Introdução à Complexidade Computacional

Parte 1

Professor Murilo V. G. da Silva

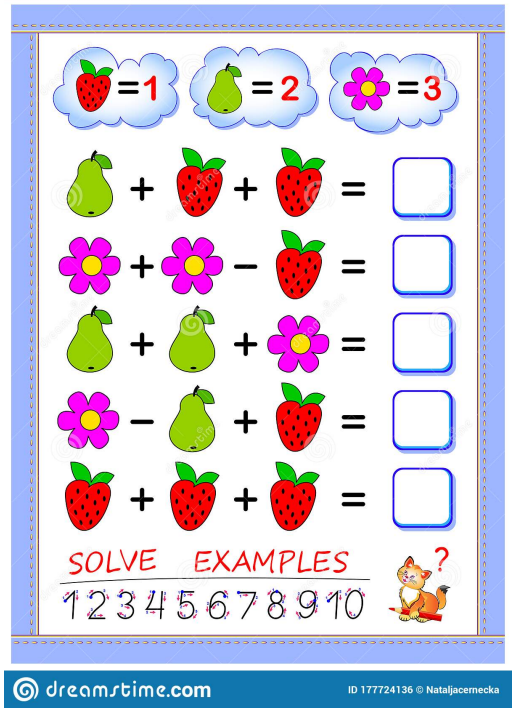
Departamento de Informática

Universidade Federal do Paraná

2025 / 2

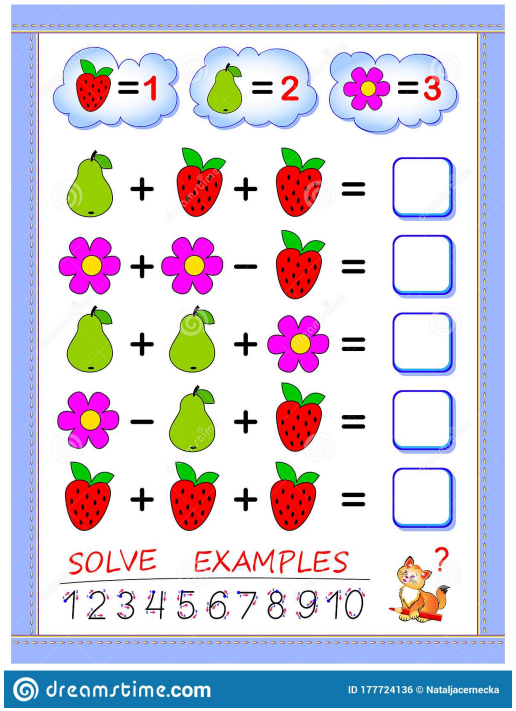
Dificuldade de Problemas

Alguns problemas são mais difíceis que outros?



Dificuldade de Problemas

Alguns problemas são mais difíceis que outros?

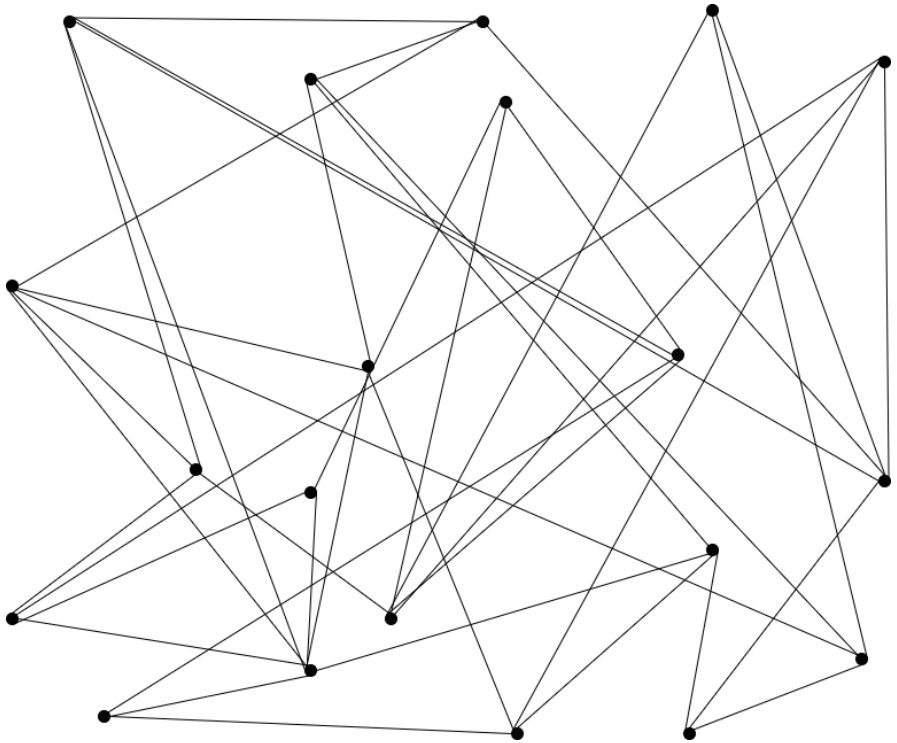


Dado (x, y, z), é verdade que $x + y = z$?

Existe xeque-mate garantido para as brancas?

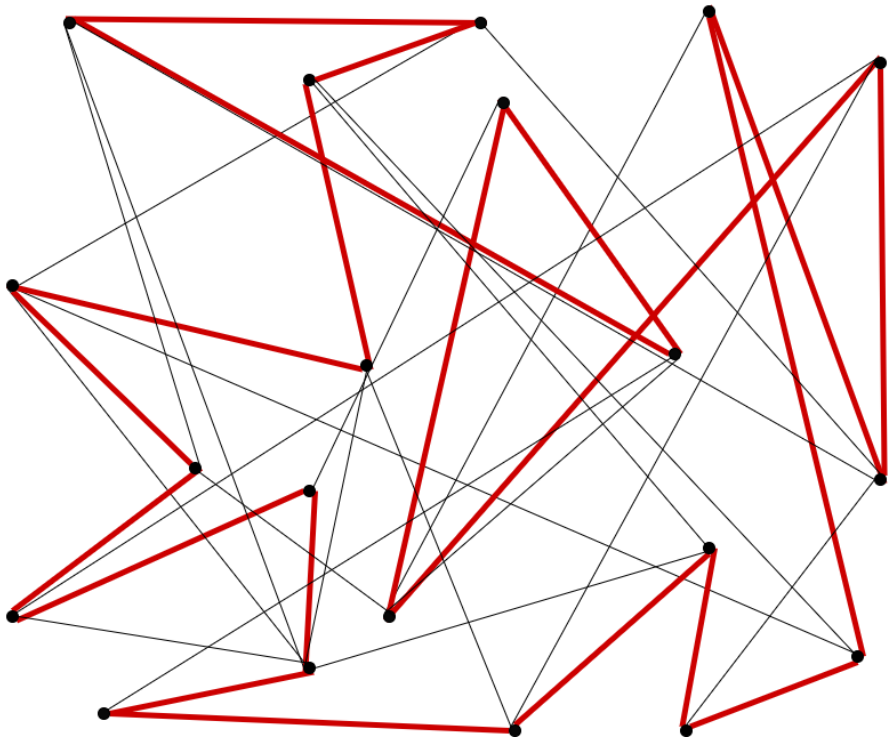
Dificuldade de Problemas

Considere o problema do Grafo Hamiltoniano:



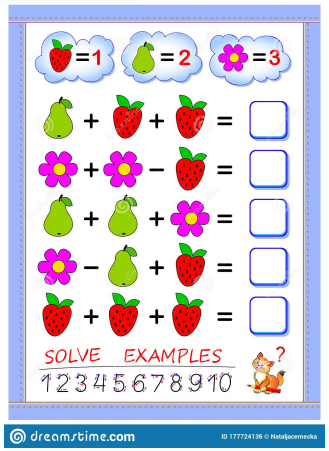
Dificuldade de Problemas

Considere o problema do Grafo Hamiltoniano:

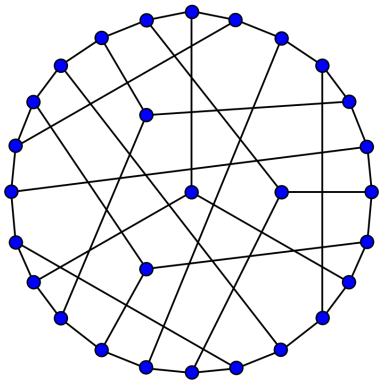


Dificuldade de Problemas

Decisão polinomial *vs* Verificação polinomial *vs* Exponencial



Soma de inteiros



Grafo Hamiltoniano

Xadrez Generalizado

Classes de Complexidade

Exponencial é ruim?

Classes de Complexidade

Exponencial é ruim? Explicação visual:

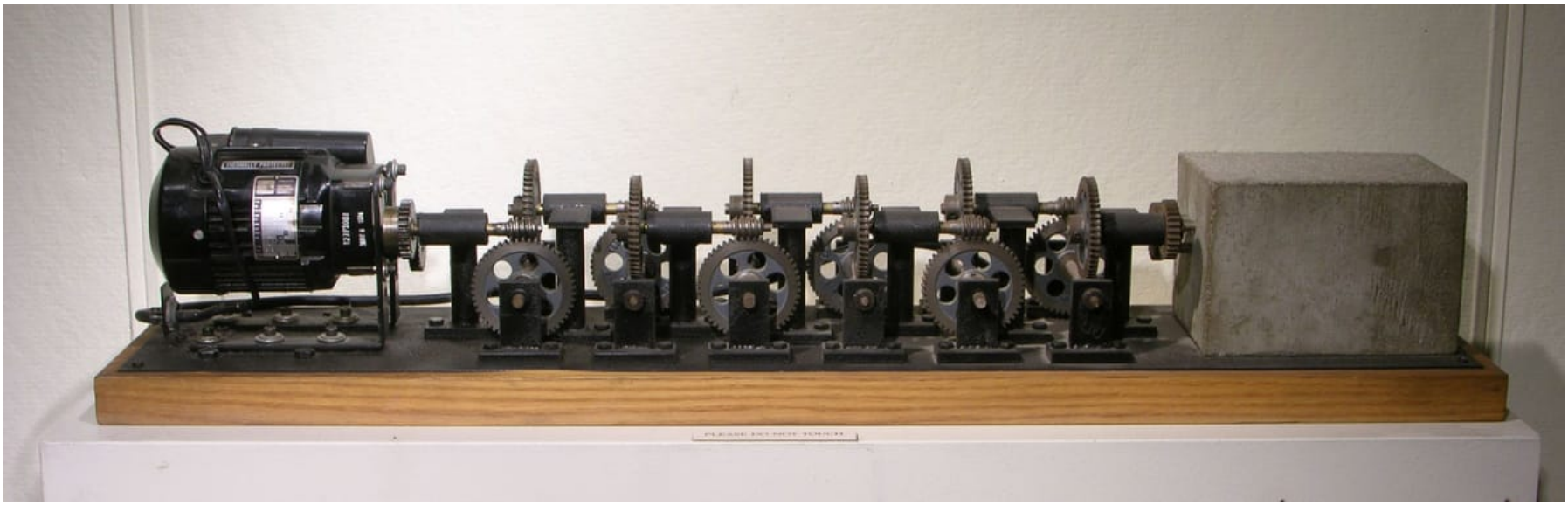
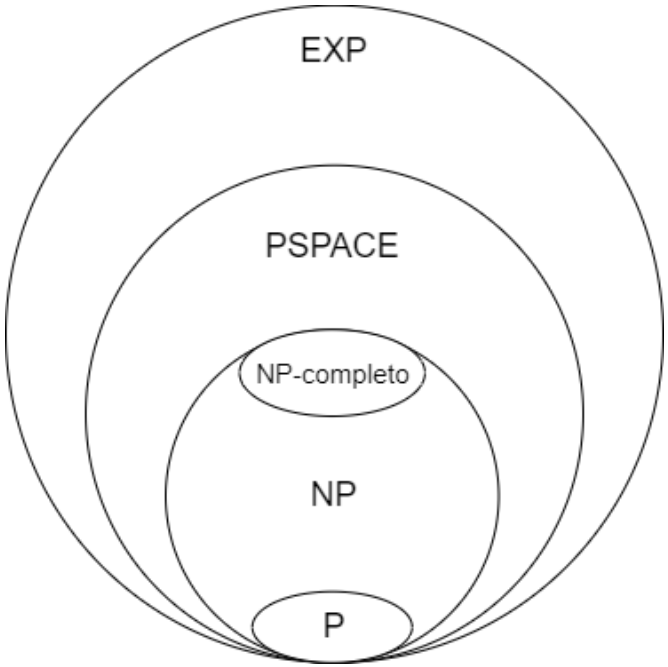


Figura: Barak & Arora, pág. 5

Classes de Complexidade

Algumas classes de complexidade que vamos estudar:



Existem muito mais classes de complexidade!

Complexity Zoo

complexityzoo.net/Complexity_Zoo

SEI · Murilo V. G. da Silva · gmail · Calendar · YouTube · Drive · CMSC 652 --- Com... · 15-855: Graduate C... · CS 151 · Seminários Online · Outros favoritos

Create account Log in

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

MediaWiki

Main page
Recent changes
Random page
Help about MediaWiki

Tools

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

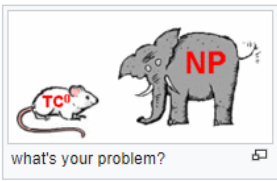
Complexity Zoo

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.

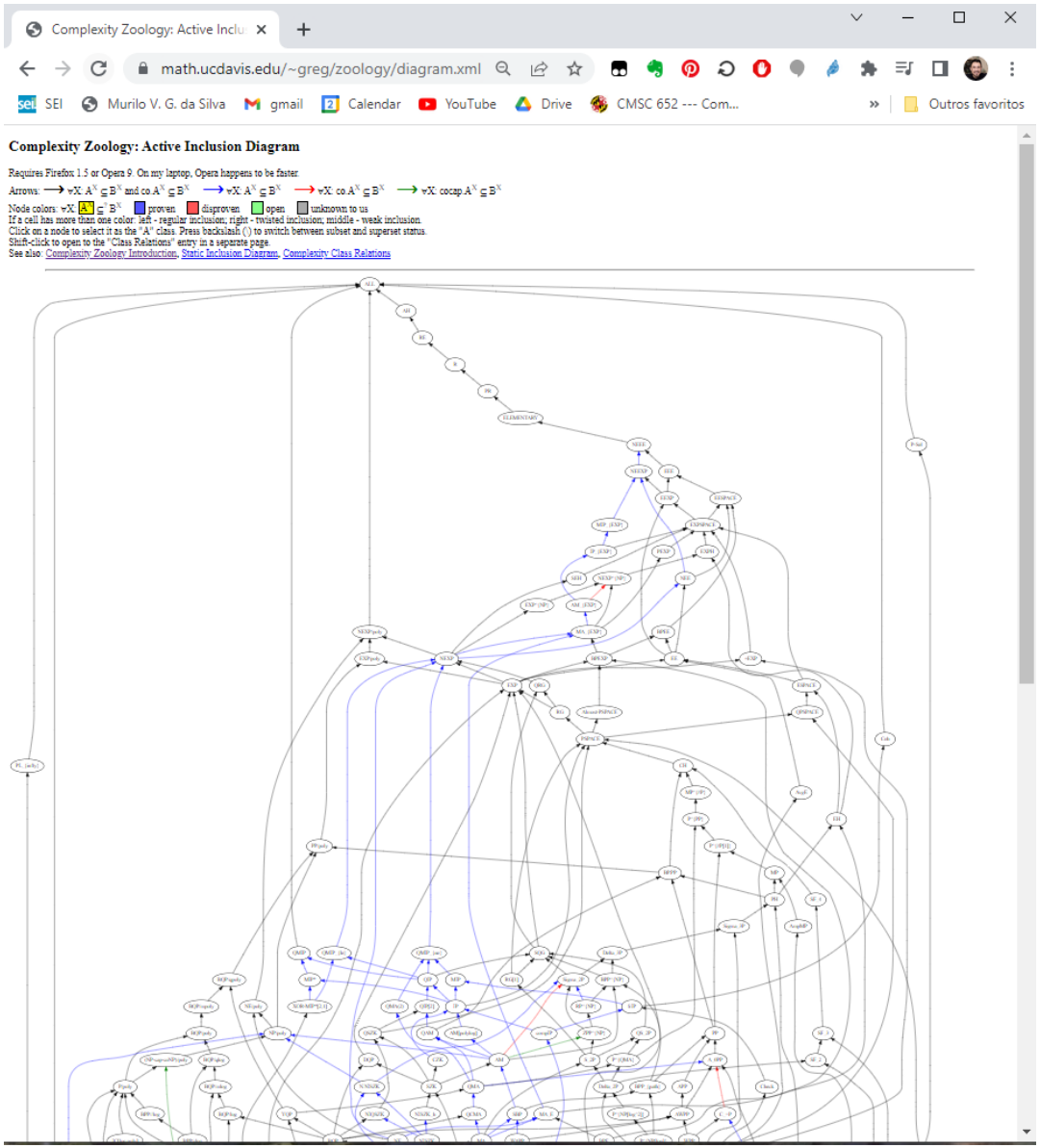
See Also

Introductory Resources

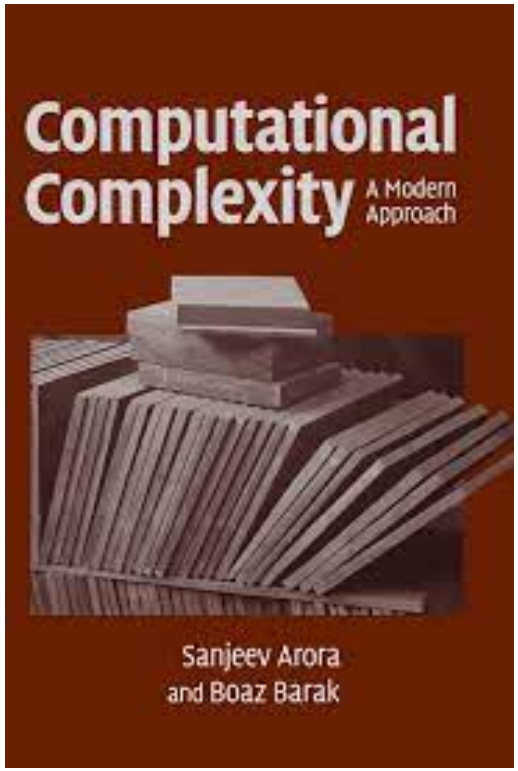
- Introductory Essay: New visitors may want to stop here and see what the Zoo is all about.
- Petting Zoo: A more gentle version of the Zoo with fewer classes, meant for new initiates in complexity. (If you're looking for where the Most Important Classes went, look in the Petting Zoo.)

Other Collections and Resources

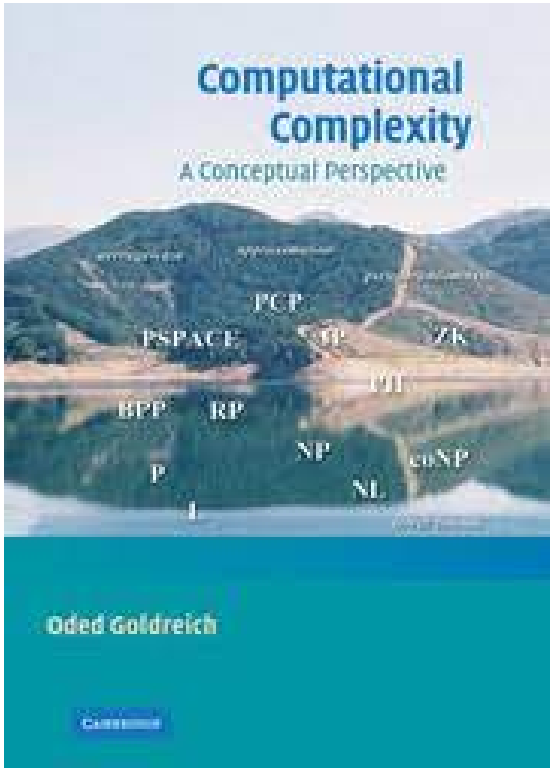
Existem muito mais classes de complexidade!



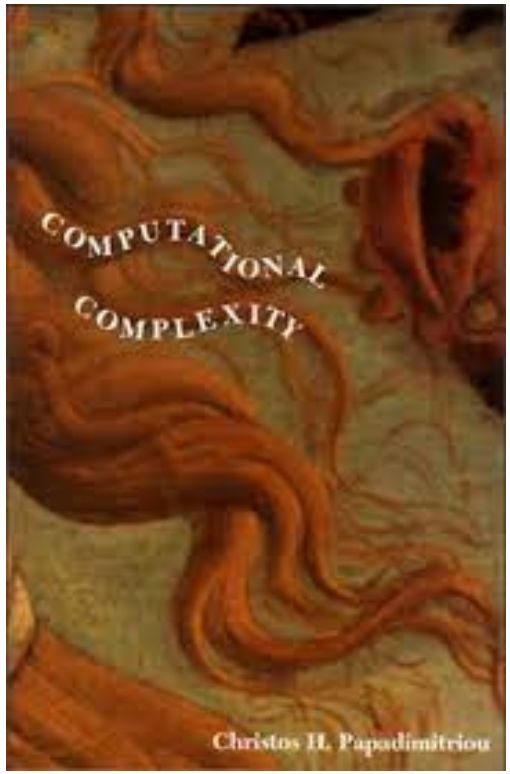
Livros especializados na área



Barak/Arora



Goldreich



Papadimitriou

Complexidade Computacional

Considere os problemas de busca:

Complexidade Computacional

Considere os problemas de busca:

(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

Considere os problemas de busca:

(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 encontrar z, tal que $z = x + y$ é um problema simples.

Complexidade Computacional

Considere os problemas de busca:

(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 encontrar z, tal que $z = x + y$ é um problema simples.
- Porém, z tem pelo menos 64 dígitos

Complexidade Computacional

Considere os problemas de busca:

(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 encontrar z, tal 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

Considere os problemas de busca:

(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 encontrar z, tal 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" ($n \times n$).

Complexidade Computacional

Considere os problemas de busca:

(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.

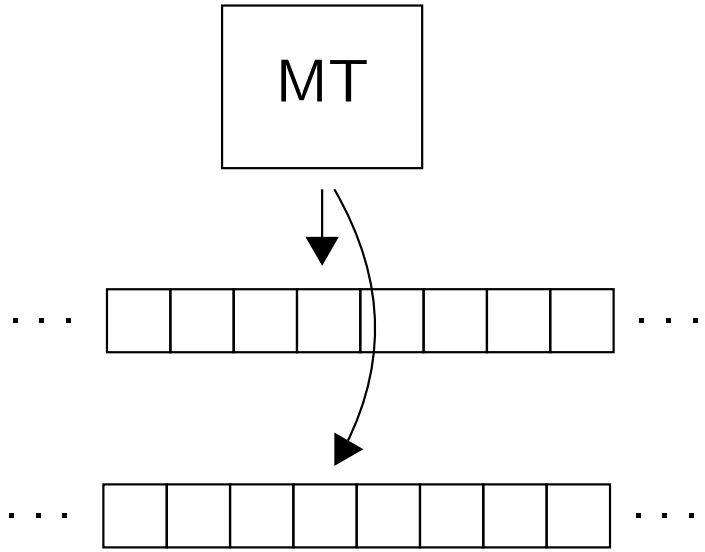
- Parece "óbvio" que encontrar z, tal que $z = x + y$ é um problema simples.
- Porém, z tem pelo menos 64 dígitos (existem 10^{64} números com o tamanho de z).
- Mais importante: Somar com n dígitos *vs* "xadrez generalizado" ($n \times n$).
- Obs: Não confundir *problemas indecidíveis* com *problemas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

Modelo de Computação

Máquina de Turing com fita de entrada (apenas para leitura) e uma fita de trabalho.

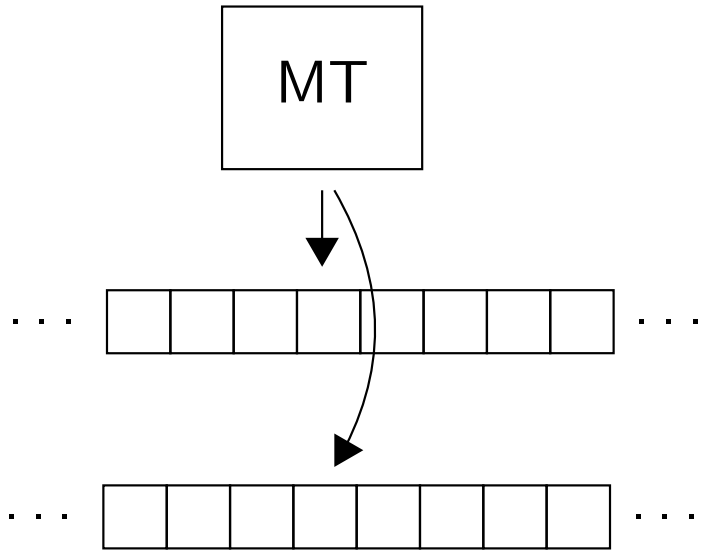
Modelo de Computação

Máquina de Turing com fita de entrada (apenas para leitura) e uma fita de trabalho.



Modelo de Computação

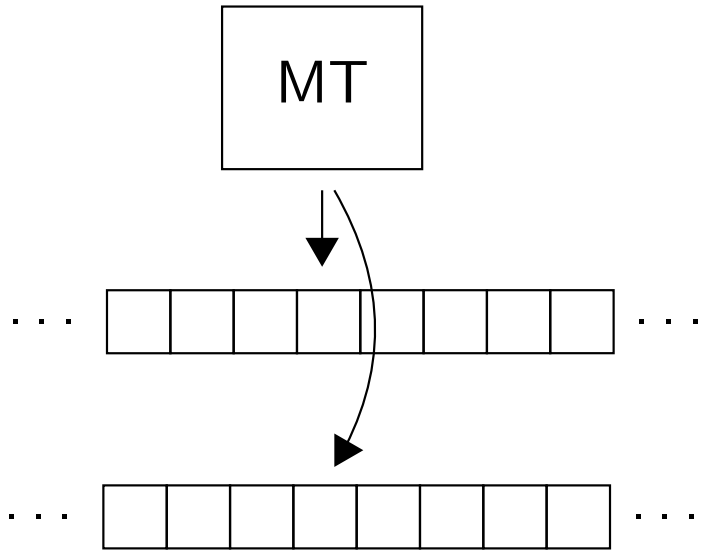
Máquina de Turing com fita de entrada (apenas para leitura) e uma fita de trabalho.



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

Modelo de Computação

Máquina de Turing com fita de entrada (apenas para leitura) e uma fita de trabalho.



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

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

Complexidade de Tempo de Máquinas de Turing

Seja M uma MT,

Complexidade de Tempo de Máquinas de Turing

Seja M uma MT,

Complexidade de tempo

A complexidade de tempo de M

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 entrada de tamanho n,

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 entrada de tamanho n, M 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 entrada de tamanho n, M para depois de executar no máximo $t_{\mathrm{M}}(n)$ transições.

Exemplo: Considere a MT M_{SOMA} para somar que vimos anteriormente.

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 entrada de tamanho n, M para depois de executar no máximo $t_{\mathrm{M}}(n)$ transições.

Exemplo: Considere a MT M_{SOMA} para somar que vimos anteriormente.

- Para qualquer entrada de tamanho n, M_{SOMA} sempre para depois de fazer, no máximo, $2n + 1$ 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 entrada de tamanho n, M para depois de executar no máximo $t_{\mathrm{M}}(n)$ transições.

Exemplo: Considere a MT M_{SOMA} para somar que vimos anteriormente.

- Para qualquer entrada de tamanho n, M_{SOMA} sempre para depois de fazer, no máximo, $2n + 1$ transições.
- Portanto, a complexidade de tempo de M_{SOMA} é a função $t_{M_{\mathrm{SOMA}}}(n) = 2n + 1$.

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 entrada de tamanho n, M para depois de executar no máximo $t_{\mathrm{M}}(n)$ transições.

Exemplo: Considere a MT M_{SOMA} para somar que vimos anteriormente.

- Para qualquer entrada de tamanho n, M_{SOMA} sempre para depois de fazer, no máximo, $2n + 1$ transições.
- Portanto, a complexidade de tempo de M_{SOMA} é a função $t_{M_{\mathrm{SOMA}}}(n) = 2n + 1$.

Simplificando, dizemos "M_{SOMA} tem complexidade de tempo $2n + 1$"

Complexidade de Espaço de Máquinas de Turing

Seja M uma MT,

Complexidade de Espaço de Máquinas de Turing

Seja M uma MT,

Complexidade de espaço

A complexidade de espaço de M

Complexidade de Espaço de Máquinas de Turing

Seja M uma MT,

Complexidade de espaço

A complexidade de espaço de M é uma função $s_M : \mathbb{N} \to \mathbb{N}$

Complexidade de Espaço de Máquinas de Turing

Seja M uma MT,

Complexidade de espaço

A complexidade de espaço de M é uma função $s_M : \mathbb{N} \to \mathbb{N}$ tal que, para qualquer entrada de tamanho n,

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 entrada de tamanho n, 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 $s_{\mathrm{M}} : \mathbb{N} \to \mathbb{N}$ tal que, para qualquer entrada de tamanho n, M para usando no máximo $s_{\mathrm{M}}(n)$ posições da fita 2.

Exemplo:

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 entrada de tamanho n, M para usando no máximo $s_{\mathrm{M}}(n)$ posições da fita 2.

Exemplo: Suponha uma MT M tal que para toda entrada de tamanho n, M sempre para usando no máximo $\log n + 7$ 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 $s_{\mathrm{M}} : \mathbb{N} \to \mathbb{N}$ tal que, para qualquer entrada de tamanho n, M para usando no máximo $s_{\mathrm{M}}(n)$ posições da fita 2.

Exemplo: Suponha uma MT M tal que para toda entrada de tamanho n, M sempre para usando no máximo $\log n + 7$ posições da fita 2.

Neste caso dizemos que a complexidade de espaço de M é $\log n + 7$.

Mais definições

Notação: Se $f(n) = O(n^r)$ para algum $r \in \mathbb{N}$ constante,

Mais definições

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

Mais definições

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

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

Mais definições

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

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

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*.

MTNs polinomiais:

Mais definições

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

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*.

MTNs polinomiais: *todos os ramos da árvore de computações possíveis tem profundidade $poli(n)$.*

Mais definições

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

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*.

MTNs polinomiais: *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 um problema de decisão.

Decidindo problemas

Seja L um problema de decisão.

- Se existe uma MT polinomial que decide L, então dizemos que *L é decidível em tempo polinomial*

Decidindo problemas

Seja L um problema de decisão.

- Se existe uma MT polinomial que decide L, então dizemos que *L é decidível em tempo polinomial*
- Se existe uma MTN polinomial que decide L, então dizemos que *L é decidível em tempo polinomial não determinístico*

Decidindo problemas

Seja L um problema de decisão.

- Se existe uma MT polinomial que decide L, então dizemos que *L é decidível em tempo polinomial*
- Se existe uma MTN polinomial que decide L, então dizemos que *L é decidível em tempo polinomial não determinístico*

Decidindo problemas

Seja L um problema de decisão.

- Se existe uma MT polinomial que decide L, então dizemos que *L é decidível em tempo polinomial*
- Se existe uma MTN polinomial que decide L, então dizemos que *L é decidível em tempo polinomial não determinístico*

Ver no livro definições para problemas decidíveis em:

Decidindo problemas

Seja L um problema de decisão.

- Se existe uma MT polinomial que decide L, então dizemos que *L é decidível em tempo polinomial*
- Se existe uma MTN polinomial que decide L, então dizemos que *L é decidível em tempo polinomial não determinístico*

Ver no livro definições para problemas decidíveis em:

- *espaço polinomial*;

Decidindo problemas

Seja L um problema de decisão.

- Se existe uma MT polinomial que decide L, então dizemos que *L é decidível em tempo polinomial*
- Se existe uma MTN polinomial que decide L, então dizemos que *L é decidível em tempo polinomial não determinístico*

Ver no livro definições para problemas decidíveis em:

- *espaço polinomial*;
- *espaço polinomial não determinístico*;

Decidindo problemas

Seja L um problema de decisão.

- Se existe uma MT polinomial que decide L, então dizemos que *L é decidível em tempo polinomial*
- Se existe uma MTN polinomial que decide L, então dizemos que *L é decidível em tempo polinomial não determinístico*

Ver no livro definições para problemas decidíveis em:

- *espaço polinomial*;
- *espaço polinomial não determinístico*;
- *tempo exponencial*.

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

As classes P, NP e PSPACE e EXP

A classe **P**: problemas decidíveis em tempo polinomial.

As classes P, NP e PSPACE e EXP

A classe **P**: problemas decidíveis em tempo polinomial.

A classe **NP**: problemas decidíveis em tempo polinomial não determinístico.

As classes P, NP e PSPACE e EXP

A classe **P**: problemas decidíveis em tempo polinomial.

A classe **NP**: problemas decidíveis em tempo polinomial não determinístico.

A classe **EXP**: problemas decidíveis em tempo exponencial.

As classes P, NP e PSPACE e EXP

A classe **P**: problemas decidíveis em tempo polinomial.

A classe **NP**: problemas decidíveis em tempo polinomial não determinístico.

A classe **EXP**: problemas decidíveis em tempo exponencial.

A classe **PSPACE**: problemas decidíveis em espaço polinomial.

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

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

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

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

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

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.

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 $N = (Q, \Sigma, \Gamma, (\delta, \delta), q_0, B, F)$.

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.

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

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

- Problema em aberto: $\mathrm{P} \neq \mathrm{NP}$?

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

- Problema em aberto: $\mathrm{P} \neq \mathrm{NP}$? (i.e., é verdade que $\mathrm{P} \subsetneq \mathrm{NP}$?)
- E PSPACE e NPSPACE?

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

- Problema em aberto: $\mathrm{P} \neq \mathrm{NP}$? (i.e., é verdade que $\mathrm{P} \subsetneq \mathrm{NP}$?)
- E PSPACE e NPSPACE? (vejam em Tópicos em Complexidade: PSPACE = NPSPACE)

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

- Problema em aberto: $\mathrm{P} \neq \mathrm{NP}$? (i.e., é verdade que $\mathrm{P} \subsetneq \mathrm{NP}$?)
- E PSPACE e NPSPACE? (vejam em Tópicos em Complexidade: PSPACE = NPSPACE)

Teorema: $\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP}$

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

- Problema em aberto: $\mathrm{P} \neq \mathrm{NP}$? (i.e., é verdade que $\mathrm{P} \subsetneq \mathrm{NP}$?)
- E PSPACE e NPSPACE? (vejam em Tópicos em Complexidade: PSPACE = NPSPACE)

Teorema: $\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP}$

Prova: (opcional)

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

- Problema em aberto: $\mathrm{P} \neq \mathrm{NP}$? (i.e., é verdade que $\mathrm{P} \subsetneq \mathrm{NP}$?)
- E PSPACE e NPSPACE? (vejam em Tópicos em Complexidade: PSPACE = NPSPACE)

Teorema: $\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP}$

Prova: (opcional)

- É verdade que $\mathrm{NP} \subsetneq \mathrm{PSPACE}$?

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

- Problema em aberto: $\mathrm{P} \neq \mathrm{NP}$? (i.e., é verdade que $\mathrm{P} \subsetneq \mathrm{NP}$?)
- E PSPACE e NPSPACE? (vejam em Tópicos em Complexidade: PSPACE = NPSPACE)

Teorema: $\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP}$

Prova: (opcional)

- É verdade que $\mathrm{NP} \subsetneq \mathrm{PSPACE}$? (problema em aberto)

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

- Problema em aberto: $\mathrm{P} \neq \mathrm{NP}$? (i.e., é verdade que $\mathrm{P} \subsetneq \mathrm{NP}$?)
- E PSPACE e NPSPACE? (vejam em Tópicos em Complexidade: PSPACE = NPSPACE)

Teorema: $\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP}$

Prova: (opcional)

- É verdade que $\mathrm{NP} \subsetneq \mathrm{PSPACE}$? (problema em aberto)
- É verdade que $\mathrm{PSPACE} \subsetneq \mathrm{EXP}$?

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

- Problema em aberto: $\mathrm{P} \neq \mathrm{NP}$? (i.e., é verdade que $\mathrm{P} \subsetneq \mathrm{NP}$?)
- E PSPACE e NPSPACE? (vejam em Tópicos em Complexidade: PSPACE = NPSPACE)

Teorema: $\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP}$

Prova: (opcional)

- É verdade que $\mathrm{NP} \subsetneq \mathrm{PSPACE}$? (problema em aberto)
- É verdade que $\mathrm{PSPACE} \subsetneq \mathrm{EXP}$? (problema em aberto)

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

- Problema em aberto: $\mathrm{P} \neq \mathrm{NP}$? (i.e., é verdade que $\mathrm{P} \subsetneq \mathrm{NP}$?)
- E PSPACE e NPSPACE? (vejam em Tópicos em Complexidade: PSPACE = NPSPACE)

Teorema: $\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP}$

Prova: (opcional)

- É verdade que $\mathrm{NP} \subsetneq \mathrm{PSPACE}$? (problema em aberto)
- É verdade que $\mathrm{PSPACE} \subsetneq \mathrm{EXP}$? (problema em aberto)

(vejam em Tópicos em Complexidade: $\mathrm{P} \subsetneq \mathrm{EXP}$)