Finite State Incompressible Infinite Sequences

Cristian S. Calude[1], Ludwig Staiger[2] and Frank Stephan[3]

[1]University of Auckland
[2]Martin-Luther-Universität Halle-Wittenberg
[3]National University of Singapore

Algorithmic Randomness, Singapore, 9 June 2014

The incomputability of all descriptional complexities is an obstacle towards more "down-to-earth" applications of AIT (e.g. for practical compression).

To avoid incomputability we can

- restrict the resources available to the universal Turing machine, or
- restrict the computational power of the machines used (e.g. use context-free grammars or straight-line programs) instead of Turing machines.

Here we use the second approach with finite transducers instead of Turing machines.

The lack of a universal finite transducer is not an obstacle.

The incomputability of all descriptional complexities is an obstacle towards more "down-to-earth" applications of AIT (e.g. for practical compression).

To avoid incomputability we can

- restrict the resources available to the universal Turing machine, or
- restrict the computational power of the machines used (e.g. use context-free grammars or straight-line programs) instead of Turing machines.

Here we use the second approach with finite transducers instead of Turing machines.

The lack of a universal finite transducer is not an obstacle.

The incomputability of all descriptional complexities is an obstacle towards more "down-to-earth" applications of AIT (e.g. for practical compression).

To avoid incomputability we can

- restrict the resources available to the universal Turing machine, or
- restrict the computational power of the machines used (e.g. use context-free grammars or straight-line programs) instead of Turing machines.

Here we use the second approach with finite transducers instead of Turing machines.

The lack of a universal finite transducer is not an obstacle.

The incomputability of all descriptional complexities is an obstacle towards more "down-to-earth" applications of AIT (e.g. for practical compression).

To avoid incomputability we can

- restrict the resources available to the universal Turing machine, or
- restrict the computational power of the machines used (e.g. use context-free grammars or straight-line programs) instead of Turing machines.

Here we use the second approach with finite transducers instead of Turing machines.

The lack of a universal finite transducer is not an obstacle.

The incomputability of all descriptional complexities is an obstacle towards more “down-to-earth” applications of AIT (e.g. for practical compression).

To avoid incomputability we can

- restrict the resources available to the universal Turing machine, or
- restrict the computational power of the machines used (e.g. use context-free grammars or straight-line programs) instead of Turing machines.

Here we use the second approach with finite transducers instead of Turing machines.

The lack of a universal finite transducer is not an obstacle.

- **[Plain] Machine.** A machine or enumeration is a partially computable function **U** from binary strings to binary strings.
- **Prefix-Free Machine.** A prefix-free machine is a machine **M** such that for any two strings σ, τ with $\tau \neq \varepsilon$, if $\mathsf{M}(\sigma)$ is defined then $\mathsf{M}(\sigma\tau)$ is undefined.
- **Process Machine.** A process machine is a machine **W** such that for all σ, τ with $\sigma, \sigma\tau \in \mathrm{dom}(\mathsf{W})$, the string $\mathsf{W}(\sigma)$ is a prefix of $\mathsf{W}(\sigma\tau)$.
- **Universal Machine.** The machine (plain/prefix-free/process) **U** is universal if for every (plain/prefix-free/process) machine U' there is a constant **c** such that for every σ there exists an τ with $|\tau| \leq |\sigma| + \mathsf{c}$ and $\mathsf{U}(\tau) = \mathsf{U}'(\sigma)$.

- **[Plain] Machine.** A machine or enumeration is a partially computable function U from binary strings to binary strings.
- **Prefix-Free Machine.** A prefix-free machine is a machine M such that for any two strings σ, τ with $\tau \neq \varepsilon$, if $\mathsf{M}(\sigma)$ is defined then $\mathsf{M}(\sigma\tau)$ is undefined.
- **Process Machine.** A process machine is a machine W such that for all σ, τ with $\sigma, \sigma\tau \in \mathrm{dom}(\mathsf{W})$, the string $\mathsf{W}(\sigma)$ is a prefix of $\mathsf{W}(\sigma\tau)$.
- **Universal Machine.** The machine (plain/prefix-free/process) U is universal if for every (plain/prefix-free/process) machine U' there is a constant c such that for every σ there exists an τ with $|\tau| \leq |\sigma| + \mathsf{c}$ and $\mathsf{U}(\tau) = \mathsf{U}'(\sigma)$.

- **[Plain] Machine.** A machine or enumeration is a partially computable function **U** from binary strings to binary strings.
- **Prefix-Free Machine.** A prefix-free machine is a machine **M** such that for any two strings σ, τ with $\tau \neq \varepsilon$, if $\mathbf{M}(\sigma)$ is defined then $\mathbf{M}(\sigma\tau)$ is undefined.
- **Process Machine.** A process machine is a machine **W** such that for all σ, τ with $\sigma, \sigma\tau \in \mathrm{dom}(\mathbf{W})$, the string $\mathbf{W}(\sigma)$ is a prefix of $\mathbf{W}(\sigma\tau)$.
- **Universal Machine.** The machine (plain/prefix-free/process) **U** is universal if for every (plain/prefix-free/process) machine **U′** there is a constant **c** such that for every σ there exists an τ with $|\tau| \leq |\sigma| + \mathbf{c}$ and $\mathbf{U}(\tau) = \mathbf{U}'(\sigma)$.

- **[Plain] Machine.** A machine or enumeration is a partially computable function **U** from binary strings to binary strings.
- **Prefix-Free Machine.** A prefix-free machine is a machine **M** such that for any two strings σ, τ with $\tau \neq \varepsilon$, if $\mathbf{M}(\sigma)$ is defined then $\mathbf{M}(\sigma\tau)$ is undefined.
- **Process Machine.** A process machine is a machine **W** such that for all σ, τ with $\sigma, \sigma\tau \in \mathrm{dom}(\mathbf{W})$, the string $\mathbf{W}(\sigma)$ is a prefix of $\mathbf{W}(\sigma\tau)$.
- **Universal Machine.** The machine (plain/prefix-free/process) **U** is universal if for every (plain/prefix-free/process) machine $\mathbf{U}'$ there is a constant **c** such that for every σ there exists an τ with $|\tau| \leq |\sigma| + \mathbf{c}$ and $\mathbf{U}(\tau) = \mathbf{U}'(\sigma)$.

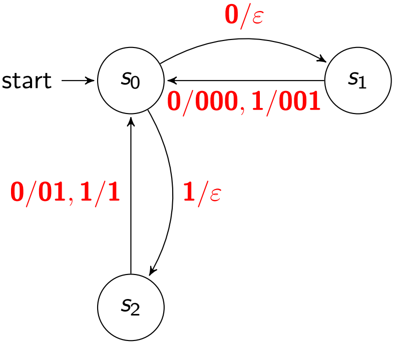
- **Kolmogorov Complexity.** Fix a universal machine. The plain Kolmogorov complexity of the string $\mathbf{x}$ is the length of the shortest $\sigma \in \mathbf{dom(U)}$ with $\mathbf{U}(\sigma) = \mathbf{x}$.
- **Prefix-free Kolmogorov Complexity.** The prefix-free Kolmogorov complexity is the Kolmogorov complexity based on a universal prefix-free machine.
- **Process Complexity.** The process complexity is the Kolmogorov complexity based on a universal process machine.

- **Kolmogorov Complexity.** Fix a universal machine. The plain Kolmogorov complexity of the string **x** is the length of the shortest $\sigma \in \mathbf{dom(U)}$ with $\mathbf{U}(\sigma) = \mathbf{x}$.
- **Prefix-free Kolmogorov Complexity.** The prefix-free Kolmogorov complexity is the Kolmogorov complexity based on a universal prefix-free machine.
- **Process Complexity.** The process complexity is the Kolmogorov complexity based on a universal process machine.

- **Kolmogorov Complexity.** Fix a universal machine. The plain Kolmogorov complexity of the string $\mathbf{x}$ is the length of the shortest $\sigma \in \mathbf{dom(U)}$ with $\mathbf{U}(\sigma) = \mathbf{x}$.
- **Prefix-free Kolmogorov Complexity.** The prefix-free Kolmogorov complexity is the Kolmogorov complexity based on a universal prefix-free machine.
- **Process Complexity.** The process complexity is the Kolmogorov complexity based on a universal process machine.

A sequence $\mathbf{A}$ is Martin-Löf random iff the prefix-free Kolmogorov complexity $\mathbf{H}$ of binary strings satisfies $\mathbf{H}(\mathbf{A} \upharpoonright \mathbf{n}) \geq \mathbf{n}$ for almost all $\mathbf{n}$.

An admissible transducer, short **transducer**, consists of a finite state-set $\mathbf{Q}$ and a transition function mapping each state $\mathbf{s}$ and bit $\mathbf{b} \in \{\mathbf{0}, \mathbf{1}\}$ to a new state $\mathbf{s}'$ and output word $\mathbf{w}$.



$\mathbf{Tr}(\mathbf{0110}) = \mathbf{00101}$

Normal Sequences. A sequence $\mathbf{A}$ is normal iff for every string σ, the number of occurrences of σ within the first $\mathbf{n}$ bits of $\mathbf{A}$ converges to $\mathbf{2}^{-|\sigma|}$ for $\mathbf{n} \to \infty$.

The **finite state complexity** of the transducer $\mathbf{Tr}$—denoted by $\mathbf{C_{Tr}(x)}$—is defined by the length of the shortest $\mathbf{y}$ with $\mathbf{Tr(y) = x}$.

Fact. A sequence is normal iff there is no transducer $\mathbf{Tr}$ and no constant $\mathbf{c < 1}$ such that $\mathbf{C_{Tr}(A \upharpoonright n) < n \cdot c}$, for infinitely many $\mathbf{n}$.

Normal Sequences. A sequence $\mathbf{A}$ is normal iff for every string σ, the number of occurrences of σ within the first $\mathbf{n}$ bits of $\mathbf{A}$ converges to $\mathbf{2}^{-|\sigma|}$ for $\mathbf{n} \to \infty$.

The **finite state complexity** of the transducer $\mathbf{Tr}$—denoted by $\mathbf{C_{Tr}(x)}$—is defined by the length of the shortest $\mathbf{y}$ with $\mathbf{Tr(y) = x}$.

Fact. A sequence is normal iff there is no transducer $\mathbf{Tr}$ and no constant $\mathbf{c} < 1$ such that $\mathbf{C_{Tr}(A \upharpoonright n) < n \cdot c}$, for infinitely many $\mathbf{n}$.

Normal Sequences. A sequence $\mathbf{A}$ is normal iff for every string σ, the number of occurrences of σ within the first $\mathbf{n}$ bits of $\mathbf{A}$ converges to $\mathbf{2}^{-|\sigma|}$ for $\mathbf{n} \to \infty$.

The **finite state complexity** of the transducer $\mathbf{Tr}$—denoted by $\mathbf{C_{Tr}(x)}$—is defined by the length of the shortest $\mathbf{y}$ with $\mathbf{Tr(y) = x}$.

Fact. A sequence is normal iff there is no transducer $\mathbf{Tr}$ and no constant $\mathbf{c} < \mathbf{1}$ such that $\mathbf{C_{Tr}(A \upharpoonright n) < n \cdot c}$, for infinitely many $\mathbf{n}$.

A partially computable function **S** mapping binary strings to transducers $\sigma \mapsto \mathsf{Tr}_\sigma^{\mathsf{S}}$ is called an **enumeration** provided every transducer **Tr** has a string $\sigma \in \mathrm{dom}(\mathsf{S})$.

Given a enumeration S of transducers the **finite state complexity** $C_S(x)$ is defined (Calude, Salomaa and Roblot [2011,2012]) by

$$C_S(x) = \min\{|\sigma| + |y| : \mathsf{Tr}_\sigma^{\mathsf{S}}(y) = x\}.$$

Fact. For every numeration S there is a constant c_S such that for all x,

$$C_S(x) \leq |x| + c_s.$$

Fact. Let S be a enumeration of transducers and let $\mathrm{dom}(\mathsf{S})$ be computable. Then the mapping $x \mapsto C_S(x)$ is computable.

A partially computable function **S** mapping binary strings to transducers $\sigma \mapsto \mathbf{Tr}^{\mathbf{S}}_{\sigma}$ is called an **enumeration** provided every transducer **Tr** has a string $\sigma \in \mathrm{dom}(\mathbf{S})$.

Given a enumeration **S** of transducers the **finite state complexity** $C_{\mathbf{S}}(x)$ is defined (Calude, Salomaa and Roblot [2011,2012]) by

$$C_{\mathbf{S}}(x) = \min\{|\sigma| + |y| : \mathbf{Tr}^{\mathbf{S}}_{\sigma}(y) = x\}.$$

Fact. For every numeration **S** there is a constant $c_{\mathbf{S}}$ such that for all x,

$$C_{\mathbf{S}}(x) \leq |x| + c_{\mathbf{s}}.$$

Fact. Let **S** be a enumeration of transducers and let $\mathrm{dom}(\mathbf{S})$ be computable. Then the mapping $x \mapsto C_{\mathbf{S}}(x)$ is computable.

A partially computable function **S** mapping binary strings to transducers $\sigma \mapsto \mathbf{Tr}_\sigma^{\mathbf{S}}$ is called an **enumeration** provided every transducer **Tr** has a string $\sigma \in \mathrm{dom}(\mathbf{S})$.

Given a enumeration **S** of transducers the **finite state complexity** $C_S(x)$ is defined (Calude, Salomaa and Roblot [2011,2012]) by

$$C_S(x) = \min\{|\sigma| + |y| : \mathbf{Tr}_\sigma^{\mathbf{S}}(y) = x\}.$$

Fact. For every numeration **S** there is a constant c_S such that for all x,

$$C_S(x) \leq |x| + c_s.$$

Fact. Let **S** be a enumeration of transducers and let $\mathrm{dom}(\mathbf{S})$ be computable. Then the mapping $x \mapsto C_S(x)$ is computable.

A partially computable function $\mathbf{S}$ mapping binary strings to transducers $\sigma \mapsto \mathbf{Tr}_\sigma^{\mathbf{S}}$ is called an **enumeration** provided every transducer **Tr** has a string $\sigma \in \mathrm{dom}(\mathbf{S})$.

Given a enumeration $\mathbf{S}$ of transducers the **finite state complexity** $\mathsf{C}_{\mathbf{S}}(\mathsf{x})$ is defined (Calude, Salomaa and Roblot [2011,2012]) by

$$\mathsf{C}_{\mathbf{S}}(\mathsf{x}) = \min\{|\sigma| + |\mathbf{y}| : \mathbf{Tr}_\sigma^{\mathbf{S}}(\mathbf{y}) = \mathsf{x}\}.$$

Fact. For every numeration $\mathbf{S}$ there is a constant $\mathsf{c}_{\mathbf{S}}$ such that for all x,

$$\mathsf{C}_{\mathbf{S}}(\mathsf{x}) \leq |\mathsf{x}| + \mathsf{c}_{\mathsf{s}}.$$

Fact. Let $\mathbf{S}$ be a enumeration of transducers and let $\mathrm{dom}(\mathbf{S})$ be computable. Then the mapping $\mathsf{x} \mapsto \mathsf{C}_{\mathbf{S}}(\mathsf{x})$ is computable.

A **perfect enumeration S** of all transducers is a partially computable function with a prefix-free and computable domain mapping each binary string $\sigma \in \mathrm{dom}(\mathsf{S})$ to a transducer $\mathsf{T}_{\sigma}^{\mathsf{S}}$ in a one-one and onto way.

A **universal enumeration S** of all transducers is a partially computable function with prefix-free domain such that for each other prefix-free enumeration S' of transducers there exists a constant c such that for all σ' in the domain of S', the transducer $\mathsf{T}_{\sigma'}^{\mathsf{S}'}$ equals some transducer $\mathsf{T}_{\sigma}^{\mathsf{S}}$ with $\sigma \in \mathrm{dom}(\mathsf{S})$ and

$$|\sigma| \leq |\sigma'| + \mathsf{c}.$$

A **perfect enumeration $\mathbf{S}$** of all transducers is a partially computable function with a prefix-free and computable domain mapping each binary string $\sigma \in \mathrm{dom}(\mathbf{S})$ to a transducer $\mathbf{T}_\sigma^{\mathbf{S}}$ in a one-one and onto way.

A **universal enumeration $\mathbf{S}$** of all transducers is a partially computable function with prefix-free domain such that for each other prefix-free enumeration $\mathbf{S}'$ of transducers there exists a constant $\mathbf{c}$ such that for all σ' in the domain of $\mathbf{S}'$, the transducer $\mathbf{T}_{\sigma'}^{\mathbf{S}'}$ equals some transducer $\mathbf{T}_\sigma^{\mathbf{S}}$ with $\sigma \in \mathrm{dom}(\mathbf{S})$ and

$$|\sigma| \leq |\sigma'| + \mathbf{c}.$$

Theorem. Let S be a universal machine enumerating all transducers. Then C_S is bounded:

- from above by the prefix-free Kolmogorov complexity, and
- from below by both, the plain Kolmogorov complexity of x and the process complexity of x.

Theorem. Let S be a universal machine enumerating all transducers. Then C_S is bounded:

- from above by the prefix-free Kolmogorov complexity, and
- from below by both, the plain Kolmogorov complexity of x and the process complexity of x.

Theorem. Let S be a universal machine enumerating all transducers. Then C_S is bounded:

- from above by the prefix-free Kolmogorov complexity, and
- from below by both, the plain Kolmogorov complexity of x and the process complexity of x.

Theorem. The following statements are equivalent to the sequence A not being Martin-Löf random:

- There is a perfect S such that for every c, almost all n satisfy $C_S(A \upharpoonright n) < n - c$.
- There is a perfect S such that for every c there is an n satisfying $C_S(A \upharpoonright n) < n - c$.
- For every universal S and every c, almost all n satisfy $C_S(A \upharpoonright n) < n - c$.
- For every universal S and every c there is an n satisfying $C_S(A \upharpoonright n) < n - c$.

Theorem. The following statements are equivalent to the sequence A not being Martin-Löf random:

- There is a perfect S such that for every c, almost all n satisfy $C_S(A \upharpoonright n) < n - c$.
- There is a perfect S such that for every c there is an n satisfying $C_S(A \upharpoonright n) < n - c$.
- For every universal S and every c, almost all n satisfy $C_S(A \upharpoonright n) < n - c$.
- For every universal S and every c there is an n satisfying $C_S(A \upharpoonright n) < n - c$.

Theorem. The following statements are equivalent to the sequence A not being Martin-Löf random:

- There is a perfect S such that for every c, almost all n satisfy $C_S(A \upharpoonright n) < n - c$.
- There is a perfect S such that for every c there is an n satisfying $C_S(A \upharpoonright n) < n - c$.
- For every universal S and every c, almost all n satisfy $C_S(A \upharpoonright n) < n - c$.
- For every universal S and every c there is an n satisfying $C_S(A \upharpoonright n) < n - c$.

Theorem. The following statements are equivalent to the sequence **A** not being Martin-Löf random:

- There is a perfect **S** such that for every **c**, almost all **n** satisfy $C_S(A \upharpoonright n) < n - c$.
- There is a perfect **S** such that for every **c** there is an **n** satisfying $C_S(A \upharpoonright n) < n - c$.
- For every universal **S** and every **c**, almost all **n** satisfy $C_S(A \upharpoonright n) < n - c$.
- For every universal **S** and every **c** there is an **n** satisfying $C_S(A \upharpoonright n) < n - c$.

Theorem. The following statements are equivalent to the sequence **A** not being Martin-Löf random:

- There is a perfect **S** such that for every **c**, almost all **n** satisfy $C_S(A \upharpoonright n) < n - c$.
- There is a perfect **S** such that for every **c** there is an **n** satisfying $C_S(A \upharpoonright n) < n - c$.
- For every universal **S** and every **c**, almost all **n** satisfy $C_S(A \upharpoonright n) < n - c$.
- For every universal **S** and every **c** there is an **n** satisfying $C_S(A \upharpoonright n) < n - c$.

If we drop the prefix-freeness condition the complexity $\mathbf{C_S}$ can behave in a different way. For example, as in the case of plain (Kolmogorov) complexity, in every sequence there exist infinitely many complexity dips.

Theorem. There exist enumerations $\mathbf{S}$ such that for every infinite sequence $\mathbf{A}$ there are infinitely many prefixes $v_i \sqsubset A$ such that

$$|\mathbf{v_i}| - \mathbf{C_S}(\mathbf{v_i}) > \mathbf{i}.$$

Complexity dips cannot be avoided even when we consider only transducers for which the output can always be at most $\mathbf{m}$ times as long as the input.

If we drop the prefix-freeness condition the complexity $\mathbf{C_S}$ can behave in a different way. For example, as in the case of plain (Kolmogorov) complexity, in every sequence there exist infinitely many complexity dips.

Theorem. There exist enumerations **S** such that for every infinite sequence **A** there are infinitely many prefixes $v_i \sqsubset A$ such that

$$|\mathbf{v_i}| - \mathbf{C_S}(\mathbf{v_i}) > \mathbf{i}.$$

Complexity dips cannot be avoided even when we consider only transducers for which the output can always be at most **m** times as long as the input.

If we drop the prefix-freeness condition the complexity $\mathbf{C_S}$ can behave in a different way. For example, as in the case of plain (Kolmogorov) complexity, in every sequence there exist infinitely many complexity dips.

Theorem. There exist enumerations **S** such that for every infinite sequence **A** there are infinitely many prefixes $v_i \sqsubset A$ such that

$$|\mathbf{v_i}| - \mathbf{C_S}(\mathbf{v_i}) > \mathbf{i}.$$

Complexity dips cannot be avoided even when we consider only transducers for which the output can always be at most **m** times as long as the input.

Definition. A sequence **A** is called C_S**–incompressible** if

$$\liminf_n C_S(A \upharpoonright n)/n = 1.$$

Theorem. For every enumeration **S**, every Martin-Löf random sequence is C_S–incompressible, but the converse implication is not true.

Indeed, there are normal sequences which are simultaneously C_S–compressible and **Liouville numbers** Definition .

This proves that C_S–incompressibility is **stronger** than all other known forms of finite automata based incompressibility.

Definition. A sequence **A** is called ***C_S*–incompressible** if

$$\mathbf{liminf_n C_S(A \upharpoonright n)/n = 1}.$$

Theorem. For every enumeration **S**, every Martin-Löf random sequence is $\mathbf{C_S}$–incompressible, but the converse implication is not true.

Indeed, there are normal sequences which are simultaneously $\mathbf{C_S}$–compressible and **Liouville numbers** ▸ Definition .

This proves that $\mathbf{C_S}$–incompressibility is **stronger** than all other known forms of finite automata based incompressibility.

Definition. A sequence **A** is called C_S**–incompressible** if

$$\mathbf{liminf_n C_S(A \upharpoonright n)/n = 1.}$$

Theorem. For every enumeration **S**, every Martin-Löf random sequence is $\mathbf{C_S}$–incompressible, but the converse implication is not true.

Indeed, there are normal sequences which are simultaneously $\mathbf{C_S}$–compressible and **Liouville numbers** ▸ Definition .

This proves that $\mathbf{C_S}$–incompressibility is **stronger** than all other known forms of finite automata based incompressibility.

Definition. A sequence **A** is called C_S**–incompressible** if

$$\mathbf{liminf_n C_S(A \upharpoonright n)/n = 1}.$$

Theorem. For every enumeration **S**, every Martin-Löf random sequence is $\mathbf{C_S}$–incompressible, but the converse implication is not true.

Indeed, there are normal sequences which are simultaneously $\mathbf{C_S}$–compressible and **Liouville numbers** Definition .

This proves that $\mathbf{C_S}$–incompressibility is **stronger** than all other known forms of finite automata based incompressibility.

Theorem. For every enumeration $\mathbf{S}$, every $\mathbf{C_S}$-incompressible sequence is normal.

Theorem. For every enumeration $\mathbf{S}$, there are normal (computable or incomputable) sequences $\mathbf{A}$ such that

$$\lim_{\mathbf{n}\to\infty} \mathbf{C_S}(\mathbf{A} \upharpoonright \mathbf{n})/\mathbf{n} = \mathbf{0},$$

so $\mathbf{C_S}$-compressible. ▸ Proof

Theorem. There is a normal and computable sequence which is $\mathbf{C_S}$-compressible for all enumerations $\mathbf{S}$.

Theorem. There exist a perfect enumeration $\mathbf{S}$ and a computable normal and $\mathbf{C_S}$-incompressible sequence.

Theorem. For every enumeration S, every C_S-incompressible sequence is normal.

Theorem. For every enumeration S, there are normal (computable or incomputable) sequences A such that

$$\lim_{n\to\infty} C_S(A \restriction n)/n = 0,$$

so C_S-compressible. ▸ Proof

Theorem. There is a normal and computable sequence which is C_S-compressible for all enumerations S.

Theorem. There exist a perfect enumeration S and a computable normal and C_S-incompressible sequence.

Theorem. For every enumeration $\mathbf{S}$, every $\mathbf{C_S}$-incompressible sequence is normal.

Theorem. For every enumeration $\mathbf{S}$, there are normal (computable or incomputable) sequences $\mathbf{A}$ such that

$$\lim_{n\to\infty} \mathbf{C_S}(\mathbf{A} \restriction \mathbf{n})/\mathbf{n} = \mathbf{0},$$

so $\mathbf{C_S}$-compressible. ▸ Proof

Theorem. There is a normal and computable sequence which is $\mathbf{C_S}$-compressible for all enumerations $\mathbf{S}$.

Theorem. There exist a perfect enumeration $\mathbf{S}$ and a computable normal and $\mathbf{C_S}$-incompressible sequence.

Theorem. For every enumeration $\mathbf{S}$, every $\mathbf{C_S}$-incompressible sequence is normal.

Theorem. For every enumeration $\mathbf{S}$, there are normal (computable or incomputable) sequences $\mathbf{A}$ such that

$$\lim_{n\to\infty} \mathbf{C_S}(\mathbf{A} \upharpoonright \mathbf{n})/\mathbf{n} = \mathbf{0},$$

so $\mathbf{C_S}$-compressible. ▸ Proof

Theorem. There is a normal and computable sequence which is $\mathbf{C_S}$-compressible for all enumerations $\mathbf{S}$.

Theorem. There exist a perfect enumeration $\mathbf{S}$ and a computable normal and $\mathbf{C_S}$-incompressible sequence.

Enumerations are computable listings of all transducers. Two types of enumerations have been defined: **universal and perfect**.

Characterisations of Martin-Löf randomness in terms of C_S-complexity for both types of enumerations S.

Relations between finite state complexity and other descriptional complexities have been obtained. In particular, finite state complexities based on some exotic enumerations behave like the plain (Kolmogorov) complexity.

The notion of C_S-incompressibility was investigated and related to normality and (in)computability. C_S-incompressibility implies normality but the converse fails.

Main fact. **Enumerations matter more than processing units.**

Enumerations are computable listings of all transducers. Two types of enumerations have been defined: **universal and perfect**.

Characterisations of Martin-Löf randomness in terms of C_S-complexity for both types of enumerations S.

Relations between finite state complexity and other descriptional complexities have been obtained. In particular, finite state complexities based on some exotic enumerations behave like the plain (Kolmogorov) complexity.

The notion of C_S-incompressibility was investigated and related to normality and (in)computability. C_S-incompressibility implies normality but the converse fails.

Main fact. **Enumerations matter more than processing units.**

Enumerations are computable listings of all transducers. Two types of enumerations have been defined: **universal and perfect**.

Characterisations of Martin-Löf randomness in terms of C_S-complexity for both types of enumerations S.

Relations between finite state complexity and other descriptional complexities have been obtained. In particular, finite state complexities based on some exotic enumerations behave like the plain (Kolmogorov) complexity.

The notion of C_S-incompressibility was investigated and related to normality and (in)computability. C_S-incompressibility implies normality but the converse fails.

Main fact. **Enumerations matter more than processing units.**

Enumerations are computable listings of all transducers. Two types of enumerations have been defined: **universal and perfect**.

Characterisations of Martin-Löf randomness in terms of C_S-complexity for both types of enumerations S.

Relations between finite state complexity and other descriptional complexities have been obtained. In particular, finite state complexities based on some exotic enumerations behave like the plain (Kolmogorov) complexity.

The notion of C_S-incompressibility was investigated and related to normality and (in)computability. C_S-incompressibility implies normality but the converse fails.

Main fact. **Enumerations matter more than processing units.**

Enumerations are computable listings of all transducers. Two types of enumerations have been defined: **universal and perfect**.

Characterisations of Martin-Löf randomness in terms of C_S-complexity for both types of enumerations S.

Relations between finite state complexity and other descriptional complexities have been obtained. In particular, finite state complexities based on some exotic enumerations behave like the plain (Kolmogorov) complexity.

The notion of C_S-incompressibility was investigated and related to normality and (in)computability. C_S-incompressibility implies normality but the converse fails.

Main fact. Enumerations matter more than processing units.

- Is there a perfect enumeration and a computable sequence $\mathbf{A}$ such that $\mathbf{C_S(A \upharpoonright n) \geq n - c}$, for some $\mathbf{c}$ and all $\mathbf{n}$?
- Study the relations between $\mathbf{C_S}$-incompressibility and other notions of randomness, in particular ε-randomness?

- Is there a perfect enumeration and a computable sequence $\mathbf{A}$ such that $\mathbf{C_S(A \upharpoonright n) \geq n - c}$, for some $\mathbf{c}$ and all $\mathbf{n}$?
- Study the relations between $\mathbf{C_S}$-incompressibility and other notions of randomness, in particular ε-randomness?

References

C. S. Calude, K. Salomaa, T. K. Roblot. Finite state complexity, *Theoretical Computer Science* 412: 5668–5677 (2011).

C. S. Calude, K. Salomaa, T. K. Roblot. State-size hierarchy for FS-complexity, *International Journal of Foundations of Computer Science*, 23, 1: 37–50 (2012).

C. S. Calude, L. Staiger, F. Stephan. Finite state incompressible infinite sequences, in T. V. Gopal, M. Agrawal, A. Li, B. S. Cooper (eds). *Proceedings of the 11th Annual Conference on Theory and Applications of Models of Computation*, LNCS 8402, Springer, 2014, 50–66.

A **Liouvile number** is a transcendental real number α such that for every positive integer $\mathbf{n}$, there exist integers $\mathbf{p}$ and $\mathbf{q}$ with $\mathbf{q} > 1$ such that

$$0 < |\alpha - \frac{p}{q}| < q^{-n}.$$

For example,

$$\sum_{k=1}^{\infty} 2^{-k!} = 0.110001000000000000000000100\ldots$$

Incompressibility

Definition of Liouvile Number

A **Liouvile number** is a transcendental real number α such that for every positive integer $\mathbf{n}$, there exist integers $\mathbf{p}$ and $\mathbf{q}$ with $\mathbf{q} > \mathbf{1}$ such that

$$\mathbf{0} < |\alpha - \frac{\mathbf{p}}{\mathbf{q}}| < \mathbf{q}^{-\mathbf{n}}.$$

For example,

$$\sum_{\mathbf{k}=\mathbf{1}}^{\infty} \mathbf{2}^{-\mathbf{k}!} = \mathbf{0.1100010000000000000000100}\ldots$$

▸ Incompressibility

Proof Sketch

Denote by $B(r)$ the prefix of length 2^r of a de Bruijn string of order r (i.e. a string of length $2^r + r - 1$ containing every string of length r as a contiguous substring exactly once). For example, $B(2) = 0011$ and $B(3) = 00010111$.

Lemma. If the function f is increasing and $f(i) \geq i^i$, then the sequence

$$A_f = B(1)^{f(1)} B(2)^{f(2)} \cdots B(n)^{f(n)} \cdots$$

is normal and the real number $0.A_f$ is a Liouville number.

Proof Sketch

Denote by $\mathbf{B(r)}$ the prefix of length $\mathbf{2^r}$ of a de Bruijn string of order $\mathbf{r}$ (i.e. a string of length $\mathbf{2^r + r - 1}$ containing every string of length $\mathbf{r}$ as a contiguous substring exactly once). For example, $\mathbf{B(2) = 0011}$ and $\mathbf{B(3) = 00010111}$.

Lemma. If the function $\mathbf{f}$ is increasing and $\mathbf{f(i) \geq i^i}$, then the sequence

$$\mathbf{A_f = B(1)^{f(1)} B(2)^{f(2)} \ldots B(n)^{f(n)} \ldots}$$

is normal and the real number $\mathbf{0.A_f}$ is a Liouville number.

Consider the transducers $\mathbf{T_n}$

$$\begin{aligned}
\delta_n(s_i, 0) &= s_i, & \mu_n(s_i, 0) &= B(i), & \text{for } i \le n, \\
\delta_n(s_i, 1) &= s_{i+1}, & \mu_n(s_i, 1) &= B(i), & \text{for } i \le n, \\
\delta_n(s_{n+1}, a) &= s_{n+1}, & \mu_n(s_{n+1}, a) &= a, & \text{for } a \in \{0, 1\}.
\end{aligned}$$

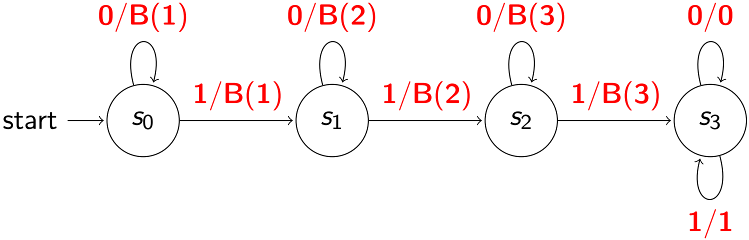
For example, $\mathbf{T_3}$ is

Proof Sketch (Continued)

Consider the transducers $\mathbf{T_n}$

$$
\begin{aligned}
\delta_n(s_i, 0) &= s_i, & \mu_n(s_i, 0) &= B(i), & \text{for } i \leq n, \\
\delta_n(s_i, 1) &= s_{i+1}, & \mu_n(s_i, 1) &= B(i), & \text{for } i \leq n, \\
\delta_n(s_{n+1}, a) &= s_{n+1}, & \mu_n(s_{n+1}, a) &= a, & \text{for } a \in \{0, 1\}\,.
\end{aligned}
$$

For example, $\mathbf{T_3}$ is



Proof Sketch (Continued)

For every prefix σ of $\mathbf{A_f}$ of the form

$$\sigma = \mathbf{B(1)^{f(1)} \cdots B(n-1)^{f(n-1)} \cdot B(n)^{j}} \cdot \tau,$$

we have $\mathbf{B(1)} \sqsubset \sigma \sqsubset \mathbf{A_f}$, so

$$\frac{\mathbf{C_S}(\sigma)}{|\sigma|} \le \frac{\mathbf{4f(n-1)+j}}{\mathbf{2^{n-1}f(n-1)+2^{n}j}} \le \frac{\mathbf{4}}{\mathbf{2^{n-1}}},$$

hence

$$\lim_{\mathbf{n}\to\infty} \mathbf{C_S(A(\upharpoonright n)/|n| = 0}.$$

▸ IncompressibilityNormality

For every prefix σ of $\mathbf{A_f}$ of the form

$$\sigma = \mathbf{B(1)^{f(1)}} \cdots \mathbf{B(n-1)^{f(n-1)}} \cdot \mathbf{B(n)^{j}} \cdot \tau,$$

we have $\mathbf{B(1)} \sqsubset \sigma \sqsubset \mathbf{A_f}$, so

$$\frac{\mathbf{C_S}(\sigma)}{|\sigma|} \leq \frac{\mathbf{4f(n-1)+j}}{\mathbf{2^{n-1}f(n-1)+2^{n}j}} \leq \frac{\mathbf{4}}{\mathbf{2^{n-1}}},$$

hence

$$\lim_{\mathbf{n}\to\infty} \mathbf{C_S}(\mathbf{A}(\restriction \mathbf{n})/|\mathbf{n}| = \mathbf{0}.$$

IncompressibilityNormality