

The intensional content of Rice's Theorem

Andrea Asperti

Department of Computer Science, University of Bologna
Mura Anteo Zamboni 7, 40127, Bologna, ITALY
asperti@cs.unibo.it

Content

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity
- 3 Similarity and Complexity Cliques
- 4 Rice-Shapiro's Theorem
 - Monotonicity
 - compactness
- 5 Corollaries
- 6 Kleene's Fixed Point Theorem
- 7 Conclusions
 - Main results
 - Future works and applications

Content

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity
- 3 Similarity and Complexity Cliques
- 4 Rice-Shapiro's Theorem
 - Monotonicity
 - compactness
- 5 Corollaries
- 6 Kleene's Fixed Point Theorem
- 7 Conclusions
 - Main results
 - Future works and applications

Content

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity
- 3 Similarity and Complexity Cliques
- 4 Rice-Shapiro's Theorem
 - Monotonicity
 - compactness
- 5 Corollaries
- 6 Kleene's Fixed Point Theorem
- 7 Conclusions
 - Main results
 - Future works and applications

Content

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity
- 3 Similarity and Complexity Cliques
- 4 Rice-Shapiro's Theorem
 - Monotonicity
 - compactness
- 5 Corollaries
- 6 Kleene's Fixed Point Theorem
- 7 Conclusions
 - Main results
 - Future works and applications

Content

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity
- 3 Similarity and Complexity Cliques
- 4 Rice-Shapiro's Theorem
 - Monotonicity
 - compactness
- 5 Corollaries
- 6 Kleene's Fixed Point Theorem
- 7 Conclusions
 - Main results
 - Future works and applications

Content

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity
- 3 Similarity and Complexity Cliques
- 4 Rice-Shapiro's Theorem
 - Monotonicity
 - compactness
- 5 Corollaries
- 6 Kleene's Fixed Point Theorem
- 7 Conclusions
 - Main results
 - Future works and applications

Content

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity
- 3 Similarity and Complexity Cliques
- 4 Rice-Shapiro's Theorem
 - Monotonicity
 - compactness
- 5 Corollaries
- 6 Kleene's Fixed Point Theorem
- 7 Conclusions
 - Main results
 - Future works and applications

Outline

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity
- 3 Similarity and Complexity Cliques
- 4 Rice-Shapiro's Theorem
 - Monotonicity
 - compactness
- 5 Corollaries
- 6 Kleene's Fixed Point Theorem
- 7 Conclusions
 - Main results
 - Future works and applications

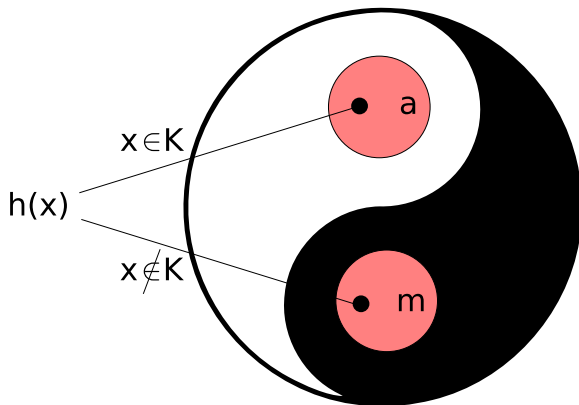
Rice's Theorem

Rice 1953

An estensional property of programs is decidable if and only if it is trivial.

estensional = closed w.r.t. estensional equivalence

Rice's Yin Yang

 $\forall x, \phi_m(x) \uparrow$ 

the function h

Let $K = \text{dom}(\phi_k)$, and define

$$\phi_{h(x)}(y) = \phi_k(x); \phi_a(y)$$

Clearly, if ϕ_m is the everywhere divergent function,

$$\phi_{h(x)} \approx \begin{cases} \phi_a & \text{if } x \in K \\ \phi_m & \text{if } x \notin K \end{cases}$$

Does h preserve any other property, in addition to extensional equivalence? **Yes, complexity!**

Next: investigates the complexity assumptions needed to formalize such result.

the function h

Let $K = \text{dom}(\phi_k)$, and define

$$\phi_{h(x)}(y) = \phi_k(x); \phi_a(y)$$

Clearly, if ϕ_m is the everywhere divergent function,

$$\phi_{h(x)} \approx \begin{cases} \phi_a & \text{if } x \in K \\ \phi_m & \text{if } x \notin K \end{cases}$$

Does h preserve any other property, in addition to extensional equivalence? **Yes, complexity!**

Next: investigates the complexity assumptions needed to formalize such result.

the function h

Let $K = \text{dom}(\phi_k)$, and define

$$\phi_{h(x)}(y) = \phi_k(x); \phi_a(y)$$

Clearly, if ϕ_m is the everywhere divergent function,

$$\phi_{h(x)} \approx \begin{cases} \phi_a & \text{if } x \in K \\ \phi_m & \text{if } x \notin K \end{cases}$$

Does h preserve any other property, in addition to extensional equivalence? **Yes, complexity!**

Next: investigates the complexity assumptions needed to formalize such result.

the function h

Let $K = \text{dom}(\phi_k)$, and define

$$\phi_{h(x)}(y) = \phi_k(x); \phi_a(y)$$

Clearly, if ϕ_m is the everywhere divergent function,

$$\phi_{h(x)} \approx \begin{cases} \phi_a & \text{if } x \in K \\ \phi_m & \text{if } x \notin K \end{cases}$$

Does h preserve any other property, in addition to extensional equivalence? **Yes, complexity!**

Next: investigates the complexity assumptions needed to formalize such result.

the function h

Let $K = \text{dom}(\phi_k)$, and define

$$\phi_{h(x)}(y) = \phi_k(x); \phi_a(y)$$

Clearly, if ϕ_m is the everywhere divergent function,

$$\phi_{h(x)} \approx \begin{cases} \phi_a & \text{if } x \in K \\ \phi_m & \text{if } x \notin K \end{cases}$$

Does h preserve any other property, in addition to extensional equivalence? **Yes, complexity!**

Next: investigates the complexity assumptions needed to formalize such result.

Outline

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity**
- 3 Similarity and Complexity Cliques
- 4 Rice-Shapiro's Theorem
 - Monotonicity
 - compactness
- 5 Corollaries
- 6 Kleene's Fixed Point Theorem
- 7 Conclusions
 - Main results
 - Future works and applications

Blum's Abstract Complexity

A pair $\langle \phi, \Phi \rangle$ is a *computational complexity measure* if ϕ is a principal effective enumeration of partial recursive functions and Φ satisfies Blum's axioms (Blum 1967):

- (a) $\phi_i(\vec{n}) \downarrow \leftrightarrow \Phi_i(\vec{n}) \downarrow$
- (b) the predicate $\Phi_i(\vec{n}) = m$ is decidable

Outline

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity
- 3 Similarity and Complexity Cliques**
- 4 Rice-Shapiro's Theorem
 - Monotonicity
 - compactness
- 5 Corollaries
- 6 Kleene's Fixed Point Theorem
- 7 Conclusions
 - Main results
 - Future works and applications

Big O notation

Big O remind:

- 1 $f \in O(g)$ if and only if there exist n and c such that for any $m \geq n$, if $g(m) \downarrow$ then $f(m) \leq cg(m)$;
- 2 $f \in \Theta(g)$ if and only if $f \in O(g)$ and $g \in O(f)$.

Similarity and Complexity Clique

Definition Two programs i and j are *similar* (write $i \approx j$) if and only if

$$\phi_j \cong \phi_i \wedge \Phi_j \in \Theta(\Phi_i)$$

Similarity is an equivalence relation.

Definition Let $\langle \phi, \Phi \rangle$ be an abstract complexity measure. A set P of natural numbers is a *Complexity Clique*, if and only if for all i and j

$$i \in P \wedge j \approx i \rightarrow j \in P$$

Examples of Complexity Cliques

- 1 \emptyset and ω ;
- 2 for any index i , $[i]_{\approx}$;
- 3 for any index i , $\{j \mid \Phi_j \in O(\Phi_i)\}$.
- 4 all programs with polynomial (exponential, ...) complexity.

Warning: not every Complexity Class is a Complexity Cliques.

Complexity Cliques are closed w.r.t to union, intersection, and complementation.

Complexity Assumptions: s-m-n

Definition A pair $\langle \phi, \Phi \rangle$ has the *s-m-n property* if for all m and n there exists a recursive function s_m^n such that, for any i and all x_1, \dots, x_m

$$(a) \phi_{s_m^n(i, x_1, \dots, x_m)} \cong \lambda y_1, \dots, y_n. \phi_i(x_1, \dots, x_m, y_1, \dots, y_n)$$

$$(b) \Phi_{s_m^n(i, x_1, \dots, x_m)} \in \Theta(\lambda y_1, \dots, y_n. \Phi_i(x_1, \dots, x_m, y_1, \dots, y_n))$$

Complexity Assumptions: s-m-n

Definition A pair $\langle \phi, \Phi \rangle$ has the *s-m-n property* if for all m and n there exists a recursive function s_m^n such that, for any i and all x_1, \dots, x_m

$$(a) \phi_{s_m^n(i, x_1, \dots, x_m)} \cong \lambda y_1, \dots, y_n. \phi_i(x_1, \dots, x_m, y_1, \dots, y_n)$$

$$(b) \Phi_{s_m^n(i, x_1, \dots, x_m)} \in \Theta(\lambda y_1, \dots, y_n. \Phi_i(x_1, \dots, x_m, y_1, \dots, y_n))$$

Complexity Assumptions: composition

Definition A pair $\langle \phi, \Phi \rangle$ has the composition property if there exists a total computable function h such that

$$(a) \phi_{h(i,j)} \cong \phi_i \circ \phi_j$$

$$(b) \Phi_{h(i,j)} \in \Theta(\max\{\Phi_i \circ \phi_j, \Phi_j\})$$

we only ask that **there exists** a way of composing functions with the above complexity.

Complexity Assumptions: composition

Definition A pair $\langle \phi, \Phi \rangle$ has the composition property if there exists a total computable function h such that

$$(a) \phi_{h(i,j)} \cong \phi_i \circ \phi_j$$

$$(b) \Phi_{h(i,j)} \in \Theta(\max\{\Phi_i \circ \phi_j, \Phi_j\})$$

we only ask that **there exists** a way of composing functions with the above complexity.

Complexity Assumptions: composition

Definition A pair $\langle \phi, \Phi \rangle$ has the composition property if there exists a total computable function h such that

$$(a) \phi_{h(i,j)} \cong \phi_i \circ \phi_j$$

$$(b) \Phi_{h(i,j)} \in \Theta(\max\{\Phi_i \circ \phi_j, \Phi_j\})$$

we only ask that **there exists** a way of composing functions with the above complexity.

Generalized Rice's Theorem

Asperti 2008

Under the s-m-n and the composition assumptions, a Complexity Clique P is recursive if and only if it is trivial, i.e. $P = \emptyset \vee P = \omega$.

Outline

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity
- 3 Similarity and Complexity Cliques
- 4 Rice-Shapiro's Theorem**
 - Monotonicity
 - compactness
- 5 Corollaries
- 6 Kleene's Fixed Point Theorem
- 7 Conclusions
 - Main results
 - Future works and applications

Rice-Shapiro's Theorem

Shapiro 1956

If P is a r.e extensional property of programs then

$$i \in P \Leftrightarrow \exists u \in P \phi_u \text{ is finite} \wedge \phi_u \leq \phi_i$$

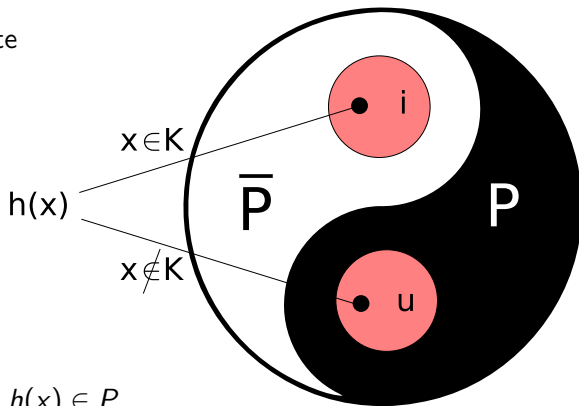
\Leftarrow monotonicity

\Rightarrow compactness

Rice-Shapiro's Yin Yang (monotonicity)

$$\phi_u \leq \phi_i$$

ϕ_u is finite



$$x \notin K \Leftrightarrow h(x) \in P$$

the function h

$$\phi_i(x) \upharpoonright \phi_j(x) = \begin{cases} \phi_i(x) & \text{if } \Phi_i(x) \leq \Phi_j(x) \\ \phi_j(x) & \text{otherwise} \end{cases}$$

Let $K = \text{dom}(\phi_k)$. Then

$$\phi_{h(x)}(y) = \phi_u(y) \upharpoonright \phi_k(x); \phi_i(y)$$

Clearly,

$$\phi_{h(x)} \approx \begin{cases} \phi_u & \text{if } x \notin K \\ \phi_i & \text{if } x \in K \end{cases}$$

the function h

$$\phi_i(x) \upharpoonright \phi_j(x) = \begin{cases} \phi_i(x) & \text{if } \Phi_i(x) \leq \Phi_j(x) \\ \phi_j(x) & \text{otherwise} \end{cases}$$

Let $K = \text{dom}(\phi_k)$. Then

$$\phi_{h(x)}(y) = \phi_u(y) \upharpoonright \phi_k(x); \phi_i(y)$$

Clearly,

$$\phi_{h(x)} \approx \begin{cases} \phi_u & \text{if } x \notin K \\ \phi_i & \text{if } x \in K \end{cases}$$

the function h

$$\phi_i(x) \upharpoonright \phi_j(x) = \begin{cases} \phi_i(x) & \text{if } \Phi_i(x) \leq \Phi_j(x) \\ \phi_j(x) & \text{otherwise} \end{cases}$$

Let $K = \text{dom}(\phi_k)$. Then

$$\phi_{h(x)}(y) = \phi_u(y) \upharpoonright \phi_k(x); \phi_i(y)$$

Clearly,

$$\phi_{h(x)} \approx \begin{cases} \phi_u & \text{if } x \notin K \\ \phi_i & \text{if } x \in K \end{cases}$$

parallel computation property

Definition (Landweber and Robertson, 1972)

A pair $\langle \phi, \Phi \rangle$ has the *parallel computation* property if there exists a total computable function h such that

$$(a) \quad \phi_{h(i,j)}(x) = \begin{cases} \phi_i(x) & \text{if } \Phi_i(x) \leq \Phi_j(x) \\ \phi_j(x) & \text{otherwise} \end{cases}$$
$$(b) \quad \Phi_{h(i,j)} \in \Theta(\lambda x. \min\{\Phi_i(x), \Phi_j(x)\})$$

Assuming the parallel computation property we may generalize monotonicity to r.e. Complexity Cliques.

parallel computation property

Definition (Landweber and Robertson, 1972)

A pair $\langle \phi, \Phi \rangle$ has the *parallel computation* property if there exists a total computable function h such that

$$(a) \quad \phi_{h(i,j)}(x) = \begin{cases} \phi_i(x) & \text{if } \Phi_i(x) \leq \Phi_j(x) \\ \phi_j(x) & \text{otherwise} \end{cases}$$
$$(b) \quad \Phi_{h(i,j)} \in \Theta(\lambda x. \min\{\Phi_i(x), \Phi_j(x)\})$$

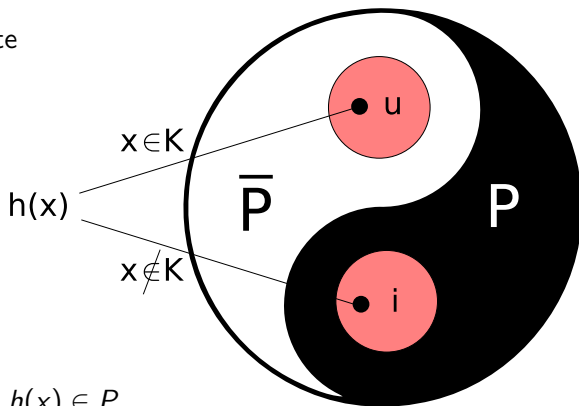
Assuming the parallel computation property we may generalize monotonicity to r.e. Complexity Cliques.

Rice-Shapiro's Yin Yang (compactness)

For some u

$$\phi_u \leq \phi_i$$

ϕ_u is finite



$$x \notin K \Leftrightarrow h(x) \in P$$

Let $K = \text{dom}(\phi_k)$.

$$\phi_{h(x)}(y) = \begin{array}{l} \text{match } FST(\phi_k(x)) | SND(\phi_i(y)) \text{ with} \\ | FST_ \Rightarrow \uparrow \\ | SND(a) \Rightarrow a \end{array}$$

If $\Phi_i \notin O(1)$, and $\phi_k(x) \downarrow$, $\Phi_i(y) > \Phi_k(x)$ almost everywhere.
Hence

$$\phi_{h(x)} \approx \begin{cases} \phi_i & \text{if } x \notin K \\ \text{some finite subfunction of } \phi_i & \text{if } x \in K \end{cases}$$

generalized parallel computation

Definition A pair $\langle \phi, \Phi \rangle$ has the *generalized parallel computation* property if there exists a total computable function p such that for all i, i', j, j'

$$(a) \quad \phi_{p(i,i',j,j')}(x) = \begin{cases} \phi_{i'}(\phi_i(x)) & \text{if } \Phi_i(x) \leq \Phi_j(x) \\ \phi_{j'}(\phi_j(x)) & \text{otherwise} \end{cases}$$

$$(b) \quad \Phi_{p(i,i',j,j')} \in \Theta \left(\lambda x. \begin{cases} \Phi_{h(i',i)}(x) & \text{if } \Phi_i(x) \leq \Phi_j(x) \\ \Phi_{h(j',j)}(x) & \text{otherwise} \end{cases} \right)$$

Assuming the parallel computation property we may prove that for any r.e. Complexity Cliques P , if $i \in P$ and $\Phi_i \notin O(1)$ then there exists $u \in P$ such that ϕ_u is finite and $\phi_u < \phi_i$.

generalized parallel computation

Definition A pair $\langle \phi, \Phi \rangle$ has the *generalized parallel computation* property if there exists a total computable function p such that for all i, i', j, j'

$$(a) \quad \phi_{p(i,i',j,j')}(x) = \begin{cases} \phi_{i'}(\phi_i(x)) & \text{if } \Phi_i(x) \leq \Phi_j(x) \\ \phi_{j'}(\phi_j(x)) & \text{otherwise} \end{cases}$$

$$(b) \quad \Phi_{p(i,i',j,j')} \in \Theta \left(\lambda x. \begin{cases} \Phi_{h(i',i)}(x) & \text{if } \Phi_i(x) \leq \Phi_j(x) \\ \Phi_{h(j',j)}(x) & \text{otherwise} \end{cases} \right)$$

Assuming the parallel computation property we may prove that for any r.e. Complexity Cliques P , if $i \in P$ and $\Phi_i \notin O(1)$ then there exists $u \in P$ such that ϕ_u is finite and $\phi_u < \phi_i$.

Outline

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity
- 3 Similarity and Complexity Cliques
- 4 Rice-Shapiro's Theorem
 - Monotonicity
 - compactness
- 5 Corollaries**
- 6 Kleene's Fixed Point Theorem
- 7 Conclusions
 - Main results
 - Future works and applications

Corollaries

Corollary Let P be a r.e. Complexity Clique. If $i \in P$ and $\Phi_i \notin O(1)$ then for every j such that $\phi_j \cong \phi_i$ we have $j \in P$.

Proof By compactness, there exists a finite sub-function $\phi_r \leq \phi_i$ such that $r \in P$, and by monotonicity, any j such that $\phi_r \leq \phi_j$, independently from its complexity Φ_j , must belong to P .

Corollary No Complexity Clique of total functions and containing (indices of) programs with non constant complexity can be r.e.

Proof By compactness.

Corollaries

Corollary Let P be a r.e. Complexity Clique. If $i \in P$ and $\Phi_i \notin O(1)$ then for every j such that $\phi_j \cong \phi_i$ we have $j \in P$.

Proof By compactness, there exists a finite sub-function $\phi_r \leq \phi_i$ such that $r \in P$, and by monotonicity, any j such that $\phi_r \leq \phi_j$, *independently from its complexity* Φ_j , must belong to P .

Corollary No Complexity Clique of total functions and containing (indices of) programs with non constant complexity can be r.e.

Proof By compactness.

Corollaries

Corollary Let P be a r.e. Complexity Clique. If $i \in P$ and $\Phi_i \notin O(1)$ then for every j such that $\phi_j \cong \phi_i$ we have $j \in P$.

Proof By compactness, there exists a finite sub-function $\phi_r \leq \phi_i$ such that $r \in P$, and by monotonicity, any j such that $\phi_r \leq \phi_j$, independently from its complexity Φ_j , must belong to P .

Corollary No Complexity Clique of total functions and containing (indices of) programs with non constant complexity can be r.e.

Proof By compactness.

Corollaries

Corollary Let P be a r.e. Complexity Clique. If $i \in P$ and $\Phi_i \notin O(1)$ then for every j such that $\phi_j \cong \phi_i$ we have $j \in P$.

Proof By compactness, there exists a finite sub-function $\phi_r \leq \phi_i$ such that $r \in P$, and by monotonicity, any j such that $\phi_r \leq \phi_j$, independently from its complexity Φ_j , must belong to P .

Corollary No Complexity Clique of total functions and containing (indices of) programs with non constant complexity can be r.e.

Proof By compactness.

Outline

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity
- 3 Similarity and Complexity Cliques
- 4 Rice-Shapiro's Theorem
 - Monotonicity
 - compactness
- 5 Corollaries
- 6 Kleene's Fixed Point Theorem**
- 7 Conclusions
 - Main results
 - Future works and applications

Kleene's Fixed Point Theorem

Kleene 1952

For any total recursive function f , there exists a such that

$$\phi_a \cong \phi_{f(a)}$$

Can we always choose a such that $\phi_a \in \Theta(\phi_{f(a)})$?

Kleene's Fixed Point Theorem

Kleene 1952

For any total recursive function f , there exists a such that

$$\phi_a \cong \phi_{f(a)}$$

Can we always choose a such that $\Phi_a \in \Theta(\Phi_{f(a)})$?

Complexity Theoretic version of Kleene's Theorem

Theorem Let $\langle \phi, \Phi \rangle$ be an abstract complexity measure with the s-m-n property, and let u be an index for the universal function. Then for any total recursive function ϕ_i there exists an index m such that, for any x ,

- (1) $\phi_m \cong \phi_{\phi_i(m)}$
- (2) $\Phi_m \in \Theta(\lambda y. \Phi_u(\phi_i(m), y))$

But what about the complexity of the interpreter u ?

Complexity Theoretic version of Kleene's Theorem

Theorem Let $\langle \phi, \Phi \rangle$ be an abstract complexity measure with the s-m-n property, and let u be an index for the universal function. Then for any total recursive function ϕ_i there exists an index m such that, for any x ,

- (1) $\phi_m \cong \phi_{\phi_i(m)}$
- (2) $\Phi_m \in \Theta(\lambda y. \Phi_u(\phi_i(m), y))$

But what about the complexity of the interpreter u ?

Fair Interpreters

Definition We say that a universal function ϕ_u is *fair* if for any x

$$\lambda y. \Phi_u(x, y) \in \Theta(\Phi_x)$$

Corollary Let $\langle \phi, \Phi \rangle$ be an abstract complexity measure with the s-m-n property. If it admits a fair universal function u then for any total recursive function ϕ_i there exists an index m such that, for any x ,

- (1) $\phi_m \cong \phi_{\phi_i(m)}$
- (2) $\Phi_m \in \Theta(\Phi_{\phi_i(m)})$

Outline

- 1 Rice's Theorem
- 2 Blum's Abstract Complexity
- 3 Similarity and Complexity Cliques
- 4 Rice-Shapiro's Theorem
 - Monotonicity
 - compactness
- 5 Corollaries
- 6 Kleene's Fixed Point Theorem
- 7 Conclusions**
 - Main results
 - Future works and applications

- Complexity Cliques generalize estensional sets
- Complexity Cliques in Δ_1^0 are trivial
- Complexity Cliques in Σ_1^0 and Π_1^0 have trivial complexities.

- Complexity Cliques vs. Complexity Classes
- Complexity-theoretic revisitation of Recursion Theory
- Complexity-theoretic aspects of the metatheory of programming languages
- Old Quest for a Machine Independent Theory of Complexity