Algoritmos e complexidade: uma introdução

Marcus Ritt

Instituto de Informática
Universidade Federal do Rio Grande do Sul
Porto Alegre, Brasil

Agenda

1. Parte 1: Modelos computacionais e analise assintótica
2. Parte 2: Analise na prática
3. Parte 3: Complexidade de problemas e classes de complexidade

Parte I

Modelos e analise

Agenda

1. Complexidade
2. Notação assintôtica
3. Modelos de computação
4. Um fundamento para a análise prática

Multiplicação

$$3 \times 5 = ?$$

Multiplicação

$$3 \times 5 = 15$$

Multiplicação...

$$
\begin{aligned}
&334780716989568987860441698482126908177 \\
&047949837137685689124313889828837938780 \\
&022876147116525317430877378144679994 89 \\
&\times \\
&367460436667995904282446337996279526322 \\
&791581643430876426760322838157396665112 \\
&792333734171433968102700927987363089 17 = ?
\end{aligned}
$$

Multiplicação...

$$
\begin{aligned}
&334780716989568987860441698482126908177\\
&047949837137685689124313889828837938780\\
&022876147116525317430877378144679994 89\\
&\times\\
&367460436667995904282446337996279526322\\
&791581643430876426760322838157396665112\\
&792333734171433968102700927987363089 17\\
&=\\
&123018668453011775513049495838496272077\\
&285356959533479219732245215172640050726\\
&365751874520219978646938995647494277406\\
&384592519255732630345373154826850791702\\
&612214291346167042921431160222124047927\\
&473779408066535141959745985690214341 3
\end{aligned}
$$

Versus: Fatoração

$$21 = ? \times ?$$

Versus: Fatoração

$$21 = 3 \times 7$$

Versus: Fatoração...

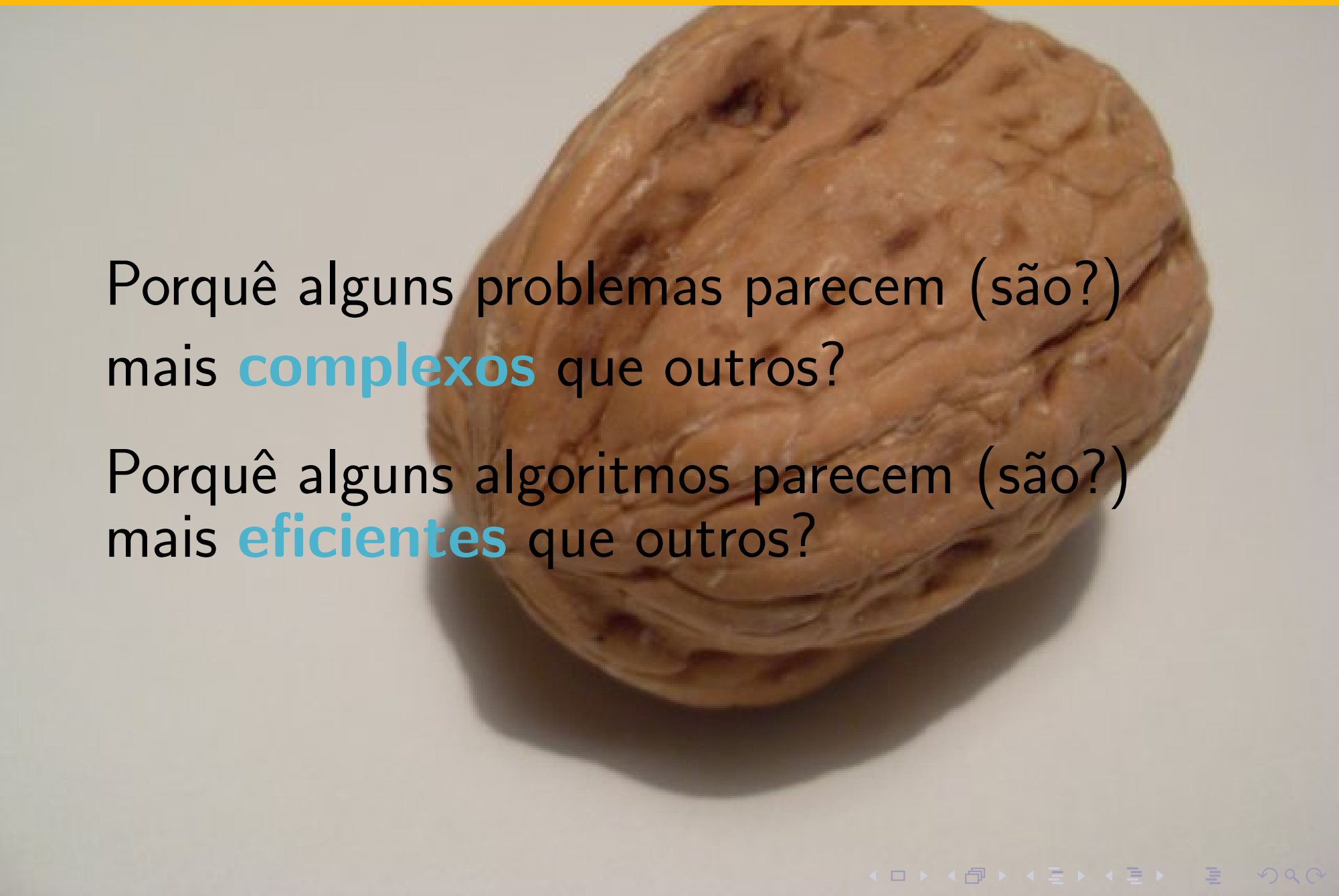
$$
\begin{aligned}
&1230186684530117755130494958384962720772853569595334792197322452151726400507263657518745202199786469389956474942774063845925192557326303453731548268507917026122142913461670429214311602221240479272=?\times ?
\end{aligned}
$$

Versus: Fatoração...

$$
\begin{aligned}
&1230186684530117755130494958384962720772853569595334792197322452151726400507263657518745202199786469389956474942774063845925192557326303453731548268507917026122142913461670429214311602212404792747377940806653514195974595685690214341 3\\
&= \\
&3347807169895689878604416948212690817704794983713768568912431388982883793878002287614711652531743087737814467999489\\
&\times \\
&3674604366679959042824463379962795263227915816434308764267603228381573966651127923337341714339681027009279873630891 7
\end{aligned}
$$

Custo: 2000 2.2GHz-Opteron–anos CPU

Complexidade?



Porquê alguns problemas parecem (são?) mais **complexos** que outros?

Porquê alguns algoritmos parecem (são?) mais **eficientes** que outros?

Nossos exemplos

Multiplicação de números inteiros de $O(n)$ bits:

- Método "escola": $O(n^2)$
- Melhor método: $n \log n \, 2^{O(\log^* n)}$ (Fürer, 2007)

Fatoração de um número inteiro com $O(n)$ bits:

- Crivo de Eratóstenes: $O(2^n n \log n)$.
- General prime number sieve: $O(e^{(c+o(1))n^{1/3} \log^{2/3} n})$

(

Agenda

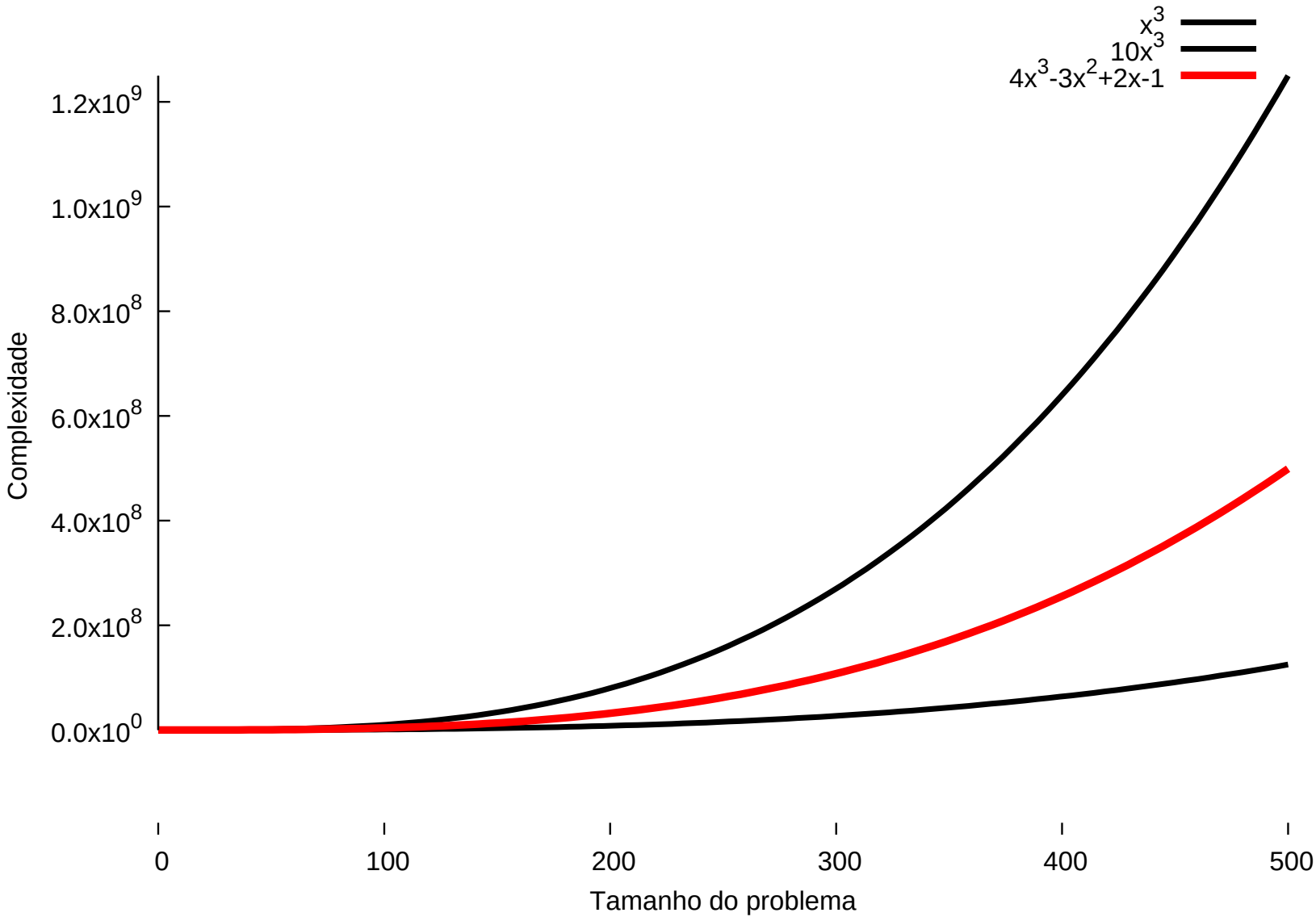
1. Complexidade
2. Notação assintôtica
3. Modelos de computação
4. Um fundamento para a análise prática

Notação assintôtica

- Frequentemente desejável: só considerar a **ordem de crescimento** de uma função
- Exemplo: $4n^3 - 3n^3 + 2n - 1$ cresce "como" n^3

Notação:

- Limite superior: $4n^3 - 3n^3 + 2n - 1 \in O(n^3)$: cresce "não mais" que n^3
- Limite inferior: $4n^3 - 3n^3 + 2n - 1 \in \Omega(n^3)$: cresce "mais" que n^3
- $4n^3 - 3n^3 + 2n - 1 \in \Theta(n^3)$: cresce "igual" a n^3


x^3
$10x^3$
$4x^3-3x^2+2x-1$
1.2x10^9
1.0x10^9
8.0x10^8
6.0x10^8
4.0x10^8
2.0x10^8
0.0x10^0
Complexidade
0
100
200
300
400
500
Tamanho do problema

Mais formal...

Classes de crescimento

Para funções $f, g : \mathbb{N} \to \mathbb{R}^+$:

$$O(g(n)) = \{f \mid \exists c > 0 \exists n_0 \forall n > n_0 : \ f(n) \leq cg(n)\}$$
$$\Omega(g(n)) = \{f \mid \exists c > 0 \exists n_0 \forall n > n_0 : \ f(n) \geq cg(n)\}$$
$$\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$$
$$o(g(n)) = \{f \mid \forall c > 0 \exists n_0 \forall n > n_0 : \ f(n) \leq cg(n)\}$$
$$\omega(g(n)) = \{f \mid \forall c > 0 \exists n_0 \forall n > n_0 : \ f(n) \geq cg(n)\}$$

Exemplos

- $\operatorname{sen}(n) \in O(1)$
- $n^2 \in O(n^3)$
- $n^{O(1)}$ (tempo polinomial)
- $n! \in \sqrt{2\pi n}\left(\frac{n}{e}\right)^n(1 + O(1/n))$ (Stirling, 1730)
- $\pi(n) \in \Theta(n/\ln n)$ (Hadamard, 1896; de la Vallée Poussin, 1896)

Notações adicionais

- Logaritmo iterado

$$\log^* n = \begin{cases} 0 & \text{se } n \leq 1 \\ 1 + \log^*(\log n) & \text{caso contrário} \end{cases}$$

- Desconsiderar fatores logarítmicos (soft-O, O-suave)

$$f \in \tilde{O}(g) \implies \exists k : f \in O(g \log^k g)$$

Algumas características

$$
\begin{aligned}
f &= O(f) \\
cO(f) &= O(f) \\
O(f) + O(f) &= O(f) \\
O(O(f)) &= O(f) \\
O(f)O(g) &= O(fg) \\
O(fg) &= fO(g) \\
g = O(f) &\implies f + g = \Theta(f)
\end{aligned}
$$

)

Eficiente: em que?

- Tempo de execução
- Memôria consumida
- Número de operações E/S
- Memôria acessada e número de falhas de cache
- Energia consumida
- Energia dissipada

Problema básico: qual algoritmo é mais eficiente?

O que é eficiente?

Uma tentativa:

Definition (Kleinberg and Tardos (2005))

An algorithm is efficient if, when implemented, it runs quickly on real input instances.

Abordagem experimental

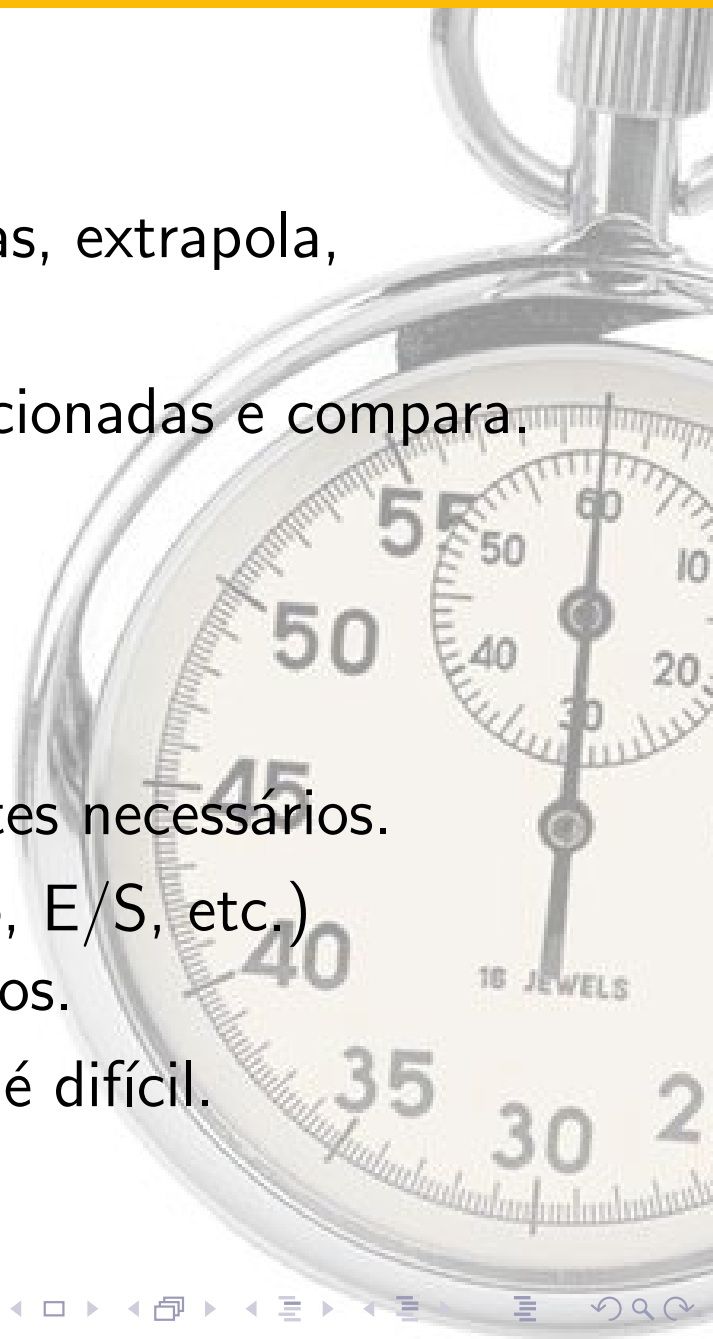
- Mede um grande número de instâncias, extrapola, compara.
- Mede um conjunto de instâncias selecionadas e compara.

Vantagens:

- Relativamente simples

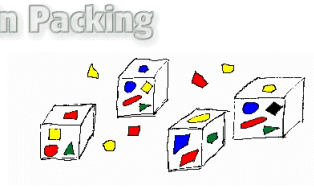
Desvantagens:

- Pode ser inviável pelo número de testes necessários.
- Muitas dependências (máquina, ruído, E/S, etc.) dificultam a comparação dos resultados.
- Escolha de instâncias representativas é difícil.



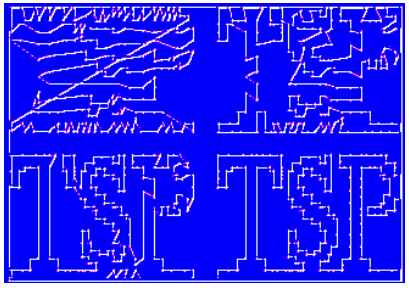
Alguns exemplos de coleções de instâncias

QAPLIB



SteinLib

VRP web



VRPLIB

SATLIB

MIPLIB

Exemplo de um fracasso

It is all to easy to make predictions which are quite at variance with observed performance.
It is also easy to assume that relative performance on one computer will apply to another computer.
Perhaps even worse, it is possible to optimize away the worst case [...] at the cost of penalizing the usual case.

Fenwick, Some perils of performance prediction: a case study in pattern matching.

Agenda

1. Complexidade
2. Notação assintôtica
3. Modelos de computação
   - A máquina de Turing
   - A máquina RAM
4. Um fundamento para a análise prática

Abordagem teórica

Define um modelo teórico, e compara neste modelo.

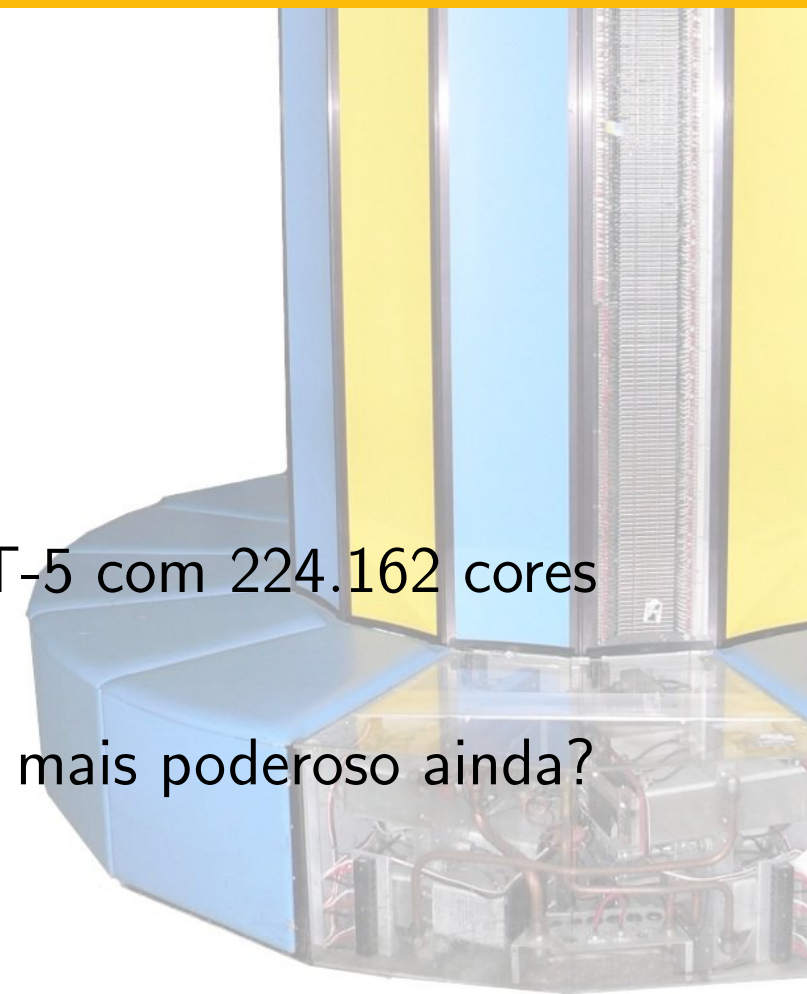
Vantagens:

- Resultados comparáveis.

Desvantagens:

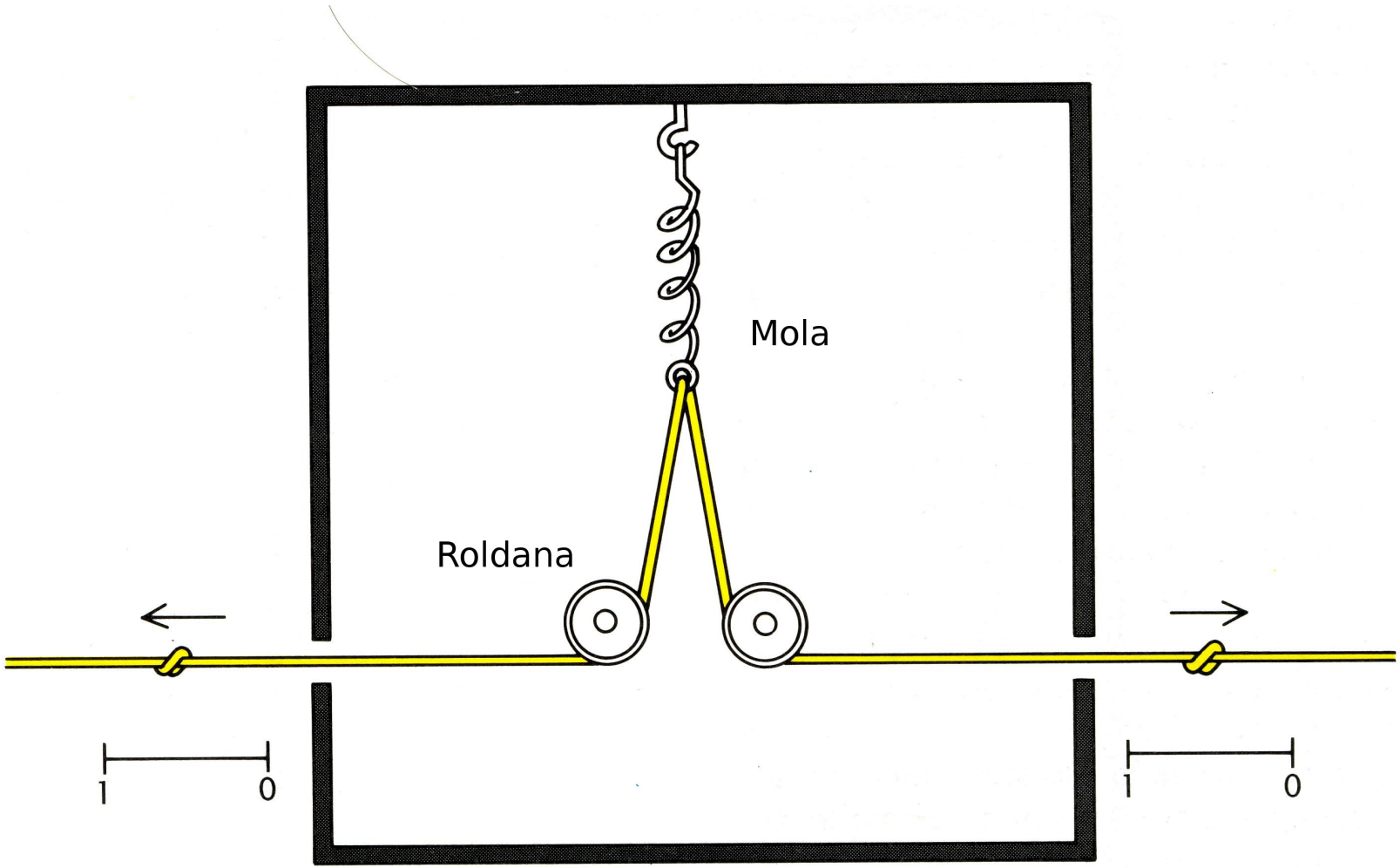
- Modelos podem ser elementares demais para algoritmos práticos (máquina Turing).
- Não é obviou qual o modelo certo (quais instruções?)

Qual o modelo de computação adequado ?

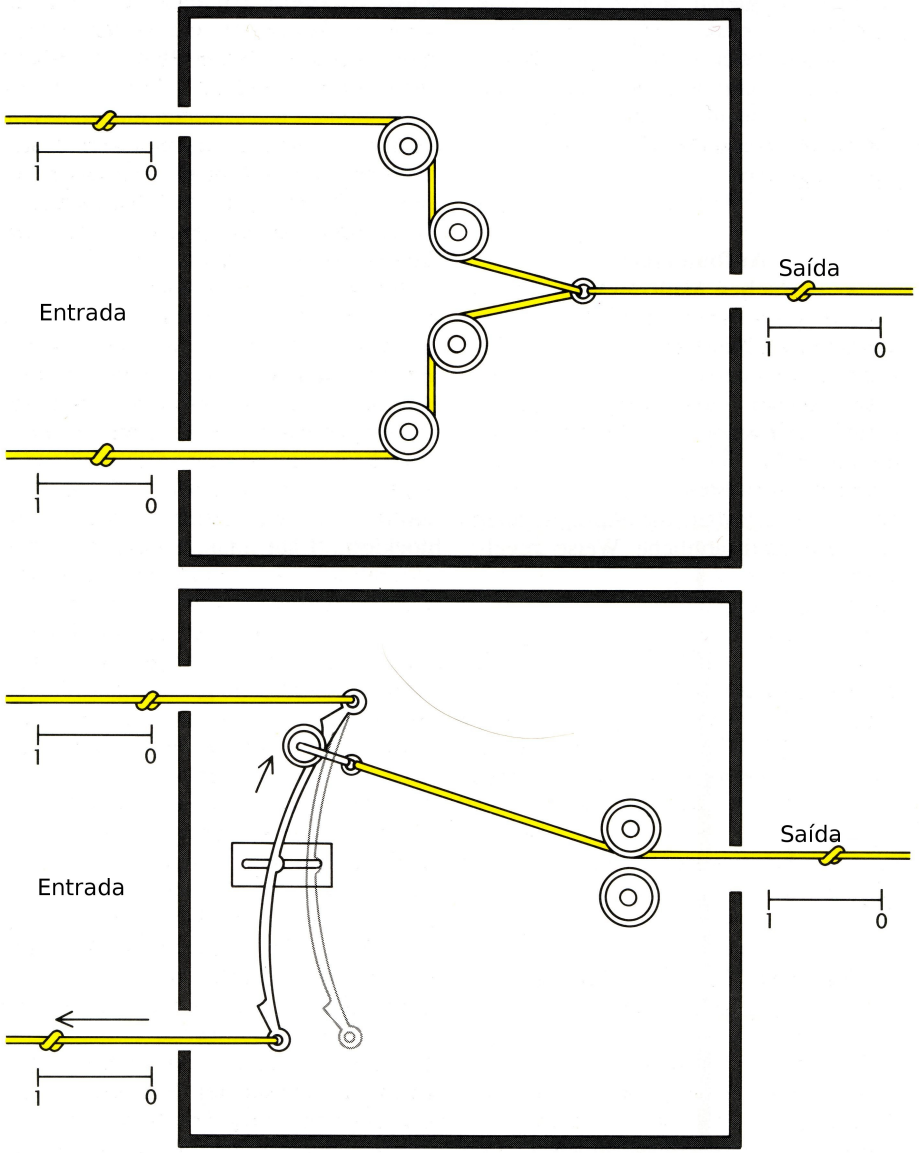


- O computador de von Neumann?
- Um super-computador paralelo?
  Top 500, 9/2009: Jaguar Cray XT-5 com 224.162 cores
- Um computador quântico?
- Um outro modelo de computação mais poderoso ainda?
  O universo?

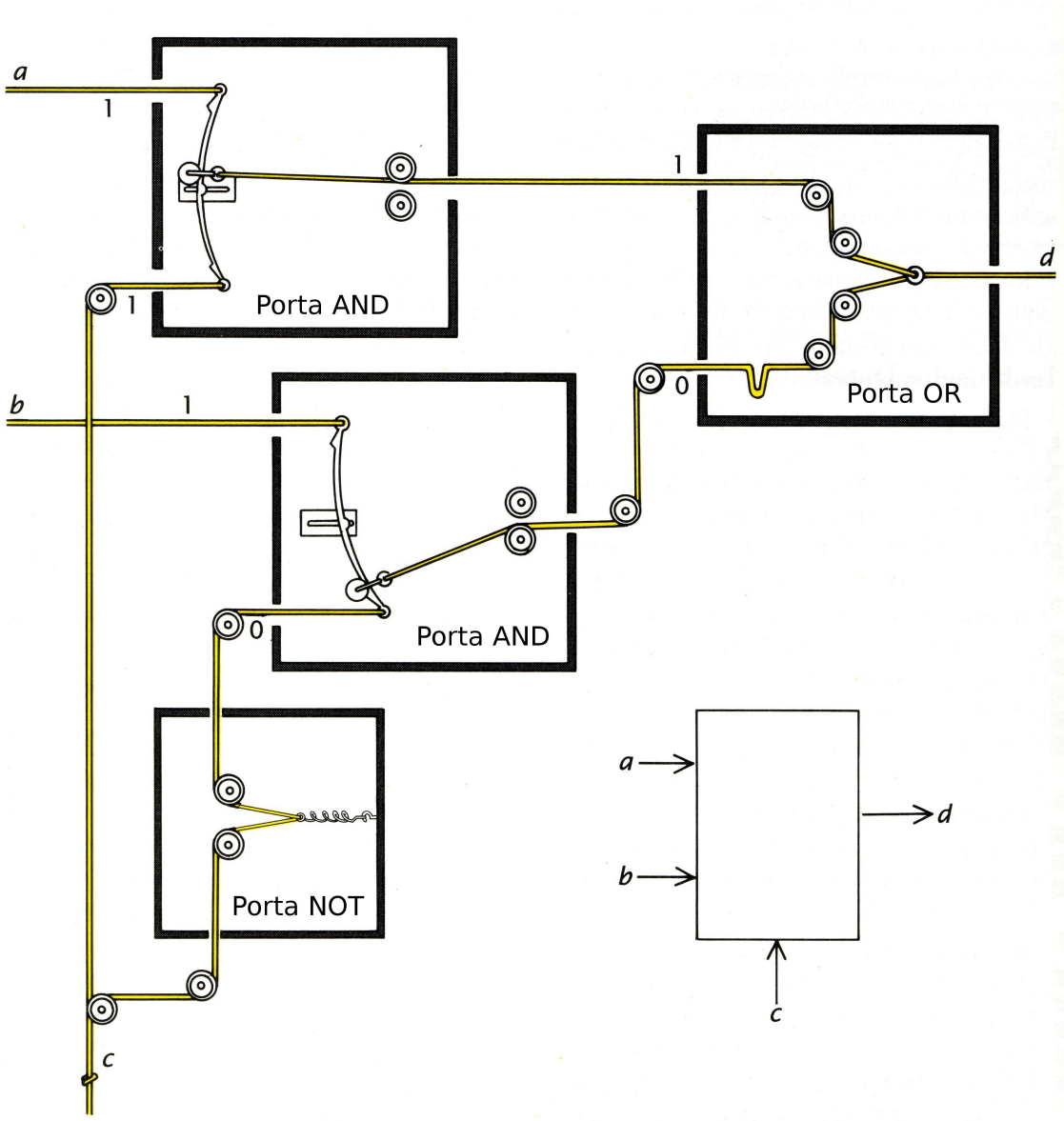
Qual o modelo certo?



Qual o modelo certo?

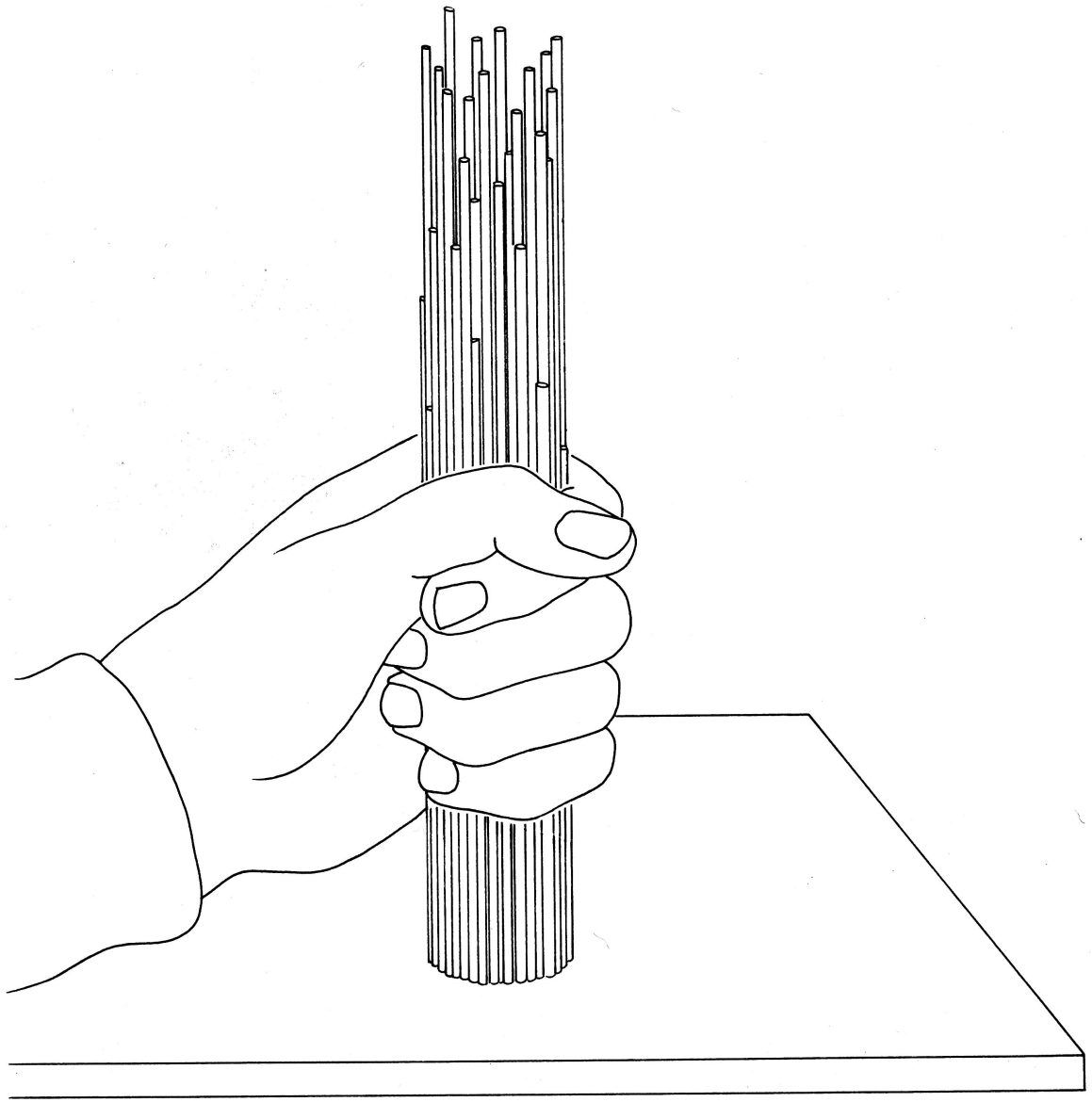


Qual o modelo certo?



Qual o modelo certo?

Ordenar n números em tempo linear?



O universo é o computador? Fatos

Idade	$13.7 \pm 0.2 \times 10^9$ anos $\approx 43.5 \times 10^{16}$ s
Tamanho	$\geq 78 \times 10^9$ anos-luz
Densidade	$9.9 \times 10^{-30} g/cm^3$
Número de átomos	10^{80}
Número de bits	10^{120}
Operações lógicas elementares até hoje	10^{120}
Operações/s	$\approx 2 \times 10^{102}$

(Pelo consenso científico atual. Fontes principais: Lloyd (2002); WMA)

Resgate 1

Tese de Church-Turing

As funções efetivamente computáveis sobre os números inteiros positivos são precisamente as funções computáveis pela máquina de Turing.

- Verdadeiro para todos modelos conhecidos

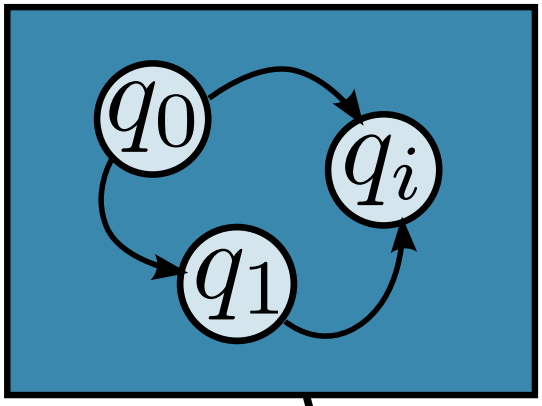
  Maquina de Turing, cálculo lambda, máquina RAM, máquina pontador, circuitos lógicos, autômatos celulares (Conway), avaliação de templates em C++, computador billiard, computador quântico, ...

A máquina de Turing

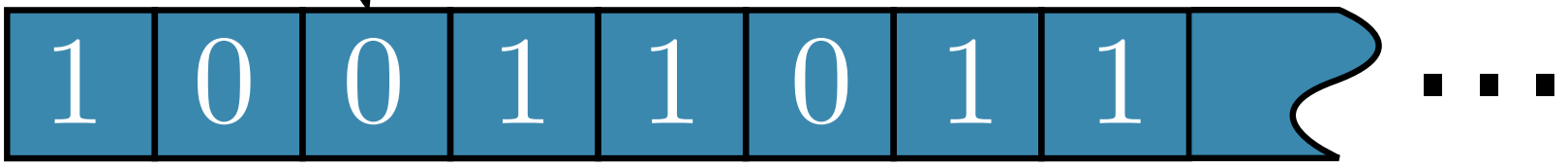
Computing is normally done by writing certain symbols on paper. "We may suppose this paper is divided into squares like a child's arithmetic book. In elementary arithmetic the two-dimensional character of the paper is sometimes used. But such a use is always avoidable, and I think that it will be agreed that the two-dimensional character of paper is no essential of computation. I assume then that the computation is carried out on one-dimensional paper, i.e. on a tape divided into squares. I shall also suppose that the number of symbols which may be printed is finite. If we were to allow an infinity of symbols, then there would be symbols differing to an arbitrarily small extent. The effect of this restriction of the number of symbols is not very serious. It is always possible to use sequences of symbols in the place of single symbols. Thus an Arabic numeral such as 17 or 999999999999999 is normally treated as a single symbol. Similarly in any European language words are treated as single symbols (Chinese, however, attempts to have an enumerable infinity of symbols). The differences from our point of view between the single and compound symbols is that the compound symbols, if they are too lengthy, cannot be observed at one glance. This is in accordance with experience. We cannot tell at a glance whether 9999999999999999 and 999999999999999 are the same. (Turing, 1936).

Alan Mathison Turing (*1912, +1954)

Um exemplo – a máquina de Turing



Cabeça de leitura
e escritura



Fita infinita

A máquina RAM

A **máquina RAM** (random access machine) é o modelo padrão para análise de algoritmos. Ela possui

- um processador com um ou mais registros, e com apontador de instruções,
- uma memória infinita de números inteiros e
- instruções elementares (controle,transferência inclusive endereçamento indireto,aritmética).

A máquina RAM

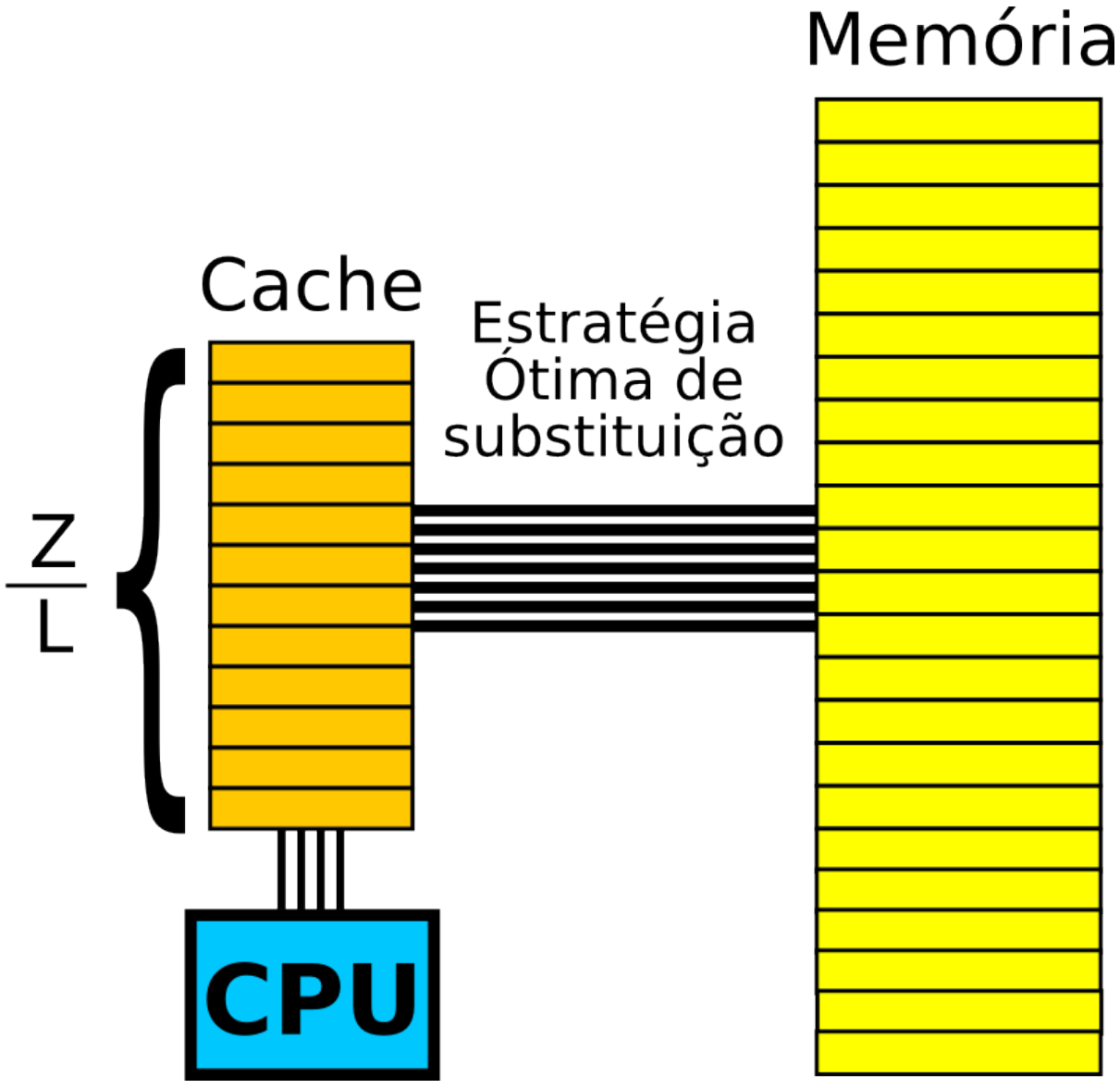
Existem RAMs com diferentes tipos de instruções aritméticas

- **SRAM**: somente sucessor
- **RAM**: adição e subtração
- **MRAM**: multiplicação e divisão

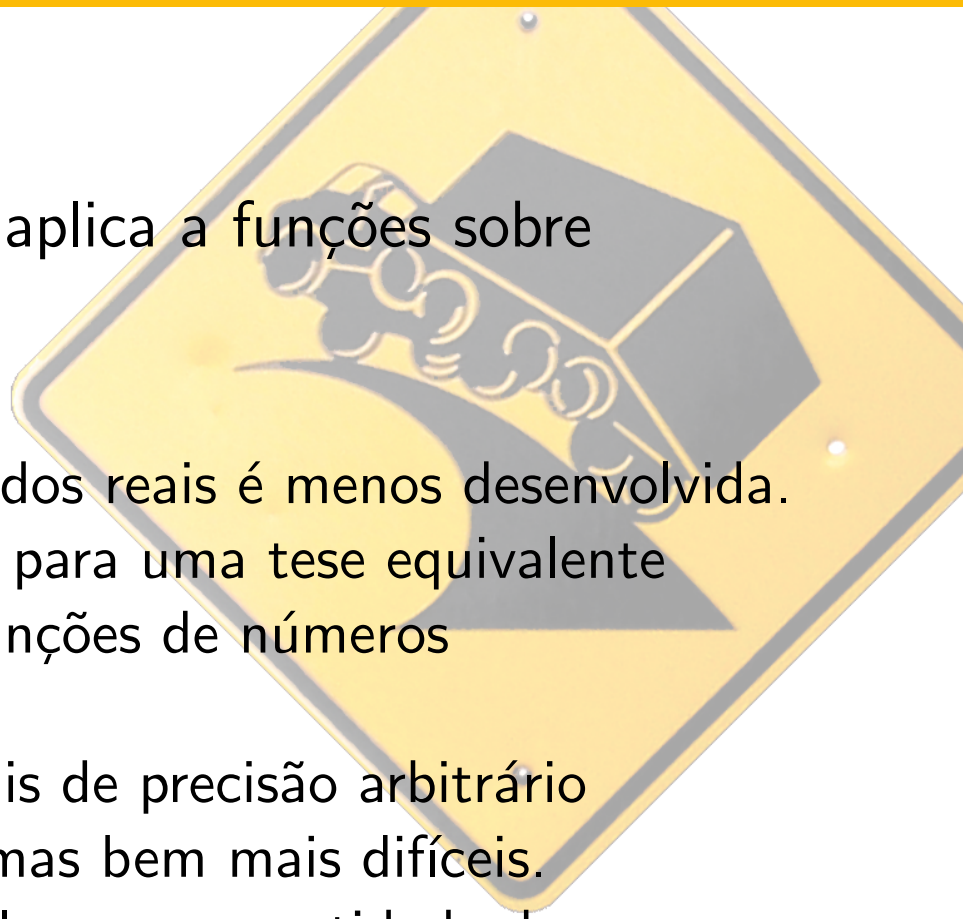
e com diferentes tipos de custos

- Custo **uniforme**: cada operação em $O(1)$
- Custo **logarítmico**: proporcional ao número de bits dos operandos

Máquina RAM com cache

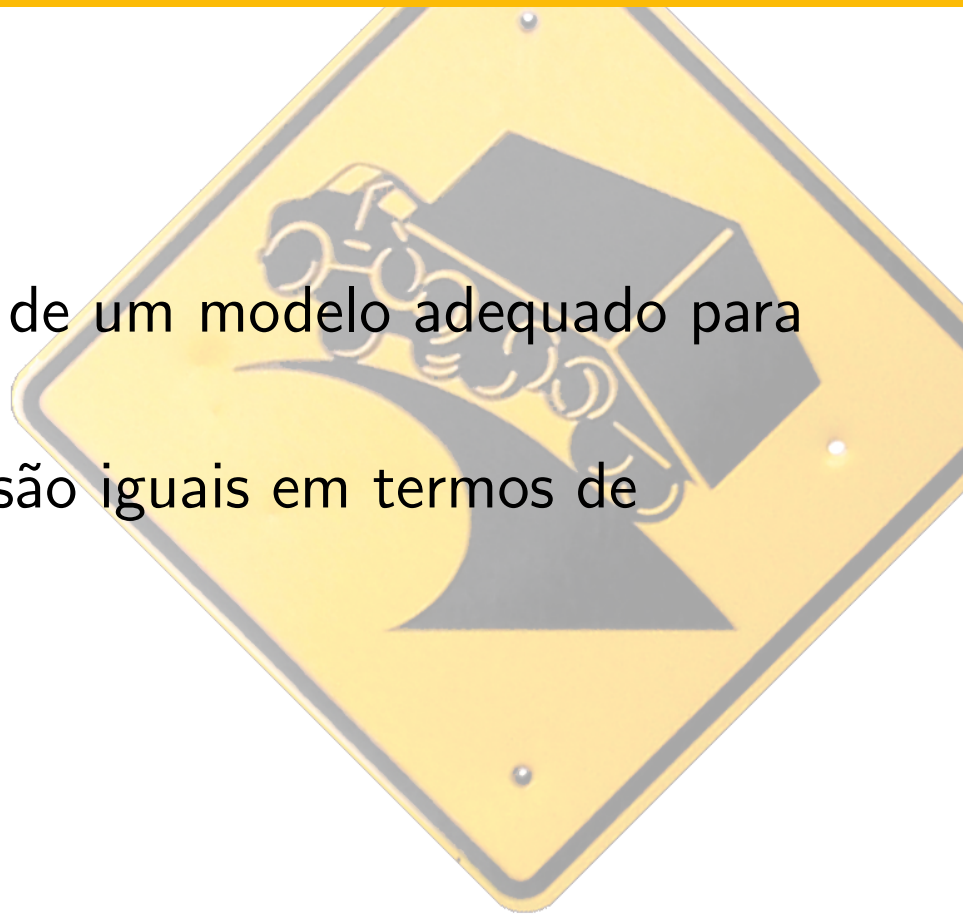


Perigo?



- A tese de Church-Turing se aplica a funções sobre números inteiros.
- E números reais?
  - A teoria da computação dos reais é menos desenvolvida.
  - Exemplo: falta evidência para uma tese equivalente sobre as funções sobre funções de números inteiros (Mitchell, 1996).
  - Com operações sobre reais de precisão arbitrário podemos resolver problemas bem mais difíceis.
  - Práticamente não é problema: a quantidade de informação por volume (ou energia) é limitada (Bekenstein, 1981).

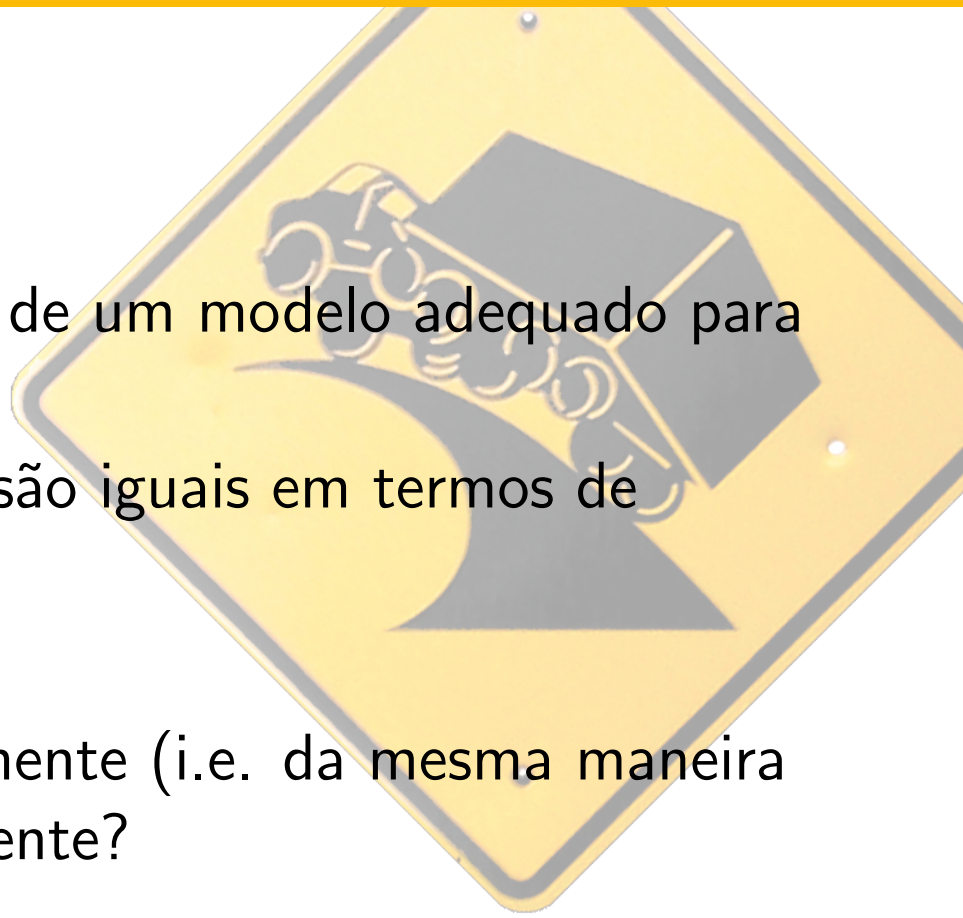
Perigo?



Isso ainda não garante a escolha de um modelo adequado para analise de eficiência!

- Será que todos os modelos são iguais em termos de eficiência?
- Não!

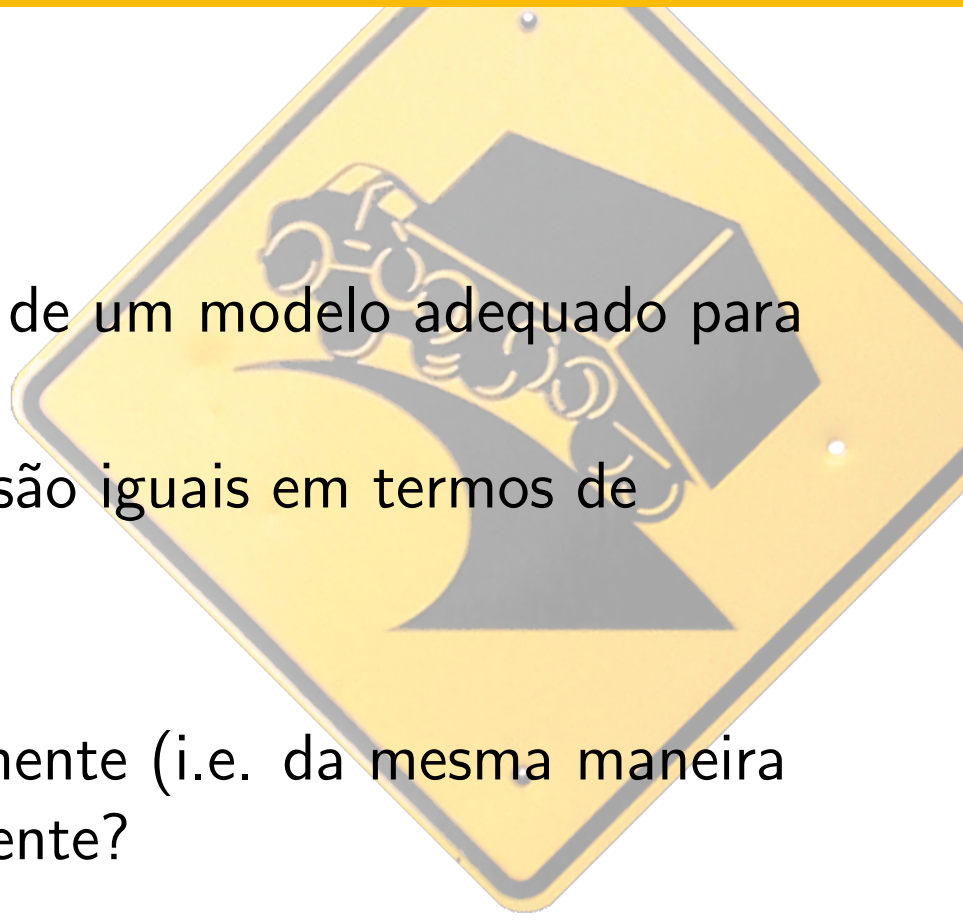
Perigo?



Isso ainda não garante a escolha de um modelo adequado para analise de eficiência!

- Será que todos os modelos são iguais em termos de eficiência?
- Não!
- Será que eles são uniformemente (i.e. da mesma maneira para todos problemas) diferente?

Perigo?



Isso ainda não garante a escolha de um modelo adequado para analise de eficiência!

- Será que todos os modelos são iguais em termos de eficiência?
- Não!
- Será que eles são uniformemente (i.e. da mesma maneira para todos problemas) diferente?
- Não!

Exemplos de simulação

Theorem (van Leeuwen (1990))

$$\begin{aligned}
\mathrm{m-tapes} &\leq 1-\mathrm{tape}(\textit{time } kn^2 \ \&\ \textit{space } Lin)\\
\mathrm{m-tapes} &\leq 2-\mathrm{tape}(\textit{time } kn\log n \ \&\ \textit{space } Lin)\\
\mathrm{SRAM-UTIME} &\leq T(\textit{time } n^2\log n)\\
\mathrm{RAM-UTIME} &\leq T(\textit{time } n^3)\\
\mathrm{MRAM-UTIME} &\leq T(\textit{time } Exp)\\
\mathrm{SRAM-LTIME} &\leq T(\textit{time } n^2)\\
\mathrm{RAM-LTIME} &\leq T(\textit{time } n^2)\\
\mathrm{MRAM-LTIME} &\leq T(\textit{time } Poly)
\end{aligned}$$

Resgate 2

Tese estendida de Church-Turing

Qualquer modelo de computação universal é equivalente à máquina de Turing com

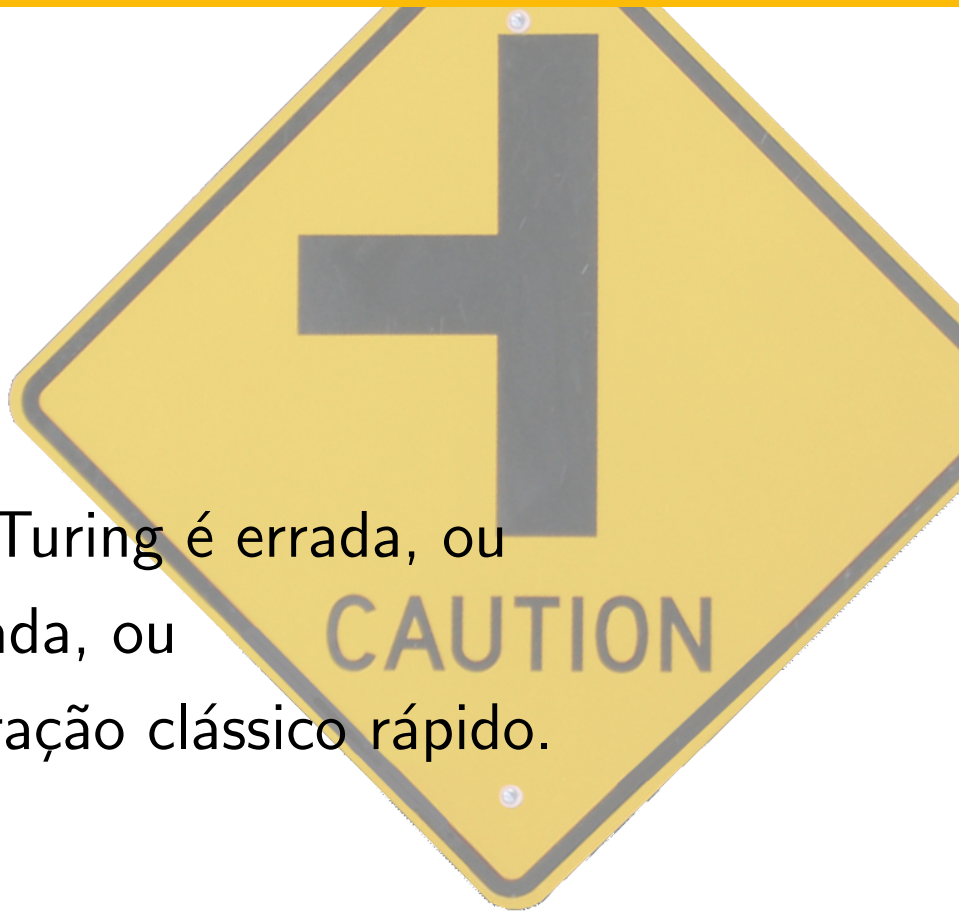
- custo adicional de tempo no máximo polinomial
- custo adicional de espaço no máximo constante

- Equivalência definido por simulação mutual.
- Verdadeiro para **quase** todos modelos conhecidos:

  Maquina de Turing, cálculo lambda, máquina RAM, máquina pontador, circuitos lógicos, autômatos celulares (Conway), avaliação de templates em C++, computador billiard, ...

- Computador quântico?

Consequência: Shor's trilemma



Ou

- a tese estendida de Church-Turing é errada, ou
- a física quântica atual é errada, ou
- existe um algoritmo de fatoração clássico rápido.