáles son las soluciones a nuestros problemas?

Definición

Una **solución** para un problema de decisión es un **algoritmo** tal que para todo input *I* responde **SI** si, y solo si, *I* satisface la pregunta.

El algoritmo puede ser:

- Una serie de instrucciones (pseudo-código).
- Un programa en Java, Python, etc.
- Una Máquina de Turing.
- ...

> *"An algorithm is a finite answer*
> *to an infinite number of questions."*
>
> Stephen Kleene

Outline

¿qué es un problema?

¿cuál es la complejidad de un problema?

¿cuáles son los problemas difíciles?

¿cuáles son los problemas más difíciles?

¿cómo medimos la complejidad de nuestro problema?

Definición (complejidad de una instancia)

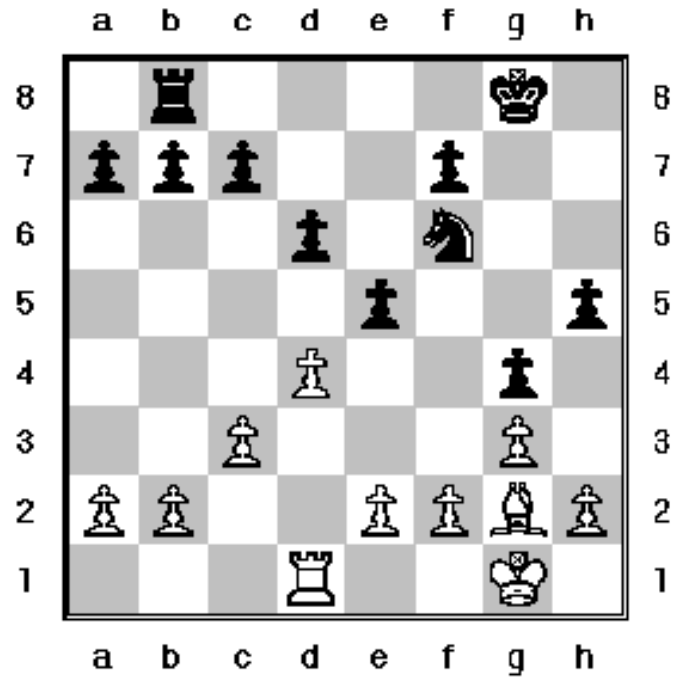
Para un problema de decisión (inputs + preguntas)
se define $|I|$ como el **tamaño** del input I.

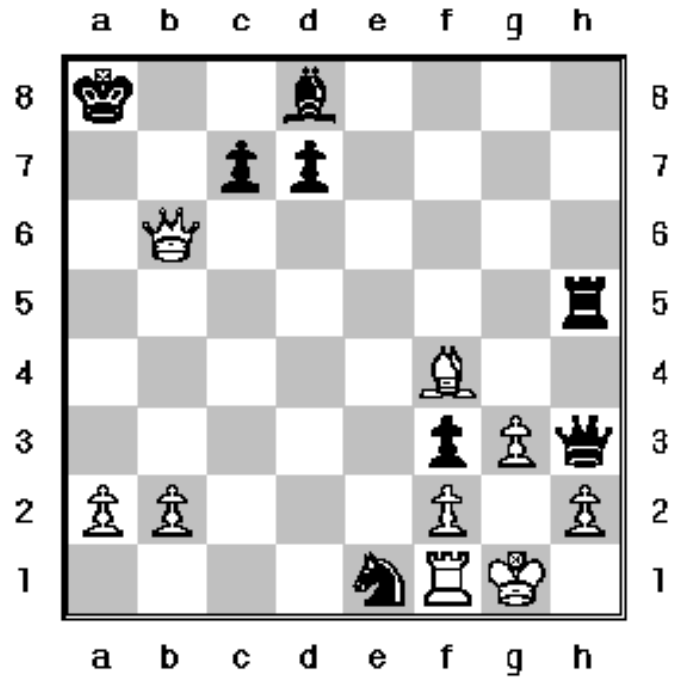
Ejemplos

- Para un **número** N, $|N|$ es el número bits en su representación binaria.
- Para un **grafo** $\mathcal{G}$, $|\mathcal{G}|$ es el número de nodos y aristas.
- Para un **palabra** w, $|w|$ es el largo de la palabra.

¿cuál es el tamaño del problema "Ajedrez"?

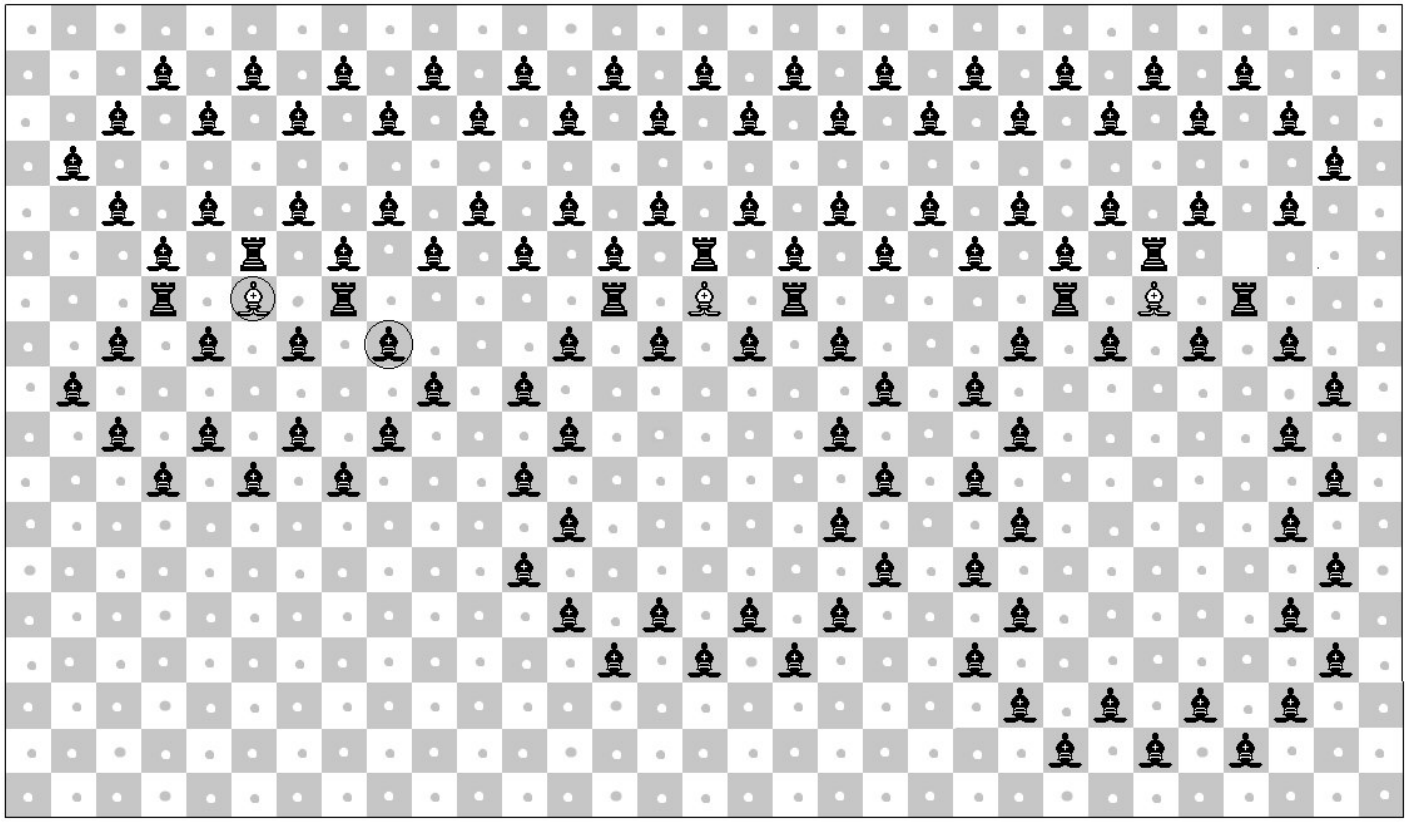
¿cómo formalizamos AJEDREZ como un problema?





- ¿cuáles son los posibles **inputs** de AJEDREZ?
- ¿cuál es una posible **pregunta** de decisión para AJEDREZ?

¿cómo formalizamos AJEDREZ como un problema?



AJEDREZ

Input: Un tablero de $N \times N$ y la posición de las fichas

Pregunta: ¿el jugador blanco puede ganar siempre desde esa posición?

¿cómo medimos la eficiencia de nuestras soluciones?

1. Tiempo.
2. Espacio.
3. Energía.
4. Accesos a disco duro.
5. Operaciones aritméticas.
6. Paralelización.
7. ...

¿cómo medimos estos factores?

¿cómo medimos la eficiencia de nuestras soluciones?

Definición

Para un algoritmo $\mathcal{A}$ y una función $T : \mathbb{N} \to \mathbb{N}$ decimos que:

"el algoritmo $\mathcal{A}$ toma tiempo $\mathcal{O}(T)$"

si para cada instancia I, el **número de pasos** N_I de $\mathcal{A}$ sobre I cumple:

$$N_I \leq T(|I|).$$

Ejemplo

- $\mathcal{O}(n)$: tiempo **lineal**.
- $\mathcal{O}(n^2)$: tiempo **quadratico**.
- $\mathcal{O}(\log(n))$: tiempo **logaritmico**.
- $\mathcal{O}(p(n))$: tiempo **polinomial** donde p es un polinomio.
- $\mathcal{O}(2^n)$: tiempo **exponencial**.

Estamos suponiendo la eficiencia en el **peor caso**

Suponemos la existencia de un demonio
que puede entregarnos la peor instancia posible!

¿cuáles son los problemas fáciles?

Definición

- Decimos que un problema se puede resolver **eficientemente** si existe un algoritmo que toma tiempo **polinomial**.
- Definimos la clase **P** de todos los problemas que se pueden resolver en tiempo **polinomial**.
- Definimos la clase **EXP** de todos los problemas que se pueden resolver en tiempo **exponencial**.

Es fácil ver que:

$$\mathbf{P} \subseteq \mathbf{EXP}$$

¿cuáles son los problemas fáciles?

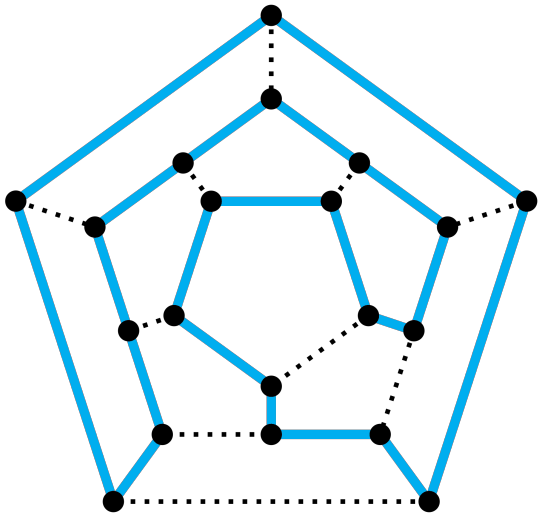
Ejemplo

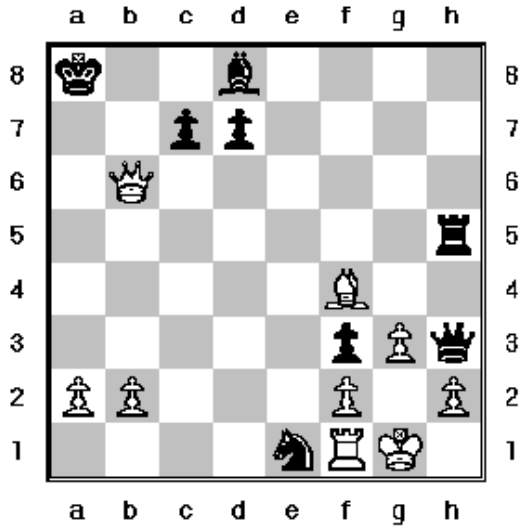
- Camino Euleriano esta en **P**.
- Camino Hamiltoniano esta en **EXP**.
- Ajedrez esta en **EXP**.

- ¿qué otros problemas conocen en la clase **P**?
- ¿si un problema esta en **EXP** quiere decir que es un problema **difícil**?
- ¿cómo sabemos si mi problema es **difícil** o no?

Outline

¿tienen algún parecido estos problemas?





¿son los dos igual de difíciles?

Camino Hamiltoniano es mas fácil en este sentido...

Usted

Un demonio

¿Tiene mi grafo G un camino hamiltoniano?

Si, tiene un camino hamiltoniano

Demuestrame que tiene un camino hamiltoniano

$v_1, v_2, v_3, \ldots, v_N$ (camino de largo N)

Ustedes lo verifican en tiempo polinomial ✓

Camino Hamiltoniano es mas fácil en este sentido...

Usted

Un demonio

¿Tiene mi grafo G un camino hamiltoniano?

Si, si tiene (miente)

Demuestrame que tiene un camino hamiltoniano

$v_1, v_2, v_3, \ldots, v_N$ (camino de largo N)

Ustedes lo verifican en tiempo polinomial y miente!! ✕

Camino Hamiltoniano es mas fácil en este sentido...

Usted

Un demonio

¿Tiene mi grafo G un camino hamiltoniano?

NO, no tiene un camino hamiltoniano

Demuestrame que NO tiene un camino hamiltoniano

. . . . . .

¿cómo les demuestra el demonio que NO hay un camino hamiltoniano?

Veamos que pasa con AJEDREZ...

Usted

Un demonio

¿Tiene blanco una estrategia ganadora en el tablero?

Si, blanco tiene una estrategia

Demuestrame que tiene una estrategia ganadora

.....

¿cómo les demuestra el demonio que hay una estrategia?

La clase de problemas **NP**

Definición

- Decimos que un problema es **eficiente de verificar** si existe un algoritmo que verifica en tiempo **polinomial** si una instancia y un certificado (de tamaño polinomial) satisfacen la pregunta.
- Definimos la clase **NP** de todos los problemas que se pueden **verificar** en tiempo **polinomial**.

Ejemplo

- Camino Hamiltoniano esta en **NP**.
- (Al parecer) Ajedrez NO esta en **NP**.

¿qué otros problemas conocen en la clase **NP**?

¿cuál es la pregunta abierta mas importante en CS?

$$\mathbf{P} \stackrel{?}{=} \mathbf{NP}$$

"If the solution to a problem can be quickly verified by a computer, can the computer also solve that problem quickly?"

Wikipedia.

¿cuál es la pregunta abierta mas importante en CS?

Uno de los 8 "Millennium Prize Problems":

1. Yang–Mills and Mass Gap
2. Riemann Hypothesis
3. **P vs NP Problem**
4. Navier–Stokes Equation
5. Hodge Conjecture
6. ~~Poincaré Conjecture~~
7. Birch and Swinnerton-Dyer Conjecture

propuestos por *The Clay Mathematics Institute*.

¿cuál es la pregunta abierta mas importante en CS?

$$\mathbf{P} \stackrel{?}{=} \mathbf{NP}$$

US$ 1 millón a quien lo resuelva.

"Aside from being an important problem in computational theory, a proof either way would have **profound implications** *for mathematics, cryptography, algorithm research, artificial intelligence, game theory, multimedia processing, philosophy, economics and many other fields."*

Wikipedia.

Outline

¿qué es un problema?

¿cuál es la complejidad de un problema?

¿cuáles son los problemas difíciles?

¿cuáles son los problemas más difíciles?

¿cuál es el problema más dificil en **NP**?

Definiciones

- Decimos que un problema P^* es **difícil** en **NP** (**NP**-hard) si para todo problema $P \in$ **NP**, cada instancia del problema P se puede **reducir** a un instancia del problema en P^*.

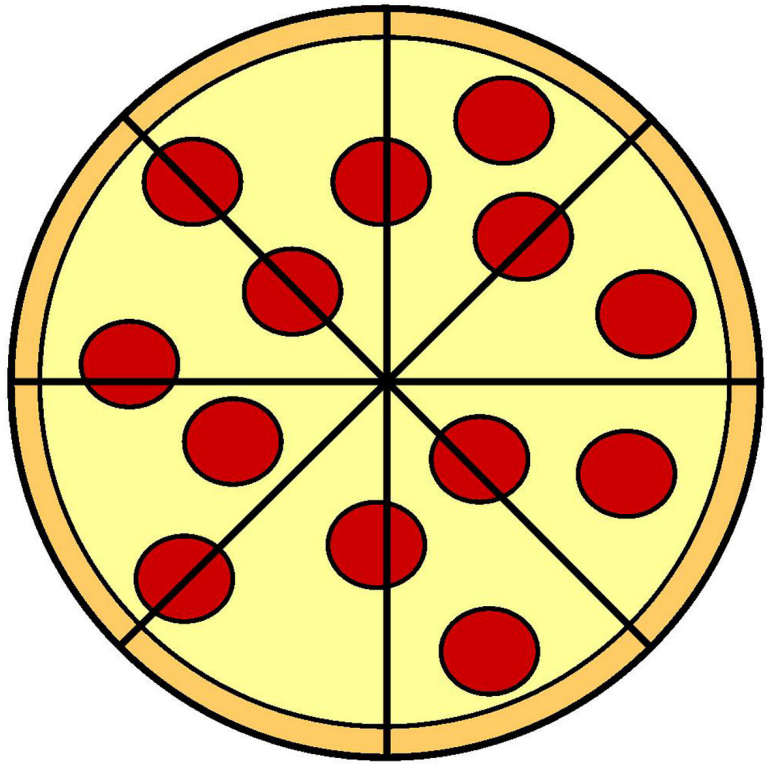
- Decimos que un problema P^* es **completo** en **NP** (**NP**-completo) si P^* esta en **NP** y es difícil en **NP**.

Los problemas **NP**-completo son los problemas más **difíciles** en **NP**

¿cuál es el problema más dificil en **NP**?

¿existe algún problema **NP**-completo en **NP**?

¿cómo pedir una pizza entre amigos?



¿cómo pedir una pizza entre amigos?

- Usted esta en un grupo de amigos y quieren pedir **solo** una pizza.
- Cada amigo espera que se cumplan sus preferencias.

Marcelo: $($ (queso OR peperonni) AND $\overline{\text{anchoas}}$ $)$ AND

Juan: $($ $\overline{\text{peperonni}}$ OR $\overline{\text{anchoas}}$ $)$ AND

Denis: $($ $\overline{\text{queso}}$ OR anchoas OR piña $)$

¿cómo podemos **satisfacer** a todos los amigos?

¿cómo pedir una pizza entre amigos?

Definición

Una formula **proposicional** P es una formula compuesta por:

- Variables (v) que puede toman valor `true` o `false`.
- Variables **negadas** ($\overline{v}$) que toman el valor opuesto.
- Conectores `AND` y `OR` .

Ejemplo

$$((q \texttt{ OR } p) \texttt{ AND } \overline{a}) \texttt{ AND } (\overline{p} \texttt{ OR } \overline{a}) \texttt{ AND } (\overline{q} \texttt{ OR } a \texttt{ OR } r)$$

¿cómo pedir una pizza entre amigos?

Definición

Una formula proposicional es **satisfacible** si existe
una asignación de valores a las variables que hace la formula verdadera.

Ejemplo

Marcelo: $($ (queso OR peperonni) AND $\overline{\text{anchoas}}$ $)$ AND

Juan: $($ $\overline{\text{peperonni}}$ OR $\overline{\text{anchoas}}$ $)$ AND

Denis: $($ $\overline{\text{queso}}$ OR anchoas OR piña $)$

La formula se satisface con la siguiente asignación de valores:

$$\begin{aligned} \text{queso} &= \texttt{true} \\ \text{peperonni} &= \texttt{true} \\ \text{anchoas} &= \texttt{false} \\ \text{piña} &= \texttt{true} \end{aligned}$$

Satisfacibilidad

SAT
Input: Una formula proposicional φ.
Pregunta: ¿es φ satisfacible por alguna valuación?

- ¿és SAT un problema en **NP**?
- ¿se parece SAT a Camino Hamiltoniano?

¿cuál es el problema más dificil en **NP**?

¿existe algún problema **NP**-completo en **NP**?

Teorema (1971)

SAT es un problema **NP**-completo.

Por lo tanto:

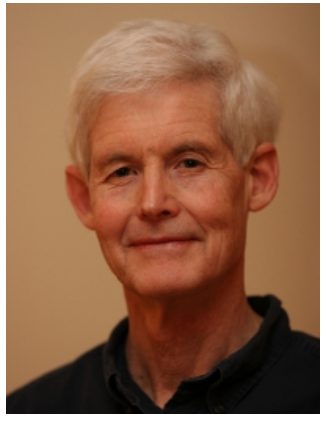
- SAT es uno de los problemas más difíciles en **NP**.
- SAT "contiene" todos los problemas en **NP**.

¿cuál es el problema más dificil en **NP**?

¿existe algún problema **NP**-completo en **NP**?

Teorema (1971)

SAT es un problema **NP**-completo.



Stephen Cook

Leonid Levin

¿es SAT el único problema **NP**-completo?

Richard Karp

Idea de Karp (1972)

Si tengo un problema P^* en **NP** y SAT se puede **reducir** a P^*, entonces P^* es también **NP**-completo.

¿es SAT el único problema **NP**-completo?

Muchos problemas son **NP**-completos como:

- Camino Hamiltoneano.
- Problema del vendedor viajero.
- Optimización discreta.
- Problemas de planificación.
- Sudoku, busca minas, ...
- Problema en bases de datos.
- Problema en inteligencia artificial.
- Match entre secuencias de DNA.
- etc...

¿qué ocurre si alguien encuentra alguna solución eficiente a un problema NP-completo?

Demostraría que **NP** = **P**

En particular, todos estos problemas se podrían resolver eficientemente:

- Camino Hamiltoneano.
- Problema del vendedor viajero.
- Optimización discreta.
- Problemas de planificación.
- Sudoku, busca minas, ...
- Problema en bases de datos.
- Problema en inteligencia artificial.
- Match entre secuencias de DNA.
- etc...

Algunas conclusiones sobre complejidad computacional

1. La complejidad computacional es un fenómeno en la naturaleza.

2. Es posible comparar y medir la complejidad de los problemas.

3. La complejidad computacional es uno de los desafios matemáticos y computacionales más importantes de este siglo.

¿donde puedo saber más sobre estos temas?

Algunos cursos:

1. Matemáticas discretas.
2. Lógica para ciencia de la computación.
3. Teoría de autómatas y lenguajes formales.
4. Complejidad computacional.
5. Teoría de modelos finitos.

Laboratorio de datos

Marcelo Arenas

- Profesor Titular (desde 1999).

Juan Reutter

- Profesor Asistente (desde 2013).

Cristian Riveros

- Profesor Asistente (desde 2013).

Domagoj Vrgoc

- Profesor Asistente (desde 2016).

Temas de investigación

Bases de datos:

- Relacional.
- BD de grafos.
- Intercambio de datos.

Semantic Web:

- RDF.
- SPARQL.

Teoría de la Computación:

- Teoría de automatas.
- Complejidad computacional.

Lógica para CS:

- Teoría de modelos finitos.
- Rep. del conocimiento.

Center for Semantic Web Research

CIWS

Centro de investigación
de la web semántica

Mas información en ciws.cl