

# Decidibilidad y Computabilidad

Alan Reyes-Figueroa

Teoría de la Computación

(Aula 26) 13.noviembre.2023

Decidibilidad  
Computabilidad  
Otros modelos

# Clases de Complejidad

- ◆ En la teoría de la computación, una *clase de complejidad* es un conjunto de problemas computacionales de complejidad relacionada basada en recursos.
- ◆ Los dos recursos más comúnmente analizados son el **tiempo** y la **memoria**.

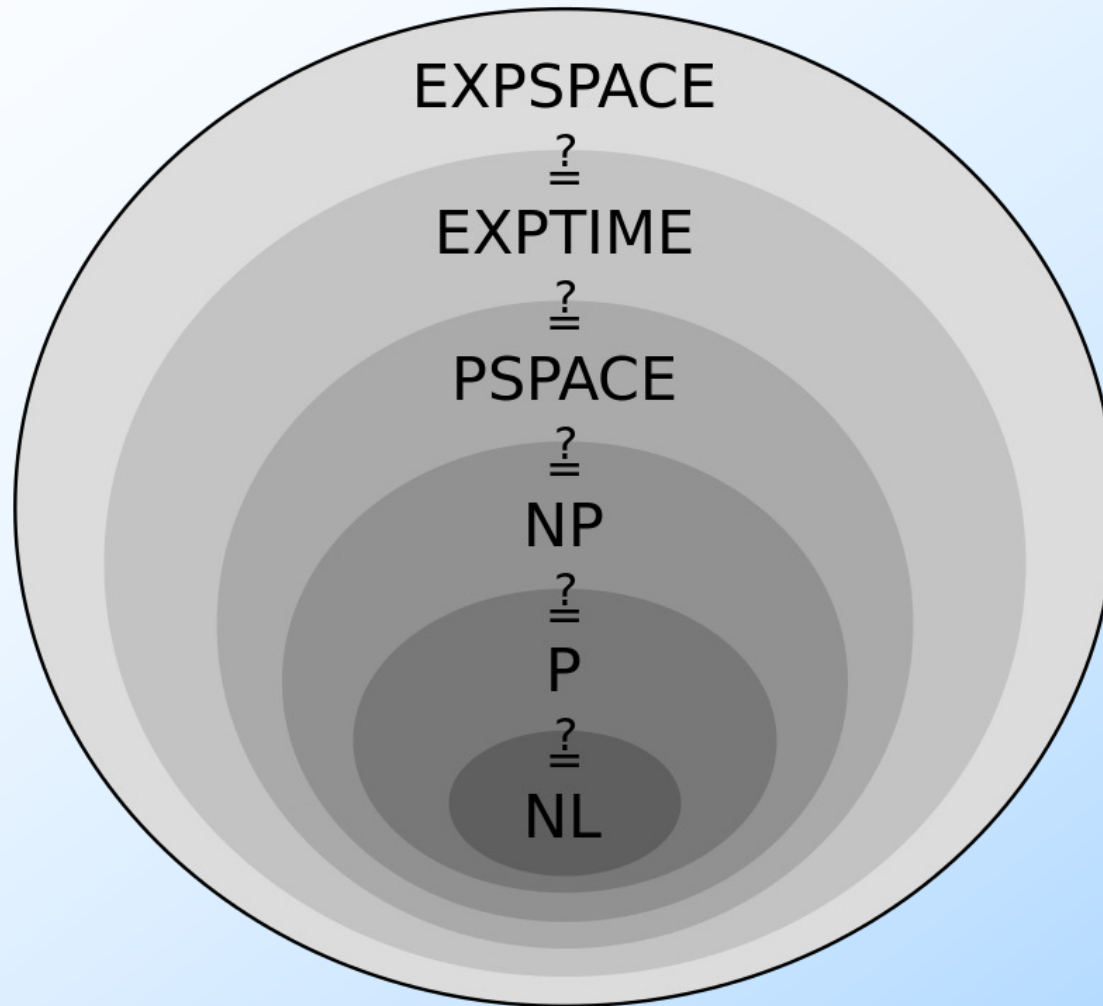
# Clases de Complejidad

- ◆ **Clase de complejidad:** se define en términos de un tipo de problema computacional, un modelo de computación y un recurso acotado (e.g. tiempo o memoria).
- ◆ La mayoría de las clases de complejidad consisten en problemas de decisión que se pueden resolver con una máquina de Turing y se diferencian por sus requisitos de tiempo o espacio (memoria).

# Clases de Complejidad

- ◆ Por ejemplo, la clase P: es el conjunto de problemas de decisión que se pueden resolver mediante una máquina de Turing determinista en tiempo polinomial.
- ◆ Existen muchas clases de complejidad definidas en términos de otros tipos de problemas (por ejemplo, problemas de conteo y problemas de funciones) y utilizando otros modelos de computación (e.g., MT probabilísticas, sistemas de prueba interactivos, circuitos booleanos y computadoras cuánticas).

# Clases de Complejidad



# Computabilidad

- ◆ La *teoría de la computabilidad*, o **teoría de la recursión**, es una rama de la lógica matemática, la informática y la teoría de la computación, iniciada en los 1930s con el estudio de las funciones computables y los grados de Turing.
- ◆ Preguntas básicas:
  - ◆ ¿Qué significa que una función sobre los números naturales sea computable?
  - ◆ ¿Cómo se pueden clasificar las funciones no computables en una jerarquía según su nivel de no computabilidad?

# Computabilidad

- ◆ La *computabilidad* es la capacidad de resolver un problema de manera eficaz.
- ◆ La computabilidad de un problema está íntimamente ligada a la existencia de un algoritmo para resolver el problema.
- ◆ Algoritmo = modelo computacional. Hoy en día para “formalizar” la noción de algoritmo se construye una abstracción (modelo) de una máquina computacional.

# Computabilidad

- ◆ Los modelos de computabilidad más ampliamente estudiados son:
  - ▶ Autómatas (DFA, NFA, PDA, ...)
  - ▶ Máquinas de Turing
  - ▶ Funciones recursivas  $\mu$
  - ▶ Cálculo lambda
  - ▶ Máquinas de Post
  - ▶ Máquinas registradoras
  - ▶ Máquinas de Turing multi-cinta
  - ▶ Máquinas de Turing probabilísticas
  - ▶ Computación cuántica

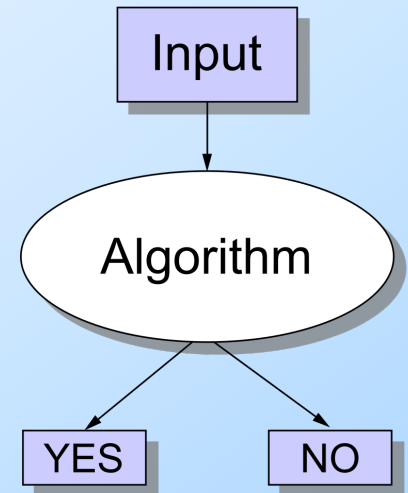


# Problemas

- ◆ Hay dos tipos principales de problemas:
  - ▶ Problemas de decisión
  - ▶ Un problema de computar una función.
- ◆ Otros tipos de problemas incluyen:
  - ▶ problemas de búsqueda,
  - ▶ problemas de optimización.
- ◆ Uno de los objetivos de la teoría de la computabilidad es determinar qué problemas, o clases de problemas, pueden resolverse en cada modelo de computación.

# Decidibilidad

- ◆ Un *problema de decisión* es un problema computacional que se puede plantear como una pregunta de sí o no a partir de los inputs.
- ◆ Un método para resolver un problema de decisión, dado en forma de algoritmo, se llama un **procedimiento de decisión**.
- ◆ Un problema de decisión que puede ser resuelto por un algoritmo se llama *decidable*.



# Funciones Computables

- ◆ Las *funciones computables* son los objetos básicos de estudio en la **teoría de la computabilidad**.
- ◆ Son el análogo formalizado de la noción intuitiva de algoritmos, en el sentido que una función es computable si existe un algoritmo que puede hacer el trabajo de la función: calcular su output.

# Funciones Computables

- ◆ La *computabilidad* de una función es una noción informal.
- ◆ Una forma de describirlo es decir que una función es computable si su valor puede obtenerse mediante un **procedimiento efectivo**.
- ◆ Una función  $f : \mathbb{N}^k \rightarrow \mathbb{N}$  es *computable* si y sólo si hay un procedimiento efectivo que, dada cualquier  $k$ -tupla  $\mathbf{x} \in \mathbb{N}^k$  de números naturales, producirá el valor  $f(\mathbf{x})$ .

# Funciones Computables

- ◆ Las máquinas de Turing tienen una caracterización (descripción) matemática precisa, y esto impone una limitante igualmente precisa sobre las funciones que pueden computar.
- ◆ Serán éstas las **funciones computables**. Identificamos al conjunto de funciones computables con el de las **efectivamente calculables**.

# Computabilidad

Problemas de computar funciones:

Potencia de las máquinas de estados finitos

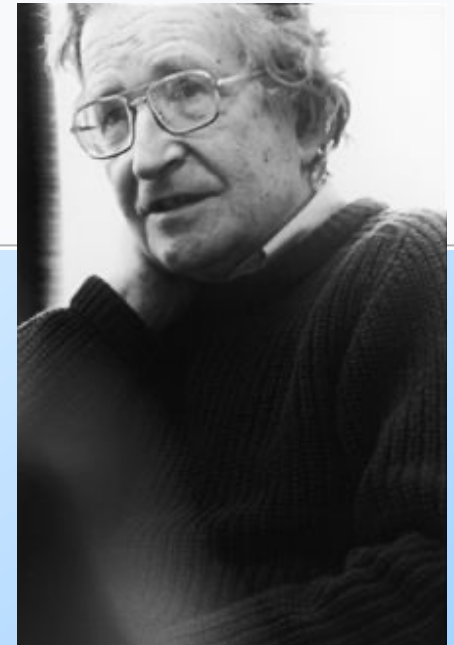
- ▶ *Lenguaje regular* = cualquier lenguaje que pueda ser aceptado por un autómata finito determinista (DFA).
- ▶ *Lenguaje libre de contexto* = cualquier lenguaje que pueda ser aceptado por un autómata de pila (PDA).
- ▶ *Lenguaje sensible al contexto* = cualquier lenguaje que pueda ser aceptado por un autómata linealmente acotado (LBA).
- ▶ *Lenguaje recursivamente enumerable* = aceptado por cualquier máquina de Turing.

# Jerarquía de Chomsky

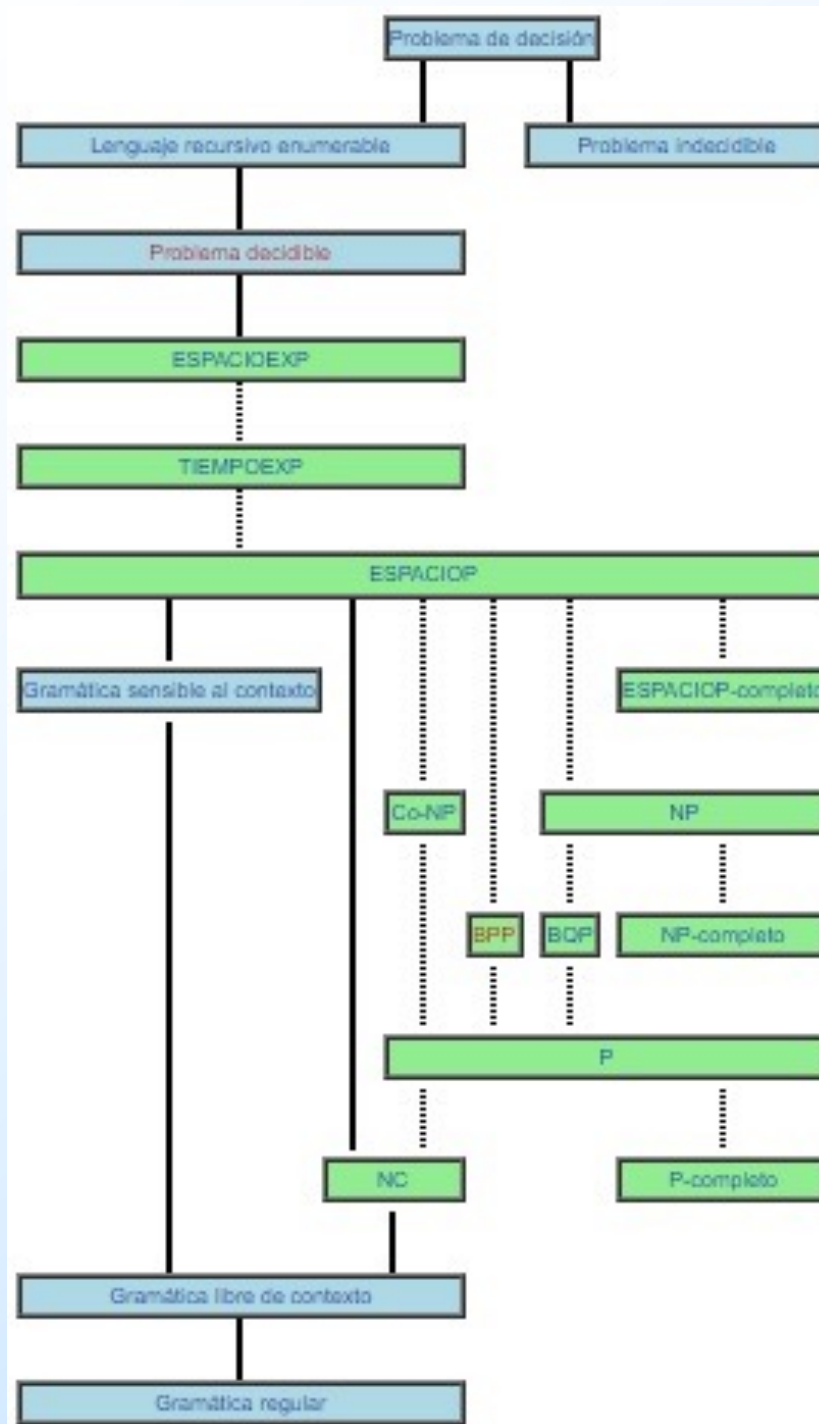
Tipo	Lenguaje	Autómata	Normas de producción de gramáticas	Ejemplos
0	recursivamente enumerable (LRE)	Máquina de Turing	$\alpha A \beta \rightarrow \delta$	$L = \{w   w \text{ describe una máquina de Turing}\}$
1	dependiente del contexto (LSC)	Autómata linealmente acotado	$\alpha A \beta \rightarrow \alpha \gamma \beta$	$L = \{a^n b^n c^n   n > 0\}$
2	independiente del contexto (LLC)	Autómata con pila	$A \rightarrow \gamma$	$L = \{a^n b^n   n > 0\}$
3	regular (LR)	Autómata finito	$A \rightarrow a\gamma$ $A \rightarrow aB$	$L = \{a^n   n \geq 0\}$

Significado de los símbolos:

- $a$  = terminal
- $A, B$  = no terminal
- $\alpha, \beta, \gamma$  = cadena de terminales y/o no terminales
  - $\alpha, \beta, \delta$  = cadena posiblemente vacía
  - $\gamma$  = cadena no vacía



- ◆ Clasificación de las gramáticas
- ◆ (Tipo 0, Tipo 1, Tipo 2, Tipo 3).





# Tesis de Church-Turing

- ◆ En la teoría de la computabilidad, la *tesis de Church-Turing* (también conocida como tesis de la computabilidad, tesis de Turing-Church, conjetura de Church-Turing, tesis de Church, conjetura de Church y tesis de Turing) es una tesis sobre la naturaleza de funciones computables.
- ◆ Establece que una función sobre los números naturales puede calcularse mediante un método efectivo si y solo si es computable por una máquina de Turing.

# Tesis de Church-Turing

- ◆ La tesis de Church-Turing conjetura que no existe un modelo efectivo de computación que pueda calcular más funciones matemáticas que una máquina de Turing.
- ◆ Hoy en día, los científicos de la computación han imaginado muchas variedades de hiperordenadores (*hypercomputers*), modelos de computación que van más allá de la computabilidad de Turing:
  - ◆ Random Turing machines,
  - ◆ quantum models, ...

# Otros modelos de computación

- ▶ Autómatas (DFA, NFA, PDA, ...)
  - ▶ Máquinas de Turing
  - ▶ Funciones recursivas  $\mu$
  - ▶ Cálculo lambda
- 
- ▶ Máquinas de Post
  - ▶ Máquinas registradoras
  - ▶ Máquinas de Turing multi-cinta
  - ▶ Máquinas de Turing probabilísticas
  - ▶ Computación cuántica