# Algorithmic game theory - IIC3810

Jorge Salas

Pontificia Universidad Católica

*jusalas@uc.cl*

25 de mayo de 2016

# Overview

1. Motivación
   - Problemas de conteo
   - Funciones
2. Clases de complejidad
   - #P
   - FP
   - FNP
   - TFNP
   - PPAD

# Problemas de conteo

- Sea $\varphi$ una fórmula proposicional. ¿Cuántas valuaciones la hacen verdadera?
- Dado un grafo $G$, ¿cuántos caminos hamiltonianos posee?
- Dado un grafo $G$ y dos nodos $s, t$. ¿Cuántos caminos simples conectan a $s$ con $t$?
- Entre otros.

# Funciones

- Sean $A$ y $B$ dos matrices de $n \times n$. Calcular $C = AB$.
- Dado un número natural $n$, encontrar su factor primo mayor.
- Sea $b \in \mathbb{N}$ y $n \in \mathbb{N}$, calcular $b^n$.
- Sea $w \in \{0,1\}^*$, calcular la cantidad de 1's en la palabra.
- Entre otros.

# Clases de complejidad

## Definición

Dada una máquina de Turing no determinista $M$, se define $\#M(w)$ como la cantidad de ejecuciones de aceptación para un input $w$.

## Definición (#P)

Sea $f : \Sigma^* \to \mathbb{N}$. Decimos que $f$ pertenece a la clase $\#P$ si existe una NTM $M$ que funciona en tiempo polinomial tal que $f(w) = \#M(w)$.

¿Qué tan difícil es la clase $\#P$?

# Clases de complejidad

**Definición**

Dada una máquina de Turing no determinista $M$, se define $\#M(w)$ como la cantidad de ejecuciones de aceptación para un input $w$.

**Definición (#P)**

Sea $f : \Sigma^* \to \mathbb{N}$. Decimos que $f$ pertenece a la clase $\#P$ si existe una NTM $M$ que funciona en tiempo polinomial tal que $f(w) = \#M(w)$.

¿Qué tan difícil es la clase $\#P$?

**Ejemplo**

#SAT pertenece a #P.

# Clases de complejidad

## Definición (FP)

Se define la clase $FP$ como todas las funciones $f(x)$ computables por una máquina de Turing **determinista** en tiempo polinomial.

## Ejemplo

Visto anteriormente.

¿Cuál es la relación entre $\#P$ y $FP$?

# Clases de complejidad

Para responder esta pregunta debemos restringirnos un poco.

**Definición**

Se define $FP_{\mathbb{N}}$ cómo la restricción de $FP$ a funciones con recorrido en $\mathbb{N}$.

¿Por qué hacer esta restricción?

# Clases de complejidad

**Teorema**

$FP_{\mathbb{N}} \subseteq \#P$

Demostración en pizarra.

# Clases de complejidad

**Teorema**

$FP_{\mathbb{N}} \subseteq \#P$

Demostración en pizarra.

Ahora tenemos una relación entre estas clases.

# Clases de complejidad

## Definición

Dadas $f, g$ en $\#P$. Decimos que existe una reducción parsimoniosa $h \in FP$ entre $f$ y $g$ si $f(x) = g(h(x))$.

# Clases de complejidad

## Definición

Dadas $f, g$ en $\#P$. Decimos que existe una reducción parsimoniosa $h \in FP$ entre $f$ y $g$ si $f(x) = g(h(x))$.

## Teorema

$\#SAT$ es $\#P-$Completo mediante reducción parsimoniosa.

## Demosotración

En pizarra.

# Clases de complejidad

## Definición

Sean $f, g$ en $\#P$. Decimos que $f$ es reducible a $g$ (en el sentido de Cook) si $f \in FP^g$.

## Teorema

$\#DNF$ es $\#P-$Completo mediante reducción de Cook.

# Clases de complejidad

## Definición

Sea $P(x, y)$ un predicado. $P$ es reconocible en tiempo polinomial si dado un certificado $(x, y)$, una máquina de turing determinista verifica $P(x, y)$ en tiempo polinomial.

# Clases de complejidad

## Definición

Sea $P(x, y)$ un predicado. $P$ es reconocible en tiempo polinomial si dado un certificado $(x, y)$ una máquina de turing determinista verifica $P(x, y)$ en tiempo polinomial.

## Definición

Decimos que $P$ es polinomialmente balanceado si dado que $P(x, y)$ se cumple, existe un polinomio $p$ tal que $|y| \leq p(|x|)$.

# Clases de complejidad

## Definición

Dado un predicado $P(x, y)$. Se define el problema $\Pi_P$: Dado $x \in \Sigma^*$, encontrar $y$ tal que $P(x, y)$.

# Clases de complejidad

## Definición

Dado un predicado $P(x, y)$. Se define el problema $\Pi_P$: Dado $x \in \Sigma^*$, encontrar $y$ tal que $P(x, y)$.

## Definición (FNP)

Se define la clase *FNP* como los problemas $\Pi_P$ donde $P(x, y)$ es un predicado reconocible en tiempo polinomial y polinomialmente balanceado.

# Clases de complejidad

## Definición

Dado un predicado $P(x, y)$. Se define el problema $\Pi_P$: Dado $x \in \Sigma^*$, encontrar $y$ tal que $P(x, y)$.

## Definición (FNP)

Se define la clase *FNP* como los problemas $\Pi_P$ donde $P(x, y)$ es un predicado reconocible en tiempo polinomial y polinomialmente balanceado.

## Ejemplo

- $P(x, y) := x = 2y, \quad y \in \mathbb{N}.$
- $P(\varphi, \sigma) := \sigma(\varphi) = 1.$

¿Cuál es la relación entre *FNP* y $\#P$?.

# Clases de complejidad

## Definición (TFNP)

Sea $P(x, y)$ un predicado. $\Pi_P$ pertenece a la clase *TFNP* si $\Pi_P$ pertenece a *FNP* y además para todo $x \in \Sigma^*$ existe $y \in \Sigma^*$ tal que $P(x, y)$ se cumple.

# Clases de complejidad

## Definición (TFNP)

Sea $P(x, y)$ un predicado. $\Pi_P$ pertenece a la clase *TFNP* si $\Pi_P$ pertenece a *FNP* y además para todo $x \in \Sigma^*$ existe $y \in \Sigma^*$ tal que $P(x, y)$ se cumple.

## Ejemplo

- $P(x, y) :=: x \in \mathbb{N}$, $y$ es la factorización en primos de $x$.
- $P(f, x, y) := f := \{0,1\}^n \to \{0,1\}^{n-1}$, $f(x) = f(y)$.

# Clases de complejidad

**Definición**

Sean $\Pi_P, \Pi_S$ en $FNP$. Decimos que $\Pi_P$ es reducible a $\Pi_S$ si existen funciones $f$ y $g$ en $FP$ tal que para todo $x \in \Sigma^*$, $(x, g(y)) \in P$ si y sólo si $(f(x), y) \in S$.

¿Existirá algún problema $TFNP-$Completo?

Cada problema $\Pi_P$ en *TFNP* debe tener algún tipo de "garantía" de su totalidad. La idea es clasificar los problemas en *TFNP* de acorde a esta "garantía".

# Clases de complejidad

**Lemma (argumento de paridad para grafos dirigidos)**

Si un grafo dirigido (finito) posee un nodo desbalanceado (cuyo grado de incidencia es distinto a su grado de salida), entonces posee otro nodo desbalanceado.

# Clases de complejidad

**Lema (argumento de paridad para grafos dirigidos)**

Si un grafo dirigido (finito) posee un nodo desbalanceado (cuyo grado de incidencia es distinto a su grado de salida), entonces posee otro nodo desbalanceado.

**Definición (*PPAD*)**

Definimos la clase *PPAD* como los problemas $\Pi_P \in TFNP$ tal que la existencia de una solución se demuestra utilizando el lema de paridad para grafos dirigidos.

¿Qué tan difícil parece esta clase?

# Clases de complejidad

**Definición**

END OF THE LINE: Dados dos circuitos booleanos $S$ y $P$, cada uno con $n$ inputs y $n$ outputs, tal que $S(P(0^n)) \neq 0^n$ y $P(S(0^n)) = 0^n$. Encontrar $x \in \{0,1\}^n$ tal que $S(P(x)) \neq x \neq 0^n$ o $P(S(x)) \neq x$.

¿Cómo se utiliza el argumento de paridad?

# Clases de complejidad

**Definición (*PPAD* alternativa)**

Se define la clase *PPAD* como todos los problemas $\Pi_P \in TFNP$ tal que $\Pi_P$ es reducible al problema $\mathrm{END\ OF\ THE\ LINE}$.

# References


Leslie G. Valiant (1979)

The Complexity of Enumeration and Reliability Problems

*Siam J. Comput.* 8(3), 410 – 421.

Christos H. Papadimitriou (1994)

On the complexity of the Parity Argument and Other Inefficient Proof of Existence

*Journal of Computer and System Sciences* 48, 498 – 532.

Leslie G. Valiant (1977)

The Complexity of Computing the Permanent

*Theorical Computer Science* 8, 189 – 201.

# References


N. Meggido and L.H. Papadimitriou (1991)

On Total Functions, Existence Theorems and Computational Complexity

*Theorical Computer Science* 81, 317 – 324.

Noam Nisan et al. (2007)

Algorithmic Game Theory

Constantinos Daskalakis (2009)

Nash Equilibria: Complexity, Symmetries, and Approximation

*CSAIL, MIT*