# Introdução à Complexidade Computacional

## Problemas fáceis vs Problemas difíceis

**Professor Murilo V. G. da Silva**

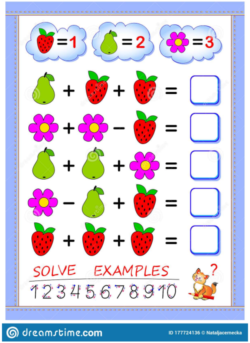
Departamento de Informática

Universidade Federal do Paraná

2025 / 2

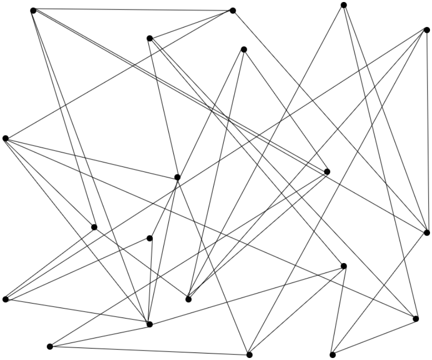
# Dificuldade de Problemas

Alguns problemas são mais difíceis que outros?



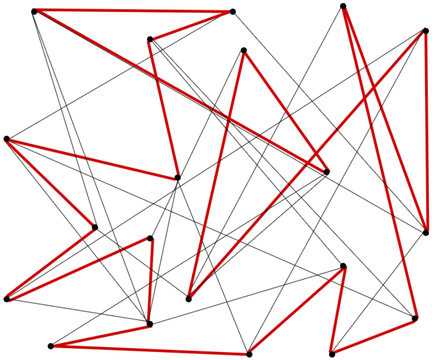

# Dificuldade de Problemas

Considere o problema do Grafo Hamiltoniano:



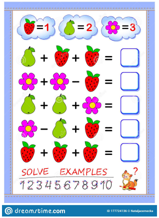
# Dificuldade de Problemas

Considere o problema do Grafo Hamiltoniano:

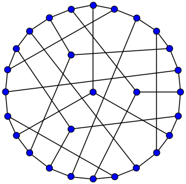


# Dificuldade de Problemas

Decisão polinomial *vs* Verificação polinomial *vs* Exponencial (até para verificar?)




Soma de inteiros



Grafo Hamiltoniano

Xadrez Generalizado

# Classes de Complexidade

Exponencial é ruim?

Exponencial é ruim? Explicação visual:

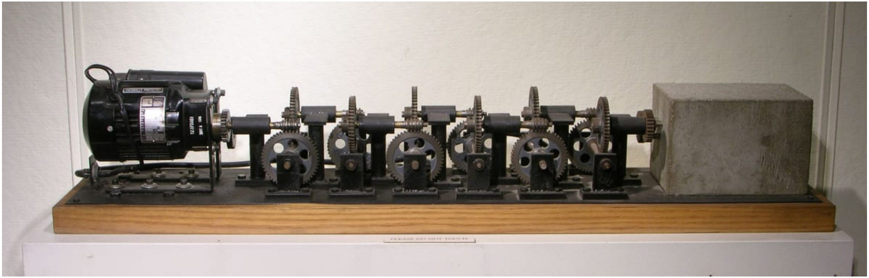
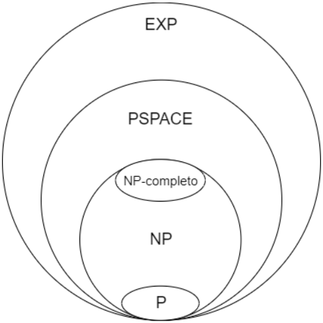


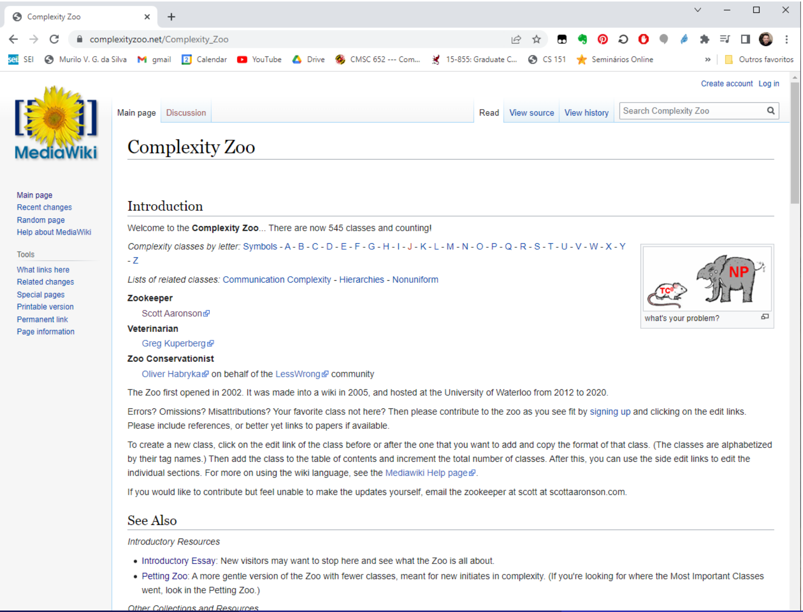
Figura: Barak & Arora, pág. 5

# Classes de Complexidade

Algumas classes de complexidade que vamos estudar:



# Existem muito mais classes de complexidade!



Complexity Zoo

complexityzoo.net/Complexity_Zoo

Create account Log in

Main page | Discussion | Read | View source | View history | Search Complexity Zoo

Main page
Recent changes
Random page
Help about MediaWiki

Tools

What links here
Related changes
Special pages
Printable version
Permanent link
Page information

# Complexity Zoo

## Introduction

Welcome to the **Complexity Zoo**... There are now 545 classes and counting!

*Complexity classes by letter:* Symbols - A - B - C - D - E - F - G - H - I - J - K - L - M - N - O - P - Q - R - S - T - U - V - W - X - Y - Z

*Lists of related classes:* Communication Complexity - Hierarchies - Nonuniform



what's your problem?

**Zookeeper**

Scott Aaronson

**Veterinarian**

Greg Kuperberg

**Zoo Conservationist**

Oliver Habryka on behalf of the LessWrong community

The Zoo first opened in 2002. It was made into a wiki in 2005, and hosted at the University of Waterloo from 2012 to 2020.

Errors? Omissions? Misattributions? Your favorite class not here? Then please contribute to the zoo as you see fit by signing up and clicking on the edit links. Please include references, or better yet links to papers if available.

To create a new class, click on the edit link of the class before or after the one that you want to add and copy the format of that class. (The classes are alphabetized by their tag names.) Then add the class to the table of contents and increment the total number of classes. After this, you can use the side edit links to edit the individual sections. For more on using the wiki language, see the Mediawiki Help page.

If you would like to contribute but feel unable to make the updates yourself, email the zookeeper at scott at scottaaronson.com.

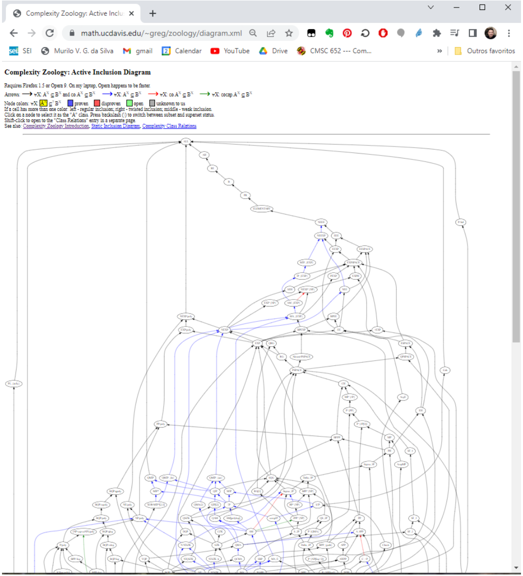
## See Also

*Introductory Resources*

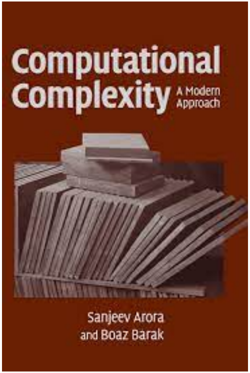
- Introductory Essay: New visitors may want to stop here and see what the Zoo is all about.
- Petting Zoo: A more gentle version of the Zoo with fewer classes, meant for new initiates in complexity. (If you're looking for where the Most Important Classes went, look in the Petting Zoo.)

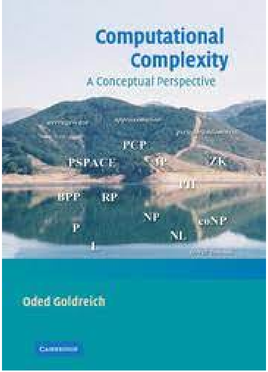
*Other Collections and Resources*

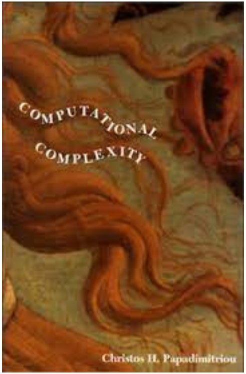
# Existem muito mais classes de complexidade!



# Livros especializados na área




**Barak/Arora** Goldreich **Papadimitriou**

# Complexidade Computacional

(1) Dados dois números inteiros, calcular a soma dos dois números;
(2) Dado um tabuleiro de xadrez em que as peças brancas têm a vez de jogar, determinar a jogada ótima para as peças brancas.

# Complexidade Computacional

(1) Dados dois números inteiros, calcular a soma dos dois números;

(2) Dado um tabuleiro de xadrez em que as peças brancas têm a vez de jogar, determinar a jogada ótima para as peças brancas.

- Parece "óbvio" que $z = x + y$ é um problema simples.

# Complexidade Computacional

(1) Dados dois números inteiros, calcular a soma dos dois números;

(2) Dado um tabuleiro de xadrez em que as peças brancas têm a vez de jogar, determinar a jogada ótima para as peças brancas.

- Parece "óbvio" que $z = x + y$ é um problema simples.
- Porém, $z$ tem pelo menos 64 dígitos (existem $10^{64}$ números com o tamanho de $z$).

# Complexidade Computacional

(1) Dados dois números inteiros, calcular a soma dos dois números;

(2) Dado um tabuleiro de xadrez em que as peças brancas têm a vez de jogar, determinar a jogada ótima para as peças brancas.

- Parece "óbvio" que $z = x + y$ é um problema simples.
- Porém, $z$ tem pelo menos 64 dígitos (existem $10^{64}$ números com o tamanho de $z$).
- Mais importante: Somar com $n$ dígitos *vs* "xadrez generalizado" (tamanho $n$).

# Complexidade Computacional

(1) Dados dois números inteiros, calcular a soma dos dois números;

(2) Dado um tabuleiro de xadrez em que as peças brancas têm a vez de jogar, determinar a jogada ótima para as peças brancas.

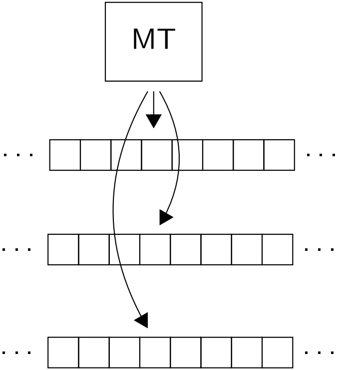
- Parece "óbvio" que $z = x + y$ é um problema simples.
- Porém, $z$ tem pelo menos 64 dígitos (existem $10^{64}$ números com o tamanho de $z$).
- Mais importante: Somar com $n$ dígitos *vs* "xadrez generalizado" (tamanho $n$).
- Obs: Não confundir *problemas indecidíveis* com *problemas intratáveis*.

# Modelo de Computação

Máquina de Turing com fita de entrada, de trabalho e de saída.

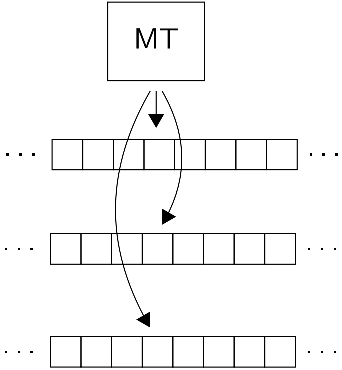
# Modelo de Computação

Máquina de Turing com fita de entrada, de trabalho e de saída.



# Modelo de Computação

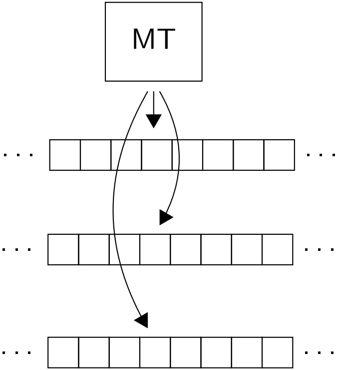
Máquina de Turing com fita de entrada, de trabalho e de saída.




Neste modelo quando escrevemos $M(x) = y$ queremos dizer:

# Modelo de Computação

Máquina de Turing com fita de entrada, de trabalho e de saída.




Neste modelo quando escrevemos $M(x) = y$ queremos dizer:

- Quando $x$ é colocada na primeira fita, a máquina termina com $y$ na terceira fita.

# Complexidade de Tempo de Máquinas de Turing

Seja $M$ uma MT,

## Complexidade de tempo

A complexidade de tempo de $M$ é uma função $t_{\mathrm{M}} : \mathbb{N} \to \mathbb{N}$ tal que, para qualquer string de entrada de tamanho $n$, a máquina para depois de executar no máximo $t_{\mathrm{M}}(n)$ transições.

# Complexidade de Tempo de Máquinas de Turing

Seja $M$ uma MT,

## Complexidade de tempo

A complexidade de tempo de $M$ é uma função $t_{\mathrm{M}} : \mathbb{N} \to \mathbb{N}$ tal que, para qualquer string de entrada de tamanho $n$, a máquina para depois de executar no máximo $t_{\mathrm{M}}(n)$ transições.

Exemplo: Se $\forall w \in \Sigma^*$ com $|w| = n$, $M$ sempre para depois de fazer, no máximo, $n^2 + 3n$ transições, dizemos que a complexidade de tempo de $M$ é $n^2 + 3n$.

# Complexidade de Espaço de Máquinas de Turing

Seja $M$ uma MT,

## Complexidade de espaço

A complexidade de espaço de $M$ é uma função $s_{\mathrm{M}} : \mathbb{N} \to \mathbb{N}$ tal que, para qualquer string de entrada de tamanho $n$, a máquina $M$ para usando no máximo $s_{\mathrm{M}}(n)$ posições da fita 2.

# Complexidade de Espaço de Máquinas de Turing

Seja $M$ uma MT,

## Complexidade de espaço

A complexidade de espaço de $M$ é uma função $s_{\mathrm{M}} : \mathbb{N} \rightarrow \mathbb{N}$ tal que, para qualquer string de entrada de tamanho $n$, a máquina $M$ para usando no máximo $s_{\mathrm{M}}(n)$ posições da fita 2.

Exemplo: Se $\forall w \in \Sigma^*$ com $|w| = n$, $M$ sempre para usando no máximo $\log n + 7$ posições da fita 2, dizemos que a complexidade de espaço de $M$ é $\log n + 7$.

# Mais definições

## Notação: $poli(n)$

Se $f(n) = O(n^r)$ para algum $r \in \mathbb{N}$ constante, então dizemos que $f(n) = poli(n)$.

# Mais definições

## Notação: *poli*$(n)$

Se $f(n) = O(n^r)$ para algum $r \in \mathbb{N}$ constante, então dizemos que $f(n) = poli(n)$.

## MT com complexidade polinomial

Se $M$ tem complexidade de tempo $poli(n)$, então dizemos que *$M$ é polinomial*.

Se $M$ tem complexidade de espaço $poli(n)$, dizemos que *$M$ é de espaço polinomial*.

# Mais definições

## Notação: *poli*$(n)$

Se $f(n) = O(n^r)$ para algum $r \in \mathbb{N}$ constante, então dizemos que $f(n) = poli(n)$.

## MT com complexidade polinomial

Se $M$ tem complexidade de tempo $poli(n)$, então dizemos que *$M$ é polinomial*.

Se $M$ tem complexidade de espaço $poli(n)$, dizemos que *$M$ é de espaço polinomial*.

## Complexidade em MTs não determinísticas

Uma MTN $N$ é polinomial se dada uma entrada de tamanho $n$, *todos os ramos da árvore de computações possíveis tem profundidade* $poli(n)$.

Note: Independente das escolhas não determinísticas, ela sempre faz $poli(n)$ transições

# Decidindo problemas

Seja $L$ uma linguagem.

# Decidindo problemas

Seja $L$ uma linguagem.

## Decisão em tempo polinomial

- Se $\exists$ MT polinomial que decide $L$, então dizemos que $L$ pode ser *decidida deterministicamente em tempo polinomial*
- Se $\exists$ MTN polinomial que decide $L$, então dizemos que $L$ pode ser *decidida não deterministicamente em tempo polinomial*.

# Decidindo problemas

Seja $L$ uma linguagem.

## Decisão em tempo polinomial

- Se $\exists$ MT polinomial que decide $L$, então dizemos que $L$ pode ser *decidida deterministicamente em tempo polinomial*
- Se $\exists$ MTN polinomial que decide $L$, então dizemos que $L$ pode ser *decidida não deterministicamente em tempo polinomial*.

## Decisão em espaço polinomial

- Se $\exists$ MT de espaço polinomial que decide $L$, então dizemos que $L$ pode ser *decidida deterministicamente em espaço polinomial*
- Se $\exists$ MTN de espaço polinomial que decide $L$, então dizemos que $L$ pode ser *decidida não deterministicamente em espaço polinomial*.

# Decidindo problemas

Seja $L$ uma linguagem.

## Decisão em tempo polinomial

- Se $\exists$ MT polinomial que decide $L$, então dizemos que $L$ pode ser *decidida deterministicamente em tempo polinomial*
- Se $\exists$ MTN polinomial que decide $L$, então dizemos que $L$ pode ser *decidida não deterministicamente em tempo polinomial*.

## Decisão em espaço polinomial

- Se $\exists$ MT de espaço polinomial que decide $L$, então dizemos que $L$ pode ser *decidida deterministicamente em espaço polinomial*
- Se $\exists$ MTN de espaço polinomial que decide $L$, então dizemos que $L$ pode ser *decidida não deterministicamente em espaço polinomial*.

Decisão em tempo ou espaço exponencial:

# Decidindo problemas

Seja $L$ uma linguagem.

## Decisão em tempo polinomial

- Se $\exists$ MT polinomial que decide $L$, então dizemos que $L$ pode ser *decidida deterministicamente em tempo polinomial*
- Se $\exists$ MTN polinomial que decide $L$, então dizemos que $L$ pode ser *decidida não deterministicamente em tempo polinomial*.

## Decisão em espaço polinomial

- Se $\exists$ MT de espaço polinomial que decide $L$, então dizemos que $L$ pode ser *decidida deterministicamente em espaço polinomial*
- Se $\exists$ MTN de espaço polinomial que decide $L$, então dizemos que $L$ pode ser *decidida não deterministicamente em espaço polinomial*.

Decisão em tempo ou espaço exponencial: definições análogas

# As classes **P**, **NP** e **PSPACE** e **EXP**

## A classe **P**

Linguagens decidíveis deterministicamente em tempo polinomial.

# As classes **P**, **NP** e **PSPACE** e **EXP**

## A classe **P**

Linguagens decidíveis deterministicamente em tempo polinomial.

## A classe **NP**

Linguagens decidíveis não deterministicamente em tempo polinomial.

# As classes **P**, **NP** e **PSPACE** e **EXP**

## A classe **P**

Linguagens decidíveis deterministicamente em tempo polinomial.

## A classe **NP**

Linguagens decidíveis não deterministicamente em tempo polinomial.

## A classe **EXP**

Linguagens decidíveis deterministicamente em tempo exponencial.

# As classes **P**, **NP** e **PSPACE** e **EXP**

### A classe **P**

Linguagens decidíveis deterministicamente em tempo polinomial.

### A classe **NP**

Linguagens decidíveis não deterministicamente em tempo polinomial.

### A classe **EXP**

Linguagens decidíveis deterministicamente em tempo exponencial.

### A classe **PSPACE**

Linguagens decidíveis deterministicamente em espaço polinomial.

# O problema **P** *vs* **NP**

**Teorema**

$\mathsf{P} \subseteq \mathsf{NP}$

**Teorema**

$\mathrm{P} \subseteq \mathrm{NP}$

**Prova:** Seja $L \in \mathrm{P}$.

**Teorema**

P $\subseteq$ NP

**Prova:** Seja $L \in$ P.

1 Existe uma MT polinomial $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ que decide $L$.

# O problema **P** *vs* **NP**

**Teorema**

$\mathsf{P} \subseteq \mathsf{NP}$

**Prova:** Seja $L \in \mathsf{P}$.

1. Existe uma MT polinomial $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ que decide $L$.
2. Agora considere a Máquina de Turing não determinística $N = (Q, \Sigma, \Gamma, (\delta, \delta), q_0, B, F)$.

# O problema P *vs* NP

**Teorema**

P $\subseteq$ NP

**Prova:** Seja $L \in$ P.

1 Existe uma MT polinomial $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ que decide $L$.

2 Agora considere a Máquina de Turing não determinística $N = (Q, \Sigma, \Gamma, (\delta, \delta), q_0, B, F)$.

3 A máquina $N$ comporta-se exatamente da mesma maneira que $M$, portanto $N$ também decide $L$ em tempo polinomial.

# O problema **P** *vs* **NP**

**Teorema**

P $\subseteq$ NP

**Prova:** Seja $L \in$ P.

1. Existe uma MT polinomial $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ que decide $L$.
2. Agora considere a Máquina de Turing não determinística $N = (Q, \Sigma, \Gamma, (\delta, \delta), q_0, B, F)$.
3. A máquina $N$ comporta-se exatamente da mesma maneira que $M$, portanto $N$ também decide $L$ em tempo polinomial.
4. Logo $L \in$ NP e consquentemente P $\subseteq$ NP.

# O problema **P** *vs* **NP**

**Teorema**

$\mathrm{P} \subseteq \mathrm{NP}$

**Prova:** Seja $L \in \mathrm{P}$.

1. Existe uma MT polinomial $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ que decide $L$.
2. Agora considere a Máquina de Turing não determinística $N = (Q, \Sigma, \Gamma, (\delta, \delta), q_0, B, F)$.
3. A máquina $N$ comporta-se exatamente da mesma maneira que $M$, portanto $N$ também decide $L$ em tempo polinomial.
4. Logo $L \in \mathrm{NP}$ e consquentemente $\mathrm{P} \subseteq \mathrm{NP}$.

- $\mathrm{P} \neq \mathrm{NP}$?

# O problema **P** *vs* **NP**

**Teorema**

P $\subseteq$ NP

**Prova:** Seja $L \in$ P.

1. Existe uma MT polinomial $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ que decide $L$.
2. Agora considere a Máquina de Turing não determinística $N = (Q, \Sigma, \Gamma, (\delta, \delta), q_0, B, F)$.
3. A máquina $N$ comporta-se exatamente da mesma maneira que $M$, portanto $N$ também decide $L$ em tempo polinomial.
4. Logo $L \in$ NP e consquentemente P $\subseteq$ NP.

- P $\neq$ NP?
- E as classes P-space e NP-space?