Acceptable Complexity Measures of Theorems

Bruno Grenet

École Normale Supérieure de Lyon
http://perso.ens-lyon.fr/bruno.grenet/

November 12, 2008
LIF, Marseille

Historical Overview

Historical Overview

- 1931: Gödel publishes his Incompleteness Theorem

Historical Overview

- 1931: Gödel publishes his Incompleteness Theorem

Some true mathematical statements are unprovable.

Historical Overview

- 1931: Gödel publishes his Incompleteness Theorem

Some true mathematical statements are unprovable.

  - Are there many such statements?

Historical Overview

- 1931: Gödel publishes his Incompleteness Theorem

Some true mathematical statements are unprovable.

  - Are there many such statements?
  - Are there natural such statements?

Historical Overview

- 1931: Gödel publishes his Incompleteness Theorem

Some true mathematical statements are unprovable.

- Are there many such statements?
- Are there natural such statements?
- Why are they unprovable?

Historical Overview

- 1931: Gödel publishes his Incompleteness Theorem

Some true mathematical statements are unprovable.

  - Are there many such statements?
  - Are there natural such statements?
  - Why are they unprovable?

- 1974: Chaitin proposes his "heuristic principle"

Historical Overview

- 1931: Gödel publishes his Incompleteness Theorem

Some true mathematical statements are unprovable.

  - Are there many such statements?
  - Are there natural such statements?
  - Why are they unprovable?

- 1974: Chaitin proposes his “heuristic principle”

The theorems of a finitely-specified theory cannot be significantly more complex than the theory itself.

Historical Overview

- 1931: Gödel publishes his Incompleteness Theorem

Some true mathematical statements are unprovable.

  - Are there many such statements?
  - Are there natural such statements?
  - Why are they unprovable?

- 1974: Chaitin proposes his “heuristic principle”

The theorems of a finitely-specified theory cannot be significantly more complex than the theory itself.

- 2005: Calude and Jürgensen prove the “heuristic principle”

Goal

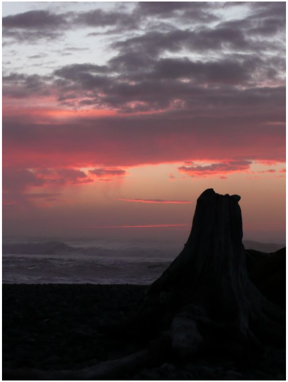
- $\delta(x) = H(x) - |x|$ where H is the *program-size* complexity.

Goal

- $\delta(x) = H(x) - |x|$ where H is the *program-size* complexity.
- Is it the only measure satisfying the heuristic principle?

Goal

- $\delta(x) = H(x) - |x|$ where H is the *program-size* complexity.
- Is it the only measure satisfying the heuristic principle?



Gillepsie Beach, South Island

Outline

Outline

1. A few definitions
2. About δ
3. Acceptable Complexity Measures
4. An Independence Result
5. Other measures?

Aphabets and strings

For $i \geq 2$,

- X_i: alphabet with i elements

Aphabets and strings

For $i \geq 2$,

- X_i: alphabet with i elements
- X_i^*: set of finite strings on X_i, including the empty string λ

Aphabets and strings

For $i \geq 2$,

- X_i: alphabet with i elements
- X_i^*: set of finite strings on X_i, including the empty string λ
- $|w|_i$: length of w

Aphabets and strings

For $i \geq 2$,

- X_i: alphabet with i elements
- X_i^*: set of finite strings on X_i, including the empty string λ
- $|w|_i$: length of w
- Gödel numbering for the language L: computable one-to-one function $g : L \to X_2^*$

Aphabets and strings

For $i \geq 2$,

- X_i: alphabet with i elements
- X_i^*: set of finite strings on X_i, including the empty string λ
- $|w|_i$: length of w
- Gödel numbering for the language L: computable one-to-one function $g : L \to X_2^*$
- G: set of all the Gödel numberings

Self-delimiting Turing Machines

- Prefix-free set: $u \in S$ implies that $uv \notin S$ $(v \neq \lambda)$

Self-delimiting Turing Machines

- Prefix-free set: $u \in S$ implies that $uv \notin S$ $(v \neq \lambda)$
- $PROG_T = \{x \in X_i^* : T(x) \downarrow\}$

Self-delimiting Turing Machines

- Prefix-free set: $u \in S$ implies that $uv \notin S$ $(v \neq \lambda)$
- $PROG_T = \{x \in X_i^* : T(x) \downarrow\}$
- Self-delimiting Turing Machine: $PROG_T$ is prefix-free

Self-delimiting Turing Machines

- Prefix-free set: $u \in S$ implies that $uv \notin S$ $(v \neq \lambda)$
- $PROG_T = \{x \in X_i^* : T(x) \downarrow\}$
- Self-delimiting Turing Machine: $PROG_T$ is prefix-free
- Kraft's inequality: for a prefix-free set S, note $r_k = \operatorname{card}\{x \in S : |x|_i = k\}$. Then

$$\sum_{k=1}^{\infty} r_k \cdot i^{-k} \leq 1.$$

Kraft-Chaitin Theorem

Let $(n_k)_{k\in\mathbb{N}}$ be a computable sequence of non-negative integers such that

$$\sum_{k=1}^{\infty} i^{-n_k} \leq 1.$$

Then we can effectively construct a prefix-free sequence of strings $(w_k)_{k\in\mathbb{N}}$ such that for each $k \geq 1$, $|w_k|_i = n_k$.

Program-size complexity

Definition

$$H_{i,T}(x) = \min\left\{|y|_i : y \in X_i^* \text{ and } T(y) = x\right\}$$

Program-size complexity

Definition

$H_{i,T}(x) = \min \{|y|_i : y \in X_i^* \text{ and } T(y) = x\}$

Invariance Theorem

There exists a universal machine U_i such that for every T, there exists c such that

$$H_{i,U_i}(x) \leq H_{i,T}(x) + c$$

Program-size complexity

Definition

$H_{i,T}(x) = \min \{|y|_i : y \in X_i^* \text{ and } T(y) = x\}$

Invariance Theorem

There exists a universal machine U_i such that for every T, there exists c such that

$$H_{i,U_i}(x) \leq H_{i,T}(x) + c$$

$$H_i \triangleq H_{i,U_i}$$

Program-size complexity

Definition

$H_{i,T}(x) = \min\{|y|_i : y \in X_i^* \text{ and } T(y) = x\}$

Invariance Theorem

There exists a universal machine U_i such that for every T, there exists c such that

$$H_{i,U_i}(x) \leq H_{i,T}(x) + c$$

$$H_i \triangleq H_{i,U_i}$$

Definition

x^* is the lexicographically first string of length $H_i(x)$ such that $U_i(x^*) = x$.

Outline

Definitions

Definition

$$\delta_i(x) = H_i(x) - |x|_i \, , i \geq 2$$

Definitions

Definition

$$\delta_i(x) = H_i(x) - |x|_i \, , i \geq 2$$

Definition

$$\delta_g(u) = H_2(g(u)) - \lceil \log_2(i) \cdot |x|_i \rceil \, ,$$

where g is a Gödel numbering.

Invariance of the measure

Theorem

There exists a constant c such that

$$|H_2(g(u)) - \log_2(i) \cdot H_i(u)| \leq c.$$

Invariance of the measure

Theorem

There exists a constant c such that

$$|H_2(g(u)) - \log_2(i) \cdot H_i(u)| \leq c.$$

Corollary

- With the same constant c as in the theorem, it holds that

$$|\delta_g(u) - \log_2(i) \cdot \delta_i(u)| \leq c + 1.$$

Invariance of the measure

Theorem

There exists a constant c such that

$$|H_2(g(u)) - \log_2(i) \cdot H_i(u)| \leq c.$$

Corollary

- With the same constant c as in the theorem, it holds that

$$|\delta_g(u) - \log_2(i) \cdot \delta_i(u)| \leq c + 1.$$

- For every g and g', there exists a constant d such that

$$\left|H_2(g(u)) - H_2(g'(u))\right| \leq d \text{ and } \left|\delta_g(u) - \delta_{g'}(u)\right| \leq d + 1.$$

Proof sketch for the theorem - 1

$$H_2(g(u)) \leq \log_2(i) \cdot H_i(u) + c_1.$$

Proof sketch for the theorem - 1

$$H_2(g(u)) \leq \log_2(i) \cdot H_i(u) + c_1.$$

- $n_w \triangleq \lceil \log_2(i) \cdot |w|_i \rceil$

Proof sketch for the theorem - 1

$$H_2(g(u)) \leq \log_2(i) \cdot H_i(u) + c_1.$$

- $n_w \triangleq \lceil \log_2(i) \cdot |w|_i \rceil$

$$\sum_{w \in PROG_{U_i}} 2^{-n_w} = \sum_{w \in PROG_{U_i}} 2^{-\lceil \log_2(i) \cdot |w|_i \rceil} \leq \sum_{w \in PROG_{U_i}} i^{-|w|_i} \leq 1$$

Proof sketch for the theorem - 1

$$H_2(g(u)) \leq \log_2(i) \cdot H_i(u) + c_1.$$

- $n_w \triangleq \lceil \log_2(i) \cdot |w|_i \rceil$

$$\sum_{w \in PROG_{U_i}} 2^{-n_w} = \sum_{w \in PROG_{U_i}} 2^{-\lceil \log_2(i) \cdot |w|_i \rceil} \leq \sum_{w \in PROG_{U_i}} i^{-|w|_i} \leq 1$$

- By Kraft-Chaitin Theorem, we can construct $\{s_w \in X_2^* : w \in PROG_{U_i}, |s_w|_2 = n_w\}$, prefix-free and c.e.

Proof sketch for the theorem - 1

$$H_2(g(u)) \leq \log_2(i) \cdot H_i(u) + c_1.$$

- $n_w \triangleq \lceil \log_2(i) \cdot |w|_i \rceil$

$$\sum_{w \in PROG_{U_i}} 2^{-n_w} = \sum_{w \in PROG_{U_i}} 2^{-\lceil \log_2(i) \cdot |w|_i \rceil} \leq \sum_{w \in PROG_{U_i}} i^{-|w|_i} \leq 1$$

- By Kraft-Chaitin Theorem, we can construct $\{s_w \in X_2^* : w \in PROG_{U_i}, |s_w|_2 = n_w\}$, prefix-free and c.e.
- We define a machine C such that $C(s_w) = g(U_i(w))$.

Proof sketch for the theorem - 1

$$H_2(g(u)) \leq \log_2(i) \cdot H_i(u) + c_1.$$

- $n_w \triangleq \lceil \log_2(i) \cdot |w|_i \rceil$

$$\sum_{w \in PROG_{U_i}} 2^{-n_w} = \sum_{w \in PROG_{U_i}} 2^{-\lceil \log_2(i) \cdot |w|_i \rceil} \leq \sum_{w \in PROG_{U_i}} i^{-|w|_i} \leq 1$$

- By Kraft-Chaitin Theorem, we can construct $\{s_w \in X_2^* : w \in PROG_{U_i}, |s_w|_2 = n_w\}$, prefix-free and c.e.
- We define a machine C such that $C(s_w) = g(U_i(w))$.
- Note that $C(s_{w^*}) = g(U_i(w^*)) = g(w)$.

Proof sketch for the theorem - 1

$$H_2(g(u)) \leq \log_2(i) \cdot H_i(u) + c_1.$$

- $n_w \triangleq \lceil \log_2(i) \cdot |w|_i \rceil$

$$\sum_{w \in PROG_{U_i}} 2^{-n_w} = \sum_{w \in PROG_{U_i}} 2^{-\lceil \log_2(i) \cdot |w|_i \rceil} \leq \sum_{w \in PROG_{U_i}} i^{-|w|_i} \leq 1$$

- By Kraft-Chaitin Theorem, we can construct $\{s_w \in X_2^* : w \in PROG_{U_i}, |s_w|_2 = n_w\}$, prefix-free and c.e.
- We define a machine C such that $C(s_w) = g(U_i(w))$.
- Note that $C(s_{w^*}) = g(U_i(w^*)) = g(w)$.

$$\begin{aligned} H_C(g(w)) &\leq |s_{w^*}|_2 = \lceil \log_2(i) \cdot |w^*|_i \rceil = \lceil \log_2(i) \cdot H_i(w) \rceil \\ &\leq \log_2(i) \cdot H_i(w) + 1 \end{aligned}$$

Proof sketch for the theorem - 2

$$\log_2(i) \cdot H_i(u) \leq H_2(g(u)) + c_2$$

Proof sketch for the theorem - 2

$$\log_2(i) \cdot H_i(u) \leq H_2(g(u)) + c_2$$

- $m_w \triangleq \lceil \log_i(2) \cdot |w|_2 \rceil$

Proof sketch for the theorem - 2

$$\log_2(i) \cdot H_i(u) \leq H_2(g(u)) + c_2$$

- $m_w \triangleq \lceil \log_i(2) \cdot |w|_2 \rceil$

$$\sum_{w \in PROG_{U_2}} i^{-m_w} \leq \sum_{w \in PROG_{U_2}} 2^{-|w|_2} \leq 1$$

Proof sketch for the theorem - 2

$$\log_2(i) \cdot H_i(u) \leq H_2(g(u)) + c_2$$

- $m_w \triangleq \lceil \log_i(2) \cdot |w|_2 \rceil$

$$\sum_{w \in PROG_{U_2}} i^{-m_w} \leq \sum_{w \in PROG_{U_2}} 2^{-|w|_2} \leq 1$$

- By Kraft-Chaitin Theorem, we can construct $\{t_w \in X_i^* : w \in PROG_{U_2}, |t_w|_i = m_w\}$, prefix-free and c.e.

Proof sketch for the theorem - 2

$$\log_2(i) \cdot H_i(u) \leq H_2(g(u)) + c_2$$

- $m_w \triangleq \lceil \log_i(2) \cdot |w|_2 \rceil$

$$\sum_{w \in PROG_{U_2}} i^{-m_w} \leq \sum_{w \in PROG_{U_2}} 2^{-|w|_2} \leq 1$$

- By Kraft-Chaitin Theorem, we can construct $\{t_w \in X_i^* : w \in PROG_{U_2}, |t_w|_i = m_w\}$, prefix-free and c.e.
- We define a machine D such that $D(t_w) = u$ if $U_2(w) = g(u)$ (possible because g is $1-1$).

Proof sketch for the theorem - 2

$$\log_2(i) \cdot H_i(u) \leq H_2(g(u)) + c_2$$

- $m_w \triangleq \lceil \log_i(2) \cdot |w|_2 \rceil$

$$\sum_{w \in PROG_{U_2}} i^{-m_w} \leq \sum_{w \in PROG_{U_2}} 2^{-|w|_2} \leq 1$$

- By Kraft-Chaitin Theorem, we can construct $\{t_w \in X_i^* : w \in PROG_{U_2}, |t_w|_i = m_w\}$, prefix-free and c.e.
- We define a machine D such that $D(t_w) = u$ if $U_2(w) = g(u)$ (possible because g is $1-1$).
- If $U_2(w) = g(u)$,

$$H_D(u) \leq \lceil \log_i(2) \cdot |w|_2 \rceil \leq \log_i(2) \cdot |w|_2 + 1 \leq \log_i(2) \cdot H_2(g(u)) + d$$

Complexity of well-formed formulae

Lemma

Let x be a wff. Then $H_i(x) \leq |x|_i + \mathcal{O}(1)$.

Complexity of well-formed formulae

Lemma

Let x be a wff. Then $H_i(x) \leq |x|_i + \mathcal{O}(1)$.

- We define a machine C such that $H_C(x) \leq |x|_i + 2$.

Complexity of well-formed formulae

Lemma

Let x be a wff. Then $H_i(x) \leq |x|_i + \mathcal{O}(1)$.

- We define a machine C such that $H_C(x) \leq |x|_i + 2$.
- Define C by $C(x) = x$ if x is well-formed, $C(x) =\uparrow$ else.

Complexity of well-formed formulae

Lemma

Let x be a wff. Then $H_i(x) \leq |x|_i + \mathcal{O}(1)$.

- We define a machine C such that $H_C(x) \leq |x|_i + 2$.
- Define C by $C(x) = x$ if x is well-formed, $C(x) = \uparrow$ else.
- $PROG_C$ is not prefix-free.

Complexity of well-formed formulae

Lemma

Let x be a wff. Then $H_i(x) \leq |x|_i + \mathcal{O}(1)$.

- We define a machine C such that $H_C(x) \leq |x|_i + 2$.
- Define C by $C(x) = x$ if x is well-formed, $C(x) =\uparrow$ else.
- $PROG_C$ is not prefix-free.
- Change in the definition: $C(xy) = x$ if x is well-formed, $C(z) =\uparrow$ in all other cases. Here $y = ++$ or any ill-formed formula such that xyz is ill-formed.

Complexity of well-formed formulae

Lemma

Let x be a wff. Then $H_i(x) \leq |x|_i + \mathcal{O}(1)$.

- We define a machine C such that $H_C(x) \leq |x|_i + 2$.
- Define C by $C(x) = x$ if x is well-formed, $C(x) = \uparrow$ else.
- $PROG_C$ is not prefix-free.
- Change in the definition: $C(xy) = x$ if x is well-formed, $C(z) = \uparrow$ in all other cases. Here $y = ++$ or any ill-formed formula such that xyz is ill-formed.

Can we improve the bound?

Main theorem on δ_g

- $\mathcal{F}$: finitely-specified, arithmetically sound and consistent theory, strong enough to formalize arithmetic.

Main theorem on δ_g

- $\mathcal{F}$: finitely-specified, arithmetically sound and consistent theory, strong enough to formalize arithmetic.
- $\mathcal{T}$: set of theorems that $\mathcal{F}$ proves.

Main theorem on δ_g

- $\mathcal{F}$: finitely-specified, arithmetically sound and consistent theory, strong enough to formalize arithmetic.
- $\mathcal{T}$: set of theorems that $\mathcal{F}$ proves.

Theorem

There exists a constant $N_{\mathcal{F}}$ such that for all $x \in \mathcal{T}$, $\delta_g(x) < N_{\mathcal{F}}$.

Main theorem on δ_g

- $\mathcal{F}$: finitely-specified, arithmetically sound and consistent theory, strong enough to formalize arithmetic.
- $\mathcal{T}$: set of theorems that $\mathcal{F}$ proves.

Theorem

There exists a constant $N_\mathcal{F}$ such that for all $x \in \mathcal{T}$, $\delta_g(x) < N_\mathcal{F}$.

- By the previous lemma, for every $x \in \mathcal{T}$, $\delta_i(x) \leq c$.

Main theorem on δ_g

- $\mathcal{F}$: finitely-specified, arithmetically sound and consistent theory, strong enough to formalize arithmetic.
- $\mathcal{T}$: set of theorems that $\mathcal{F}$ proves.

Theorem

There exists a constant $N_{\mathcal{F}}$ such that for all $x \in \mathcal{T}$, $\delta_g(x) < N_{\mathcal{F}}$.

- By the previous lemma, for every $x \in \mathcal{T}$, $\delta_i(x) \leq c$.
- As $|\delta_g(x) - \log_2(i) \cdot \delta_i(x)| \leq d$, $\delta_g(x) \leq d + \log_2(i) \cdot c$.

Main theorem on δ_g

- $\mathcal{F}$: finitely-specified, arithmetically sound and consistent theory, strong enough to formalize arithmetic.
- $\mathcal{T}$: set of theorems that $\mathcal{F}$ proves.

Theorem

There exists a constant $N_{\mathcal{F}}$ such that for all $x \in \mathcal{T}$, $\delta_g(x) < N_{\mathcal{F}}$.

- By the previous lemma, for every $x \in \mathcal{T}$, $\delta_i(x) \leq c$.
- As $|\delta_g(x) - \log_2(i) \cdot \delta_i(x)| \leq d$, $\delta_g(x) \leq d + \log_2(i) \cdot c$.

Proposition

$\forall N > 0,\ \lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\left\{x \in X_i^* : |x|_i = n, \delta_g(x) \leq N\right\} = 0$

Outline

Introduction

Are there other measures satisfying the heuristic principle?

Introduction

Are there other measures satisfying the heuristic principle?

- Definition of a notion of *acceptable* complexity measure

Introduction

Are there other measures satisfying the heuristic principle?

- Definition of a notion of *acceptable* complexity measure
- Properties of those measures

Introduction

Are there other measures satisfying the heuristic principle?

- Definition of a notion of *acceptable* complexity measure
- Properties of those measures
- Which measures are acceptable?

Complexity Measure Builder

Definition

Let $\hat{\rho}_i : \mathbb{N} \times \mathbb{N} \to \mathbb{Q}$ be a computable function. Then we define the *complexity measure builder* ρ by

$$\begin{array}{rcl} \rho : G & \to & [X_i^* \to \mathbb{Q}] \\ g & \mapsto & \rho_g \end{array}$$

where $\rho_g(u) = \hat{\rho}_i(H_2(g(u)), |u|_i)$.

Complexity Measure Builder

Definition

Let $\hat{\rho}_i : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{Q}$ be a computable function. Then we define the *complexity measure builder* ρ by

$$\begin{aligned} \rho : G &\rightarrow [X_i^* \rightarrow \mathbb{Q}] \\ g &\mapsto \rho_g \end{aligned}$$

where $\rho_g(u) = \hat{\rho}_i(H_2(g(u)), |u|_i)$.

- $\hat{\rho}_i$: *witness* of the builder

Complexity Measure Builder

Definition

Let $\hat{\rho}_i : \mathbb{N} \times \mathbb{N} \to \mathbb{Q}$ be a computable function. Then we define the *complexity measure builder* ρ by

$$\begin{array}{rcl} \rho : G & \to & [X_i^* \to \mathbb{Q}] \\ g & \mapsto & \rho_g \end{array}$$

where $\rho_g(u) = \hat{\rho}_i(H_2(g(u)), |u|_i)$.

- $\hat{\rho}_i$: *witness* of the builder
- ρ_g: complexity measure

Acceptable Builder

(i) If $\mathcal{F} \vdash x$, then $\rho_g(x) < N_{\mathcal{F}}$.

Acceptable Builder

(i) If $\mathcal{F} \vdash x$, then $\rho_g(x) < N_{\mathcal{F}}$.

- Heuristic principle

Acceptable Builder

(i) If $\mathcal{F} \vdash x$, then $\rho_g(x) < N_{\mathcal{F}}$.

- Heuristic principle

(ii) $\lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\left\{x \in X_i^* : |x|_i = n \text{ and } \rho_g(x) \leq N\right\} = 0$

Acceptable Builder

(i) If $\mathcal{F} \vdash x$, then $\rho_g(x) < N_{\mathcal{F}}$.

- Heuristic principle

(ii) $\lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\,\{x \in X_i^* : |x|_i = n \text{ and } \rho_g(x) \leq N\} = 0$

- Lower bound on the complexity

Acceptable Builder

(i) If $\mathcal{F} \vdash x$, then $\rho_g(x) < N_{\mathcal{F}}$.

- Heuristic principle

(ii) $\lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\,\{x \in X_i^* : |x|_i = n \text{ and } \rho_g(x) \leq N\} = 0$

- Lower bound on the complexity

(iii) $\left|\rho_g(x) - \rho_{g'}(x)\right| \leq c$

Acceptable Builder

(i) If $\mathcal{F} \vdash x$, then $\rho_g(x) < N_{\mathcal{F}}$.

- Heuristic principle

(ii) $\lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\,\{x \in X_i^* : |x|_i = n \text{ and } \rho_g(x) \leq N\} = 0$

- Lower bound on the complexity

(iii) $\left|\rho_g(x) - \rho_{g'}(x)\right| \leq c$

- Independence on the Gödel numbering

Acceptable Builder

(i) If $\mathcal{F} \vdash x$, then $\rho_g(x) < N_{\mathcal{F}}$.

- Heuristic principle

(ii) $\lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\,\{x \in X_i^* : |x|_i = n \text{ and } \rho_g(x) \leq N\} = 0$

- Lower bound on the complexity

(iii) $\left|\rho_g(x) - \rho_{g'}(x)\right| \leq c$

- Independence on the Gödel numbering

Proposition

There exists N such that for all $M \geq N$, $\{x \in X_i^* : \rho_g(x) \leq M\}$ is infinite.

Acceptable Builder

(i) If $\mathcal{F} \vdash x$, then $\rho_g(x) < N_{\mathcal{F}}$.

- Heuristic principle

(ii) $\lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\,\{x \in X_i^* : |x|_i = n \text{ and } \rho_g(x) \leq N\} = 0$

- Lower bound on the complexity

(iii) $\left|\rho_g(x) - \rho_{g'}(x)\right| \leq c$

- Independence on the Gödel numbering

Proposition

There exists N such that for all $M \geq N$, $\{x \in X_i^* : \rho_g(x) \leq M\}$ is infinite.

Proposition

The function δ_g is an acceptable complexity measure.

And what about H?

Proposition

The program-size complexity is not an acceptable complexity measure.

And what about H?

Proposition

The program-size complexity is not an acceptable complexity measure.

(i) If $\mathcal{F} \vdash x$, then $H_2(g(x)) < N_{\mathcal{F}}$.

And what about H?

Proposition

The program-size complexity is not an acceptable complexity measure.

(i) ✗ $\text{card}\,\{x \in X_i^* : H_2(g(x)) \leq N\} \leq 2^N$

And what about H?

Proposition

The program-size complexity is not an acceptable complexity measure.

(i) ✗ $\mathrm{card}\,\{x \in X_i^* : H_2(g(x)) \leq N\} \leq 2^N$

(ii) $\lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\,\{x \in X_i^* : |x|_i = n \text{ and } H_2(g(x)) \leq N\} = 0$

And what about H?

Proposition

The program-size complexity is not an acceptable complexity measure.

(i) ✗ $\mathrm{card}\,\{x \in X_i^* : H_2(g(x)) \leq N\} \leq 2^N$

(ii) ✓ $\{x \in X_i^* : |x|_i = n, H_2(g(x)) \leq N\} = \emptyset$ for large enough n

And what about H?

Proposition

The program-size complexity is not an acceptable complexity measure.

(i) ✗ $\mathrm{card}\,\{x \in X_i^* : H_2(g(x)) \leq N\} \leq 2^N$

(ii) ✓ $\{x \in X_i^* : |x|_i = n, H_2(g(x)) \leq N\} = \emptyset$ for large enough n

(iii) $|H_2(g(x)) - H_2(g'(x))| \leq c$

And what about H?

Proposition

The program-size complexity is not an acceptable complexity measure.

(i) ✗ $\mathrm{card}\{x \in X_i^* : H_2(g(x)) \leq N\} \leq 2^N$

(ii) ✓ $\{x \in X_i^* : |x|_i = n, H_2(g(x)) \leq N\} = \emptyset$ for large enough n

(iii) ✓ Already seen as a corollary.

Outline

Introduction

- Study of two complexity builders, not acceptable.

Introduction

- Study of two complexity builders, not acceptable.
- Independence of the three conditions in the definition.

First example

Definition

- $$\hat{\rho}_i^1(x,y) = \begin{cases} x/y, & \text{if } y \neq 0, \\ 0, & \text{else.} \end{cases}$$

First example

Definition

- $$\hat{\rho}_i^1(x,y) = \begin{cases} x/y, & \text{if } y \neq 0, \\ 0, & \text{else.} \end{cases}$$

- $$\rho_g^1(x) = \begin{cases} \frac{H_2(g(x))}{|x|_i}, & \text{if } x \neq \lambda, \\ 0, & \text{else.} \end{cases}$$

First example

Definition

- $$\hat{\rho}_i^1(x,y) = \begin{cases} x/y, & \text{if } y \neq 0, \\ 0, & \text{else.} \end{cases}$$
- $$\rho_g^1(x) = \begin{cases} \frac{H_2(g(x))}{|x|_i}, & \text{if } x \neq \lambda, \\ 0, & \text{else.} \end{cases}$$

Definition

$$\rho_i^1(x) = \begin{cases} \frac{H_i(x)}{|x|_i}, & \text{if } x \neq \lambda, \\ 0, & \text{else.} \end{cases}$$

Second example

Definition

- $$\hat{\rho}_i^2(x,y) = \begin{cases} x/\lceil \log_i y \rceil, & \text{if } y > 1, \\ 0, & \text{else.} \end{cases}$$

Second example

Definition

- $$\hat{\rho}_i^2(x,y) = \begin{cases} x/\left\lceil \log_i y \right\rceil, & \text{if } y > 1, \\ 0, & \text{else.} \end{cases}$$
- $$\rho_g^2(x) = \begin{cases} \dfrac{H_2(g(x))}{\left\lceil \log_i |x|_i \right\rceil}, & \text{if } |x|_i > 1, \\ 0, & \text{else.} \end{cases}$$

Second example

Definition

- $$\hat\rho_i^2(x,y) = \begin{cases} x/\lceil \log_i y \rceil, & \text{if } y > 1, \\ 0, & \text{else.} \end{cases}$$

- $$\rho_g^2(x) = \begin{cases} \frac{H_2(g(x))}{\lceil \log_i |x|_i \rceil}, & \text{if } |x|_i > 1, \\ 0, & \text{else.} \end{cases}$$

Definition

$$\rho_i^2(x) = \begin{cases} \frac{H_i(x)}{\lceil \log_i |x|_i \rceil}, & \text{if } |x|_i > 1, \\ 0, & \text{else.} \end{cases}$$

Invariance of the both measures

Lemma

$$\left|\rho_g^1(u) - \log_2(i) \cdot \rho_i^1(u)\right| \leq c_1$$

Invariance of the both measures

Lemma

$$\left|\rho_g^1(u) - \log_2(i) \cdot \rho_i^1(u)\right| \leq c_1$$

Lemma

$$\left|\rho_g^2(u) - \log_2(i) \cdot \rho_i^2(u)\right| \leq c_2$$

Invariance of the both measures

Lemma

$$\left|\rho_g^1(u) - \log_2(i)\cdot\rho_i^1(u)\right| \leq c_1$$

Lemma

$$\left|\rho_g^2(u) - \log_2(i)\cdot\rho_i^2(u)\right| \leq c_2$$

- We can use the results about δ_g.

ρ_g^1 is not acceptable

Lemma

There exists M such that for all $x \in X_i^*$, $\rho_g^1(x) \leq M$.

ρ_g^1 is not acceptable

Lemma

There exists M such that for all $x \in X_i^*$, $\rho_g^1(x) \leq M$.

- $H_i(x) \leq |x|_i + \alpha \cdot \log_i |x|_i + \beta$

ρ_g^1 is not acceptable

Lemma

There exists M such that for all $x \in X_i^*$, $\rho_g^1(x) \le M$.

- $H_i(x) \le |x|_i + \alpha \cdot \log_i |x|_i + \beta$

Proposition

(i) If $\mathcal{F} \vdash x$, then $\rho_g^1(x) < N_{\mathcal{F}}$.

ρ_g^1 is not acceptable

Lemma

There exists M such that for all $x \in X_i^*$, $\rho_g^1(x) \leq M$.

- $H_i(x) \leq |x|_i + \alpha \cdot \log_i |x|_i + \beta$

Proposition

(i) ✓ The bound is always valid.

ρ_g^1 is not acceptable

Lemma

There exists M such that for all $x \in X_i^*$, $\rho_g^1(x) \leq M$.

- $H_i(x) \leq |x|_i + \alpha \cdot \log_i |x|_i + \beta$

Proposition

(i) ✓ The bound is always valid.

(ii) $\lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\left\{x \in X_i^* : |x|_i = n \text{ and } \rho_g^1(x) \leq N\right\} = 0$

ρ_g^1 is not acceptable

Lemma

There exists M such that for all $x \in X_i^*$, $\rho_g^1(x) \leq M$.

- $H_i(x) \leq |x|_i + \alpha \cdot \log_i |x|_i + \beta$

Proposition

(i) ✓ The bound is always valid.

(ii) ✗ $\{x \in X_i^* : |x|_i = n, \rho_g^1(x) \leq N\} = X_i^n$ for N big enough.

ρ_g^1 is not acceptable

Lemma

There exists M such that for all $x \in X_i^*$, $\rho_g^1(x) \leq M$.

- $H_i(x) \leq |x|_i + \alpha \cdot \log_i |x|_i + \beta$

Proposition

(i) ✓ The bound is always valid.

(ii) ✗ $\{x \in X_i^* : |x|_i = n, \rho_g^1(x) \leq N\} = X_i^n$ for N big enough.

(iii) $\left|\rho_g^1(x) - \rho_{g'}^1(x)\right| \leq c$

ρ_g^1 is not acceptable

Lemma

There exists M such that for all $x \in X_i^*$, $\rho_g^1(x) \leq M$.

- $H_i(x) \leq |x|_i + \alpha \cdot \log_i |x|_i + \beta$

Proposition

(i) ✓ The bound is always valid.

(ii) ✗ $\{x \in X_i^* : |x|_i = n, \rho_g^1(x) \leq N\} = X_i^n$ for N big enough.

(iii) ✓ As for δ.

ρ_g^2 is not acceptable either

Proposition

(i) If $\mathcal{F} \vdash x$, then $\rho_g^2(x) < N_{\mathcal{F}}$.

ρ_g^2 is not acceptable either

Proposition

(i) ✗ See below.

ρ_g^2 is not acceptable either

Proposition

(i) ✗ See below.

(ii) $\lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\left\{x \in X_i^* : |x|_i = n \text{ and } \rho_g^2(x) \leq N\right\} = 0$

ρ_g^2 is not acceptable either

Proposition

(i) ✗ See below.

(ii) ✓ Long proof (*via* Kraft-Chaitin Theorem).

ρ^2_g is not acceptable either

Proposition

(i) ✗ See below.

(ii) ✓ Long proof (*via* Kraft-Chaitin Theorem).

(iii) $\left|\rho^2_g(x) - \rho^2_{g'}(x)\right| \leq c$

ρ_g^2 is not acceptable either

Proposition

(i) ✗ See below.

(ii) ✓ Long proof (*via* Kraft-Chaitin Theorem).

(iii) ✓ *Cf* previous slide.

ρ_g^2 is not acceptable either

Proposition

(i) ✗ See below.

(ii) ✓ Long proof (*via* Kraft-Chaitin Theorem).

(iii) ✓ *Cf* previous slide.

- If (i) holds, $\mathrm{card}\,\{x \in \mathcal{T} : |x| = n\} \leq \alpha \cdot n^{\beta \cdot N_{\mathcal{F}}}$.

ρ_g^2 is not acceptable either

Proposition

(i) ✗ See below.

(ii) ✓ Long proof (*via* Kraft-Chaitin Theorem).

(iii) ✓ *Cf* previous slide.

- If (i) holds, $\mathrm{card}\,\{x \in \mathcal{T} : |x| = n\} \leq \alpha \cdot n^{\beta \cdot N_{\mathcal{F}}}$.
- There is an exponential number of provable formulae like

$$\forall x_1 \exists x_2 \exists x_3 \ldots \forall x_k \bigwedge_{l=1}^{k} (x_l = x_l).$$

Intuitive Results and Independence

- ρ^1 is "too small" and ρ^2 is "too big".

Intuitive Results and Independence

- ρ^1 is “too small” and ρ^2 is “too big”.

(i) Upper bound: the complexity of the theorems has to be bounded.

Intuitive Results and Independence

- ρ^1 is "too small" and ρ^2 is "too big".

(i) Upper bound: the complexity of the theorems has to be bounded.

(ii) Lower bound: avoid trivial measures.

Intuitive Results and Independence

- ρ^1 is "too small" and ρ^2 is "too big".

(i) Upper bound: the complexity of the theorems has to be bounded.
(ii) Lower bound: avoid trivial measures.
(iii) Independence from the chosen language.

Intuitive Results and Independence

- ρ^1 is "too small" and ρ^2 is "too big".

(i) Upper bound: the complexity of the theorems has to be bounded.
(ii) Lower bound: avoid trivial measures.
(iii) Independence from the chosen language.

Theorem

The three conditions are independent from each other.

Proof of the independence (1)

If $H_2(g(x)) = H_2(g'(x))$ hold for all but finitely many $x \in X_i^*$.

Proof of the independence (1)

If $H_2(g(x)) = H_2(g'(x))$ hold for all but finitely many $x \in X_i^*$.

- $\rho_g(x) = \hat{\rho}_i(H_2(g(x)), |x|_i) = \hat{\rho}_i(H_2(g'(x)), |x|_i) = \rho_{g'}(x)$

Proof of the independence (1)

If $H_2(g(x)) = H_2(g'(x))$ hold for all but finitely many $x \in X_i^*$.

- $\rho_g(x) = \hat{\rho}_i(H_2(g(x)), |x|_i) = \hat{\rho}_i(H_2(g'(x)), |x|_i) = \rho_{g'}(x)$
- $\max\left\{\left|\rho_g(x) - \rho_{g'}(x)\right| : x \in X_i^*\right\} = c < \infty$

Proof of the independence (1)

If $H_2(g(x)) = H_2(g'(x))$ hold for all but finitely many $x \in X_i^*$.

- $\rho_g(x) = \hat{\rho}_i(H_2(g(x)), |x|_i) = \hat{\rho}_i(H_2(g'(x)), |x|_i) = \rho_{g'}(x)$
- $\max\left\{\left|\rho_g(x) - \rho_{g'}(x)\right| : x \in X_i^*\right\} = c < \infty$
- For all $x \in X_i^*$, $\left|\rho_g(x) - \rho_{g'}(x)\right| \leq c$

Proof of the independence (1)

If $H_2(g(x)) = H_2(g'(x))$ hold for all but finitely many $x \in X_i^*$.

- $\rho_g(x) = \hat{\rho}_i(H_2(g(x)), |x|_i) = \hat{\rho}_i(H_2(g'(x)), |x|_i) = \rho_{g'}(x)$
- $\max\left\{\left|\rho_g(x) - \rho_{g'}(x)\right| : x \in X_i^*\right\} = c < \infty$
- For all $x \in X_i^*$, $\left|\rho_g(x) - \rho_{g'}(x)\right| \leq c$
- ρ satisfy (iii).

Proof of the independence (2)

If $H_2(g(x)) \neq H_2(g'(x))$ hold for infinitely many $x \in X_i^*$ (*).

Proof of the independence (2)

If $H_2(g(x)) \neq H_2(g'(x))$ hold for infinitely many $x \in X_i^*$ (*).

- Define ρ_g by $x \mapsto \delta_g(x)^2$.

Proof of the independence (2)

If $H_2(g(x)) \neq H_2(g'(x))$ hold for infinitely many $x \in X_i^*$ (*).

- Define ρ_g by $x \mapsto \delta_g(x)^2$.

(i) If $\mathcal{F} \vdash x$, then $\rho_g(x) < N_{\mathcal{F}}$.

Proof of the independence (2)

If $H_2(g(x)) \neq H_2(g'(x))$ hold for infinitely many $x \in X_i^*$ (*).

- Define ρ_g by $x \mapsto \delta_g(x)^2$.

(i) ✓ $\delta_g(x) < N_{\mathcal{F}} \implies \rho_g(x) < N_{\mathcal{F}}^2$.

Proof of the independence (2)

If $H_2(g(x)) \neq H_2(g'(x))$ hold for infinitely many $x \in X_i^*$ (*).

- Define ρ_g by $x \mapsto \delta_g(x)^2$.

(i) ✓ $\delta_g(x) < N_{\mathcal{F}} \implies \rho_g(x) < N_{\mathcal{F}}^2$.

(ii) $\lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\left\{x \in X_i^* : |x|_i = n \text{ and } \rho_g(x) \leq N\right\} = 0$

Proof of the independence (2)

If $H_2(g(x)) \neq H_2(g'(x))$ hold for infinitely many $x \in X_i^*$ (*).

- Define ρ_g by $x \mapsto \delta_g(x)^2$.

(i) ✓ $\delta_g(x) < N_{\mathcal{F}} \implies \rho_g(x) < N_{\mathcal{F}}^2$.

(ii) ✓ $\leq \lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\left\{x \in X_i^* : |x|_i = n \text{ and } \delta_g(x) \leq \sqrt{N}\right\} = 0$

Proof of the independence (2)

If $H_2(g(x)) \neq H_2(g'(x))$ hold for infinitely many $x \in X_i^*$ (*).

- Define ρ_g by $x \mapsto \delta_g(x)^2$.

(i) ✓ $\delta_g(x) < N_{\mathcal{F}} \implies \rho_g(x) < N_{\mathcal{F}}^2$.

(ii) ✓ $\leq \lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\left\{x \in X_i^* : |x|_i = n \text{ and } \delta_g(x) \leq \sqrt{N}\right\} = 0$

(iii) $\left|\rho_g(x) - \rho_{g'}(x)\right| \leq c$

Proof of the independence (2)

If $H_2(g(x)) \neq H_2(g'(x))$ hold for infinitely many $x \in X_i^*$ (*).

- Define ρ_g by $x \mapsto \delta_g(x)^2$.

(i) ✓ $\delta_g(x) < N_{\mathcal{F}} \implies \rho_g(x) < N_{\mathcal{F}}^2$.

(ii) ✓ $\leq \lim_{n\to\infty} i^{-n} \cdot \mathrm{card}\left\{x \in X_i^* : |x|_i = n \text{ and } \delta_g(x) \leq \sqrt{N}\right\} = 0$

(iii) ✗ Else, (*) is false.

Proof of the independence (3)

- ρ^1 satisfies (i) and (iii) but not (ii).

Proof of the independence (3)

- ρ^1 satisfies (i) and (iii) but not (ii).
- ρ^2 satisfies (ii) and (iii) but not (i).

Proof of the independence (3)

- ρ^1 satisfies (i) and (iii) but not (ii).
- ρ^2 satisfies (ii) and (iii) but not (i).
- Either (iii) is always satisfied, or δ^2 satisfies (i) and (ii) but not (iii).

Outline

Introduction

Can we find other acceptable measures of complexity?

Introduction

Can we find other acceptable measures of complexity?

- We study two kinds of measures, defined by two kinds of witnesses:

Introduction

Can we find other acceptable measures of complexity?

- We study two kinds of measures, defined by two kinds of witnesses:
  - linear in both variables,

Introduction

Can we find other acceptable measures of complexity?

- We study two kinds of measures, defined by two kinds of witnesses:
  - linear in both variables,
  - multiplicative variation of the program-size complexity.

Introduction

Can we find other acceptable measures of complexity?

- We study two kinds of measures, defined by two kinds of witnesses:
  - linear in both variables,
  - multiplicative variation of the program-size complexity.

Proposition

Suppose that ρ_g is acceptable. Then so is $\alpha \cdot \rho_g + \beta$, $\alpha, \beta \in \mathbb{Q}$, $\alpha > 0$.

Linear variations of the program-size complexity

Proposition

Let $\hat{\rho}_i : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{Q}$ be a computable function, linear in both variables. If it defines an acceptable complexity measure, then

$$\hat{\rho}_i(x, y) = a \cdot (x - \varepsilon \cdot \lceil \log_2(i) \cdot y \rceil) + b,$$

where $1/2 \leq \varepsilon \leq 1$.

Linear variations of the program-size complexity

Proposition

Let $\hat{\rho}_i : \mathbb{N} \times \mathbb{N} \to \mathbb{Q}$ be a computable function, linear in both variables. If it defines an acceptable complexity measure, then

$$\hat{\rho}_i(x, y) = \qquad x - \varepsilon \cdot \lceil \log_2(i) \cdot y \rceil \qquad ,$$

where $1/2 \leq \varepsilon \leq 1$.

Linear variations of the program-size complexity

Proposition

Let $\hat{\rho}_i : \mathbb{N} \times \mathbb{N} \to \mathbb{Q}$ be a computable function, linear in both variables. If it defines an acceptable complexity measure, then

$$\hat{\rho}_i(x, y) = \qquad x - \varepsilon \cdot \lceil \log_2(i) \cdot y \rceil \qquad ,$$

where $1/2 \leq \varepsilon \leq 1$.

- If $\varepsilon > 1$, then (ii) is not verified.

Linear variations of the program-size complexity

Proposition

Let $\hat{\rho}_i : \mathbb{N} \times \mathbb{N} \to \mathbb{Q}$ be a computable function, linear in both variables. If it defines an acceptable complexity measure, then

$$\hat{\rho}_i(x, y) = \qquad x - \varepsilon \cdot \lceil \log_2(i) \cdot y \rceil \qquad ,$$

where $1/2 \leq \varepsilon \leq 1$.

- If $\varepsilon > 1$, then (ii) is not verified.
- If $\varepsilon < 1/2$, then (i) is not verified.

Linear variations of the program-size complexity

Proposition

Let $\hat{\rho}_i : \mathbb{N} \times \mathbb{N} \to \mathbb{Q}$ be a computable function, linear in both variables. If it defines an acceptable complexity measure, then

$$\hat{\rho}_i(x, y) = \quad x - \varepsilon \cdot \lceil \log_2(i) \cdot y \rceil \quad ,$$

where $1/2 \leq \varepsilon \leq 1$.

- If $\varepsilon > 1$, then (ii) is not verified.
- If $\varepsilon < 1/2$, then (i) is not verified.
- Between $1/2$ and 1, your ideas are welcome!

Multiplicative variations of the program-size complexity

Proposition

Let $\rho_g(x) = H_2(g(x))/f(|x|_i)$ where f is computable. Then ρ_g is not acceptable.

Multiplicative variations of the program-size complexity

Proposition

Let $\rho_g(x) = H_2(g(x))/f(|x|_i)$ where f is computable. Then ρ_g is not acceptable.

- We suppose that ρ_g satisfies (i), and prove that it does not satisfy (ii).

Multiplicative variations of the program-size complexity

Proposition

Let $\rho_g(x) = H_2(g(x))/f(|x|_i)$ where f is computable. Then ρ_g is not acceptable.

- We suppose that ρ_g satisfies (i), and prove that it does not satisfy (ii).
- $2^{c \cdot n} \leq \operatorname{card}\{x \in \mathcal{T} : |x|_i = n\}$

Multiplicative variations of the program-size complexity

Proposition

Let $\rho_g(x) = H_2(g(x))/f(|x|_i)$ where f is computable. Then ρ_g is not acceptable.

- We suppose that ρ_g satisfies (i), and prove that it does not satisfy (ii).
- $2^{c \cdot n} \leq \mathrm{card}\left\{x \in \mathcal{T} : |x|_i = n\right\} \leq 2^{N_{\mathcal{F}} \cdot f(n)}$

Multiplicative variations of the program-size complexity

Proposition

Let $\rho_g(x) = H_2(g(x))/f(|x|_i)$ where f is computable. Then ρ_g is not acceptable.

- We suppose that ρ_g satisfies (i), and prove that it does not satisfy (ii).
- $2^{c \cdot n} \leq \operatorname{card}\{x \in \mathcal{T} : |x|_i = n\} \leq 2^{N_{\mathcal{F}} \cdot f(n)}$

Multiplicative variations of the program-size complexity

Proposition

Let $\rho_g(x) = H_2(g(x))/f(|x|_i)$ where f is computable. Then ρ_g is not acceptable.

- We suppose that ρ_g satisfies (i), and prove that it does not satisfy (ii).
- $2^{c \cdot n} \leq \operatorname{card}\{x \in \mathcal{T} : |x|_i = n\} \leq 2^{N_{\mathcal{F}} \cdot f(n)}$
- $c \cdot n \leq N_{\mathcal{F}} \cdot f(n)$

Multiplicative variations of the program-size complexity

Proposition

Let $\rho_g(x) = H_2(g(x))/f(|x|_i)$ where f is computable. Then ρ_g is not acceptable.

- We suppose that ρ_g satisfies (i), and prove that it does not satisfy (ii).
- $2^{c\cdot n} \leq \mathrm{card}\,\{x \in \mathcal{T} : |x|_i = n\} \leq 2^{N_{\mathcal{F}}\cdot f(n)}$
- $c \cdot n \leq N_{\mathcal{F}} \cdot f(n)$
- $\{x \in X_i^* : |x|_i = n \text{ and } \rho_g(x) \leq N_{\mathcal{F}}\} = X_i^n$

Multiplicative variations of the program-size complexity

Proposition

Let $\rho_g(x) = H_2(g(x))/f(|x|_i)$ where f is computable. Then ρ_g is not acceptable.

- We suppose that ρ_g satisfies (i), and prove that it does not satisfy (ii).
- $2^{c \cdot n} \leq \mathrm{card}\,\{x \in \mathcal{T} : |x|_i = n\} \leq 2^{N_{\mathcal{F}} \cdot f(n)}$
- $c \cdot n \leq N_{\mathcal{F}} \cdot f(n)$
- $\{x \in X_i^* : |x|_i = n \text{ and } \rho_g(x) \leq N_{\mathcal{F}}\} = X_i^n$
- (ii) is not verified.

Summary of the work

- Studying the results about δ_g

Summary of the work

- Studying the results about δ_g
  - Some corrections

Summary of the work

- Studying the results about δ_g
  - Some corrections
  - Key elements in the proofs

Summary of the work

- Studying the results about δ_g
  - Some corrections
  - Key elements in the proofs
- Proposition of a general definition of *acceptable complexity measure of theorems*

Summary of the work

- Studying the results about δ_g
  - Some corrections
  - Key elements in the proofs
- Proposition of a general definition of *acceptable complexity measure of theorems*
- Studying those acceptable measures to find other ones (in progress)


The End
Thank you very much!