

Introdução à Teoria da Computação

Complexidade de Kolmogorov

Professor Murilo V. G. da Silva

Departamento de Informática
Universidade Federal do Paraná

2026 / 2

Descrevendo uma string

Considere as três strings w_1 , w_2 e w_3 de 480 bits abaixo:

Descrevendo uma string

Considere as três strings w_1 , w_2 e w_3 de 480 bits abaixo:

[illegible]

Descrevendo uma string

Considere as três strings w_1 , w_2 e w_3 de 480 bits abaixo:

$w_1 =$

[illegible]

Descrevendo uma string

Considere as três strings w_1 , w_2 e w_3 de 480 bits abaixo:

[illegible][illegible]

$w_3 =$ 011010000101001101010000010101010010111011101000010100001101110110011011011011
11010010010011010111010000111010101010011101001011010100011101010101101011111011
001110101010111101011110011101110111010010010011000100011010111110110100101110
10001010100110101010010101010101000101100010011101111101101010011110101110011
10001010100110101010010101010101000101000100111011111011010100111101011110011
110011101000111010100110100011110101011110101010100111010010110101000111010

Descrevendo uma string

[illegible][illegible]

$w_3 =$

```
011010000101001101010000010101010010111011101000010100001101110110011011011011  
1101001001001101011010000111010101010011101001011010100011101010101101011111011  
001110101010111101011110011101110111010010010011000100011010111110110100101110  
100010101001101010100101010101101000101100010011101111101101010011110101110011  
1000101010011010101001010101010100010110001001110111110110100111101011110011  
1100111010001110101001101000111101010101111010101010011101001010101000111010
```

Descrevendo uma string

[illegible][illegible][illegible]

- Descrição de w_1 : “480 bits 0”

Descrevendo uma string

[illegible][illegible]

$w_3 =$ 011010000101001110101000001010101010010111011101000010100001101110110011011011011
11010010010011010110100001110101010100111010010110101000111010101101011111011
00111010101011110101111001110111011101010010010011000100011010111110110100101110
1000101010011010101001010101101000101000100111011111011010011110101110011
10001010100110101010010101011010001010001001110111110110100111101011110011
110011101000111010101101000111101010111101010111011100111010010110101000111010

- Descrição de w_1 : "480 bits 0"
- Descrição de w_2 : "80 bits 0 e 80 bits 1; repita isso 3 vezes"

Descrevendo uma string

[illegible][illegible]

$w_3 =$ 011010000101001110101000001010101010010111011101000010100001101110110011011011011
11010010010011010110100001110101010100111010010110101000111010101101011111011
00111010101011110101111001110111011101010010010011000100011010111110110100101110
1000101010011010101001010101101000101000100111011111011010011110101110011
10001010100110101010010101011010001010001001110111110110100111101011110011
110011101000111010101101000111101010111101010111011100111010010110101000111010

- Descrição de w_1 : “480 bits 0”
- Descrição de w_2 : “80 bits 0 e 80 bits 1; repita isso 3 vezes”
- Descrição de w_3 : “a sequência de bits 0110100001010011010100000101010101001011
101110100001010000110111011001101101101111010010010011010110100001110101010100
11101001011010100011101010101101011111011001110101010111101011110011101110111
01001001001100010001101011111011010010111101000101001101010100101010101010
001011000100111011111010101001111010111001111001110100011101010011010001111
010101011110101010100111010010101010001110101”

A descrição mínima de uma string

Seja x uma string.

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\lceil M \rceil w$

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\lceil M \rceil w$, sendo M uma MT e w uma string,

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\lceil M \rceil w$, sendo M uma MT e w uma string, tal que $M(w) = x$.

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\lceil M \rceil w$, sendo M uma MT e w uma string, tal que $M(w) = x$.

Importante: Seja $\mathcal{U}$ é uma MT universal.

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\langle M \rangle w$, sendo M uma MT e w uma string, tal que $M(w) = x$.

Importante: Seja $\mathcal{U}$ é uma MT universal. Se y é uma descrição de x , então $\mathcal{U}(y) = x$

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\lceil M \rceil w$, sendo M uma MT e w uma string, tal que $M(w) = x$.

Importante: Seja $\mathcal{U}$ é uma MT universal. Se y é uma *descrição* de x , então $\mathcal{U}(y) = x$

A *Descrição Mínima* de uma string x ,

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\lceil M \rceil w$, sendo M uma MT e w uma string, tal que $M(w) = x$.

Importante: Seja $\mathcal{U}$ é uma MT universal. Se y é uma *descrição* de x , então $\mathcal{U}(y) = x$

A *Descrição Mínima* de uma string x , denotada $d_m(x)$,

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\sqcup M \sqcup w$, sendo M uma MT e w uma string, tal que $M(w) = x$.

Importante: Seja $\mathcal{U}$ é uma MT universal. Se y é uma *descrição* de x , então $\mathcal{U}(y) = x$

A *Descrição Mínima* de uma string x , denotada $d_m(x)$, é a *menor string* $\sqcup M \sqcup w$

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\lceil M \rceil w$, sendo M uma MT e w uma string, tal que $M(w) = x$.

Importante: Seja $\mathcal{U}$ é uma MT universal. Se y é uma *descrição* de x , então $\mathcal{U}(y) = x$

A *Descrição Mínima* de uma string x , denotada $d_m(x)$, é a *menor string* $\lceil M \rceil w$ em ordem lexicográfica

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\lceil M \rceil w$, sendo M uma MT e w uma string, tal que $M(w) = x$.

Importante: Seja $\mathcal{U}$ é uma MT universal. Se y é uma *descrição* de x , então $\mathcal{U}(y) = x$

A *Descrição Mínima* de uma string x , denotada $d_m(x)$, é a *menor string* $\lceil M \rceil w$ em ordem lexicográfica tal que $\lceil M \rceil w$ é uma descrição de x .

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\sqcup M \sqcup w$, sendo M uma MT e w uma string, tal que $M(w) = x$.

Importante: Seja $\mathcal{U}$ é uma MT universal. Se y é uma *descrição* de x , então $\mathcal{U}(y) = x$

A *Descrição Mínima* de uma string x , denotada $d_m(x)$, é a *menor string* $\sqcup M \sqcup w$ em ordem lexicográfica tal que $\sqcup M \sqcup w$ é uma descrição de x .

A *Complexidade de Komogorov* de uma string x ,

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\sqcup M \sqcup w$, sendo M uma MT e w uma string, tal que $M(w) = x$.

Importante: Seja $\mathcal{U}$ é uma MT universal. Se y é uma *descrição* de x , então $\mathcal{U}(y) = x$

A *Descrição Mínima* de uma string x , denotada $d_m(x)$, é a *menor string* $\sqcup M \sqcup w$ em ordem lexicográfica tal que $\sqcup M \sqcup w$ é uma descrição de x .

A *Complexidade de Komogorov* de uma string x , denotada $K(x)$,

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\sqcup M \sqcup w$, sendo M uma MT e w uma string, tal que $M(w) = x$.

Importante: Seja $\mathcal{U}$ é uma MT universal. Se y é uma *descrição* de x , então $\mathcal{U}(y) = x$

A *Descrição Mínima* de uma string x , denotada $d_m(x)$, é a *menor string* $\sqcup M \sqcup w$ em ordem lexicográfica tal que $\sqcup M \sqcup w$ é uma descrição de x .

A *Complexidade de Komogorov* de uma string x , denotada $K(x)$, é o tamanho da descrição mínima de x ,

A descrição mínima de uma string

Seja x uma string. Vamos definir a descrição de x como

- Uma máquina/mecanismo/processo sistemático que produza a string x
- Ideia inicial: a descrição de x é uma MT M tal que $M(\epsilon) = x$
- Ideia mais geral: a descrição de x é um par (M, w) tal que $M(w) = x$

Uma *descrição* de uma string x é uma string $\sqcup M \sqcup w$, sendo M uma MT e w uma string, tal que $M(w) = x$.

Importante: Seja $\mathcal{U}$ é uma MT universal. Se y é uma *descrição* de x , então $\mathcal{U}(y) = x$

A *Descrição Mínima* de uma string x , denotada $d_m(x)$, é a *menor string* $\sqcup M \sqcup w$ em ordem lexicográfica tal que $\sqcup M \sqcup w$ é uma descrição de x .

A *Complexidade de Komogorov* de uma string x , denotada $K(x)$, é o tamanho da descrição mínima de x , ou seja, $K(x) = |d_m(x)|$.

Complexidade de Kolmogorov

A *Complexidade de Kolmogorov* de uma string x , denotada $K(x)$, é o tamanho da descrição mínima de x , ou seja, $K(x) = |d_m(x)|$.

Complexidade de Kolmogorov

A *Complexidade de Kolmogorov* de uma string x , denotada $K(x)$, é o tamanho da descrição mínima de x , ou seja, $K(x) = |d_m(x)|$.

Teorema: Existe uma constante c tal que $\forall x \in \Sigma^*, K(x) \leq |x| + c$.

Complexidade de Kolmogorov

A *Complexidade de Kolmogorov* de uma string x , denotada $K(x)$, é o tamanho da descrição mínima de x , ou seja, $K(x) = |d_m(x)|$.

Teorema: Existe uma constante c tal que $\forall x \in \Sigma^*, K(x) \leq |x| + c$.

Prova: Considere a MT M tal que $\forall x, M(x) = x$.

Complexidade de Kolmogorov

A *Complexidade de Kolmogorov* de uma string x , denotada $K(x)$, é o tamanho da descrição mínima de x , ou seja, $K(x) = |d_m(x)|$.

Teorema: Existe uma constante c tal que $\forall x \in \Sigma^*, K(x) \leq |x| + c$.

Prova: Considere a MT M tal que $\forall x, M(x) = x$.

- Seja $c = |LM|$.

Complexidade de Kolmogorov

A *Complexidade de Kolmogorov* de uma string x , denotada $K(x)$, é o tamanho da descrição mínima de x , ou seja, $K(x) = |d_m(x)|$.

Teorema: Existe uma constante c tal que $\forall x \in \Sigma^*, K(x) \leq |x| + c$.

Prova: Considere a MT M tal que $\forall x, M(x) = x$.

- Seja $c = |\llcorner M \lrcorner|$.
- $\llcorner M \lrcorner x$ é uma descrição de x de tamanho

Complexidade de Kolmogorov

A *Complexidade de Kolmogorov* de uma string x , denotada $K(x)$, é o tamanho da descrição mínima de x , ou seja, $K(x) = |d_m(x)|$.

Teorema: Existe uma constante c tal que $\forall x \in \Sigma^*, K(x) \leq |x| + c$.

Prova: Considere a MT M tal que $\forall x, M(x) = x$.

- Seja $c = |\llcorner M \lrcorner|$.
- $\llcorner M \lrcorner x$ é uma descrição de x de tamanho $|x| + c$

Complexidade de Kolmogorov

A *Complexidade de Kolmogorov* de uma string x , denotada $K(x)$, é o tamanho da descrição mínima de x , ou seja, $K(x) = |d_m(x)|$.

Teorema: Existe uma constante c tal que $\forall x \in \Sigma^*, K(x) \leq |x| + c$.

Prova: Considere a MT M tal que $\forall x, M(x) = x$.

- Seja $c = |\llcorner M \lrcorner|$.
- $\llcorner M \lrcorner x$ é uma descrição de x de tamanho $|x| + c$
- Portanto, $K(x) \leq |x| + c$.

Complexidade de Kolmogorov

Complexidade de Kolmogorov

Def.: Uma string x é *c -compressível* se $K(x) \leq |x| - c$.

Complexidade de Kolmogorov

Def.: Uma string x é *c-compressível* se $K(x) \leq |x| - c$. Caso x não seja *c-compressível*, dizemos que x é *c-incompressível*.

Complexidade de Kolmogorov

Def.: Uma string x é *c-compressível* se $K(x) \leq |x| - c$. Caso x não seja c -compressível, dizemos que x é *c-incompressível*.

Def.: Uma string 1-incompressível é dita uma string *incompressível*.

Complexidade de Kolmogorov

Def.: Uma string x é *c-compressível* se $K(x) \leq |x| - c$. Caso x não seja c -compressível, dizemos que x é *c-incompressível*.

Def.: Uma string 1-incompressível é dita uma string *incompressível*.

Teorema: Existem strings incompressíveis de todos os tamanhos.

Complexidade de Kolmogorov

Def.: Uma string x é *c-compressível* se $K(x) \leq |x| - c$. Caso x não seja c -compressível, dizemos que x é *c-incompressível*.

Def.: Uma string 1-incompressível é dita uma string *incompressível*.

Teorema: Existem strings incompressíveis de todos os tamanhos.

Prova: Seja $n \in \mathbb{N}$.

Complexidade de Kolmogorov

Def.: Uma string x é *c-compressível* se $K(x) \leq |x| - c$. Caso x não seja c -compressível, dizemos que x é *c-incompressível*.

Def.: Uma string 1-incompressível é dita uma string *incompressível*.

Teorema: Existem strings incompressíveis de todos os tamanhos.

Prova: Seja $n \in \mathbb{N}$. Observe o seguinte:

Complexidade de Kolmogorov

Def.: Uma string x é *c-compressível* se $K(x) \leq |x| - c$. Caso x não seja c -compressível, dizemos que x é *c-incompressível*.

Def.: Uma string 1-incompressível é dita uma string *incompressível*.

Teorema: Existem strings incompressíveis de todos os tamanhos.

Prova: Seja $n \in \mathbb{N}$. Observe o seguinte:

- Número de strings de tamanho n :

Complexidade de Kolmogorov

Def.: Uma string x é *c-compressível* se $K(x) \leq |x| - c$. Caso x não seja *c-compressível*, dizemos que x é *c-incompressível*.

Def.: Uma string 1-incompressível é dita uma string *incompressível*.

Teorema: Existem strings incompressíveis de todos os tamanhos.

Prova: Seja $n \in \mathbb{N}$. Observe o seguinte:

- Número de strings de tamanho n : 2^n .

Complexidade de Kolmogorov

Def.: Uma string x é *c-compressível* se $K(x) \leq |x| - c$. Caso x não seja *c-compressível*, dizemos que x é *c-incompressível*.

Def.: Uma string 1-incompressível é dita uma string *incompressível*.

Teorema: Existem strings incompressíveis de todos os tamanhos.

Prova: Seja $n \in \mathbb{N}$. Observe o seguinte:

- Número de strings de tamanho n : 2^n .
- Número de strings menores que n :

Complexidade de Kolmogorov

Def.: Uma string x é *c-compressível* se $K(x) \leq |x| - c$. Caso x não seja c -compressível, dizemos que x é *c-incompressível*.

Def.: Uma string 1-incompressível é dita uma string *incompressível*.

Teorema: Existem strings incompressíveis de todos os tamanhos.

Prova: Seja $n \in \mathbb{N}$. Observe o seguinte:

- Número de strings de tamanho n : 2^n .
- Número de strings menores que n : $\sum_{i=0}^{n-1} 2^i =$

Complexidade de Kolmogorov

Def.: Uma string é x é *c-compressível* se $K(x) \leq |x| - c$. Caso x não seja c -compressível, dizemos que x é *c-incompressível*.

Def.: Uma string 1-incompressível é dita uma string *incompressível*.

Teorema: Existem strings incompressíveis de todos os tamanhos.

Prova: Seja $n \in \mathbb{N}$. Observe o seguinte:

- Número de strings de tamanho n : 2^n .
- Número de strings menores que n : $\sum_{i=0}^{n-1} 2^i = 2^n - 1$.

Complexidade de Kolmogorov

Def.: Uma string x é *c-compressível* se $K(x) \leq |x| - c$. Caso x não seja *c-compressível*, dizemos que x é *c-incompressível*.

Def.: Uma string 1-incompressível é dita uma string *incompressível*.

Teorema: Existem strings incompressíveis de todos os tamanhos.

Prova: Seja $n \in \mathbb{N}$. Observe o seguinte:

- Número de strings de tamanho n : 2^n .
- Número de strings menores que n : $\sum_{i=0}^{n-1} 2^i = 2^n - 1$.
- Como $2^n > 2^n - 1$, então existem mais strings x de tamanho n do que descrições $d(x)$, tal que $|d(x)| < n$.

Complexidade de Kolmogorov

Def.: Uma string x é *c-compressível* se $K(x) \leq |x| - c$. Caso x não seja *c-compressível*, dizemos que x é *c-incompressível*.

Def.: Uma string 1-incompressível é dita uma string *incompressível*.

Teorema: Existem strings incompressíveis de todos os tamanhos.

Prova: Seja $n \in \mathbb{N}$. Observe o seguinte:

- Número de strings de tamanho n : 2^n .
- Número de strings menores que n : $\sum_{i=0}^{n-1} 2^i = 2^n - 1$.
- Como $2^n > 2^n - 1$, então existem mais strings x de tamanho n do que descrições $d(x)$, tal que $|d(x)| < n$.
- Logo, existe pelo menos uma string x de tamanho n que não admite nenhuma descrição menor do que n .

Complexidade de Kolmogorov

Def.: Uma string x é *c-compressível* se $K(x) \leq |x| - c$. Caso x não seja *c-compressível*, dizemos que x é *c-incompressível*.

Def.: Uma string 1-incompressível é dita uma string *incompressível*.

Teorema: Existem strings incompressíveis de todos os tamanhos.

Prova: Seja $n \in \mathbb{N}$. Observe o seguinte:

- Número de strings de tamanho n : 2^n .
- Número de strings menores que n : $\sum_{i=0}^{n-1} 2^i = 2^n - 1$.
- Como $2^n > 2^n - 1$, então existem mais strings x de tamanho n do que descrições $d(x)$, tal que $|d(x)| < n$.
- Logo, existe pelo menos uma string x de tamanho n que não admite nenhuma descrição menor do que n . Portanto, x é incompressível.

Aleatoridade e incompressibilidade (Seção 8.3 – Opcional)

Aleatoridade e incompressibilidade (Seção 8.3 – Opcional)

Referência opcional: [Kolmogorov Complexity and its Applications](#), Li & Vitányi.

Aleatoridade e incompressibilidade (Seção 8.3 – Opcional)

Referência opcional: [Kolmogorov Complexity and its Applications](#), Li & Vitányi.

Uma função computável $P : \Sigma^* \rightarrow \{0, 1\}$ é chamada uma [propriedade de strings](#)

Aleatoridade e incompressibilidade (Seção 8.3 – Opcional)

Referência opcional: [Kolmogorov Complexity and its Applications](#), Li & Vitányi.

Uma função computável $P : \Sigma^* \rightarrow \{0, 1\}$ é chamada uma [propriedade de strings](#)
(note: aqui propriedades são computáveis por definição)

Aleatoridade e incompressibilidade (Seção 8.3 – Opcional)

Referência opcional: [Kolmogorov Complexity and its Applications](#), Li & Vitányi.

Uma função computável $P : \Sigma^* \rightarrow \{0, 1\}$ é chamada uma [propriedade de strings](#)
(note: aqui propriedades são computáveis por definição)

Terminologia: Dado x , dizemos que “ x respeita a propriedade P ” se $P(x) = 1$.

Aleatoridade e incompressibilidade (Seção 8.3 – Opcional)

Referência opcional: [Kolmogorov Complexity and its Applications](#), Li & Vitányi.

Uma função computável $P : \Sigma^* \rightarrow \{0, 1\}$ é chamada uma [propriedade de strings](#)
(note: aqui propriedades são computáveis por definição)

Terminologia: Dado x , dizemos que “ x respeita a propriedade P ” se $P(x) = 1$.

Uma propriedade P é dita uma [propriedade de strings aleatórias](#) se

Aleatoridade e incompressibilidade (Seção 8.3 – Opcional)

Referência opcional: [Kolmogorov Complexity and its Applications](#), Li & Vitányi.

Uma função computável $P : \Sigma^* \rightarrow \{0, 1\}$ é chamada uma [propriedade de strings](#)
(note: aqui propriedades são computáveis por definição)

Terminologia: Dado x , dizemos que “ x respeita a propriedade P ” se $P(x) = 1$.

Uma propriedade P é dita uma [propriedade de strings aleatórias](#) se

- Ao sortearmos a string binária $x = x_1x_2 \cdots x_n$, $\Pr[P(x) = 1] \rightarrow 1$ quando $n \rightarrow \infty$

Aleatoridade e incompressibilidade (Seção 8.3 – Opcional)

Referência opcional: [Kolmogorov Complexity and its Applications](#), Li & Vitányi.

Uma função computável $P : \Sigma^* \rightarrow \{0, 1\}$ é chamada uma **propriedade de strings**
(note: aqui propriedades são computáveis por definição)

Terminologia: Dado x , dizemos que “ x respeita a propriedade P ” se $P(x) = 1$.

Uma propriedade P é dita uma **propriedade de strings aleatórias** se

- Ao sortearmos a string binária $x = x_1x_2 \cdots x_n$, $\Pr[P(x) = 1] \rightarrow 1$ quando $n \rightarrow \infty$

Exemplos de propriedades de uma string aleatórias:

Aleatoridade e incompressibilidade (Seção 8.3 – Opcional)

Referência opcional: [Kolmogorov Complexity and its Applications](#), Li & Vitányi.

Uma função computável $P : \Sigma^* \rightarrow \{0, 1\}$ é chamada uma **propriedade de strings**
(note: aqui propriedades são computáveis por definição)

Terminologia: Dado x , dizemos que “ x respeita a propriedade P ” se $P(x) = 1$.

Uma propriedade P é dita uma **propriedade de strings aleatórias** se

- Ao sortearmos a string binária $x = x_1 x_2 \cdots x_n$, $\Pr[P(x) = 1] \rightarrow 1$ quando $n \rightarrow \infty$

Exemplos de propriedades de uma string aleatórias:

- Quantidade 0's e 1's é equilibrada;

Aleatoridade e incompressibilidade (Seção 8.3 – Opcional)

Referência opcional: [Kolmogorov Complexity and its Applications](#), Li & Vitányi.

Uma função computável $P : \Sigma^* \rightarrow \{0, 1\}$ é chamada uma **propriedade de strings**
(note: aqui propriedades são computáveis por definição)

Terminologia: Dado x , dizemos que “ x respeita a propriedade P ” se $P(x) = 1$.

Uma propriedade P é dita uma **propriedade de strings aleatórias** se

- Ao sortearmos a string binária $x = x_1x_2 \cdots x_n$, $\Pr[P(x) = 1] \rightarrow 1$ quando $n \rightarrow \infty$

Exemplos de propriedades de uma string aleatórias:

- Quantidade 0's e 1's é equilibrada;
- Tamanho esperado da maior sequência de 0's é $\log_2 n$;

Aleatoridade e incompressibilidade (Seção 8.3 – Opcional)

Referência opcional: [Kolmogorov Complexity and its Applications](#), Li & Vitányi.

Uma função computável $P : \Sigma^* \rightarrow \{0, 1\}$ é chamada uma [propriedade de strings](#)
(note: aqui propriedades são computáveis por definição)

Terminologia: Dado x , dizemos que “ x respeita a propriedade P ” se $P(x) = 1$.

Uma propriedade P é dita uma [propriedade de strings aleatórias](#) se

- Ao sortearmos a string binária $x = x_1x_2 \cdots x_n$, $\Pr[P(x) = 1] \rightarrow 1$ quando $n \rightarrow \infty$

Exemplos de propriedades de uma string aleatórias:

- Quantidade 0's e 1's é equilibrada;
- Tamanho esperado da maior sequência de 0's é $\log_2 n$;
- Todos os “padrões” de tamanho $\leq \log_2 n$ ocorrem com mesma probabilidade.

Aleatoridade e incompressibilidade (Seção 8.3 – Opcional)

Referência opcional: [Kolmogorov Complexity and its Applications](#), Li & Vitányi.

Uma função computável $P : \Sigma^* \rightarrow \{0, 1\}$ é chamada uma **propriedade de strings**
(note: aqui propriedades são computáveis por definição)

Terminologia: Dado x , dizemos que “ x respeita a propriedade P ” se $P(x) = 1$.

Uma propriedade P é dita uma **propriedade de strings aleatórias** se

- Ao sortearmos a string binária $x = x_1x_2 \cdots x_n$, $\Pr[P(x) = 1] \rightarrow 1$ quando $n \rightarrow \infty$

Exemplos de propriedades de uma string aleatórias:

- Quantidade 0's e 1's é equilibrada;
- Tamanho esperado da maior sequência de 0's é $\log_2 n$;
- Todos os “padrões” de tamanho $\leq \log_2 n$ ocorrem com mesma probabilidade.

Teorema: Para toda propriedade P de strings aleatórias, o seguinte conjunto é finito:

Aleatoriedade e incompressibilidade (Seção 8.3 – Opcional)

Referência opcional: [Kolmogorov Complexity and its Applications](#), Li & Vitányi.

Uma função computável $P : \Sigma^* \rightarrow \{0, 1\}$ é chamada uma **propriedade de strings**
(note: aqui propriedades são computáveis por definição)

Terminologia: Dado x , dizemos que “ x respeita a propriedade P ” se $P(x) = 1$.

Uma propriedade P é dita uma **propriedade de strings aleatórias** se

- Ao sortearmos a string binária $x = x_1 x_2 \cdots x_n$, $\Pr[P(x) = 1] \rightarrow 1$ quando $n \rightarrow \infty$

Exemplos de propriedades de uma string aleatórias:

- Quantidade 0's e 1's é equilibrada;
- Tamanho esperado da maior sequência de 0's é $\log_2 n$;
- Todos os “padrões” de tamanho $\leq \log_2 n$ ocorrem com mesma probabilidade.

Teorema: Para toda propriedade P de strings aleatórias, o seguinte conjunto é finito:

$$I_P = \{x : x \text{ é incompressível e } x \text{ não respeita } P\}$$

Aleatoridade e incompressibilidade (Seção 8.3 – Opcional)

Referência opcional: Kolmogorov Complexity and its Applications, Li & Vitányi.

Uma função computável $P : \Sigma^* \rightarrow \{0, 1\}$ é chamada uma **propriedade de strings**
(note: aqui propriedades são computáveis por definição)

Terminologia: Dado x , dizemos que “ x respeita a propriedade P ” se $P(x) = 1$.

Uma propriedade P é dita uma **propriedade de strings aleatórias** se

- Ao sortearmos a string binária $x = x_1x_2 \cdots x_n$, $\Pr[P(x) = 1] \rightarrow 1$ quando $n \rightarrow \infty$

Exemplos de propriedades de uma string aleatórias:

- Quantidade 0's e 1's é equilibrada;
- Tamanho esperado da maior sequência de 0's é $\log_2 n$;
- Todos os “padrões” de tamanho $\leq \log_2 n$ ocorrem com mesma probabilidade.

Teorema: Para toda propriedade P de strings aleatórias, o seguinte conjunto é finito:

$$I_P = \{x : x \text{ é incompressível e } x \text{ não respeita } P\}$$

Corolário: Sejam $P_1, P_2, \dots, P_k$ propriedades de strings aleatórias. Então existe constante c tal que todas as strings incompressível x maiores que c respeitam todas propriedades $P_1, \dots, P_k$.

Incomputabilidade da Complexidade de Kolmogorov

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string.

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w ,

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \lfloor K(w) \rfloor$.

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \lfloor K(w) \rfloor$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \ulcorner K(w) \urcorner$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \ulcorner K(x) \urcorner$

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \ulcorner K(w) \urcorner$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \ulcorner K(x) \urcorner$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \lfloor K(w) \rfloor$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \lfloor K(x) \rfloor$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \lfloor K(w) \rfloor$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \lfloor K(x) \rfloor$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \lfloor K(w) \rfloor$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \lfloor K(x) \rfloor$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \lfloor K(w) \rfloor$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \lfloor K(x) \rfloor$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \lfloor K(w) \rfloor$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \lfloor K(x) \rfloor$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

- Isto é, para toda entrada $\lfloor n \rfloor$, $D(\lfloor n \rfloor) = w_n$, para alguma string w_n .

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \lfloor K(w) \rfloor$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \lfloor K(x) \rfloor$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

- Isto é, para toda entrada $\lfloor n \rfloor$, $D(\lfloor n \rfloor) = w_n$, para alguma string w_n .

$\therefore$

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \lfloor K(w) \rfloor$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \lfloor K(x) \rfloor$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

- Isto é, para toda entrada $\lfloor n \rfloor$, $D(\lfloor n \rfloor) = w_n$, para alguma string w_n .
 $\therefore \exists$ string w_n cuja descrição tem tamanho $|D| + |\lfloor n \rfloor|$

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \ulcorner K(w) \urcorner$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \ulcorner K(x) \urcorner$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

- Isto é, para toda entrada $\ulcorner n \urcorner$, $D(\ulcorner n \urcorner) = w_n$, para alguma string w_n .
 $\therefore \exists$ string w_n cuja descrição tem tamanho $|D| + |\ulcorner n \urcorner|$
- $|\ulcorner n \urcorner|$ tem no máximo $\log_2 n + 1$ bits

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \lfloor K(w) \rfloor$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \lfloor K(x) \rfloor$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

- Isto é, para toda entrada $\lfloor n \rfloor$, $D(\lfloor n \rfloor) = w_n$, para alguma string w_n .
 $\therefore \exists$ string w_n cuja descrição tem tamanho $|D| + |\lfloor n \rfloor|$
- $|\lfloor n \rfloor|$ tem no máximo $\log_2 n + 1$ bits $\therefore$

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \ulcorner K(w) \urcorner$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \ulcorner K(x) \urcorner$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

- Isto é, para toda entrada $\ulcorner n \urcorner$, $D(\ulcorner n \urcorner) = w_n$, para alguma string w_n .
 $\therefore \exists$ string w_n cuja descrição tem tamanho $|D| + |\ulcorner n \urcorner|$
- $|\ulcorner n \urcorner|$ tem no máximo $\log_2 n + 1$ bits $\therefore \exists c > \ulcorner D \urcorner$ tal que $K(w_n) \leq \log_2 n + c$

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \ulcorner K(w) \urcorner$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \ulcorner K(x) \urcorner$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

- Isto é, para toda entrada $\ulcorner n \urcorner$, $D(\ulcorner n \urcorner) = w_n$, para alguma string w_n .
 $\therefore \exists$ string w_n cuja descrição tem tamanho $|D| + |\ulcorner n \urcorner|$
- $|\ulcorner n \urcorner|$ tem no máximo $\log_2 n + 1$ bits $\therefore \exists c > \ulcorner D \urcorner$ tal que $K(w_n) \leq \log_2 n + c$
(a constante c absorve detalhes da codificação e separadores).

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \ulcorner K(w) \urcorner$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \ulcorner K(x) \urcorner$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

- Isto é, para toda entrada $\ulcorner n \urcorner$, $D(\ulcorner n \urcorner) = w_n$, para alguma string w_n .
 $\therefore \exists$ string w_n cuja descrição tem tamanho $|D| + |\ulcorner n \urcorner|$
- $|\ulcorner n \urcorner|$ tem no máximo $\log_2 n + 1$ bits $\therefore \exists c > \ulcorner D \urcorner$ tal que $K(w_n) \leq \log_2 n + c$
(a constante c absorve detalhes da codificação e separadores).

Pelo passo (3), temos $K(w_n) > n$

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \lfloor K(w) \rfloor$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \lfloor K(x) \rfloor$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

- Isto é, para toda entrada $\lfloor n \rfloor$, $D(\lfloor n \rfloor) = w_n$, para alguma string w_n .
 $\therefore \exists$ string w_n cuja descrição tem tamanho $|D| + |\lfloor n \rfloor|$
- $|\lfloor n \rfloor|$ tem no máximo $\log_2 n + 1$ bits $\therefore \exists c > |D|$ tal que $K(w_n) \leq \log_2 n + c$
(a constante c absorve detalhes da codificação e separadores).

Pelo passo (3), temos $K(w_n) > n \therefore$

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \lfloor K(w) \rfloor$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \lfloor K(x) \rfloor$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

- Isto é, para toda entrada $\lfloor n \rfloor$, $D(\lfloor n \rfloor) = w_n$, para alguma string w_n .
 $\therefore \exists$ string w_n cuja descrição tem tamanho $|D| + |\lfloor n \rfloor|$
- $|\lfloor n \rfloor|$ tem no máximo $\log_2 n + 1$ bits $\therefore \exists c > |D|$ tal que $K(w_n) \leq \log_2 n + c$
(a constante c absorve detalhes da codificação e separadores).

Pelo passo (3), temos $K(w_n) > n \therefore n < K(w_n) \leq c + \log_2 n$.

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \ulcorner K(w) \urcorner$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \ulcorner K(x) \urcorner$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

- Isto é, para toda entrada $\ulcorner n \urcorner$, $D(\ulcorner n \urcorner) = w_n$, para alguma string w_n .
 $\therefore \exists$ string w_n cuja descrição tem tamanho $|D| + |\ulcorner n \urcorner|$
- $|\ulcorner n \urcorner|$ tem no máximo $\log_2 n + 1$ bits $\therefore \exists c > \ulcorner D \urcorner$ tal que $K(w_n) \leq \log_2 n + c$
(a constante c absorve detalhes da codificação e separadores).

Pelo passo (3), temos $K(w_n) > n \therefore n < K(w_n) \leq c + \log_2 n$.

Para n suficientemente grande a desigualdade é impossível:

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \ulcorner K(w) \urcorner$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \ulcorner K(x) \urcorner$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

- Isto é, para toda entrada $\ulcorner n \urcorner$, $D(\ulcorner n \urcorner) = w_n$, para alguma string w_n .
 $\therefore \exists$ string w_n cuja descrição tem tamanho $|D| + |\ulcorner n \urcorner|$
- $|\ulcorner n \urcorner|$ tem no máximo $\log_2 n + 1$ bits $\therefore \exists c > \ulcorner D \urcorner$ tal que $K(w_n) \leq \log_2 n + c$
(a constante c absorve detalhes da codificação e separadores).

Pelo passo (3), temos $K(w_n) > n \therefore n < K(w_n) \leq c + \log_2 n$.

Para n suficientemente grande a desigualdade é impossível: lado esquerdo cresce linearmente
lado direito cresce como $\log n$

Incomputabilidade da Complexidade de Kolmogorov

Teorema: Seja w uma string. Não existe MT M_K que compute a complexidade de Kolmogorov de w , i.e., tal que $M_K(w) = \ulcorner K(w) \urcorner$.

Prova: Suponha que $\exists$ uma MT M_K que compute a complexidade de Kolmogorov de uma string

- Isto é, $\forall x \in \Sigma^*, M_K(x) = \ulcorner K(x) \urcorner$

Usando M_K podemos construir a uma MT D que, dado um natural n em binário, faz:

- (1) Enumera strings $x \in \{0,1\}^*$ em ordem lexicográfica;
- (2) Para cada string x enumerada, usa M_K para calcular $K(x)$;
- (3) Ao encontrar a primeira string x tal que $K(x) > n$, escreve x na fita e para.

Note: sempre existe x com $K(x) > n$, pois $\exists$ string incompressível de todo tamanho

- Isto é, para toda entrada $\ulcorner n \urcorner$, $D(\ulcorner n \urcorner) = w_n$, para alguma string w_n .
 $\therefore \exists$ string w_n cuja descrição tem tamanho $|D| + |\ulcorner n \urcorner|$
- $|\ulcorner n \urcorner|$ tem no máximo $\log_2 n + 1$ bits $\therefore \exists c > \ulcorner D \urcorner$ tal que $K(w_n) \leq \log_2 n + c$
(a constante c absorve detalhes da codificação e separadores).

Pelo passo (3), temos $K(w_n) > n \therefore n < K(w_n) \leq c + \log_2 n$.

Para n suficientemente grande a desigualdade é impossível: lado esquerdo cresce linearmente
lado direito cresce como $\log n$

Uma contradição, portanto, M_K não existe.