Normal numbers and automatic complexity

alexander.shen@lirmm.fr, www.lirmm.fr/~ashen

LIRMM CNRS & University of Montpellier

September 2017, FCT
Individual random sequence

If a coin tossing gives 00000 or 01010101, we become suspicious

Individual sequence of 0 and 1: can it be "random"/"nonrandom"?

von Mises (1919): Kollektiv: a basic notion of probability theory; frequency stability

Borel: normal numbers
If a coin tossing gives 00000... or 01010101..., we become suspicious
If a coin tossing gives 00000... or 01010101..., we become suspicious.

Individual sequence of 0 and 1: can it be “random”/“nonrandom”?
If a coin tossing gives 00000... or 01010101..., we become suspicious.

Individual sequence of 0 and 1: can it be “random” / “nonrandom”?

von Mises (1919): Kollektiv: a basic notion of probability theory; frequency stability
If a coin tossing gives 00000... or 01010101..., we become suspicious.

Individual sequence of 0 and 1: can it be “random”/“nonrandom”?

von Mises (1919): Kollektiv: a basic notion of probability theory; frequency stability

Borel: normal numbers
Normal numbers

Normal numbers

00100111010111

#(n) = number of 0 among the first n bits

simply normal: #0(n) / n → 1/2, #1(n) → 1/2

#00(n) = number of occurrences of 00 in the first n positions

#00(n) + #01(n) + #10(n) + #11(n) = n

normal: #00(n) / n → 1/4 and the same for all other blocks (any length)

Another approach: cut the sequence into k-bit blocks and count the number of blocks of each type (aligned occurrences); these two definitions are equivalent

almost all numbers are normal e, π, √2

Champernowne: 0 1 10 11 100 101 110...

Wall: α is normal, n integer ⇒ nα, α/n normal

alexander.shen@lirmm.fr, www.lirmm.fr/~ashen

Normal numbers and automatic complexity
00100111010111…

$\#_0(n) =$ number of 0 among the first $n$ bits
Normal numbers

- 00100111010111...
  \#_0(n) = number of 0 among the first n bits
- simply normal: \#_0(n)/n \to 1/2, \#_1(n) \to 1/2
Normal numbers

- 00100111010111...
  \(#_0(n)\) = number of 0 among the first \(n\) bits
- simply normal: \(#_0(n) / n \to 1/2\), \(#_1(n) \to 1/2\)
- \(#_{00}(n) =\)
  number of occurrences of 00 in the first \(n\) positions

\[\]

\[\]

\[\]

\[\]

\[\]

\[\]
Normal numbers

- 00100111010111...  
  
  #₀(𝑛) = number of 0 among the first 𝑛 bits  
- simply normal: #₀(𝑛)/𝑛 → 1/2, #₁(𝑛) → 1/2  
- #₀₀(𝑛) =  
  number of occurrences of 00 in the first 𝑛 positions  
- #₀₀(𝑛) + #₀₁(𝑛) + #₁₀(𝑛) + #₁₁(𝑛) = 𝑛
Normal numbers

- 00100111010111...
  \(\#_0(n)\) = number of 0 among the first \(n\) bits
- simply normal: \(\#_0(n)/n \to 1/2, \#_1(n) \to 1/2\)
- \(\#_{00}(n) = \) number of occurrences of 00 in the first \(n\) positions
- \(\#_{00}(n) + \#_{01}(n) + \#_{10}(n) + \#_{11}(n) = n\)
- normal: \(\#_{00}(n)/n \to 1/4\) and the same for all other blocks (any length)
00100111010111…
#₀(𝑛) = number of 0 among the first 𝑛 bits
simply normal: #₀(𝑛)/𝑛 → 1/2, #₁(𝑛) → 1/2
#₀₀(𝑛) =
number of occurrences of 00 in the first 𝑛 positions
#₀₀(𝑛) + #₀₁(𝑛) + #₁₀(𝑛) + #₁₁(𝑛) = 𝑛
normal: #₀₀(𝑛)/𝑛 → 1/4 and the same for all other blocks (any length)
Another approach: cut the sequence into 𝑘-bit blocks and count the number of blocks of each type (aligned occurrences); these two definitions are equivalent
Normal numbers

- 00100111010111...
  \#_0(n) = number of 0 among the first \( n \) bits
- simply normal: \( \#_0(n)/n \rightarrow 1/2, \#_1(n) \rightarrow 1/2 \)
- \#_{00}(n) =
  number of occurences of 00 in the first \( n \) positions
- \#_{00}(n) + \#_{01}(n) + \#_{10}(n) + \#_{11}(n) = n
- normal: \( \#_{00}(n)/n \rightarrow 1/4 \) and the same for all other blocks (any length)
- Another approach: cut the sequence into \( k \)-bit blocks and count the number of blocks of each type (aligned occurrences); these two definitions are equivalent
- almost all numbers are normal
00100111010111...

*#₀(n) = number of 0 among the first n bits*

simply normal: *#₀(n)/n → 1/2, #₁(n) → 1/2*

*#₀₀(n) = number of occurrences of 00 in the first n positions*

*#₀₀(n) + #₀₁(n) + #₁₀(n) + #₁₁(n) = n*

normal: *#₀₀(n)/n → 1/4 and the same for all other blocks (any length)*

Another approach: cut the sequence into k-bit blocks and count the number of blocks of each type (aligned occurrences); these two definitions are equivalent

almost all numbers are normal

e, π, \sqrt{2}??? Champernowne: 0 1 10 11 100 101 110...
Normal numbers

- 00100111010111... 
- $\#_0(n) =$ number of 0 among the first $n$ bits 
- simply normal: $\#_0(n)/n \rightarrow 1/2$, $\#_1(n) \rightarrow 1/2$ 
- $\#_{00}(n) =$ number of occurrences of 00 in the first $n$ positions 
- $\#_{00}(n) + \#_{01}(n) + \#_{10}(n) + \#_{11}(n) = n$ 
- normal: $\#_{00}(n)/n \rightarrow 1/4$ and the same for all other blocks (any length) 
- Another approach: cut the sequence into $k$-bit blocks and count the number of blocks of each type (aligned occurrences); these two definitions are equivalent 
- almost all numbers are normal 
- $e, \pi, \sqrt{2}??$ Champernowne: 0 1 10 11 100 101 110... 
- Wall: $\alpha$ is normal, $n$ integer $\Rightarrow$ $n\alpha$, $\alpha/n$ normal
Randomness as incompressibility

Individual random sequences: plausible as outcomes of coin tossing experiment

Normality is necessary but hardly sufficient

Martin-Löf: random ⇔ obeys all "effective laws" of probability theory

Kolmogorov, Levin, Chaitin, ...: randomness = incompressibility of prefixes

000... not random: short description: "million zeros"

What is "description"? Different answers possible

Normality = weak randomness

Limited class of descriptions: finite memory

Well-known since 1960s (Agafonov, Schnorr, Stimm, Dai, Lathrop, Lutz, Mayordomo, Becher, Heiber, ...)

our (small) contribution: clean definitions and proofs

alexander.shen@lirmm.fr, www.lirmm.fr/~ashen

Normal numbers and automatic complexity
Randomness as incompressibility

- Individual random sequences: plausible as outcomes of coin tossing experiment
- Individual random sequences: plausible as outcomes of coin tossing experiment
- Normality is necessary but hardly sufficient
Randomness as incompressibility

- Individual random sequences: plausible as outcomes of coin tossing experiment
- Normality is necessary but hardly sufficient
- Martin-Löf: random $\iff$ obeys all “effective laws” of probability theory
Randomness as incompressibility

- Individual random sequences: plausible as outcomes of coin tossing experiment
- Normality is necessary but hardly sufficient
- Martin-Löf: random ⇔ obeys all “effective laws” of probability theory
- Kolmogorov, Levin, Chaitin,…: randomness = incompressibility of prefixes
Randomness as incompressibility

- Individual random sequences: plausible as outcomes of coin tossing experiment
- Normality is necessary but hardly sufficient
- Martin-Löf: random $\iff$ obeys all “effective laws” of probability theory
- Kolmogorov, Levin, Chaitin,...: randomness $=$ incompressibility of prefixes
- 000...000 not random: short description: “million zeros”
Randomness as incompressibility

- Individual random sequences: plausible as outcomes of coin tossing experiment
- Normality is necessary but hardly sufficient
- Martin-Löf: random $\iff$ obeys all “effective laws” of probability theory
- Kolmogorov, Levin, Chaitin, . . . : randomness $\equiv$ incompressibility of prefixes
- 000 . . . 000 not random: short description: “million zeros”
- What is “description”? Different answers possible
Randomness as incompressibility

- Individual random sequences: plausible as outcomes of coin tossing experiment
- Normality is necessary but hardly sufficient
- Martin-Löf: random ⇔ obeys all “effective laws” of probability theory
- Kolmogorov, Levin, Chaitin,...: randomness = incompressibility of prefixes
- 000...000 not random: short description: “million zeros”
- What is “description”? Different answers possible
- Normality = weak randomness
Randomness as incompressibility

- Individual random sequences: plausible as outcomes of coin tossing experiment
- Normality is necessary but hardly sufficient
- Martin-Löf: random $\iff$ obeys all “effective laws” of probability theory
- Kolmogorov, Levin, Chaitin, . . . : randomness $=$ incompressibility of prefixes
- 000 . . . 000 not random: short description: “million zeros”
- What is “description”? Different answers possible
- Normality $=$ *weak* randomness
- *Limited* class of descriptions: finite memory
Randomness as incompressibility

- Individual random sequences: plausible as outcomes of coin tossing experiment
- Normality is necessary but hardly sufficient
- Martin-Löf: random $\iff$ obeys all “effective laws” of probability theory
- Kolmogorov, Levin, Chaitin, . . . : randomness = incompressibility of prefixes
- 000 . . . 000 not random: short description: “million zeros”
- What is “description”? Different answers possible
- Normality $=$ weak randomness
- Limited class of descriptions: finite memory
- Well-known since 1960s (Agafonov, Schnorr, Stimm, Dai, Lathrop, Lutz, Mayordomo, Becher, Heiber, . . . )
Randomness as incompressibility

- Individual random sequences: plausible as outcomes of coin tossing experiment
- Normality is necessary but hardly sufficient
- Martin-Löf: random ⇔ obeys all “effective laws” of probability theory
- Kolmogorov, Levin, Chaitin,...: randomness = incompressibility of prefixes
- 000...000 not random: short description: “million zeros”
- What is “description”? Different answers possible
- Normality = weak randomness
- Limited class of descriptions: finite memory
- Well-known since 1960s (Agafonov, Schnorr, Stimm, Dai, Lathrop, Lutz, Mayordomo, Becher, Heiber,...)
- our (small) contribution: clean definitions and proofs
Kolmogorov complexity: framework

Relation $D(p, x)$ on strings: "$p$ is a description of $x$"

$$C_D(x) = \min \{ |p| : D(p, x) \}$$

trivial $D$: $\Lambda$ is a description of everything, $C_D(x) = 0$

restrictions for $D$ needed

plain Kolmogorov complexity: $D$ is a c.e. functional relation (only one $x$ for each $p$)

our requirement: the relation $D$ is an $O(1)$-valued function (each description describes $O(1)$ objects) that "can be checked with finite memory"

corresponding class of complexity functions $C_D$ allows us to characterize normal sequences as incompressible
Relation $D(p, x)$ on strings: “$p$ is a description of $x$”
Relation $D(p, x)$ on strings: “$p$ is a description of $x$”

$C_D(x) = \min\{|p| : D(p, x)\}$
Kolmogorov complexity: framework

- Relation $D(p, x)$ on strings: “$p$ is a description of $x$”
- $C_D(x) = \min\{|p| : D(p, x)\}$
- trivial $D$: $\Lambda$ is a description of everything, $C_D(x) = 0$
Relation $D(p, x)$ on strings: “$p$ is a description of $x$”

$C_D(x) = \min\{|p| : D(p, x)\}$

trivial $D$: \Lambda is a description of everything, $C_D(x) = 0$

restrictions for $D$ needed
Kolmogorov complexity: framework

- Relation $D(p, x)$ on strings: “$p$ is a description of $x$”
- $C_D(x) = \min\{|p| : D(p, x)\}$
- trivial $D$: $\Lambda$ is a description of everything, $C_D(x) = 0$
- restrictions for $D$ needed
- plain Kolmogorov complexity: $D$ is a c.e. functional relation (only one $x$ for each $p$)
Relation $D(p, x)$ on strings: “$p$ is a description of $x$”

$C_D(x) = \min\{|p| : D(p, x)\}$

trivial $D$: $\Lambda$ is a description of everything, $C_D(x) = 0$

restrictions for $D$ needed

plain Kolmogorov complexity: $D$ is a c.e. functional relation (only one $x$ for each $p$)

our requirement: the relation $D$ is an $O(1)$-valued function (each description describes $O(1)$ objects) that “can be checked with finite memory”
Kolmogorov complexity: framework

- Relation $D(p, x)$ on strings: “$p$ is a description of $x$”
- $C_D(x) = \min\{|p| : D(p, x)\}$
- trivial $D$: $\Lambda$ is a description of everything, $C_D(x) = 0$
- restrictions for $D$ needed
- plain Kolmogorov complexity: $D$ is a c.e. functional relation (only one $x$ for each $p$)
- our requirement: the relation $D$ is an $O(1)$-valued function (each description describes $O(1)$ objects) that “can be checked with finite memory”
- corresponding class of complexity functions $C_D$ allows us to characterize normal sequences as incompressible
Idea: \( D(p, x) \) is automatic if it can be checked reading \( p \) and \( x \) bit by bit, with finite memory similar to rational relations but no initial/final state.

Formal definition: graph; edges labeled by \((u, v), (u, \varepsilon), (\varepsilon, u), (\varepsilon, \varepsilon)\).

Path \( \Rightarrow \) pair of strings.

\( D = \) the set of all pairs that can be read along paths.

"Automatic relations" multiplication and division by an integer constant are automatic relations.

Union/composition of two automatic relations is automatic.
Idea: $D(p, x)$ is automatic if it can be checked reading $p$ and $x$ bit by bit, with finite memory.
Idea: $D(p, x)$ is automatic if it can be checked reading $p$ and $x$ bit by bit, with finite memory

similar to rational relations but no initial/final state
Idea: $D(p, x)$ is automatic if it can be checked reading $p$ and $x$ bit by bit, with finite memory

similar to rational relations but no initial/final state

Formal definition: graph; edges labeled by $(u, v), (u, \varepsilon), (\varepsilon, u), (\varepsilon, \varepsilon)$
Idea: \( D(p, x) \) is automatic if it can be checked reading \( p \) and \( x \) bit by bit, with finite memory.

Similar to rational relations but no initial/final state.

Formal definition: graph; edges labeled by \((u, v)\), \((u, \varepsilon)\), \((\varepsilon, u)\), \((\varepsilon, \varepsilon)\).

Path \( \Rightarrow \) pair of strings.
Idea: $D(p, x)$ is *automatic* if it can be checked reading $p$ and $x$ bit by bit, with finite memory

similar to rational relations but no initial/final state

Formal definition: graph; edges labeled by $(u, v), (u, \varepsilon), (\varepsilon, u), (\varepsilon, \varepsilon)$

path $\Rightarrow$ pair of strings

$D =$ the set of all pairs that can be read along paths
Idea: $D(p, x)$ is automatic if it can be checked reading $p$ and $x$ bit by bit, with finite memory

similar to rational relations but no initial/final state

Formal definition: graph; edges labeled by $(u, v)$, $(u, \varepsilon)$, $(\varepsilon, u)$, $(\varepsilon, \varepsilon)$

path $\Rightarrow$ pair of strings

$D =$ the set of all pairs that can be read along paths

“automatic relations”
Idea: $D(p, x)$ is automatic if it can be checked reading $p$ and $x$ bit by bit, with finite memory

- similar to rational relations but no initial/final state
- Formal definition: graph; edges labeled by $(u, v), (u, \varepsilon), (\varepsilon, u), (\varepsilon, \varepsilon)$
- path $\Rightarrow$ pair of strings
- $D =$ the set of all pairs that can be read along paths
- “automatic relations”
- multiplication and division by an integer constant are automatic relations
Idea: $D(p, x)$ is automatic if it can be checked reading $p$ and $x$ bit by bit, with finite memory
similar to rational relations but no initial/final state
Formal definition: graph; edges labeled by $(u, v), (u, \varepsilon), (\varepsilon, u), (\varepsilon, \varepsilon)$
path $\Rightarrow$ pair of strings
$D =$ the set of all pairs that can be read along paths
“automatic relations”
multiplication and division by an integer constant are automatic relations
union/composition of two automatic relations is automatic
Theorem (Becher, Heiber)

A sequence $x_1x_2x_3\ldots$ is normal $\iff$

$$\liminf C_D(x_1\ldots x_n)/n \geq 1$$

for every automatic $O(1)$-valued relation $D(p,x)$
Part 1: non-normal sequences are compressible

- Assume that different k-bit blocks have different frequencies.
- Use standard block coding (Shannon, Fano, Huffman).
- Frequent blocks have shorter codes.
- Block coding uses finite memory.

Technical: select a subsequence that has limit frequencies; use these frequencies for block coding, use convexity of entropy function.
Part 1: non-normal sequences are compressible

- assume that different $k$-bit blocks have different frequencies
Part 1: non-normal sequences are compressible

- assume that different $k$-bit blocks have different frequencies
- use standard block coding (Shannon, Fano, Huffman)
Part 1: non-normal sequences are compressible

- assume that different $k$-bit blocks have different frequencies
- use standard block coding (Shannon, Fano, Huffman) [frequent blocks have shorter codes]
Part 1: non-normal sequences are compressible

- assume that different $k$-bit blocks have different frequencies
- use standard block coding (Shannon, Fano, Huffman) [frequent blocks have shorter codes]
- block coding uses finite memory
Part 1: non-normal sequences are compressible

- assume that different $k$-bit blocks have different frequencies
- use standard block coding (Shannon, Fano, Huffman) [frequent blocks have shorter codes]
- block coding uses finite memory
- Technical: select a subsequence that has limit frequencies; use these frequencies for block coding, use convexity of entropy function
Part 2: normal sequences are not compressible

Some automatic relation $D$ why $x_1 x_2 \ldots x_N$ is not compressible?

Split it into $k$-bit blocks $X_1 X_2 \ldots X_M$

description $p$ can be also split into corresponding blocks

trivial crucial lemma: $C_D(xy) \geq C_D(x) + C_D(y)$

all $k$-bit strings appear equally often among $X_1, X_2, \ldots, X_M$

most of $k$-bit strings are incompressible (even in Kolmogorov's sense)

so the economy is negligible compared to length
Part 2: normal sequences are not compressible

- Normal sequence $x_1x_2 \ldots$
Part 2: normal sequences are not compressible

- Normal sequence $x_1x_2\ldots$
- Some automatic $O(1)$ relation $D$
Part 2: normal sequences are not compressible

- Normal sequence $x_1 x_2 \ldots$
- Some automatic $O(1)$ relation $D$
- Why $x_1 x_2 \ldots x_N$ is not compressible?
Part 2: normal sequences are not compressible

- Normal sequence \( x_1x_2\ldots \)
- Some automatic \( O(1) \) relation \( D \)
- Why \( x_1x_2\ldots x_N \) is not compressible?
- Split it into \( k \)-bit blocks \( X_1X_2\ldots X_M \)
Part 2: normal sequences are not compressible

- Normal sequence $x_1x_2\ldots$
- Some automatic $O(1)$ relation $D$
- Why $x_1x_2\ldots x_N$ is not compressible?
- Split it into $k$-bit blocks $X_1X_2\ldots X_M$
- description $p$ can be also split into corresponding blocks
Part 2: normal sequences are not compressible

- Normal sequence $x_1x_2\ldots$
- Some automatic $O(1)$ relation $D$
- Why $x_1x_2\ldots x_N$ is not compressible?
- Split it into $k$-bit blocks $X_1X_2\ldots X_M$
- description $p$ can be also split into corresponding blocks
- trivial crucial lemma: $C_D(xy) \geq C_D(x) + C_D(y)$
Normal sequence $x_1 x_2 \ldots$

Some automatic $O(1)$ relation $D$

Why $x_1 x_2 \ldots x_N$ is not compressible?

Split it into $k$-bit blocks $X_1 X_2 \ldots X_M$

description $p$ can be also split into corresponding blocks

trivial crucial lemma: $C_D(xy) \geq C_D(x) + C_D(y)$

all $k$-bit strings appear equally often among $X_1, X_2, \ldots, X_M$
Normal sequence $x_1x_2\ldots$
Some automatic $O(1)$ relation $D$
Why $x_1x_2\ldots x_N$ is not compressible?
Split it into $k$-bit blocks $X_1X_2\ldots X_M$
description $p$ can be also split into corresponding blocks
trivial crucial lemma: $C_D(xy) \geq C_D(x) + C_D(y)$
all $k$-bit strings appear equally often among $X_1, X_2, \ldots, X_M$
most of $k$-bit strings are incompressible (even in Kolmogorov’s sense)
Part 2: normal sequences are not compressible

- Normal sequence $x_1 x_2 \ldots$
- Some automatic $O(1)$ relation $D$
- Why $x_1 x_2 \ldots x_N$ is not compressible?
- Split it into $k$-bit blocks $X_1 X_2 \ldots X_M$
- description $p$ can be also split into corresponding blocks
- trivial crucial lemma: $C_D(xy) \geq C_D(x) + C_D(y)$
- all $k$-bit strings appear equally often among $X_1, X_2, \ldots, X_M$
- most of $k$-bit strings are incompressible (even in Kolmogorov’s sense)
- so the economy is negligible compared to length
What do we get as byproducts?

\[ \alpha \text{ is normal, } n \text{ integer} \Rightarrow n\alpha \text{ and } \alpha/n \text{ are normal} \]

Proof: multiplication and division by a constant are \( O(1) \)-valued automatic relations and composition of automatic relations is automatic.

Aligned definition \( \equiv \) non-aligned definition

Proof: the criterion can be proven for non-aligned definition in a similar way.

Agafonov: automatic selection rule preserves normality

Proof: if a selected subsequence is compressible, this compression can be used together with uncompressed description of the remaining terms (some care needed).

Piatetski-Shapiro theorem: if no block appear \( c \) times more often then they should, the sequence is normal.

Hall: $\alpha$ is normal, $n$ integer $\Rightarrow n\alpha$ and $\alpha/n$ are normal
What do we get as byproducts?

- Hall: $\alpha$ is normal, $n$ integer $\Rightarrow n\alpha$ and $\alpha/n$ are normal
- Proof: multiplication and division by a constant are $O(1)$-valued automatic relations and composition of automatic relations is automatic
What do we get as byproducts?

- Hall: $\alpha$ is normal, $n$ integer $\Rightarrow$ $n\alpha$ and $\alpha/n$ are normal
- Proof: multiplication and division by a constant are $O(1)$-valued automatic relations and composition of automatic relations is automatic
- aligned definition $\Leftrightarrow$ non-aligned definition


alexander.shen@lirmm.fr, www.lirmm.fr/~ashen
What do we get as byproducts?

- Hall: $\alpha$ is normal, $n$ integer $\Rightarrow$ $n\alpha$ and $\alpha/n$ are normal
- Proof: multiplication and division by a constant are $O(1)$-valued automatic relations and composition of automatic relations is automatic
- aligned definition $\Leftrightarrow$ non-aligned definition
- Proof: the criterion can be proven for non-aligned definition in a similar way
What do we get as byproducts?

- Hall: $\alpha$ is normal, $n$ integer $\Rightarrow n\alpha$ and $\alpha/n$ are normal
- Proof: multiplication and division by a constant are $O(1)$-valued automatic relations and composition of automatic relations is automatic
- aligned definition $\Leftrightarrow$ non-aligned definition
- Proof: the criterion can be proven for non-aligned definition in a similar way
- Agafonov: automatic selection rule preserves normality
What do we get as byproducts?

- Hall: \( \alpha \) is normal, \( n \) integer \( \Rightarrow \) \( n\alpha \) and \( \alpha/n \) are normal
- Proof: multiplication and division by a constant are \( O(1) \)-valued automatic relations and composition of automatic relations is automatic
- aligned definition \( \iff \) non-aligned definition
- Proof: the criterion can be proven for non-aligned definition in a similar way
- Agafonov: automatic selection rule preserves normality
- Proof: if a selected subsequence is compressible, this compression can be used together with uncompressed description of the remaining terms (some care needed)


alexander.shen@lirmm.fr, www.lirmm.fr/~ashen

Normal numbers and automatic complexity
What do we get as byproducts?

- Hall: $\alpha$ is normal, $n$ integer $\Rightarrow n\alpha$ and $\alpha/n$ are normal
- Proof: multiplication and division by a constant are $O(1)$-valued automatic relations and composition of automatic relations is automatic
- aligned definition $\iff$ non-aligned definition
- Proof: the criterion can be proven for non-aligned definition in a similar way
- Agafonov: automatic selection rule preserves normality
- Proof: if a selected subsequence is compressible, this compression can be used together with uncompressed description of the remaining terms (some care needed)
- Piatetski-Shapiro theorem: if no block appear $c$ times more often then they should, the sequence is normal
What do we get as byproducts?

- Hall: $\alpha$ is normal, $n$ integer $\Rightarrow n\alpha$ and $\alpha/n$ are normal
  Proof: multiplication and division by a constant are $O(1)$-valued automatic relations and composition of automatic relations is automatic
  - aligned definition $\iff$ non-aligned definition
  Proof: the criterion can be proven for non-aligned definition in a similar way

- Agafonov: automatic selection rule preserves normality
  Proof: if a selected subsequence is compressible, this compression can be used together with uncompressed description of the remaining terms (some care needed)

- Piatetski-Shapiro theorem: if no block appear $c$ times more often then they should, the sequence is normal