Andrzej Bedychaj
Uniwersytet Jagielloński

Twierdzenie Ornsteina-Weissa i jego zastosowanie do estymacji entropii języka polskiego.

Wstęp

W historii ludzkości różne okresy zyskują przydomki pochodzące od przełomowych odkryć dokonanych w tych epokach. Człowiek zaznał epoki kamienia łupanego, brązu czy pary. Współczesność w której przyszło nam żyć z pełną świadomością może zostać określona epoką informacji. Dyscypliną matematyczną najmocniej związaną z problemami tego okresu jest teoria informacji. Dziedzina ta bazuje na rachunku prawdopodobieństwa i statystyce, a głównym problemem badanym przez matematyków, którzy się nią zajmują jest entropia, czyli miara informacji przypadająca na daną wiadomość.

Angielska strona Wikipedii poświęcona entropii przedstawia ciekawe przykłady, które ułatwiają zrozumienie tego pojęcia. Najbliższym treści poniższej pracy jest ten o entropii języka angielskiego. Język naturalny jakim jest język polski (lub właśnie angielski) często charakteryzuje się niską entropią, gdyż jest w pewien sposób przewidywalny. Oznacza to, iż niektóre związki liter są bardziej prawdopodobne niż inne. Ta sama miara jest zdecydowanie większa w przypadku całkowicie losowego ciągu znaków, gdyż nie sposób przewidzieć co nastąpi po znanej nam już części wiadomości.

W poniższej pracy skoncentrowałem się na algorytmie liczenia entropii zaproponowanym przez Donalda Samuela Ornsteina i Benjamina Weissa w kontekście języka polskiego. Według mojej najlepszej wiedzy badania entropii prowadzone w oparciu o zaproponowany algorytm nie były do tej pory prowadzone w analizie tekstów w języku polskim. Analizie poddane zostały dwa rodzaje literackie tj. liryka i epika. Zastosowana metoda z powodzeniem może zostać użyta do dalszych analiz tekstów w języku polskim, jak również innych języków naturalnych.

Początek pracy zawiera teoretyczne wprowadzenie do przedstawionych w dalszej jej części badań. W pierwszym rozdziale przedstawione zostają podstawowe definicje teorii informacji. Drugi rozdział traktuje o entropii w układach zachowujących miarę. Kluczowymi zagadnieniami poruszonymi w tej sekcji są twierdzenia Kołomogorowa-Sinaja i Shanonna-McMillana-Breimana. W kolejnym rozdziale omówiona zostaje koncepcja kompresji danych oraz kluczowe dla całej pracy twierdzenie Ornsteina-Weissa. Jego implementacja oraz zastosowanie w kontekście analizy polskich tekstów przedstawione zostaje w rozdziale czwartym. Kody źródłowe wszelkich zastosowanych programów znajdują się w aneksie na końcu pracy.

Preliminaria

Jednym z podstawowych obiektów jakie bada teoria ergodyczna są układy zachowujące miarę. Oznacza to dla nas przestrzeń $(X, \mathcal{B}, \mu)$ wraz z przekształceniem zachowującym miarę $T\colon X \to X$.

Niech X będzie zbiorem, którego elementy nazywamy punktami. Niech $\mathcal{B}$ będzie σ-algebrą podzbiorów X, której elementami są zbiory mierzalne. Załóżmy ponadto, że $T^{-1}(B) \in \mathcal{B}$ dla każdego $B \in \mathcal{B}$, a μ jest miarą proba-

bilistyczną określoną na $\mathcal{B}$ zachowaną przez odwzorowanie T, co oznacza, że $\mu\left(T^{-1}(B)\right) = \mu(B)$ dla każdego $B \in \mathcal{B}$. Innymi słowy, miara μ jest niezmiennicza dla T. Więcej informacji można znaleźć w [1] i [2].

1. Podziały zbiorów, entropia i funkcja informacji

W tej części przyjrzymy się entropii, w kontekście odwzorowań zachowujących miarę. Rozdział został opracowany w oparciu o niedokończoną jeszcze pracę *Entropy in dynamics* [2]. Rozpoczniemy od podstawowych definicji teorii informacji.

1.1. Entropia

Definicja 1. *Co najwyżej przeliczalną rodzinę zbiorów $\xi = \{A_1, A_2, A_3, ...\}$ nazywamy rozbiciem zbioru X, gdy spełnione są równocześnie następujące warunki:*

1. $\emptyset \notin \xi$,
2. $\bigcup_{A_i \in \xi} = X$,
3. *jeżeli $A, B \in \xi$ i $A \neq B$ to $A \cap B = \emptyset$.*

Z każdym rozbiciem ξ skojarzona jest najmniejsza σ-algebra zawierającą zbiory z ξ oznaczana $\sigma(\xi)$. Elementy rozbicia ξ będziemy nazywać atomami, a oznaczenie $[x]_\xi$ będzie dla nas atomem rozbicia ξ zawierającym punkt x.

Niech ξ i η będą dwoma różnymi rozbiciami. Piszemy $\xi \leqslant \eta$ jeżeli każdy atom rozbicia ξ jest sumą atomów z rozbicia η. Mówimy też, że η jest rozdrobnieniem ξ. Definiujemy również połączenie dla $\xi = \{A_1, A_2, A_3, ...\}$ i $\eta = \{B_1, B_2, B_3, ...\}$ oznaczane $\xi \vee \eta$, które jest rozbiciem na zbiory niepuste postaci $A_i \cap B_j$. Rozbicie ξ nazywamy generującym względem odwzorowania zachowującego miarę T, kiedy podzbiory postaci:

$$A_{i_0} \cap T^{-1} A_{i_1} \cap \ldots \cap T^{-(n-1)} A_{i_{n-1}}$$

generują wszystkie mierzalne podzbiory X.

Niech teraz $\xi = \{A_1, A_2, A_3, ...\}$ będzie rozbiciem. Możemy o nim myśleć jako o ponumerowanej liście możliwych wyników eksperymentu, dla którego prawdopodobieństwo zdarzenia i ma miarę $\mu(A_i)$. Pierwszym krokiem jest teraz powiązanie liczby $H(\xi)$ z miarą niepewności zdarzenia z ξ, lub równoważnie z miarą informacji pozyskanej w wyniku eksperymentu.

Rozważenie dwóch skrajnych przypadków da nam potrzebną intuicję. Po pierwsze, jeżeli dla jednego ze zbiorów A_i $\mu(A_i) = 1$, to nie ma żadnej niepewności co do wyniku eksperymentu więc miara informacji winna być równa $H(\xi) = 0$. Na drugim biegunie zachodzi przypadek w którym ξ składa się z k atomów, a atomy zawierają po jednym elemencie każdy i $\mu(A_i) = \frac{1}{k}$. W tym wypadku nasza funkcja $H(\xi)$ przyjmuje swoje maksimum, bo nie potrafimy nic powiedzieć o wyniku eksperymentu.

Definicja 2. *Entropią rozbicia $\xi = \{A_1, A_2, A_3, ...\}$ nazywamy funkcję:*

$$H_\mu(\xi) = H(\mu(A_1), \mu(A_2), \mu(A_3), \ldots) = -\sum_{i \geqslant 1} \mu(A_i) \log(\mu(A_i)) \in [0, \infty].$$

Umawiamy się, że $0 \log(0) = 0$.

Uwaga. *Będziemy zawsze zakładali, że wszystkie elementy rozważanych rozbić są dodatniej miary. Jest to zgodne z tradycją pomijania zbiorów miary zero.*

Dodatkowo definiujemy entropię warunkową:

Definicja 3. *Niech $\xi = \{A_1, A_2, A_3, ...\}$ i $\eta = \{B_1, B_2, B_3, ...\}$ będą rozbiciami. Definiujemy entropię warunkową eksperymentu ξ pod warunkiem znajomości wyniku eksperymentu η jako:*

$$H_\mu(\xi|\eta) = \sum_{j=1}^{\infty} \mu(B_j) H\left(\frac{\mu(A_1 \cap B_j)}{\mu(B_j)}, \frac{\mu(A_2 \cap B_j)}{\mu(B_j)}, \frac{\mu(A_3 \cap B_j)}{\mu(B_j)}, \ldots\right).$$

Z powyższych definicji wynika, że entropia nie zależy bezpośrednio od rozbicia ξ, lecz od wektora $(\mu(A_1), \mu(A_2), \mu(A_3), ...)$. Jeżeli nasze rozważania zawęzimy do rozbić skończonych, to funkcję H będziemy rozważać na sympleksach $\Delta_k = \{(p_1, p_2, p_3, ..., p_k) : p_i \geqslant 0, \sum p_i = 1\}$, więc $H(p_1, ..., p_k) = -\sum_{i=1}^{k} p_i \log p_i$

Własności 1. *Funkcja H ma następujące własności:*

1. *$H(p_1, p_2, ..., p_k) \geqslant 0$ i $H(p_1, p_2, ..., p_k) = 0$ wtedy i tylko wtedy, gdy istnieje takie $i \in \{1, \ldots, k\}$, że $p_i = 1$,*
2. *$H(p_1, p_2, ..., p_k, 0) = H(p_1, p_2, ..., p_k)$,*
3. *Dla każdego $k \geqslant 1$, funkcja H ograniczona do Δ_k jest ciągła, nie zależy od permutacji zmiennych $p_1, \ldots, p_k$ i osiąga maksimum równe $\log k$ w punkcie $\left(\frac{1}{k}, ..., \frac{1}{k}\right)$.*

Dowód. (1) i (2) wynikają wprost z definicji entropii.

Dla dowodu (3) odnotujmy najpierw pewien fakt:

Lemat 1. *Jeżeli ξ jest rozbiciem na k atomów, to*

$$H_\mu(\xi) \leqslant \log k.$$

Równość w powyższym zachodzi wtedy i tylko wtedy gdy $\mu(A_i) = \frac{1}{k}$ dla każdego $A_i \in \xi$.

Zdefiniujmy funkcję:

$$\phi(x) = \begin{cases} 0, & gdy\, x = 0, \\ x \log x, & gdy\, x > 0. \end{cases}$$

Jest ona ciągła i ma minimum w punkcie $\frac{1}{e}$. Ponieważ $\phi'' = \frac{1}{x} > 0$ i $(x \mapsto -\log x)'' = \frac{1}{x^2} > 0$ na $(0, \infty)$, to obie funkcje są silnie wypukłe na $(0, \infty)$.

Zauważmy teraz, że dla pewnego $A_i \in \xi$ takiego że $0 < \mu(A_i) \neq \frac{1}{k}$ mamy

$$\frac{1}{k} \log \frac{1}{k} = \phi\left(\frac{1}{k}\right) = \phi\left(\sum_{A_i \in \xi} \frac{1}{k} \mu(A_i)\right) < \sum_{A_i \in \xi} \frac{1}{k} \phi(\mu(A_i)).$$

Wynika stąd, że

$$-\sum_{A_i \in \xi} \mu(A_i) \log \mu(A_i) < \log k.$$

Jeżeli teraz $\mu(A_i) = \frac{1}{k}$ dla każdego $A_i \in \xi$, to $H_\mu(\xi) = \log k$. □

1.2. Funkcja informacji

Wprowadzimy teraz kolejną funkcję powiązaną z rozbiciem ξ i entropią $H_\mu(\xi)$.

Definicja 4. *Funkcją informacji rozbicia ξ nazywamy funkcję $I_\mu(\xi) : X \mapsto \mathbb{R}$ daną wzorem:*

$$I_\mu(\xi)(x) = -\log \mu([x]_\xi),$$

gdzie $[x]_\xi \in \xi$ jest elementem rozbicia dla którego $x \in [x]_\xi$.

Dodatkowo możemy zdefiniować informację warunkową.

Definicja 5. *Niech ξ i η będą rozbiciami. Informacją warunkową nazywamy funkcję $I_\mu(\xi|\eta) : X \mapsto \mathbb{R}$ daną wzorem:*

$$I_\mu(\xi|\eta) = -\log \frac{\mu([x]_{\xi \vee \eta})}{\mu([x]_\eta)}.$$

Własności 2. *Niech ξ i η będą skończonymi rozbiciami przestrzeni $(X, \mathcal{B}, \mu)$. Wtedy:*

1. *$H_\mu(\xi) = \int I_\mu(\xi)\, d\mu$ oraz $H_\mu(\xi|\eta) = \int I_\mu(\xi|\eta)\, d\mu$,*

2. $I_\mu(\xi \vee \eta) = I_\mu(\eta|\xi) + I_\mu(\xi)$ *i* $H_\mu(\xi \vee \eta) = H_\mu(\eta|\xi) + H_\mu(\xi)$*; dodatkowo jeżeli* $H_\mu(\xi) < +\infty$*, mamy* $H_\mu(\xi \vee \eta) - H_\mu(\xi) = H_\mu(\eta|\xi)$*,*
3. $H_\mu(\xi \vee \eta) \leqslant H_\mu(\xi) + H_\mu(\eta)$*,*
4. jeżeli η *i* ζ *mają skończoną entropię, to* $H_\mu(\xi|\eta \vee \zeta) \leqslant H_\mu(\xi|\zeta)$*.*

Dowód. Niech $\xi = \{A_1, A_2, ...\}$ i $\eta = \{B_1, B_2, ...\}$, wtedy:

$$\begin{aligned}\int I_\mu(\xi|\eta)\, d\mu &= -\sum_{A_i\in\xi; B_j\in\eta} \left(\log \frac{\mu(A_i \cap B_j)}{\mu(B_j)}\right) \mu(A_i \cap B_j) \\ &= -\sum_{B_j\in\eta} \mu(B_j) \sum_{A_i\in\xi} \frac{\mu(A_i \cap B_j)}{\mu(B_j)} \log \frac{\mu(A_i \cap B_j)}{\mu(B_j)} \\ &= \sum_{B_j\in\eta} \mu(B_j)\, H\left(\frac{\mu(A_1 \cap B_j)}{\mu(B_j)}, \frac{\mu(A_2 \cap B_j)}{\mu(B_j)}, ...\right) \\ &= H_\mu(\xi|\eta),\end{aligned}$$

co kończy dowód (1).

Teraz:

$$\begin{aligned}I_\mu(\xi \vee \eta)(x) &= -\log \mu([x]_\xi \cap [x]_\eta) \\ &= -\log \mu([x]_\xi) - \log \frac{\mu([x]_\xi \cap [x]_\eta)}{\mu([x]_\xi)} \\ &= I_\mu(\xi)(x) + I_\mu(\eta|\xi)(x),\end{aligned}$$

co po scałkowaniu kończy dowód (2).

Korzystając z wypukłości funkcji ϕ otrzymujemy:

$$\begin{aligned}H_\mu(\eta|\xi) &= -\sum_{A_i\in\xi} \sum_{B_j\in\eta} \mu(A_i)\, \phi\left(\frac{\mu(A_i \cap B_j)}{\mu(A_i)}\right) \\ &= -\sum_{B_j\in\eta} \sum_{A_i\in\xi} \mu(A_i)\, \phi\left(\frac{\mu(A_i \cap B_j)}{\mu(A_i)}\right) \\ &\leqslant -\sum_{B_j\in\eta} \phi(\mu(B_j)) = H_\mu(\eta),\end{aligned}$$

co kończy dowód (3).

Korzystając z (2) oraz (3) otrzymujemy:

$$\begin{aligned}H_\mu(\xi|\eta \vee \zeta) &= H_\mu(\xi \vee \eta \vee \zeta) - H_\mu(\eta \vee \zeta) \\ &= H_\mu(\zeta) + H_\mu(\xi \vee \eta|\zeta) - H_\mu(\zeta) - H_\mu(\eta|\zeta) \\ &= \sum_{C\in\zeta} \mu(C) \left(H_{\mu(C)^{-1}\mu|_C}(\xi \vee \eta) - H_{\mu(C)^{-1}\mu|_C}(\eta)\right) \\ &= \sum_{C\in\zeta} \mu(C)\, H_{\mu(C)^{-1}\mu|_C}(\xi|\eta) \leqslant \sum_{C\in\zeta} \mu(C)\, H_{\mu(C)^{-1}\mu|_C}(\xi) \\ &= H_\mu(\xi|\zeta),\end{aligned}$$

co daje (4) i kończy dowód całości. □

Definicja 6. *Dwa rozbicia* ξ *i* η *są niezależne (co zapisujemy* $\xi \perp \eta$ *), jeżeli* $\mu(A \cap B) = \mu(A)\,\mu(B)$ *dla każdego* $A \in \xi$ *i* $B \in \eta$*.*

Lemat 2. *Rozbicia* ξ *i* η *o skończonej entropii są niezależne wtedy i tylko wtedy, gdy* $H_\mu(\xi \vee \eta) = H_\mu(\xi) + H_\mu(\eta)$*.*

Dowód. Z (2) i założenia o skończoności entropii dla ξ i η mamy:

$$H_\mu(\xi \vee \eta) = H_\mu(\eta|\xi) + H_\mu(\xi).$$

Zauważmy teraz:

$$\begin{aligned} H_\mu(\eta|\xi) &= -\sum_{A_i\in\xi}\sum_{B_j\in\eta}\mu(A_i)\,\phi\left(\frac{\mu(A_i\cap B_j)}{\mu(A_i)}\right) \\ &= -\sum_{A_i\in\xi}\sum_{B_j\in\eta}\mu(A_i)\,\phi(\mu(B_j)) \\ &= -\sum_{A_i\in\xi}\mu(A_i)\sum_{B_j\in\eta}\mu(B_j)\log(\mu(B_j)) \\ &= \sum_{A_i\in\xi}\mu(A_i)\,H_\mu(\eta) = H_\mu(\eta)\sum_{A_i\in\xi}\mu(A_i) \\ &= H_\mu(\eta), \end{aligned}$$

co kończy dowód w jedną stronę.

Załóżmy teraz, że zachodzi $H_\mu(\xi\vee\eta) = H_\mu(\eta)+H_\mu(\xi)$. Z własności (3) i (2), mamy:

$$\begin{gathered} H_\mu(\eta|\xi) + H_\mu(\xi) \leqslant H_\mu(\eta) + H_\mu(\xi), \\ H_\mu(\eta|\xi) \leqslant H_\mu(\eta). \end{gathered}$$

Wynika stąd niezależność rozbicia η od rozbicia ξ. □

2. Entropia w układach zachowujących miarę

Podobnie jak w poprzedniej sekcji, teoria omówiona w tym rozdziale została zaczerpnięta z pracy *Entropy in dynamics* [2]. Dowód lematu Fekete podany został za *Problems and Theorems in Analysis I* [8]. Podrozdziały o przesunięciach i twierdzeniu Shannona-McMillana-Breimana opracowane zostały na podstawie *Computational Ergodic Theory* [3] i pracy *A Note on the Ergodic Theorem of Information Theory* [6].

2.1. Entropia odwzorowania

Aby wykorzystać entropię do badania układów zachowujących miarę, musimy zrozumieć jak zmienia się ona pod wpływem przekształceń działających w takich układach.

Lemat 3. *Jeżeli ξ jest rozbiciem, to przez $T^{-1}\xi$ rozumiemy rozbicie $\{T^{-1}A\colon A\in\xi\}$. Niech $(X,\mathcal{B},\mu,T)$ będzie układem zachowującym miarę i niech ξ i η będą rozbiciami. Wtedy*

$$H_\mu(\xi|\eta) = H_\mu\left(T^{-1}\xi|T^{-1}\eta\right)$$

oraz

$$I_\mu(\xi|\eta)\circ T = I_\mu\left(T^{-1}\xi|T^{-1}\eta\right).$$

Dowód. Najpierw zauważmy, że dla każdego x mamy:

$$Tx\in[Tx]_\eta \Rightarrow x\in T^{-1}[Tx]_\eta.$$

Z drugiej strony:

$$T^{-1}[Tx]_\eta \in T^{-1}\eta.$$

Dlatego:

$$x\in[x]_{T^{-1}\eta}.$$

Ostatecznie:

$$\begin{aligned} I_\mu(\xi|\eta)(Tx) &= -\log \frac{\mu([Tx]_\xi \cap [Tx]_\eta)}{\mu([Tx]_\eta)} \\ &= -\log \frac{\mu\left(T^{-1}[Tx]_\xi \cap T^{-1}[Tx]_\eta\right)}{\mu\left(T^{-1}[Tx]_\eta\right)} \\ &= -\log \frac{\mu\left([x]_{T^{-1}\xi} \cap [x]_{T^{-1}\eta}\right)}{\mu\left([x]_{T^{-1}\eta}\right)} \\ &= I_\mu\left(T^{-1}\xi|T^{-1}\eta\right)(x). \end{aligned}$$

□

Lemat 4 (Fekete)**.** *Niech (a_n) będzie ciągiem elementów z $\mathbb{R}\cup\{-\infty\}$ posiadających własność subaddytywności, czyli:*

$$a_{m+n} \leqslant a_m + a_n$$

dla każdego $m, n \geqslant 1$. Wtedy ciąg $\left(\frac{1}{n}a_n\right)$ jest zbieżny (możliwe, że do $-\infty$), oraz:

$$\lim_{n\to\infty} \frac{1}{n}a_n = \inf_{n\in\mathbb{N}} \frac{1}{n}a_n.$$

Dowód. Jeżeli $a_n = -\infty$ dla jakiegoś $n \geqslant 1$ wtedy z własności subaddytywności $a_{n+k} = -\infty$ dla każdego $k \geqslant 1$, więc twierdzenie zachodzi (z granicą równą $-\infty$).

Załóżmy teraz, że $a > -\infty$. Twierdzenie to będziemy stosować tylko gdy $a_n \geqslant 0$ dla każdego $n \geqslant 1$, a co za tym idzie $a \geqslant 0$. Ustalmy dowolny $\varepsilon > 0$. Niech m będzie takie, że $\frac{a_m}{m} < a + \varepsilon$. Teraz korzystając z subaddytywności i faktu że każdą liczbę $n \geqslant m$ można przedstawić jako $mk + r$ dla pewnego $k > 1$ i $0 \leqslant r < m$. Wówczas:

$$\begin{aligned} a_n = a_{mk+r} &\leqslant a_m + ... + a_m + a_r = ka_m + a_r, \\ \frac{a_n}{n} = \frac{a_{mk+r}}{mk+r} &\leqslant \frac{ka_m + a_r}{mk+r} = \frac{a_m}{m}\frac{km}{mk+r} + \frac{a_r}{n}, \\ a \leqslant \frac{a_n}{n} &< (a+\varepsilon)\frac{km}{mk+r} + \frac{a_r}{n}. \end{aligned}$$

Przechodząc z n do ∞ otrzymujemy tezę. □

Niech T będzie odwzorowaniem zachowującym miarę na $(X, \mathcal{B}, \mu)$, niech teraz ξ będzie podziałem X o skończonej entropii. Podział ξ możemy interpretować jako eksperyment o co najwyżej przeliczalnej liczbie wyników, które są reprezentowane przez atomy tego podziału. Entropia $H_\mu(\xi)$ mierzy średnią wartość informacji skupionej w punktach przestrzeni X (jako wyniki eksperymentów). Wielkość ta może być wyrażona za pomocą dowolnej liczby nieujemnej (także przez nieskończoność) i oczywiście jest niezależna od T (zależy tylko od ξ).

Odwzorowanie $T\colon X \to X$ możemy powiązać z ewolucją w czasie. Podział $T^{-1}\xi$ oznacza od teraz ten sam eksperyment, powtórzony w kolejnej iteracji. W tym sensie $\xi \vee T^{-1}\xi$ reprezentuje łączny wynik eksperymentu w danym momencie i jednostkę czasu później. Oznacza to, że entropia $H_\mu\left(\xi \vee T^{-1}\xi\right)$ mierzy średnią wartość informacji uzyskanej z wyników dwóch kolejnych prób. Jeżeli teraz kolejne próby - $T^{-k}\xi$ - będą od siebie niezależne (w sensie Lematu 2), to dla każdego $n \geqslant 1$ otrzymamy:

$$\begin{aligned} H_\mu\left(\xi \vee T^{-1}\xi \vee ... \vee T^{-(n-1)}\xi\right) &= H_\mu(\xi) + H_\mu\left(T^{-1}\xi\right) + ... + H_\mu\left(T^{-(n-1)}\xi\right) \\ &= nH_\mu(\xi). \end{aligned}$$

A w ogólności (tzn. bez założenia niezależności):

$$\begin{aligned} H_\mu\left(\xi \vee T^{-1}\xi \vee ... \vee T^{-(n-1)}\xi\right) &\leqslant H_\mu(\xi) + H_\mu\left(T^{-1}\xi\right) + ... + H_\mu\left(T^{-(n-1)}\xi\right) \\ &= nH_\mu(\xi). \end{aligned}$$

Możemy teraz zdefiniować ciąg (a_n) jako:

$$a_n = H_\mu\left(\xi \vee T^{-1}\xi \vee ... \vee T^{-(n-1)}\xi\right),$$

i wykazać, że posiada on własność subaddytywności (Lemat 4), a to pozwala sformułować poniższą definicję.

Definicja 7. *Niech $(X, \mathcal{B}, \mu, T)$ będzie układem zachowującym miarę, a ξ podziałem X o skończonej entropii. Entropią T przy założeniu rozbicia ξ nazywamy funkcję:*

$$h_\mu(T, \xi) = \lim_{n\to\infty} \frac{1}{n} H_\mu\left(\bigvee_{i=0}^{n-1} T^{-i}\xi\right) = \inf_{n\geqslant 1} \frac{1}{n} H_\mu\left(\bigvee_{i=0}^{n-1} T^{-i}\xi\right).$$

Entropią T nazywamy wtedy funkcję:

$$h_\mu(T) = \sup_{\{\xi\ -\ rozbicie:\ H_\mu(\xi)<\infty\}} h_\mu(T, \xi).$$

Poniższy przykład ilustruje użycie powyższej definicji.

Przykład 2.1. Niech $X_{(2)} = \{0,1\}^{\mathbb{Z}}$ z miarą Bernulliego $\left(\frac{1}{2}, \frac{1}{2}\right)$ oznaczaną dalej $\mu_{(2)}$. Jest to miara niezmiennicza dla przesunięcia $\sigma_{(2)}$. Rozważmy rozbicie stacjonarne:

$$\xi = \{[0]_0, [1]_0\},$$

gdzie $[0]_0 = \{x \in X_{(2)} \mid x_0 = 0\}$ i $[1]_0 = \{x \in X_{(2)} \mid x_0 = 1\}$ są zbiorami cylindrycznymi. Rozbicie $\sigma_{(2)}^{-k}(\xi)$ jest niezależne od $\bigvee_{j=0}^{k-1} \sigma_{(2)}^{-j}(\xi)$ dla każdego $k \geqslant 1$, więc:

$$h_{\mu_{(2)}}\left(\sigma_{(2)}, \xi\right) = \lim_{n\to\infty} \frac{1}{n} H_{\mu_{(2)}}\left(\bigvee_{j=0}^{n-1} \sigma_{(2)}^{-j}(\xi)\right) = \lim_{n\to\infty} \frac{1}{n} \log(2^n) = \log 2.$$

Nie jesteśmy jednak w stanie policzyć $h_{\mu_{(2)}}\left(\sigma_{(2)}\right)$ z poprzedniego przykładu, gdyż ta funkcja zdefiniowana jest jako supremum po wszystkich rozbiciach ξ o entropii skończonej. Aby tego dokonać musimy użyć poniższych właściwości funkcji entropii odwzorowania.

Własności 3. *Niech $(X, \mathcal{B}, \mu, T)$ będzie układem zachowującym miarę, niech ξ i η będą przeliczalnymi rozbiciami X o skończonej entropii. Wtedy:*

1. $h_\mu(T, \xi) \leqslant H_\mu(\xi)$,
2. $h_\mu(T, \xi \vee \eta) \leqslant h_\mu(T, \xi) + h_\mu(T, \eta)$,
3. $h_\mu(T, \eta) \leqslant h_\mu(T, \xi) + H_\mu(\eta|\xi)$,
4. $h_\mu(T, \xi) = h_\mu\left(T, \bigvee_{j=0}^{k} T^{-j}(\xi)\right)$ *dla każdego* $k \geqslant 1$,
5. *Jeżeli T jest odwracalna, to* $h_\mu(T, \xi) = h_\mu\left(T^{-1}, \xi\right) = h_\mu\left(T, \bigvee_{j=-k}^{k} T^{-j}(\xi)\right)$ *dla każdego* $k \geqslant 1$.

Dowód. (1) Dla każdego $n \geqslant 1$ mamy:

$$\frac{1}{n} H_\mu\left(\bigvee_{j=0}^{n-1} T^{-j}(\xi)\right) \leqslant \frac{1}{n} \sum_{j=0}^{n-1} H_\mu\left(T^{-j}(\xi)\right) = \frac{1}{n} \sum_{j=0}^{n-1} H_\mu(\xi) = H_\mu(\xi).$$

(2) Dla każdego $n \geqslant 1$ mamy:

$$\begin{aligned}\frac{1}{n} H_\mu\left(\bigvee_{j=0}^{n-1} T^{-j}(\xi \vee \eta)\right) &= \frac{1}{n} H_\mu\left(\bigvee_{j=0}^{n-1} T^{-j}(\xi)\right) \\ &\quad + \frac{1}{n} H_\mu\left(\bigvee_{j=0}^{n-1} T^{-j}(\eta) \,|\, \bigvee_{j=0}^{n-1} T^{-j}(\xi)\right) \\ &\leqslant \frac{1}{n} H_\mu\left(\bigvee_{j=0}^{n-1} T^{-j}(\xi)\right) + \frac{1}{n} H_\mu\left(\bigvee_{j=0}^{n-1} T^{-j}(\eta)\right).\end{aligned}$$

(3) Dla każdego $n \geqslant 1$ mamy:

$$\begin{aligned}
h_\mu(T,\eta) &= \lim_{n\to\infty}\frac{1}{n}H_\mu\left(\bigvee_{j=0}^{n-1}T^{-j}(\eta)\right)\\
&\leqslant \lim_{n\to\infty}\frac{1}{n}H_\mu\left(\bigvee_{j=0}^{n-1}T^{-j}(\xi\vee\eta)\right)\\
&= h_\mu(T,\xi\vee\eta)\\
&= \lim_{n\to\infty}\left(\frac{1}{n}H_\mu\left(\bigvee_{j=0}^{n-1}T^{-j}(\xi)\right)+\frac{1}{n}H_\mu\left(\bigvee_{j=0}^{n-1}T^{-j}(\eta)\,|\bigvee_{j=0}^{n-1}T^{-j}(\xi)\right)\right)\\
&\leqslant h_\mu(T,\xi)+\lim_{n\to\infty}\frac{1}{n}\sum_{j=0}^{n-1}H_\mu\left(T^{-j}(\eta)\,|T^{-j}(\xi)\right)\\
&\overset{3}{=} h_\mu(T,\xi)+H_\mu(\eta|\xi).
\end{aligned}$$

(4) Dla każdego $k \geqslant 1$ mamy:

$$\begin{aligned}
h_\mu\left(T,\bigvee_{j=0}^{k}T^{-j}(\xi)\right) &= \lim_{n\to\infty}\frac{1}{n}H_\mu\left(\bigvee_{j=0}^{n-1}T^{-j}\left(\bigvee_{i=0}^{k}T^{-i}(\xi)\right)\right)\\
&= \lim_{n\to\infty}\frac{1}{n}H_\mu\left(\bigvee_{j=0}^{k+n-1}T^{-j}(\xi)\right)\\
&= \lim_{n\to\infty}\frac{k+n-1}{n}\left(\frac{1}{k+n-1}H_\mu\left(\bigvee_{j=0}^{k+n-1}T^{-j}(\xi)\right)\right)\\
&= h_\mu(T,\xi).
\end{aligned}$$

(5) Dla każdego $n \geqslant 1$ mamy:

$$\begin{aligned}
h_\mu\left(T^{-1},\xi\right) &= \lim_{n\to\infty}\frac{1}{n}H_\mu\left(\bigvee_{j=0}^{n-1}T^{-j}(\xi)\right)\\
&= \lim_{n\to\infty}\frac{1}{n}H_\mu\left(T^{-(n-1)}\bigvee_{j=0}^{n-1}T^{j}(\xi)\right)\\
&= \lim_{n\to\infty}\frac{1}{n}H_\mu\left(\bigvee_{j=0}^{n-1}T^{-j}(\xi)\right) = h_\mu(T,\xi).
\end{aligned}$$

Drugą równość dowodzimy analogicznie do (4). Dla każdego $k \geqslant 1$ mamy:

$$\begin{aligned}
h_\mu\left(T,\bigvee_{j=-k}^{k}T^{-j}(\xi)\right) &= \lim_{n\to\infty}\frac{1}{n}H_\mu\left(\bigvee_{j=0}^{n-1}T^{-j}\left(\bigvee_{i=-k}^{k}T^{-i}(\xi)\right)\right)\\
&= \lim_{n\to\infty}\frac{1}{n}H_\mu\left(\bigvee_{j=-k}^{k+n-1}T^{-j}(\xi)\right)\\
&= \lim_{n\to\infty}\frac{2k+n-1}{n}\left(\frac{1}{2k+n-1}H_\mu\left(\bigvee_{j=-k}^{k+n-1}T^{-j}(\xi)\right)\right)\\
&= h_\mu(T,\xi).
\end{aligned}$$

□

Lemat 5. *Niech $(X, \mathcal{B}, \mu, T)$ będzie układem zachowującym miarę. Wtedy:*
1. $h_\mu\left(T^k\right) = kh_\mu(T)$ *dla każdego* $k \geqslant 1$,
2. $h_\mu(T) = h_\mu\left(T^{-1}\right)$ *jeżeli* T *jest odwracalne.*

Dowód. (1) Dla każdego podziału ξ o skończonej entropii, mamy:

$$\lim_{n\to\infty} \frac{1}{n} H_\mu\left(\bigvee_{j=0}^{n-1} T^{-kj}\left(\bigvee_{i=0}^{k-1} T^{-i}(\xi)\right)\right) = \lim_{n\to\infty} \frac{k}{nk} H_\mu\left(\bigvee_{j=0}^{kn-1} T^{-j}(\xi)\right) = kh_\mu(T,\xi).$$

Z tego wynika, że:

$$h_\mu\left(T^k, \bigvee_{j=0}^{k-1} T^{-j}(\xi)\right) = kh_\mu(T,\xi),$$

więc $h_\mu\left(T^k\right) \geqslant kh_\mu(T)$,

W drugą stronę, dla innego podziału η:

$$h_\mu\left(T^k, \eta\right) \leqslant h_\mu\left(T^k, \bigvee_{j=0}^{k-1} T^{-j}(\eta)\right) = kh_\mu(T,\eta),$$

więc $h_\mu\left(T^k\right) \leqslant kh_\mu(T)$ z lematu 4.

Punkt (2) wynika wprost z własności 5 i poprzedniego punktu. □

Lemat 6. *Entropia może zostać policzona przy użyciu skończonych podziałów, to znaczy:*

$$\sup_{\eta \text{ skończone}} h_\mu(T,\eta) = \sup_{\xi:\, H_\mu(\xi)<\infty} h_\mu(T,\xi).$$

Dowód. Skończone podziały mają skończoną entropię, stąd zawsze mamy:

$$\sup_{\eta \text{ skończone}} h_\mu(T,\eta) \leqslant \sup_{\xi:\, H_\mu(\xi)<\infty} h_\mu(T,\xi).$$

Dla odwrotnej nierówności załóżmy, że ξ jest podziałem o własności $H_\mu(\xi) < \infty$. Twierdzimy, że dla dowolnego $\varepsilon > 0$ możemy znaleźć takie η skończone i mierzalne względem $\sigma(\xi)$, że $H(\xi|\eta) < \varepsilon$. Aby to wykazać zapiszmy ξ jako $\{A_1, A_2, ...\}$ oraz zdefiniujmy $\eta = \{A_1, A_2, ..., A_N, B_N = X \setminus \bigcup_{i=1}^N A_i\}$ tak aby $\mu(B_N) \overset{N\to\infty}{\longrightarrow} 0$. Wtedy:

$$\begin{aligned}
H(\xi|\eta) &= \mu(B_N)\, H\left(\frac{\mu(A_{N+1})}{B_N}, \frac{\mu(A_{N+2})}{B_N}, ...\right) \\
&= -\sum_{j=N+1}^{\infty} \mu(A_j) \log\left(\frac{\mu(A_j)}{\mu(B_N)}\right) \\
&= -\sum_{j=N+1}^{\infty} \mu(A_j) \log(\mu(A_j)) + \mu(B_N)\log(\mu(B_N)) \\
&= -\sum_{j=N+1}^{\infty} \mu(A_j) \log(\mu(A_j)) + \phi(B_N).
\end{aligned}$$

Z założenia $H_\mu(\xi) < \infty$ i ciągłości ϕ, możliwym jest wybranie N dużego na tyle, że $H_\mu(\xi|\eta) < \varepsilon$. A z tego i własności 3 wynika:

$$h_\mu(T,\xi) \leqslant h_\mu(T,\eta) + \varepsilon.$$

□

Definicja 8. $(Y, \mathcal{B}_Y, \nu, S)$ *nazywamy faktorem* $(X, \mathcal{B}_X, \mu, T)$ *jeżeli istnieje odwzorowane zachowujące miarę* $\phi\colon X \to Y$, *takie że* $\phi(Tx) = S(\phi x)$ *dla* μ*-prawie każdego* $x \in X$.

Twierdzenie 1. *Jeżeli* $(Y, \mathcal{B}_Y, \nu, S)$ *jest faktorem* $(X, \mathcal{B}_X, \mu, T)$*, to:*

$$h_\nu(S) \leqslant h_\mu(T).$$

W szczególności entropia jest niezmiennikiem mierzalnego izomorfizmu.

Dowód. Niech $\phi\colon X \to Y$ będzie odwzorowaniem z definicji faktora. Wtedy dowolny podział ξ w Y definiuje jednoznacznie podział $\phi^{-1}(\xi)$ w X, a skoro ϕ zachowuje miarę, to:

$$H_\nu(\xi) = H_\mu\left(\phi^{-1}(\xi)\right).$$

To z kolei implikuje $h_\mu\left(T, \phi^{-1}(\xi)\right) = h_\nu(S, \xi)$, a w konsekwencji tezę twierdzenia. □

2.2. Twierdzenie Kołomogorowa-Sinaja

Definicja entropii dla transformacji zachowującej miarę zawiera suprema po zbiorze wszystkich skończonych podziałów. Aby policzyć entropię, łatwiej jest operować na jednym (konkretnym) podziale. Następujący rezultat - Twierdzenie Kołomogorowa-Sinaja - podaje warunek wystarczający dla tego ułatwienia.

Lemat 7. *Niech* $(X, \mathcal{B}, \mu, T)$ *będzie układem zachowującym miarę,* ξ *podziałem spełniającym* $\bigvee_{j=0}^{\infty} T^{-j}(\xi) = \mathcal{B}$*, a* η *podziałem* X *o skończonej entropii. Wtedy:*

$$H_\mu\left(\eta | \bigvee_{i=0}^{n} T^{-i}(\xi)\right) \overset{n\to\infty}{\longrightarrow} 0.$$

Dowód. Załóżmy najpierw, że η jest skończonym podziałem. Z założenia $\bigvee_{j=0}^{n} T^{-j}(\xi) = \mathcal{B}$ dla $n = 1, 2, ...$, stwierdzamy że dla dowolnego $\delta > 0$ i $B \in \mathcal{B}$, istnieje zbiór $A \in \sigma\left(\bigvee_{i=0}^{n} T^{-i}(\xi)\right)$ dla pewnego $n \geqslant 1$ taki że $\mu(A \triangle B) < \delta$.

Stosując powyższe rozumowanie dla każdego z elementów podziału $\eta = \{B_1, B_2, \ldots, B_m\}$ możemy znaleźć jedno n o takiej własności, że $A_i' \in \sigma\left(\bigvee_{i=0}^{n} T^{-i}(\xi)\right)$ oraz $\mu(A_i' \triangle B_i) < \frac{\delta}{m}$ dla $i = 1, 2, ..., m$ i $\bigcup_{j=1}^{m} A_j' = X$. Zapiszmy:

$$A_1 = A_1', A_2 = A_2' \setminus A_1', A_3 = A_3' \setminus (A_1' \cup A_2'), ..., A_m = A_m' \setminus \bigcup_{j=1}^{m-1} A_j'$$

i zauważmy, że:

$$\begin{aligned}
\mu(A_i \triangle B_i) &= \mu(A_i \setminus B_i) + \mu(B_i \setminus A_i) \\
&\leqslant \mu(A_i' \setminus B_i) + \mu(B_i \setminus A_i') + \mu\left(B_i \cap \bigcup_{j=1}^{i-1} A_j'\right) \\
&\leqslant \frac{\delta}{m} + \sum_{j=1}^{i-1} \mu\left(A_j' \setminus B_j\right) \leqslant \delta.
\end{aligned}$$

Niech teraz $\zeta = \{A_1, A_2, ..., A_m\}$, tak aby:

$$H_\mu\left(\eta | \bigvee_{i=1}^{n} T^{-i}(\xi)\right) \overset{\text{Fekete 4}}{\leqslant} H_\mu(\eta|\zeta)$$

$$= -\sum_{i=1}^{m} \mu(A_i \cap B_i) \log\left(\frac{\mu(A_i \cap B_i)}{\mu(A_i)}\right) - \sum_{i,j=1;\, i\neq j}^{m} \mu(A_i \cap B_j) \log\left(\frac{\mu(A_i \cap B_j)}{\mu(A_i)}\right).$$

Wyrazy pierwszej sumy są bliskie 0 ponieważ $\frac{\mu(A_i \cap B_i)}{\mu(A_i)}$ jest bliskie 1. Wyrazy drugiej sumy są bliskie 0 bo $\mu(A_i \cap B_j)$ jest bliskie 0. Innymi słowy, dla każdego $\varepsilon > 0$ dobierzemy δ wystarczająco małe (a co za tym idzie n wystarczająco duże) aby zapewnić:

$$H_\mu\left(\eta | \bigvee_{i=0}^{n} T^{-i}(\xi)\right) < \varepsilon$$

co kończy dowód pierwszej części.

Załóżmy teraz, że $\eta = \{B_1, ..., B_m, ...\}$ jest przeliczalnym podziałem ze skończoną entropią. Wtedy, dla każdego $m \geqslant 1$ możemy zdefiniować skończony podział:

$$\eta_m = \{B_1, ..., B_m, X \setminus \bigcup_{i=1}^{m} B_i\},\ gdzie\ X \setminus \bigcup_{i=1}^{m} B_i = C_m.$$

Możemy teraz zauważyć, że:

$$\begin{aligned} H_\mu(\eta|\eta_m) &= -\sum_{j=1}^{\infty}\left(\sum_{i=1}^{m}\underbrace{\left(\mu(B_j\cap B_i)\log\left(\frac{\mu(B_j\cap C_m)}{\mu(B_i)}\right)\right)}_{=0}\right. \\ &\quad \left.+\,\mu(B\cap C_m)\log\left(\frac{\mu(B_j\cap C_m)}{\mu(C_m)}\right)\right) \\ &= -\sum_{j=m+1}^{\infty}\mu(B_j)\log(\mu(B_j)) + \mu(C_m)\log(\mu(C_m)). \end{aligned}$$

Skoro $H_\mu(\eta) < \infty$, to dla każdego $\varepsilon > 0$ istnieje taki m że $H_\mu(\eta|\eta_m) < \varepsilon$. Stosując rozumowanie z pierwszej części dowodu do podziału η_m, możemy znaleźć n takie że $H_\mu\left(\eta_m|\bigvee_{i=0}^{n} T^{-i}(\xi)\right) < \varepsilon$. Łącząc dowiedzione fakty, otrzymujemy:

$$\begin{aligned} H_\mu\left(\eta|\bigvee_{i=0}^{n} T^{-i}(\xi)\right) &= H_\mu\left(\eta\vee\eta_m|\bigvee_{i=0}^{n} T^{-i}(\xi)\right) \\ &= H_\mu\left(\eta|\eta_m\vee\bigvee_{i=0}^{n} T^{-i}(\xi)\right) + H_\mu\left(\eta_m|\bigvee_{i=0}^{n} T^{-i}(\xi)\right) < 2\varepsilon. \end{aligned}$$

Pierwsza równość wynika z tego iż η_m jest indukowana z η, a druga z własności 2. Dowolność wyboru ε kończy dowód lematu. □

Twierdzenie 2 (Kołomogorow-Sinaj)**.** *Załóżmy, że $(X, \mathcal{B}, \mu, T)$ jest układem zachowującym miarę na borelowskiej przestrzeni probabilistycznej. Niech ξ będzie podziałem o skończonej entropii, który jest jednostronnym generatorem względem odwzorowania T, tzn.*

$$\bigvee_{n=0}^{\infty} T^{-n}(\xi) = \mathcal{B}.$$

Wtedy:

$$h_\mu(T) = h_\mu(T, \xi).$$

Jeżeli T jest odwracalne i ξ jest podziałem o skończonej entropii, który jest generatorem względem odwzorowania T, tzn.

$$\bigvee_{n=-\infty}^{\infty} T^{-n}(\xi) = \mathcal{B}.$$

Wtedy:

$$h_\mu(T) = h_\mu(T, \xi).$$

Dowód. Niech ξ będzie jednostronnym generatorem względem T. Dla każdego podziału η:

$$h_\mu(T,\eta) \overset{6}{\leqslant} \underbrace{h_\mu\left(T, \bigvee_{n=0}^{\infty} T^{-n}(\xi)\right)}_{\overset{4}{=} h_\mu(T,\xi)} + \underbrace{H_\mu\left(\eta|\bigvee_{i=0}^{n} T^{-i}(\xi)\right)}_{\overset{n\to\infty}{\longrightarrow} 0\ z\ lematu\ 7}$$

z tego wynika, że $h_\mu(T, \eta) \leqslant h_\mu(T, \xi)$ co dowodzi pierwszej części twierdzenia. Drugą dowodzimy analogicznie. □

2.3. Przesunięcia

Rozważmy alfabet złożony z k symboli. Niech $X = \prod_{n=1}^{\infty}\{1,\ldots,k\}$. Element $x \in X$ oznaczony jest przez ciąg $(x_1x_2x_3\ldots)$. W przypadku $k = 2$ element $x \in X$ nazywamy ciągiem binarnym. Niech $p_1,\ldots,p_k$ będą dodatnimi liczbami takimi, że $p_1+\ldots+p_k = 1$. Dla $t \geqslant 1$ definiujemy zbiór cylindryczny (nazywany dalej blokiem) długości n, określony w następujący sposób:

$$[a_1,\ldots,a_n]_{t,\ldots,t+n-1} = \{x \in X\colon x_{t+1} = a_1,\ldots,x_{t+n} = a_n\}.$$

Definiujemy również μ dla bloków jako:

$$\mu\left([a_1,\ldots,a_n]_{t,\ldots,t+n-1}\right) = p_{a_1}\ldots p_{a_n}.$$

Miara probabilistyczna na X, oznaczana μ, jest jednoznacznie zdefiniowana na σ-algebrze generowanej przez zbiory cylindryczne. Tak określone μ nazywamy miarą $(p_1,\ldots,p_k)$-Bernulliego, a X przestrzenią przesunięć Bernulliego. Jednostronne przesunięcie Bernulliego zachowujące miarę μ na X, jest zdefiniowane następująco:

$$(x_1x_2x_3\ldots) \mapsto (x_2x_3x_4\ldots).$$

Podobnie definiujemy obustronne przesunięcie Bernulliego:

$$(\ldots \overset{*}{x_0}\, x_1x_2\ldots) \mapsto (\ldots \overset{*}{x_1}\, x_2x_3\ldots)$$

na $\prod_{-\infty}^{\infty}\{1,\ldots,k\}$, gdzie $*$ oznacza zerową współrzędną ciągu.

Niech $X = \prod_{0}^{\infty}\{1,\ldots,k\}$, a $P = (p_{ij})$ będzie kwadratową macierzą stochastyczną o wymiarze k. Załóżmy, że $\boldsymbol{\pi} = (\pi_i) > 0$ spełnia $\sum_i \pi_i = 1$ i $\boldsymbol{\pi}P = \boldsymbol{\pi}$. Definiujemy μ jako:

$$\mu\left([a_1,\ldots,a_n]_{t,\ldots,t+n-1}\right) = \pi_{a_1}p_{a_1a_2}\ldots p_{a_{n-1}a_n}.$$

W tej sytuacji ponownie istnieje niezmiennicza ze względu na przesunięcie miara probabilistyczna (również oznaczana μ) na σ-algebrze generowanej przez zbiory cylindryczne. Miarę tę nazywamy miarą Markowa, a X przestrzenią przesunięć Markowa.

Niech $Pr\,(B|A)$ oznacza prawdopodobieństwo zajścia zdarzenia B pod warunkiem, że wcześniej zaobserwowano zajście zdarzenia A. Wtedy P opisuje prawdopodobieństwa przejścia od symbolu i do symbolu j, a kolejne elementy tej macierzy definiowane są w następujący sposób:

$$Pr\,(x_{n+1} = j|x_n = i) = p_{ij}.$$

Przesunięcie Markowa jest tożsame z przesunięciem Bernulliego, kiedy kolejne wiersze P są identyczne. W ogólności miara Markowa rzędu $m \geqslant 1$ jest scharakteryzowana przez warunek $Pr\,(b|a_1\ldots a_m)$, gdzie $\sum_{b=1}^{k} Pr\,(b|a_1\ldots a_m) = 1$.

Jeżeli macierz P jest nieredukowalna i aperiodyczna, wtedy wszystkie wartości własne różne od 1, mają moduł mniejszy niż 1. Nieredukowalność macierzy P implikuje ergodyczność, a wraz z aperiodycznością pociąga również za sobą własność mieszania dla przesunięcia Markowa.

Niech $\mathbf{A} = \{s_1,\ldots,s_k\}$ będzie zbiorem symboli. Rozważmy przesunięcie T na $X = \prod_1^{\infty}\mathbf{A}$, gdzie element $x \in X$ wyrażony jest przez $(x_1,\ldots,x_n,\ldots)$ dla $x_j \in \mathbf{A}$. Niech ξ będzie rozbiciem przestrzeni X, którego atomy wyglądają następująco:

$$[s_i]_\xi = \{x \in X\colon x_1 = s_i\},\ 1 \leqslant i \leqslant k.$$

Przez ξ_n oznaczmy $\bigvee_{i=0}^{n-1} T^{-i}\xi$, a atom:

$$[s_1,\ldots,s_n]_{\xi_n} = \{x \in X\colon x_1 = s_1,\ldots,x_n = s_n\}$$

dla $(s_1,\ldots,s_n) \in \mathbf{A}^n$ i rozbicie ξ jest generujące. Niech X będzie przesunięciem Bernulliego, dla którego każdy symbol s_i ma prawdopodobieństwo p_i. Wtedy:

$$H_\mu(\xi_n) = -\sum_{i_1,\ldots,i_n} p_{i_1}\ldots p_{i_n}\log\left(p_{i_1}\ldots p_{i_n}\right) = -n\sum_i p_i\log\left(p_i\right)$$

oraz $h_\mu(T) = -\sum_i p_i \log(p_i)$.

Z powyższego rozumowania wynika następujące twierdzenie:

Twierdzenie 3. *Entropia $(p_1, \ldots, p_k)$-przesunięcia Bernulliego jest równa:*

$$-\sum_{i=1}^{k} p_i \log(p_i).$$

Analogicznie możemy rozważać wartość entropii dla przesunięcia Markowa zadanego macierzą $P = (p_{ij})_{1 \leqslant i,j \leqslant k}$.

Twierdzenie 4. *Entropia przesunięcia Markowa jest równa:*

$$-\sum_{i,j} \pi_i p_{ij} \log(p_{ij}).$$

Dowód. Niech ξ będzie rozbiciem zadanym przez jednoelementowe bloki $[s]_1$, gdzie $s \in \{1, \ldots, k\}$. Definiujemy $\xi_n = \bigvee_{i=0}^{n-1} T^{-i}\xi$ zawierające bloki n-elementowe. Ponieważ $\sum_j p_{ij} = 1$ i $\boldsymbol{\pi}P = \boldsymbol{\pi}$, otrzymujemy:

$$\begin{aligned} H_\mu(\xi_{n+1}) &= -\sum_{i_0,i_1,\ldots,i_n=1}^{k} \pi_{i_0} p_{i_0 i_1} \ldots p_{i_{n-1} i_n} \log\left(\pi_{i_0} p_{i_0 i_1} \ldots p_{i_{n-1} i_n}\right) \\ &= -\sum_{i_0,i_1,\ldots,i_n=1}^{k} \pi_{i_0} p_{i_0 i_1} \ldots p_{i_{n-1} i_n} \left(\log(\pi_{i_0}) + \log(p_{i_0 i_1}) + \ldots + \log\left(p_{i_{n-1} i_n}\right)\right) \\ &= -\sum_{i=1}^{k} \pi_i \log(\pi_i) - n \sum_{i,j=1}^{k} \pi_i p_{ij} \log(p_{ij}) \end{aligned}$$

A stąd:

$$h_\mu(T) = \lim_{n\to\infty} \frac{1}{n+1} H_\mu(\xi_{n+1}) = -\sum_{i,j} \pi_i p_{ij} \log(p_{ij})$$

□

Zauważmy, że z twierdzenia 4 wprost wynika twierdzenie 3, bo przesunięcie Bernulliego będzie miało macierz Markowa o elementach $p_{ij} = p_j$ dla każdego $1 \leqslant i, j \leqslant k$. Stąd $\pi_i = p_i$ dla każdego $1 \leqslant i \leqslant k$. Ponieważ $\sum_i p_i = 1$, entropia tak zadanego przesunięcia jest równa:

$$-\sum_i \sum_j p_i p_j \log(p_j) = -\sum_j p_j \log(p_j).$$

2.4. Twierdzenie Shannona-McMillana-Breimana

Kluczowym twierdzeniem w teorii ergodycznej jest twierdzenie Birkhoffa.

Twierdzenie 5 (Birkhoff)**.** *Niech (X, μ) będzie przestrzenią probabilistyczną. Jeżeli μ jest T-niezmiennicza, a funkcja f jest całkowalna, to*

$$\lim_{n\to\infty} \frac{1}{n} \sum_{k=0}^{n-1} f\left(T^k x\right) = f^*(x)$$

dla pewnego $f^* \in L^1(X, \mu)$, gdzie $f^*(T^k x) = f^*(x)$ dla μ-prawie wszystkich x. Ponadto jeżeli T jest również ergodyczne, wtedy f^* jest stałe oraz

$$\lim_{n\to\infty} \frac{1}{n} \sum_{k=0}^{n-1} f\left(T^k x\right) = \int_X f d\mu$$

dla prawie każdego x.

Niech 1_A oznacza funkcję charakterystyczną podzbioru $A \subset X$. Wybierzmy $f(x) = 1_A$ z twierdzenia Birkhoffa. Jeżeli T jest ergodyczne, to średnia liczba odwiedzin punktu $T^k x$ w A jest równa mierze zbioru A. Innymi słowy ergodyczność implikuje normalność rozkładu. Rezultat odwrotny również zachodzi. Dla każdego podzbioru niezmienniczego A, 1_A jest funkcją niezmienniczą i lewa strona równania z powyższego twierdzenia jest równa $1_A(x)$ dla każdego n, zaś prawa strona jest równa stałej $\mu(A)$. Stąd wynika, że $\mu(A) = \{0, 1\}$.

Dowód twierdzenia Birkhoffa jest długi i techniczny, można go znaleźć m.in. w [3]. Powyższe twierdzenie wykorzystamy jedynie w dowodzie kolejnego faktu.

Niech ξ będzie generującym podziałem przestrzeni probabilistycznej (X, μ), a $T\colon X \to X$ odwzorowaniem zachowującym miarę. Niech $\xi_n = \bigvee_{i=0}^{n-1} T^{-i}\xi$, a $[x]_{\xi_n}$ oznacza atom tak zdefiniowanego podziału ξ_n. Wtedy funkcja zadana wzorem $x \mapsto \mu\left([x]_{\xi_n}\right)$ jest stała na każdym podzbiorze ξ_n oraz:

$$H_\mu(\xi_n) = -\sum_{[x]_{\xi_n} \in \xi_n} \mu\left([x]_{\xi_n}\right) \log \mu\left([x]_{\xi_n}\right) = -\int_X \log \mu\left([x]_{\xi_n}\right) d\mu$$

gdzie sumę liczymy po wszystkich atomach ξ_n. Z tego wynika:

$$h_\mu(T, \xi) = -\lim_{n\to\infty} \frac{1}{n} \int_X \log \mu\left([x]_{\xi_n}\right)$$

czyli mamy własność 4.

Twierdzenie 6 (Shannon-McMillan-Breiman)**.** *Niech T będzie przekształceniem ergodycznym na przestrzeni (X, μ) ze skończonym podziałem generującym ξ, wtedy dla prawie każdego $x \in X$ mamy:*

$$-\lim_{n\to\infty} \frac{\log \mu\left([x]_{\xi_n}\right)}{n} = h_\mu(T).$$

(Sumując lewą stronę równości po wszystkich atomach ξ_n otrzymamy definicję entropii). Pełny dowód twierdzenia znajduje się w [4] i [5]. W niniejszej pracy udowodnione zostaną tylko szczególne przypadki.

Dowód dla przesunięć Bernulliego i Markowa. Niech T oznacza przesunięcie Bernulliego na $X = \prod_1^\infty \{1, \ldots, k\}$ gdzie symbol $i \in \{1, \ldots, k\}$ związany jest z prawdopodobieństwem p_i. Wtedy:

$$[x]_{\xi_n} = \{y \in X\colon y_1 = x_1, \ldots, y_n = x_n\} = [x_1, \ldots, x_n],$$

dla ciągu $x = (x_1 x_2 x_3 \ldots) \in X$ oraz $\mu([x]_{\xi_n}) = p_{x_1} \ldots p_{x_n}$. Niech $f_k\colon X \to \mathbb{R}$ wyraża się wzorem:

$$f_k\left((x_1 x_2 x_3 \ldots)\right) = p_{x_k}.$$

Wtedy $f_k(x) = f_1\left(T^{k-1}(x)\right)$. Twierdzenie Birkhoffa implikuje, że dla każdego x otrzymujemy:

$$\begin{aligned}
\lim_{n\to\infty} \frac{1}{n} \log \mu\left([x]_{\xi_n}\right) &= \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} \log f_k(x) \\
&= \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} \log f_1\left(T^{k-1}(x)\right) \\
&= \int_X \log f_1(x)\, d\mu(x) \\
&= \sum_{i=1}^{k} p_i \log p_i.
\end{aligned}$$

Ostatnia równość wynika z faktu, że $X = \bigcup_{i=1}^k [i]$, $\mu([i]) = p_i$ i $f_1(x) = p_i$ dla każdego $i \in \{1, \ldots, k\}$.

Dowód dla przesunięć Markowa przebiega analogicznie. Niech (p_{ij}) będzie macierzą stochastyczną opisującą przesunięcie Markowa. Mamy $\mu\left([x]_{\xi_n}\right) = \pi_{x_1} p_{x_1 x_2} \dots p_{x_{n-1} x_n}$. Zdefiniujmy $f_{k,j} \colon X \to \mathbb{R}$ jako:

$$f_{k,j}\left((x_1 x_2 x_3 \dots)\right) = p_{x_k x_j}.$$

Zauważmy, że $f_{k,k+1}(x) = f_{1,2}\left(T^{k-1}(x)\right)$. Twierdzenie ergodyczne Birkhoffa implikuje, że dla każdego x:

$$\begin{aligned}
\lim_{n\to\infty} \frac{1}{n} \log \mu\left([x]_{\xi_n}\right) &= \lim_{n\to\infty} \frac{1}{n} \log\left(\pi_{x_1}\right) + \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n-1} \log f_{k,k+1}(x) \\
&= \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n-1} \log f_{1,2}\left(T^{k-1}(x)\right) \\
&= \int_X \log f_{1,2}(x)\, d\mu(x) \\
&= \sum_{i,j=1}^{k} \pi_i p_{ij} \log p_{ij}.
\end{aligned}$$

Ostatnia równość wynika z faktu, że $X = \bigcup_{i,j=1}^{k} [i,j]$, $\mu([i,j]) = \pi_i p_{ij}$ oraz $f_{1,2}(x) = p_{ij}$ dla każdego $x \in [i,j]$. □

Niech teraz $\{\dots x_{-1} x_0 x_1 \dots\}$ będzie stacjonarnym procesem stochastycznym o wartościach pochodzących z przeliczalnego zbioru $\{a_i \colon i = 1,2,3,\dots\}$ nazywanego alfabetem. Niech $p(a_{i_1}, \dots, a_{i_n}) = \mathbb{P}(x_k = a_{i_k} \colon k = 1, \dots, n)$, w skrócie będziemy zapisywać $p_i = p(a_i)$. Ustalamy:

$$g_0 = \log \frac{1}{p(x_0)},\ g_k = \log \frac{p(x_{-k} \dots x_{-1})}{p(x_{-k} \dots x_{-1} x_0)},$$
$$g_0^{(i)} = \log \frac{1}{p(a_i)},\ g_k^{(i)} = \log \frac{p(x_{-k} \dots x_{-1})}{p(x_{-k} \dots x_{-1} a_i)}.$$

Wtedy:

$$g_k \geqslant \mathbb{E}\left(g_{k+1} | x_0 \dots x_{-k}\right)$$

oraz:

$$\mathbb{E}(g_k) \leqslant -\mathbb{E}\left(\log p(x_0)\right).$$

Ponieważ tak zadany proces $\{g_k \colon k = 0,1,2,\dots\}$ jest dodatnim nadmartyngałem, jego entropia jest skończona:

$$-\mathbb{E}\left(\log p(x_0)\right) = \underbrace{-\sum_{i=1}^{\infty} p_i \log p_i}_{h_\mu(T)} < \infty.$$

Z twierdzenia o zbieżności martyngałów g_k jest zbieżny prawie na pewno, gdy $k \to +\infty$. Aby w tym przypadku udowodnić tezę z twierdzenia 6 wystarczy pokazać, że:

$$\mathbb{E}\left(\sup_{0\leqslant k<\infty} g_k\right) < \infty$$

Twierdzenie 7 (Chung)**.** *Przy oznaczeniach przyjętych powyżej, ze skończoności entropii dla procesu g_k wynika skończoność wartości oczekiwanej dla supremum tego procesu, a w konsekwencji również twierdzenie Shannona-McMillana-Breimana.*

Dowód. Zdefiniujmy:

$$E_k(m) = \{\omega \colon \sup_{0\leqslant j<k} g_j < m\,;\, g_k \geqslant m\},$$
$$E_k^{(i)}(m) = \{\omega \colon \sup_{0\leqslant j<k} g_j^{(i)} < m\,;\, g_k^{(i)} \geqslant m\},$$
$$Z_i = \{\omega \colon x_0 = a_i\}.$$

dla każdego zdarzenia elementarnego ω i $m \in \mathbb{N}$. Możemy założyć, że ciąg $\{p_i \colon i = 1, 2, \ldots\}$ jest nierosnący (możemy to osiągnąć dla każdego ciągu przez ewentualne przemianowanie odpowiednich wyrazów alfabetu). Niech $f(m) \geqslant 0$ oraz:

$$\begin{aligned}\mathbb{P}\left(E_k\left(m\right)\right) &= \sum_{i=1}^{\infty} \mathbb{P}\left(E_k\left(m\right) \cap Z_i\right) \\ &= \sum_{i \leqslant f(m)} \mathbb{P}\left(E_k\left(m\right) \cap Z_i\right) + \sum_{i > f(m)} \mathbb{P}\left(E_k\left(m\right) \cap Z_i\right).\end{aligned}$$

Ponieważ $g_k \geqslant m$ na $E_k\left(m\right)$:

$$\begin{aligned}\mathbb{P}\left(E_k\left(m\right)\right) &\leqslant 2^{-m} \mathbb{P}\left(E_k^{(i)}\right)\left(m\right); \\ \sum_{k=0}^{\infty} \sum_{i \leqslant f(m)} \mathbb{P}\left(E_k\left(m\right)\right) &\leqslant 2^{-m} \sum_{i \leqslant f(m)} \sum_{k=0}^{\infty} \mathbb{P}\left(E_k^{(i)}\left(m\right)\right) \\ &\leqslant 2^{-m} \sum_{i \leqslant f(m)} 1 \\ &\leqslant \frac{f\left(m\right)}{2^m}.\end{aligned}$$

Z drugiej strony jednak:

$$\sum_{k=0}^{\infty} \sum_{i > f(m)} \mathbb{P}\left(E_k\left(m\right) \cap Z_i\right) \leqslant \sum_{i > f(m)} \mathbb{P}\left(Z_i\right) = \sum_{i > f(m)} p_i.$$

Niech $f^{-1}\left(i\right)$ spełnia $f\left(m\right) < i$, wtedy $f^{-1}\left(i\right) \leqslant 1 + \max\{m \colon f\left(m\right) < i\}$. Sumując powyższe nierówności po wszystkich m otrzymujemy:

$$\sum_{m=0}^{\infty} \sum_{k=0}^{\infty} \mathbb{P}\left(E_k\left(m\right)\right) \leqslant \sum_{m=0}^{\infty} \frac{f\left(m\right)}{2^m} + \sum_{i=1}^{\infty} f^{-1}\left(i\right) p_i.$$

Ustalmy teraz $f\left(m\right) = \frac{2^m}{(m+1)^2}$ dla którego wyliczamy stałe dodatnie A i B, takie że $f^{-1}\left(i\right) \leqslant A \log i + B$ dla każdego $i \geqslant 1$. Ponieważ p_i jest nierosnące, otrzymujemy $ip_i \leqslant 1$ oraz:

$$\sum_{i=1}^{\infty} f^{-1}\left(i\right) p_i \leqslant \sum_{i=1}^{\infty} \left(A \log \frac{1}{p_i} + B\right) p_i = A h_\mu\left(T\right) + B.$$

A więc:

$$\sum_{m=0}^{\infty} \sum_{k=0}^{\infty} \mathbb{P}\left(E_k\left(m\right)\right) \leqslant \frac{\pi^2}{6} + A h_\mu\left(T\right) + B.$$

Ostatecznie:

$$\mathbb{E}\left(\sup_{0 \leqslant k < \infty} g_k\right) \leqslant \sum_{m=0}^{\infty} \mathbb{P}\left(\sup_{0 \leqslant k < \infty} g_k \geqslant m\right) = \sum_{m=0}^{\infty} \sum_{k=0}^{\infty} \mathbb{P}\left(E_k\left(m\right)\right).$$

□

Niech X będzie przestrzenią przesunięć, a T przesunięciem w lewo. Rozważmy podział ξ zadany przez pierwszą współrzędną. Niech $P_n\left(x\right)$ oznacza względną częstotliwość występowania n-bloku $x_1 \ldots x_n$ w ciągu $x = (x_1 x_2 x_3 \ldots)$, tzn.:

$$P_n(x) = \lim_{N \to \infty} \frac{1}{N} \#\{0 \leqslant j < N \colon x_{1+j} = x_1, \ldots, x_{n+j} = x_n\}.$$

Należy zauważyć, że $P_n\left(x\right) = P_n\left(y\right)$ wtedy i tylko wtedy, gdy $x_j = y_j$ dla każdego $1 \leqslant j < n$. Powyższa granica nie musi istnieć, ale z twierdzenia ergodycznego Birkhoffa wynika, że jeżeli μ jest ergodyczna, to $P_n\left(x\right) = \mu\left(\left[x\right]_{\xi_n}\right)$.

Z twierdzenia Shannona-McMillana-Breimana wynika, że dla prawie każdego x zachodzi:

$$h_\mu(T) = \lim_{n\to\infty} \frac{1}{n} \log\left(\frac{1}{P_n(x)}\right).$$

Wskazuje to iż typowy n-blok ma miarę bliską $2^{-nh_\mu(T)}$, a ponieważ suma miar typowych bloków powinna być bliska 1 wnioskujemy, że jest takich bloków około $2^{nh_\mu(T)}$.

Lemat 8. *Niech (X, μ) oznacza przestrzeń probabilistyczną, a μ ergodyczną i T-niezmienniczą miarę na X. Załóżmy, że ξ jest skończonym i generującym podziałem X. Dla każdego $\varepsilon > 0$ i $n \geqslant 1$ istnieją atomy w ξ_n, nazywane (n, ε)-typowymi blokami i spełniające następujące warunki:*

1. *Dla każdego typowego bloku z ξ_n zachodzi,*

$$2^{-n(h_\mu(T)+\varepsilon)} < \mu\left([x]_{\xi_n}\right) < 2^{-n(h_\mu(T)-\varepsilon)}.$$

2. *Suma mnogościowa wszystkich (n, ε)-typowych bloków ma miarę większą niż $1 - \varepsilon$.*
3. *Liczba (n, ε)-typowych bloków należy do przedziału*

$$\left((1-\varepsilon)2^{n(h_\mu(T)-\varepsilon)}, 2^{n(h_\mu(T)+\varepsilon)}\right).$$

Dowód. Po pierwsze zauważmy, że dla każdego $n \geqslant 1$ funkcja $x \mapsto \mu\left([x]_{\xi_n}\right)$ jest stała na każdym atomie $[x]_{\xi_n}$. Z twierdzenia Shannona-McMillana-Breimana wynika, iż ciąg funkcji:

$$x \mapsto -\frac{1}{n} \log \mu\left([x]_{\xi_n}\right)$$

zbiega względem miary do $h_\mu(T)$ dla $n \to \infty$. Stąd, dla każdego $\varepsilon > 0$ istnieje N takie, że dla $n \geqslant N$:

$$\mu\left(\{x \colon | -\frac{1}{n} \log \mu\left([x]_{\xi_n}\right) - h_\mu(T) | \geqslant \varepsilon\}\right) < \varepsilon.$$

Więc dla zbioru o mierze większej niż $1 - \varepsilon$, mamy:

$$2^{-n(h_\mu(T)+\varepsilon)} < \mu\left([x]_{\mu_n}\right) < 2^{-n(h_\mu(T)-\varepsilon)}.$$

Wniosek, że suma typowych bloków ma miarę większą niż $1-\varepsilon$ wynika wprost z definicji tych obiektów.

Ich liczbę szacujemy w następujący sposób: Kres górny jest osiągany przy założeniu, iż suma typowych bloków ma miarę pełną, a każdy z nich miarę równą $2^{-(h_\mu(T)+\varepsilon)n}$. Podobnie z kresem dolnym, dla którego przyjmujemy że suma typowych bloków ma miarę $1 - \varepsilon$, a każdy z tych bloków ma miarę równą $2^{-(h_\mu(T)-\varepsilon)n}$. □

Warto rozważyć powyższy lemat w kontekście przesunięcia (p, q)-Bernulliego, na przestrzeni ciągów binarnych $X = \prod_1^\infty \{0, 1\}$. Niech $x = (x_1 x_2 x_3 \ldots)$ będzie typowym ciągiem w X. Pośród n pierwszych bitów $x_1, \ldots, x_n$ z x znajdujemy około np zer i nq jedynek. Mamy więc przybliżenie $P_n(x) \approx p^{np} q^{nq}$, a w konsekwencji:

$$\log P_n(x) \approx n\left(p \log p + q \log q\right).$$

Stąd $-\frac{1}{n} \log P_n(x)$ zbiega do entropii odwzorowania.

3. Kompresja danych

Rozdział opracowany w oparciu o pracę *Computational Ergodic Theory* [3] i *Entropy in Dynamical Systems* [7].

Niech zbiór $\mathbb{A} = \{a_1, \ldots, a_N\}$ oznacza alfabet, zdefiniujmy przestrzeń $X = \prod_1^\infty \mathbb{A}$ z niezmienniczą miarą ergodyczną μ i funkcją entropii h_μ. Łańcuch skończonej długości będziemy nazywać blokiem. Blok z przestrzeni X nazywamy źródłowym. Rozważmy kolejny alfabet $\mathbb{B} = \{b_1, \ldots, b_M\}$ i związaną z nim przestrzeń $Y = \prod_1^\infty \mathbb{B}$. Blok z przestrzeni Y nazywamy blokiem kodującym.

Jako kod zdefiniujmy odwzorowanie każdego z bloków źródłowych, w bloki kodujące. Kod o ustalonej długości n, to taki kod który bloki źródłowe o długości n przekształca w słowa kodujące o długości m dla pewnych ustalonych wartości n i m. W celu uzyskania injektywnego kodowania spełniony musi zostać warunek $M^m \geqslant N^n$ lub równoważnie:

$$m \log M \geqslant n \log N.$$

Dla przykładu, aby zakodować angielski alfabet w reprezentacji binarnej potrzebujemy conajmniej 5 bitów (gdzie bit oznacza cyfrę w kodowaniu binarnym), ponieważ $N = 26$, $n = 1$ i $M = 2$ a $m \geqslant \log_2(26) \approx 4.7$. W przypadku polskiego alfabetu jest to dokładnie 5 bitów, gdyż $m \geqslant \log_2(32) = 5$.

Możemy teraz zastosować ostatni lemat z poprzedniego rozdziału (Lemat 8). Załóżmy, iż chcemy znaleźć kod który posyła bloki o relatywnie wysokim prawdopodobieństwie wystąpienia w słowa kodowe o jak najkrótszej długości. Słowa o bardzo małym prawdopodobieństwie ignorujemy. Dla dowolnego $\varepsilon > 0$ wybieramy odpowiednio długie n, takie że miara wszystkich (n, ε)-typowych bloków z X wynosi conajmniej $1 - \varepsilon$. Dobierzmy m spełniające:

$$m \log M \geqslant n \left(h_\mu(T) + \varepsilon\right).$$

Liczba (n, ε)-typowych bloków jest ograniczona przez $2^{n(h_\mu(T)+\varepsilon)}$, które z kolei jest ograniczone przez M^m. Stąd wniosek, iż każdy (n, ε)-typowy blok źródłowy z X może zostać powiązany z blokiem kodującym z Y. W najgorszym wypadku jedynie bloki źródłowe niebędące (n, ε)-typowymi blokami z X nie otrzymają unikalnego słowa kodującego. Prawdopodobieństwo, że blok źródłowy nie otrzyma własnego słowa kodującego jest mniejsze niż ε. Udoskonalenie tego podejścia może zostać zastosowane w kompresji danych. Jeżeli w zadanej przestrzeni występuje więcej wzorców (np. mniej losowości w zadanym tekście), wtedy entropia jest mniejsza i dana wiadomość może zostać bardziej skompresowana. W kompresji stosowanej w transmisji sygnałów oraz w algorytmach przechowujących dane, entropia jest współczynnikiem najlepszej kompresji.

3.1. Twierdzenie Ornsteina–Weissa

Niech $\{s_1, \ldots, s_k\}$ będzie zbiorem symboli. Rozważamy odwzorowanie T, zdefiniowane na przestrzeni $X = \prod_1^\infty \{s_1, \ldots, s_k\}$, jako $(Tx)_n = x_{n+1}$ dla $x = (x_1 x_2 x_3 \ldots)$. Niezmienniczość miary μ względem odwzorowania T wynika z równości:

$$\mu\left([a_1, \ldots, a_s]_{n,\ldots,n+s-1}\right) = \mu\left([a_1, \ldots, a_s]_{1,\ldots,s}\right),$$

gdzie $[a_1, \ldots, a_s]_{n,\ldots,n+s-1}$ jest zbiorem cylindrycznym o długości s, którego $(n+j)$-ta współrzędna jest symbolem a_j dla $n \leqslant j < n + s$.

Definicja 9. *Niech $T\colon X \to X$ będzie zachowującym miarę odwzorowaniem na przestrzeni (X, μ). Załóżmy, że dla $E \subset X$ mamy $\mu(E) > 0$. Dla $x \in E$ definiujemy czas pierwszego powrotu do E, jako:*

$$R_E(x) = \min\{n \geqslant 1\colon T^n(x) \in E\}.$$

Lemat 9 (Kac)**.** *Jeżeli T jest odwzorowaniem ergodycznym które zachowuje miarę na przestrzeni probabilistycznej (X, μ) oraz jeżeli $\mu(E) > 0$, wtedy:*

$$\int_E R_E(x)\, d\mu = 1,$$

innymi słowy, $\int_E R_E d\mu_E = \frac{1}{\mu(E)}$.

Dowód dla T odwracalnego. Dla $n \geqslant 1$, niech:

$$E_n = \{x \in E \colon R_E(x) = n\}.$$

Otrzymujemy parami rozłączną sumę $E = \bigsqcup_{n=1}^{\infty} E_n$ oraz:

$$X = \bigcup_{n=1}^{\infty} \bigcup_{k=0}^{n-1} T^k E_n.$$

Stąd:

$$\int_E R_E d\mu = \sum_{n=1}^{\infty} \int_{E_n} R_E d\mu = \sum_{n=1}^{\infty} n\mu(E_n) = \mu(X) = 1.$$

□

Dowód dla T niekoniecznie odwracalnego. Weźmy $x \in E$ i rozważmy orbitę x w zależności od odwzorowania $T \colon E \to E$:

$$x, T_E(x), \ldots, T_E^l(x), \ldots, T_E^L(x).$$

Połóżmy $N = \sum_{l=0}^{L-1} R_E\left(T_E^l(x)\right)$. Wtedy N jest czasem trwania orbity x pod odwzorowaniem T „rewizyty" E dokładnie L razy, innymi słowy:

$$\sum_{n=1}^{N} 1_E(T^n(x)) = L.$$

Stosując twierdzenie Birkhoffa dla T_E oraz T, otrzymujemy:

$$\int_E R_E d\mu_E = \lim_{L\to\infty} \frac{1}{L} \sum_{l=1}^{L} R_E\left(T_E^l(x)\right) = \lim_{N\to\infty} \frac{N}{\sum_{n=1}^{N} 1_E(T^n(x))} = \frac{1}{\mu(E)}.$$

□

Jeżeli w powyższej definicji $E = [a_1, \ldots, a_n]_{1,\ldots,n}$ i $x \in E$, wtedy:

$$R_E(x) = \min\{j \geqslant 1 \colon x_{j+1} = a_1, \ldots, x_{j+n} = a_n\}.$$

Niech $\xi = \{C_1, \ldots, C_k\}$ będzie mierzalnym podziałem X zadanym przez zbiory cylindryczne postaci:

$$C_i = [s_i]_1 = \{x \in X \colon x_1 = s_i\}.$$

Zauważmy, że ξ jest rozbiciem generującym. Niech $T^{-j}\xi$ dla $j \geqslant 1$ będzie podziałem postaci $\{T^{-j}C_1, \ldots, T^{-j}C_k\}$. Niech $[x]_{\xi_n} \subset X$ będzie jedynym podzbiorem zawierającym x, należącym do:

$$\xi_n = \xi \vee T^{-1}\xi \vee \ldots \vee T^{-(n-1)}\xi.$$

Zdefiniujmy specjalną formę czasu pierwszego powrotu jako $R_n(x) \equiv R_{[x]_{\xi_n}}(x)$, np.

$$R_n(x) = \min\{j \geqslant 1 \colon x_{j+1} = x_1, \ldots, x_{j+n} = x_n\}.$$

$$x = \underbrace{\boxed{a_1 \ldots a_n \cdots\cdots}}_{R_n(x)} a_1 \ldots a_n \cdots\cdots$$

Lemat Kaca implikuje, że średnia R_n pod warunkiem $x_i = a_i$ dla $1 \leqslant i \leqslant n$, jest równa $\frac{1}{\mathbb{P}(a_1 \ldots a_n)}$, gdzie $\mathbb{P}$ jest prawdopodobieństwem zaobserwowania $a_1 \ldots a_n$ w pierwszym n-bloku x. Lemat sugeruje również, że R_n może być przybliżone przez $\frac{1}{\mu([x]_{\xi_n})}$ w wygodny sposób. Ponieważ twierdzenie Shannona-McMillana-Breimana implikuje zbieżność $\frac{1}{\mu([x]_{\xi_n})}$ do entropii odwzorowania, oczekujemy iż $\frac{R_n}{n}$ również zbiega do wartości entropii przy $n \to \infty$.

Z drugiej strony według lematu 8 liczbę typowych podzbiorów ξ_n można przybliżyć przez $2^{nh_\mu(T)}$. Z ergodyczności orbity wynika iż typowy ciąg x odwiedza prawie wszystkie typowe podzbiory w ξ_n, a stąd czas powrotu dla prawie każdego punktu początkowego jest bliski $2^{nh_\mu(T)}$.

Zredefiniujemy teraz czas pierwszego powrotu. Niech:

$$R'_n(x) = \min\{j \geqslant n \colon x_1 \dots x_n = x_{j+1} \dots x_{j+n}\}$$

dla każdego $x = (x_1 x_2 x_3 \dots)$. Zauważmy, że R_n i R'_n są zdefiniowane prawie wszędzie i $R'_n \geqslant n$ oraz $R_n \leqslant R'_n$, a jeżeli $R_n(x) \geqslant n$, to $R_n(x) = R'_n(x)$.

Twierdzenie 8 (Ornstein-Weiss)**.** *Niech T będzie przesunięciem na $X = \prod_1^\infty \{s_1, \dots, s_k\}$ z miarą niezmienniczą i probabilistyczną μ. Załóżmy, że T jest ergodyczne. Dla $x = (x_1 x_2 x_3 \dots) \in X$ zdefiniujmy $R'_n(x) = \min\{j \geqslant n \colon x_1 \dots x_n = x_{j+1} \dots x_{j+n}\}$, wtedy dla prawie każdego x:*

$$\lim_{n\to\infty} \frac{\log\left(R'_n(x)\right)}{n} = h_\mu(T).$$

Dowód. Niech B oznacza blok długości n. Z ergodyczności T, zbiór punktów z B które powracają do B jest pełnej miary. Możemy to zapisać jako:

$$\int_B R'_n(x)\, d\mu = 1 \; lub \int_B R'_n(x)\, d\mu_B = \frac{1}{\mu(B)},$$

czyli mamy lemat Kaca (9) dla $R'_n(x)$. Stąd $R'_n(x) > \frac{e^{n\varepsilon}}{\mu(B)}$ może zachodzić na podzbiorze B o mierze μ_B co najwyżej $e^{-n\varepsilon}$. Z twierdzenia o prawdopodobieństwie całkowitym, to oszacowanie zachodzi również na całym X, co może zostać zapisane jako:

$$\mu\{x \colon \frac{\log\left(R'_n(x)\right)}{n} > I_\mu(\xi_n)(x) + \varepsilon\} < e^{-n\varepsilon}.$$

Stosując lemat Borela-Cantellego ([3] i [7]), twierdzenie Shannona-McMillana-Breimana (6) i fakt że ε jest ustalone, wykazujemy:

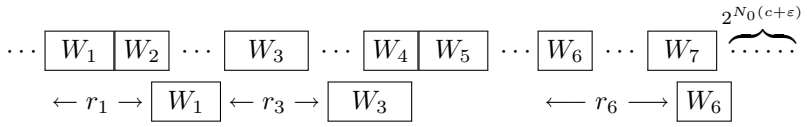
$$\limsup_{n\to\infty} \frac{\log\left(R'_n(x)\right)}{n} \leqslant h_\mu(T, \xi) \leqslant h_\mu(T)$$

μ prawie wszędzie.

Niech $\xi = \{E_1, E_2, E_3, \dots\}$ będzie rozbiciem X, przez ξ^m oznaczymy podział $\{E_1, \dots, E_{m-1}, \bigcup_{i=m}^\infty E_i\}$. Ponieważ $R'_n(x) \geqslant R'_{n-1}(T(x))$, funkcja $\liminf_{n\to\infty} \frac{\log\left(R'_n(x)\right)}{n}$ jest subniezmiennicza (μ jest subniezmiennicza, gdy $\mu\left(T^{-1}(X)\right) \leqslant \mu(X)$), a stąd wynika że stale równa c. Musimy pokazać, że $c \geqslant h_\mu(T)$. Jeżeli ξ zastąpimy przez ξ^m, czasy powrotu mogą się jedynie skrócić. Z tego powodu oraz zauważając że $h_\mu(T, \xi^m) \nearrow h_\mu(T, \xi)$, wystarczy pokazać nierówność $c \geqslant h_\mu(T, \xi)$ dla skończonych rozbić. W dalszej części dowodu $l = \#\xi$.

Ustalmy $\varepsilon > 0$ i $1 \geqslant \delta > 0$. Dla prawie każdego x istnieje $n_x > \frac{1}{\delta}$ takie, że $R_{n_x}(x) \leqslant 2^{n_x(c+\varepsilon)}$. Stąd, istnieje zbiór Z o mierze mniejszej niż $\frac{\delta}{2}$ taki, że $N_0 = \max\{n_x \colon x \notin Z\}$ jest skończone. Z twierdzenia ergodycznego ([7]), dla dostatecznie dużego m, wszystkie punkty x ze zbioru X' miary $1 - \delta$, odwiedzają Z nie więcej niż $\frac{m\delta}{2}$ razy w czasie pierwszych m iteracji. Biorąc odpowiednio duże m można założyć, że $2^{N_0(c+\varepsilon)} < \frac{m\delta}{2}$. Wtedy X' może zostać pokryte przez pewną liczbę $\mathbf{D}$ cylindrów B o długości m z których każdy ma następującą strukturę: *każdy ze zbiorów B może zostać wyrażony (niekoniecznie jednoznacznie) przez łańcuch bloków W_i o długości $n_i \in \left[\frac{1}{\delta}, N_0\right]$ i nie więcej niż $m\delta$ wpisach włączając w to $2^{N_0(c+\varepsilon)}$ końcowych wpisów, w taki sposób że każdy z bloków W_i jest powtórzony w B z prawej strony na dystansie $r_i \leqslant 2^{n_i(c+\varepsilon)}$.*

Opisana powyżej struktura wygląda następująco (zaznaczono jedynie niektóre powtórzenia):



Liczba $\mathbf{D}$ wyraża ilość bloków B o tak zadanej strukturze.

Ponieważ bloki B mają długość conajmniej $\frac{1}{\delta}$, ich liczba nie przekracza wartości $m\delta$, a stąd ich początkowa wartość może zostać wybrana na

$$2^{m\left(\delta \log \frac{1}{\delta} + (1-\delta) \log \frac{1}{1-\delta}\right)}$$

sposobów. Podobnie na tyle sposobów możemy wybrać pojedynczy wpis w B, oraz zapełnić go symbolami na:

$$l^{m\delta}$$

sposobów. Załóżmy teraz, że znane są początkowe pozycje bloków W_i. Wszystkie miejsca poza nimi również są ustalone i wypełnione symbolami (również ostatnie $2^{N_0(c+\varepsilon)}$ znaków). W tym momencie również długości n_i bloków W_i są zdeterminowane. Brakującym ogniwem w całkowitym określeniu postaci bloków B są symbole w łańcuchach W_i. Do ich jednoznacznego określenia wystarczy jednak znajomość odległości między powtórzeniami bloków W_i.

Dla każdego i odległość między W_i a jego pierwszym powtórzeniem w prawo przyjmuje wartość r_i nie większą niż $2^{n_i(c+\varepsilon)}$. Globalnie oznacza to:

$$2^{\sum_i n_i(c+\varepsilon)} \leqslant 2^{m(c+\varepsilon)}$$

możliwych wyborów r_i (dla wszystkich i). W momencie dokonania tego wyboru, blok B jest całkowicie określony, ponieważ każdy wyraz dla każdego łańcucha W_i (przesuwając się od prawej do lewej) może zostać skopiowany z pozycji odległej o r_i w prawo, czyli z określonej już części B.

Korzystając z powyższych estymat otrzymujemy:

$$\mathbf{D} \leqslant 2^{2m\left(\delta \log \frac{1}{\delta} + (1-\delta) \log \frac{1}{1-\delta}\right)} \cdot l^{m\delta} \cdot 2^{m(c+\varepsilon)}$$

$$\log \mathbf{D} \leqslant 2m \left(\delta \log \frac{1}{\delta} + (1-\delta) \log \frac{1}{1-\delta} \right) + m\delta \log l + m\,(c+\varepsilon) \leqslant m\,(c+2\varepsilon)$$

przy odpowiednim wyborze δ. Ponieważ dopełnienie X' jest pokryte przez nie więcej niż l^m cylindrów długości m, entropia ξ^m nie przekracza:

$$(1-\delta)\, m\,(c+2\varepsilon) + \delta m \log l + \left(\delta \log \frac{1}{\delta} + (1-\delta) \log \frac{1}{1-\delta} \right) < m\,(c+3\varepsilon)\,.$$

Ponieważ $H_\mu\left(\xi^m\right) \geqslant m h_\mu\left(T, \xi\right)$ dla każdego m, ξ i przy ustalonym ε, należy uznać że $c \geqslant h_\mu\left(T\right)$.

□

Powyższa formuła umożliwia nam estymowanie entropii poprzez obserwacje typowego podciągu. W rzeczywistych zastosowaniach powyższego twierdzenia nie uświadczymy nieskończonych ciągów binarnych. Jeżeli plik lub cyfrowa wiadomość jest wystarczająco długa, możemy przyjąć iż jest częścią nieskończonego bloku i powyższa formuła może zostać zastosowana. Najmocniejszym punktem twierdzenia Ornsteina-Weissa jest łatwość komputerowej implementacji.

Ponieważ zbiory R'_n i R_n są podobne, możemy zakładać iż formuła zachodzi również w drugim wypadku.

Lemat 10. *Niech T będzie odwzorowaniem ergodycznym (przesunięciem) na przestrzeni X o skończonej liczbie symboli.*

1. *Jeżeli $R_n\left(x\right) < n$, to istnieje k, $\frac{n}{3} \leqslant k < n$, takie że $R'_k\left(x\right) = k$.*

2. Jeżeli T ma dodatnią entropię, wtedy dla prawie każdego x istnieje $N = N(x)$ takie, że $R'_n(x) = R_n(x)$ dla $n \geqslant N$.

Dowód. (1.) Jeżeli $R_n(x) = k$ dla pewnego $k < n$, wtedy $x_1 \dots x_k = x_{k+1} \dots x_{2k}$ i $R'_k(x) = k$. Jeżeli $k \geqslant \frac{n}{3}$, wtedy wszystko jasne. Jeżeli $k < \frac{n}{3}$, to:

$$x_1 \dots x_{n+k} = x_1 \dots x_k x_1 \dots x_k x_1 \dots x_k \dots x_1 \dots x_l$$

dla pewnego $l \leqslant k$. Stąd $R'_{mk}(x) = mk$ dla pewnego m takiego, że $\frac{n}{3} \leqslant mk < n$.

(2.) Niech $E = \{x \in X \colon \limsup_{n\to\infty}(R'_n(x) - R_n(x)) > 0\}$. Jeżeli $x \in E$, wtedy $R'_n(x) > R_n(x)$ dla nieskończenie wielu n. Stąd $R_n(x) < n$ dla nieskończenie wielu n. Pierwsza część lematu implikuje, że $R'_k(x) = k$ dla nieskończenie wielu k oraz $\liminf_{k\to\infty} \frac{\log R'_k(x)}{k} = 0$. Następnie formuła z twierdzenia Ornsteina-Weissa implikuje, że zbiór E ma miarę zero, co kończy dowód. □

Twierdzenie 9. *Dla przesunięcia ergodycznego T na zbiorze o skończonej liczbie symboli,*

$$\lim_{n\to\infty} \frac{\log R_n(x)}{n} = h_\mu(T)$$

dla prawie każdego x.

Dowód. Jeżeli entropia jest większa niż 0, używamy lematu 10 (2.). Jeżeli entropia jest równa 0, zauważmy że:

$$\limsup_{n\to\infty} \frac{\log R_n(x)}{n} \leqslant \limsup_{n\to\infty} \frac{\log R'_n(x)}{n} = \lim_{n\to\infty} \frac{\log R'_n(x)}{n} = 0.$$

□

Wniosek 1. *Załóżmy, że każdy blok w $\prod_1^\infty \{0,1\}$ ma niezerowe prawdopodobieństwo. Lemat Kaca implikuje:*

$$\mathbb{E}(R_n) = \sum_{B\in\xi_n} \mathbb{E}(R_n|B)\,\mathbb{P}(B) = \sum_{B\in\xi_n} \frac{1}{\mathbb{P}(B)}\mathbb{P}(B) = \sum_{B\in\xi_n} 1 = 2^n.$$

4. Implementacja twierdzenia Ornsteina-Weissa

Opisane w poprzednim rozdziale wyniki zaimplementowane zostały w języku programowania Python.

Przykład 4.1. Na początku rozważmy przesunięcia $(1-p,p)$-Bernulliego na ciągach binarnych.

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  import math
4  import random
5  import time
6
7  k = 1  # Przesuniecie Bernulliego (1/(k+1),1-1/(k+1));
8  prob = 1. / (k + 1)  # Prawdopodobienstwo symbolu '1';
9  entropy = -(1. - prob) * math.log(1. - prob, 2.) \
10           - prob * math.log(prob, 2.)
```

W pierwszych liniach kodu importujemy niezbędne biblioteki. Dokumentacja kolejnych pakietów znajduje się pod adresami [9], [10], [11], [12] i [13]. Następnie ustalone zostają zmienne pomocnicze oraz policzona zostaje teoretyczna entropia przesunięcia $(1-p,p)$-Bernulliego. Dla $k = 1$ zadanego w 7 linijce będzie ona wynosiła 1.

Kolejnym działaniem będzie ustalenie długości szukanego bloku oraz długości całkowitego ciągu binarnego.

```
11 Max_Block = 25
12 N = round(2 ** (entropy * Max_Block))
```

Dla $k = 1$ i $Max\_Block = 25$, nasza "lokalna nieskończoność" wynosi 33554432. Tej długości ciąg binarny losujemy w następujący sposób:

```
13 x = list()
14 for i in range(0, N):
15     x.append(math.trunc(random.randint(0, k) / k))
```

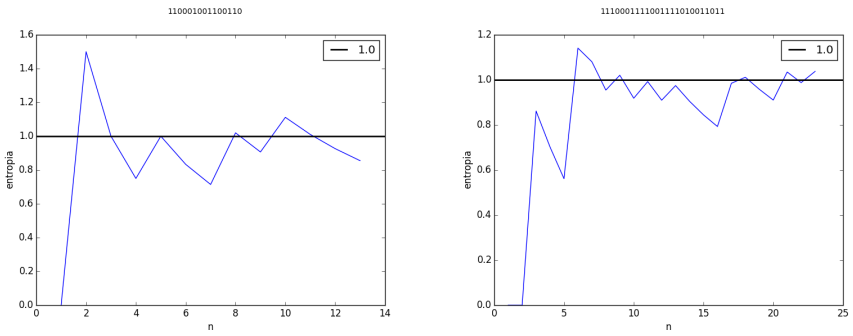
Zmienna x jest listą do której w pętli doklejamy losowo 0 lub 1 z prawdopodobieństwem $\frac{1}{k+1}$ - bo zaczynamy od 0. Metoda *trunc* rzutuje wartość rzeczywistą na całkowitą usuwając miejsca po przecinku, w przypadku $k = 1$ jest zbędna, gdyż liczby całkowite losowane są z przedziału $[0, 1]$.

```
16 seq = x[0: Max_Block]
17 t = seq.count(1) / Max_Block
18 empiricalProb = x.count(1) / N
```

Ustalamy zmienną *seq* jako początek wylosowanego ciągu, a następnie przypisujemy do t prawdopodobieństwo wystąpienia 1 w typowym podzbiorze.

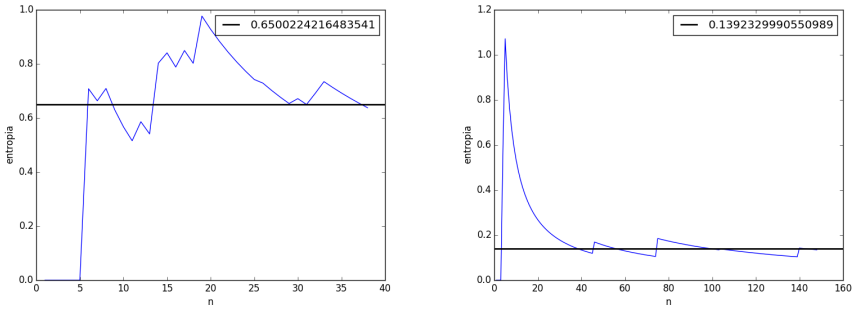
W celach diagnostycznych warto sprawdzić czy prawdopodobieństwo wystąpienia 1 w wylosowanym ciągu jest bliskie zakładanemu wcześniej. Podczas kilkuset prób przeprowadzonych przy pisaniu pracy magisterskiej, wartości prawdopodobieństw w losowanych ciągach - znacznie dłuższych niż 1000 - zawsze były bliskie wartości $\frac{1}{k+1}$. Ewentualne odchylenia od tej wartości nie wpływały na dokładność algorytmu, a jedynie na szybkość jego zbieżności.

```
19 fullBinarySeries = ''.join([str(tempX) for tempX in x])
20 tempR = list()
21 for i in range(0, Max_Block - 1):
22     ReturnIndex = fullBinarySeries.find(fullBinarySeries[0:i + 1],
23                                                                 1)
24     if ReturnIndex != -1:
25         tempR.append(ReturnIndex)
26     else:
27         break
28 g1 = {i: abs(math.log(tempR[i - 1], 2.)) / i
29           for i in range(1, len(tempR))}
```



Rysunek 1. Przykładowe wykresy dla $Max\_Block = 15$ (lewy) i $Max\_Block = 25$ (prawy). W obu przypadkach teoretyczne prawdopodobieństwo wystąpienia 1 w ciagu binarnym wynosiło $\frac{1}{2}$.

Powyższy fragment kodu stanowi adaptację algorytmu z pracy [3] w języku Python. W linijce 19 stworzoną wcześniej listę zer i jedynek przerabiamy na ciąg znaków (string). Jest to wygodniejsze, gdyż język Python lepiej radzi sobie z tego typu obiektem aniżeli z listą. Świadczy o tym m.in. wbudowana i zoptymalizowana metoda *find* która zwraca indeks pierwszego pojawienia się podanego podciągu. Pętlę w której poszukujemy coraz dłuższych podciagów kończymy po znalezieniu wszystkich o długości mniejszej lub równej Max_Block lub w momencie kiedy poszukiwany podciąg już się nie pojawia (metoda zwraca wtedy indeks -1). W ostatnich linijkach generujemy mapę w której kluczami są długości



Rysunek 2. Algorytm jest zbieżny również w innych przypadkach. Po lewej prawdopodobieństwo wystąpienia jedynki wynosiło $\frac{1}{6}$, a losowany ciąg (symulujący nieskończoność na potrzeby twierdzenia Ornsteina-Weissa) miał długość 67150596, gdyż rozważano tutaj podciąg $Max\_Block = 40$. Po prawej przesunięcie $\left(\frac{1}{51}, \frac{50}{51}\right)$-Bernulliego i blok o długości 150. Należy zaznaczyć, że w przypadku tak małego prawdopodobieństwa wystąpienia jedynki, nawet tak długi Max_Block bywał ciągiem samych 0.

podciągów a wartościami wyniki formuły z twierdzenia Ornsteina-Weissa, czyli $\frac{\log(R_i)}{i}$.

Rysunki 1 i 2 obrazują wyniki działania powyższego programu.

Przykład 4.2. Skrajne przypadki (jak ten opisany w tekście pod 2) podsuwają pomysł na "uśrednioną" wersję algorytmu. Zamiast rozważać jedynie $\log \frac{R_n}{n}$, możemy spróbować dwóch przypadków $Ave\left(\log \frac{R_n}{n}\right)$ i $\log \frac{Ave(R_n)}{n}$, gdzie funkcja Ave oznacza średnią arytmetyczną liczoną po określonym offsecie.

Pierwsze linijki programu liczącego zmodyfikowaną wersję nie różnią się od przykładu 4.1. Różnice pojawiają się dopiero w linijkach z właściwym algorytmem. W celu policzenia średniej logarytmów postępujemy w następujący sposób:

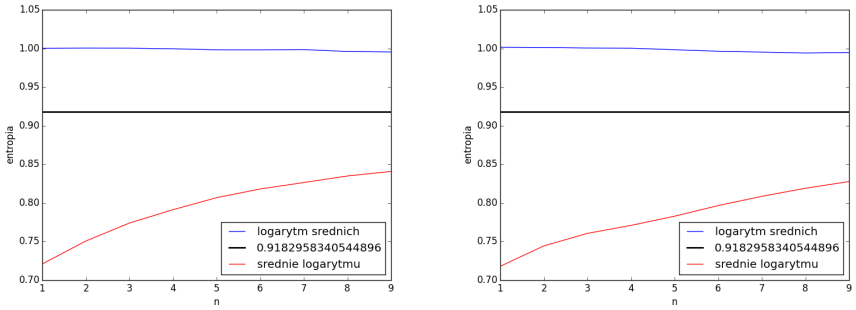
```
12  N = round(2 ** (
13      entropy * Max_Block)) * 200

    ...

19  binarySeries = ''.join([str(tempX) for tempX in x])
20
21  S = 5000
22
23  R = [list() for _ in range(0, S)]
24
25  for i in range(0, S):
26      for b in range(0, Max_Block):
27          ReturnIndex = binarySeries.find(binarySeries[i:i + b + 1],
28                                                                  i + 1)
29          if ReturnIndex != -1:
30              R[i].append(ReturnIndex - i)
31          else:
32              break # R[i].append(N)
```

W pierwszych linijkach ustawiamy wartość o którą będziemy przesuwać nasz ciąg. Im ta wartość większa tym lepsze dostaniemy przybliżenie entropii, jednak zbyt duża wartość spowoduje że najdłuższe podciągi (dodatkowo przesunięte o wartość S) mogą nie pojawić się w ciągu $binarySeries$. Wynika to z faktu, iż całkowita długość ciągu binarnego symulującego nieskończoność wynosi $2^{Max\_Block \cdot h_\mu(T)} \cdot B$, gdzie B jest pewną stałą wybraną na potrzeby symulacji (linijka 12 i 13). Aby mieć pewność, że nawet najdłuższy podciąg pojawi się po przesunięciu należałoby zmienić wykładnik na $(S \cdot Max\_Block) \cdot h_\mu(T)$, co jednak daje ogromne długości ciągu $bianrySeries$.

W linijce 23 tworzymy listę list (nie musi to być pełna macierz $S \times Max\_Block$), a dalej uzupełniamy ją w analogiczny sposób jak w przykładzie 4.1, uwzględniając przesunięcie. Można w tym momencie zastosować dwa podejścia i przy braku pojawienia się podciągu przerywać pętlę lub odkładać jakąś dużą wartość. W pierwszym przypadku otrzymamy nieco zaburzoną średnią (gdyż i-ta kolumna może nie być w pełni uzupełniona). W drugim wypadku dostaniemy nieco zaburzony wynik, przy długich podciągach które szybko przestaną się pojawiać entropia będzie dążyła do 0. Różnice w obu przypadkach zobrazowane zostały na rysunkach:



Rysunek 3. Porównanie wyników $Ave\left(\log \frac{R_n}{n}\right)$ i $\log \frac{Ave(R_n)}{n}$ dla przesunięcia $\left(\frac{1}{3}, \frac{2}{3}\right)$-Bernulliego przy wykorzystaniu komendy *break* (lewo) i wstawianiu długości całego ciągu (prawy). Jak widać różnice w obu podejściach nie są zbyt duże. Warto zwrócić uwagę na skalę która na obu wykresach jest zdecydowanie większa niż w przypadku rysunków 1 i 2. Offset w obu przypadkach wynosił $S = 5000$.

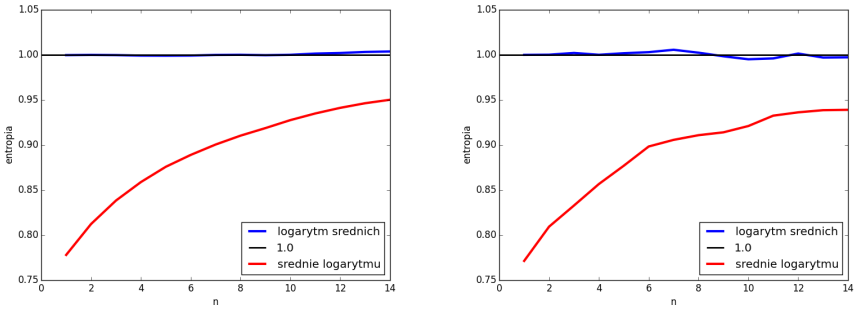
W dalszej części programu następuje właściwe wyliczanie wartości funkcji $\log \frac{Ave(R_n)}{n}$:

```
33 AveragesOfR = [0 for _ in range(0, Max_Block)]
34 for i in range(0, Max_Block):
35     tempAve = list()
36     for j in range(0, S):
37         if len(R[j]) > i:
38             tempAve.append(R[j][i])
39     AveragesOfR[i] = sum(tempAve) / len(tempAve)
40
41 g1 = {i: abs(math.log(AveragesOfR[i - 1], 2.)) / i
42                 for i in range(1, len(AveragesOfR))}
```

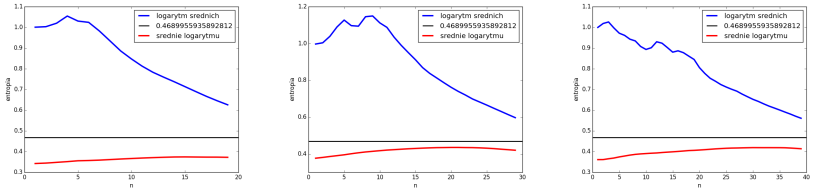
i $Ave\left(\log \frac{R_n}{n}\right)$:

```
43 AveLog = [0 for _ in range(0, Max_Block)]
44 for i in range(0, Max_Block):
45     tempAveLog = list()
46     for j in range(0, S):
47         if len(R[j]) > i:
48             tempAveLog.append(R[j][i])
49     temp = [abs(math.log(tempAveLog[t], 2.)) / (i + 1)
50                 for t in range(0, len(tempAveLog))]
51     if sum(temp) == 0 or len(temp) == 0:
52         continue
53     else:
54         AveLog[i] = sum(temp) / len(temp)
55
56 g2 = {i: AveLog[i] for i in range(1, len(AveLog))}
```

Z nierówności Jensena średnia z logarytmu jest mniejsza lub równa logarytmowi ze średniej, stąd powyższe funkcje stanowią ograniczenie szukanej entropii. Poniżej kilka przykładów:



Rysunek 4. Po prawej szkic dla przesunięcia $\left(\frac{1}{3}, \frac{2}{3}\right)$-Bernulliego, $Max\_Block = 15$ i $S = 5000$, po lewej analogiczny szkic z $S = 1000$. Widać, że większy offset nieznacznie poprawia dokładność jednakże wyraźnie wygładza obie funkcje.



Rysunek 5. Trzy szkice obrazują jak średnia logarytmów i logarytm średnich zachowują się w przypadku przesunięcia $\left(\frac{1}{10}, \frac{9}{10}\right)$-Bernulliego ze względu na długość szukanego podciągu. We wszystkich przypadkach $S = 5000$. Max_Block odpowiednio 20 (po prawej), 30 (w środku) i 40 (po lewej). Te długości podciągów szukane były odpowiednio w ciągu *binarySeries* o długościach równych 133200, 3439400 i 88776800 - co pokazuje że wzrost całkowitego ciągu, nie jest proporcjonalny do osiąganej przez algorytm dokładności.

5. Analiza entropii języka polskiego w oparciu o twierdzenie Ornsteina-Weissa

Powyższe przykłady stały się bazą dla programów służących analizie języka polskiego.

Przykład 5.1. Teksty do analizy zostały pozyskane za pomocą prostych crawlerów (robotów internetowych) ze strony (http://wolnelektury.pl/) projektu wolne lektury. Przykładowy kod takiego robota został umieszczony w 6.

Właściwy program rozpoczyna się podobnie jak w poprzednich przykładach, od importu odpowiednich bibliotek i ustawienia później wykorzystywanych zmiennych:

```
1  # -*- coding: utf-8 -*-
2  import math
3  import re
4  import glob
5
6  import numpy as np
7  import matplotlib.pyplot as plt
8
9  pattern = re.compile('([^\s\w]|_)+')
10 Teksty = glob.glob('Liryka/*.txt')
11 # Teksty = glob.glob('Epika/*.txt')
```

Jedynym nowym modułem importowanym w tym przypadku jest *glob*, którego dokumentacja znajduję się pod [14]. Ułatwia on odczyt wielu plików z zadanej ścieżki - w naszym przypadku wielu wierszy z katalogu *Liryka*. W 9 linijce tworzone jest wyrażenie regularne, które używane będzie w celu usunięcia wszystkiego co nie jest białym znakiem, literą lub cyfrą. Poczynione założenie miało na celu ujednolicenie badanych tekstów. Nie jest natomiast w żaden sposób modyfikowana np. wielkość liter.

Przechodzimy teraz do głównej pętli skryptu:

```
12 for tekst in Teksty:
13     helper = re.search('\\\(.*)\.txt', tekst, flags=0)
14     toAnalyze = open(''.join(['Liryka/', helper.group(1), '.txt']), 'r',
15                                 encoding='utf-8')
16
17     f = open(''.join(['TekstyLirykaAnaliza/', helper.group(1), '.log']),
18                 'w')
19
20     k = "".join(line for line in toAnalyze if not line.isspace())
21     k = pattern.sub('', k)
22
23     binary = ' '.join(format(ord(x), 'b') for x in k).replace(' ', '')
24     dlugoscTekstu = len(binary)
25     print(dlugoscTekstu, file=f)
26
27     prob = binary.count('1') / dlugoscTekstu
28     entropy = -(1. - prob) * math.log(1. - prob, 2.)\
29                   - prob * math.log(prob, 2.)
30     print(prob, file=f)
31     print(entropy, file=f)
32     print(abs(math.log(dlugoscTekstu, 2. * entropy)), file=f)
33
34     Max_Block = math.ceil(abs(math.log(dlugoscTekstu, 2. * entropy)))
35     S = 5000
```

W linijce numer 15 wyłuskujemy nazwę dzieła, która posłuży nam jako nazwa pliku do logowania działań w skrypcie (plik ten tworzony jest w linijce 19, a każda funkcja *print* z argumentem $file = f$ będzie zapisywała do niego wyjście). Zmienna *toAnalyze* inicjalizowana przy każdym przejściu pętli zawiera tekst dzieła, który w linijce 22 pozbawiony zostaje pustych linii, a w linijce 23 wszystkiego poza literami, cyframi i białymi znakami. Następną czynnością jest tłumaczenie tekstu na alfabet binarny, tak aby móc wykorzystać algorytmy z poprzednich przykładów.

W kolejnych liniach inicjalizowane są wartości pomocnicze:

— prawdopodobieństwo wystąpienia jedynki w tłumaczeniu binarnym (linia 29),

— entropia doświadczalna policzona na podstawie wzoru dla przesunięcia $(p, 1-p)$-Bernulliego (linia 30),

— zmienna Max_Block, będąca maksymalną długością podciągu biorącego udział w analizie (linia 36),

— przesunięcie podciągu, biorące udział w badaniu średniej logarytmu i logarytmu średnich (linia 37).

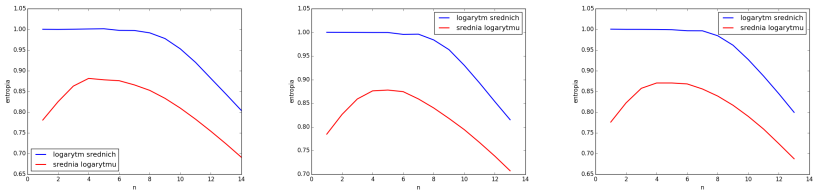
W dalszej części pętli *for* wykorzystujemy algorytmy z przykładu 4.2. Pełny kod programu znajduję się w 6.

Przykład 5.2. Przyjrzymy się teraz wynikom dla poezji:

Trzy przedstawione teksty miały długość (już po przetłumaczeniu na alfabet binarny):

1. 15833 (*Oda do Młodości, Adam Mickiewicz*),
2. 12045 (*Testament mój, Juliusz Słowacki*),
3. 9609 (*W żłobie leży, Piotr Skarga*).

Oznacza to, iż przyjęta dla przesunięcia podciągu wartość ($S = 5000$) była stosunkowo duża (przekraczająca nawet 50%, dla ostatniego tekstu). Wnioskując



Rysunek 6. Trzy powyższe wykresy powstały z analizy dzieł *Oda do Młodości* (lewy), *Testament mój* (środkowy) i kolendy *W żłobie leży* (prawy).

jednak z testów przeprowadzonych w 4.2, nie wpływa to znacząco na dokładność analiz.

W badanych wypadkach podciągi miały również zbliżoną długość:

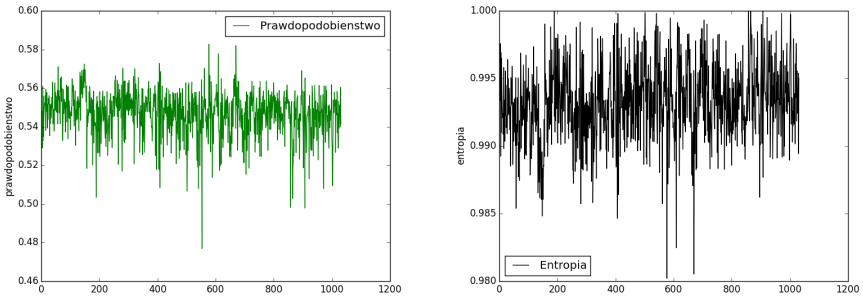
1. 15 (*Oda do Młodości, Adam Mickiewicz*),
2. 14 (*Testament mój, Juliusz Słowacki*),
3. 14 (*W żłobie leży, Piotr Skarga*).

Wartości te wyliczone zostały ze wzoru

$$\left\lceil \left| \log_{2h_\mu(T)} (dlugoscTekstu) \right| \right\rceil,$$

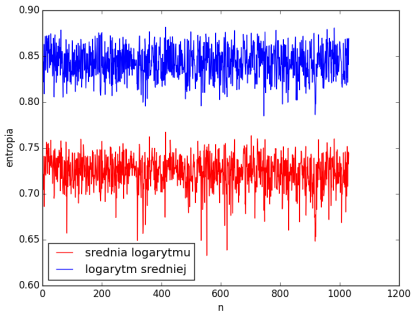
gdzie $h_\mu(T)$ jest entropią doświadczalną dla każdego z tych dzieł.

Tak skonstruowanemu badaniu poddano 1032 teksty w języku polskim z rodzaju liryki. Wyniki tych dociekań przedstawiają kolejne szkice:



Rysunek 7. Prawdopodobieństwo wystąpienia jedynki w ciągu binarnym będącym tłumaczeniem tekstu polskiego (lewy) oraz entropia wyliczona na podstawie tego prawdopodobieństwa dla przesunięcia $(p, 1-p)$-Bernulliego (prawy).

Warto zwrócić uwagę na skalę w obu wykresach. Prawdopodobieństwo wystąpienia symbolu '1' waha się w przedziale $[48\%, 58\%]$, a entropia (teoretyczna, policzona z wzoru dla przesunięć Bernulliego) jedynie w $[0.98, 1]$.

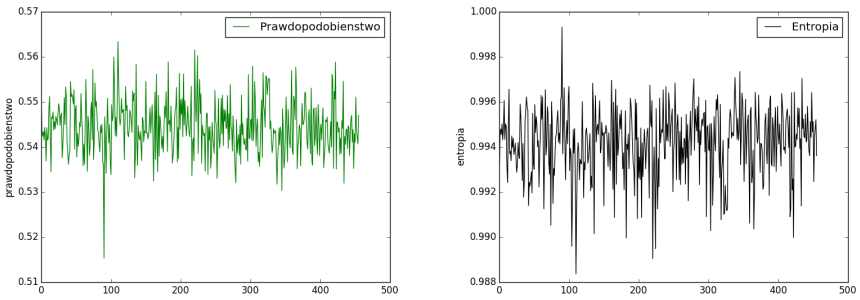


Rysunek 8. Wyniki analizy liryki w języku polskim

Ostateczny wynik dla tekstów liryki polskiej wskazuje znacznie mniejszą entropię, niż ta na rysunku 7. Wynika to z faktu, iż teksty poetyckie nie są losowymi ciągami znaków. Nawet górne ograniczenie zadane przez $\log \frac{Ave(R_n)}{n}$ leży znacząco poniżej krzywej entropii dla ciągu całkowicie losowego.

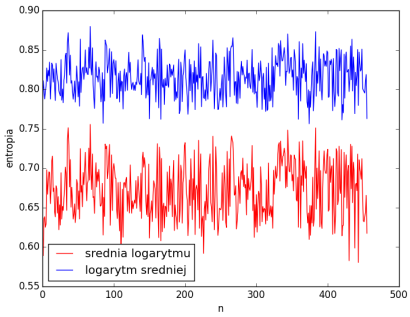
Powyższe wyniki chcielibyśmy umieścić w pewnym kontekście, więc dla porównania analizie poddano również teksty z rodzaju epiki.

Przykład 5.3. Badania tekstów epiki zostały przeprowadzone na próbce 456 tekstów polskich dzieł i tłumaczeń. Średnia długość badanych ciągów (już po tłumaczeniu na alfabet binarny) wynosiła około 1145969 znaków, wynika z tego iż podciągi wyszukiwane podczas analizy miały średnio 19 znaków. W przypadku liryki powyższe wartości były równe odpowiednio 22789 i 14 znaków.



Rysunek 9. Prawdopodobieństwo jedynki i entropia empiryczna dla tekstów z rodzaju epiki.

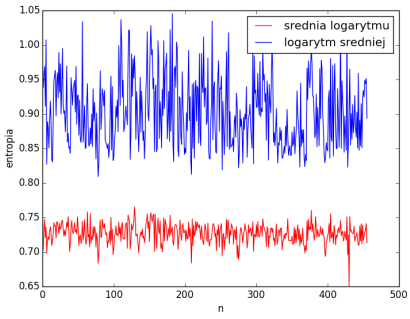
Na powyższych szkicach widać, iż prawdopodobieństwo symbolu '1' w tekstach epiki (podobnie jak w przypadku liryki) jest bliskie 50%. Również entropia ma zbliżoną do liryki, niewielką amplitudę.



Rysunek 10. Wyniki analizy epiki w języku polskim

Badania dla funkcji $Ave\left(\log \frac{R_n}{n}\right)$ i $\log \frac{Ave(R_n)}{n}$ w przypadku epiki nieznacznie różnią się od wyników osiągniętych w przypadku liryki. Entropia liryki bliższa była wartości 0.8, dla epiki jest to już 0.75. Faktem jest jednak, iż w przypadku epiki analizie poddane zostały nieco dłuższe podciągi niż w przypadku liryki, więc i ilość informacji powinna być większa (mniejsza entropia).

Jeżeli dane dla epiki ograniczymy do podciągów długości $\leqslant 14$, otrzymujemy poniższą analizę:



Rysunek 11. Wyniki analizy epiki w języku polskim przy ograniczeniu badanego podciągu do 14 znaków.

Na powyższym szkicu widać, iż dolna granica zbliżona jest do tej którą obserwowaliśmy dla tekstów liryki. Zauważalne jest jednak spore odchylenie dla funkcji log $\frac{Ave(R_n)}{n}$ stanowiącej górną granicę szukanej entropii.

Podobnie jak w przypadku liryki, także i tu mamy do czynienia ze znaczną różnicą między entropią wyliczoną z formuły Ornsteina-Weissa, a zakładaną dla ciągu całkowicie losowego.

Przytoczone przykłady obarczone są pewnym błędem, mogącym wpływać na obliczoną wartość entropii. Przesunięcie w tłumaczeniu binarnym następuje o bit, a nie o literę tekstu bazowego. Kolejny przykład obrazuje próbę wyeliminowania tego uchybienia.

Przykład 5.4. Kodowanie binarne zastosowane w 5.1 i 5.3 nie jest deterministyczne. Oznacza to, iż różne litery mogą mieć innej długości zapis w postaci dwójkowej. Istnieje kilka metod umożliwiających wykluczenie takich przypadków. W niniejszej pracy wyeliminowałem całkowicie kodowanie i dostosowałem algorytm przedstawiony wcześniej do naturalnego alfabetu języka polskiego.

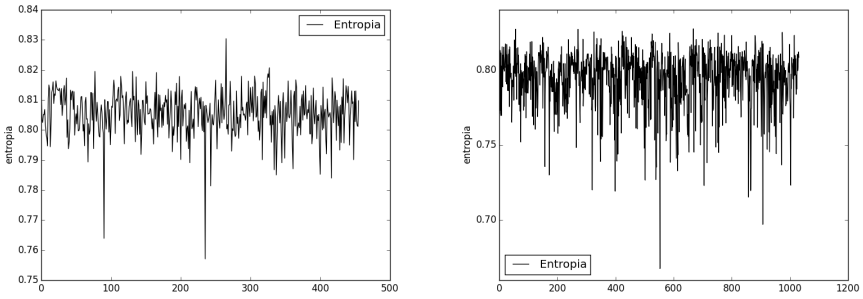
W programie zadeklarowana zostaje dodatkowa tablica zawierająca wszystkie litery:

```
13 polishAlphabet = [
14     'a', 'ą', 'b', 'c', 'ć', 'd',
15     'e', 'ę', 'f', 'g', 'h', 'i',
16     'j', 'k', 'l', 'ł', 'm', 'n',
17     'ń', 'o', 'ó', 'p', 'r', 's',
18     'ś', 't', 'u', 'w', 'y', 'z',
19     'ź', 'ż'
20 ]
```

W wypadku analizy niezakodowanych tekstów ujednoliceniu poddane zostają nie tylko znaki interpunkcyjne i łamania w tekście, ale również wielkości każdego symbolu (do tego celu użyto wbudowanej metody *lower*). Inaczej liczymy również dodatkowe zmienne takie jak prawdopodobieństwo wystąpienia znaków i entropia:

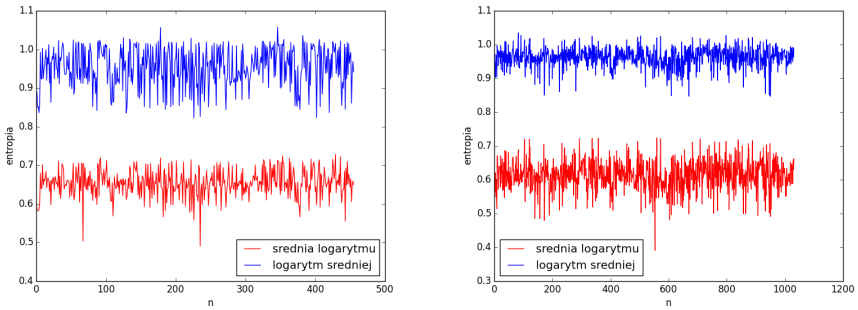
```
35 prob = [k.count(str(t)) / dlugoscTekstu for t in polishAlphabet]
36 entropy = sum([0 if t == 0 else -t * math.log(t, len(polishAlphabet))
37                 for t in prob])
```

Reszta kodu programu nie ulega większym zmianom. Cały kod znajduje się w 6.



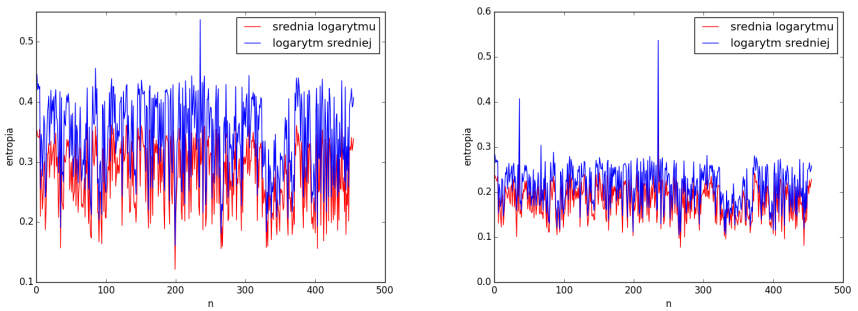
Rysunek 12. Entropia teoretyczna dla epiki (lewy) i liryki (prawy), policzona dla tekstów niezakodowanych binarnie.

Z powyższych rysunków wynika, że gdyby zadane teksty były generowane zupełnie losowo ich entropia oscylowałaby w okolicach 0.8 tak dla epiki jak i dla liryki.



Rysunek 13. Wyniki analizy epiki (lewy) i liryki (prawy), dla tekstów bez tłumaczenia na alfabet dwójkowy.

Ograniczenia zadane przez średnią logarytmów i logarytm średnich na powyższych szkicach zdają się pokazywać pełną losowość badanych tekstów. Wynika to stąd, iż w algorytmie po zmianach do badań brane są bardzo krótkie podciągi znaków, co wynika ze wzoru na Max_Block. Jest to kolejny problem, który należałoby przezwyciężyć. Najprostszym sposobem jest tutaj podanie jednej konkretnej długości podciągu.



Rysunek 14. W przypadku tekstów epiki ustalono wartość Max_Block na 10 (lewy) i 15 (prawy).

Analizowana entropia wyraźnie się poprawiła, jednak widać brak jej stabilizacji - wynika to z dowodu twierdzenia 9.

Powyższe przykłady mogą z powodzeniem zostać zastosowane do analizy innych źródeł języka polskiego. Literatura stanowi jedynie pewien reprezentatywny wycinek.

W przypadku stenogramów programów telewizyjnych, forów internetowych czy dramatów w języku polskim, głównym problemem nie powinno być obliczenie zadanej miary a obróbka badanego tekstu.

Zasygnalizowane problemy wymagają wypracowania odpowiednich metod, jednak w dalszej perspektywie powinny zostać rozwiązane w ramach znanego już dziś warsztatu matematyki i informatyki.

6. Aneks

Poniżej znajdują się pełne kody programów użytych w pracy. Repozytorium dostępne jest również pod adresem: https://github.com/GitHub098123/Entropy.

6.1. FirstReturnTime.py

```
import numpy as np
import matplotlib.pyplot as plt
import math
import random
import time

k = 5
prob = 1. / (k + 1)
entropy = -(1. - prob) * math.log(1. - prob, 2.) \
          - prob * math.log(prob, 2.)

f = open(''.join(['FirstReturnTime/FirstTime', 'Prawdopodobienstwo1_',
        str(prob), '_', str(int(time.time())), '.log']), 'w')
print(entropy, file=f)

Max_Block = 15
N = round(2 ** (
    entropy * Max_Block))
print(N, file=f)

x = list()
for i in range(1, N + 1):
    x.append(math.trunc(random.randint(0, k) / k))

empiricalProb = x.count(1) / N
print(empiricalProb, file=f)

seq = x[0: Max_Block]
print(seq, file=f)

t = seq.count(1) / Max_Block
print(t, file=f)

fullBinarySeries = ''.join([str(tempX) for tempX in x])
tempR = list()
for i in range(0, Max_Block - 1):
    ReturnIndex = fullBinarySeries.find(fullBinarySeries[0:i + 1],
                                        1)
    if ReturnIndex != -1:
        tempR.append(ReturnIndex)
    else:
        break

print(tempR, file=f)
```

```
45
46 # Algorytm z pracy Cho - nieoptymalny w Pythonie
47 # R = list()
48 # k = 1
49 # R.append(k)
50 # for n in range(0, Max_Block - 1):
51 #     if x[n] != x[n + 1]:
52 #         k += 1
53 #     R.append(k)
54 # print(R)
55
56 g1 = {i: abs(math.log(tempR[i - 1], 2.)) / i
57       for i in range(1, len(tempR))}
58 print(g1, file=f)
59
60 plt.plot(np.arange(1, len(tempR), 1), list(g1.values()))
61 plt.axhline(entropy, lw=2, color='black', label=str(entropy))
62 plt.ylabel('entropia')
63 plt.xlabel('n')
64 plt.legend(loc='best')
65 plt.savefig(''.join(['FirstReturnTime/FirstTime',
66                      'Prawdopodobienstwo1_',
67                      str(prob), '_', str(int(time.time())),
68                      '.png']))
69 f.close()
```

6.2. AveragesReturnTimes.py

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  import math
4  import random
5  import time
6
7  k = 9
8  prob = 1. / (k + 1)
9  entropy = -(1. - prob) * math.log(1. - prob, 2.)\
10                   - prob * math.log(prob, 2.)
11
12 f = open(''.join(['AvReturnTime/AveTimes', 'Prawdopodobienstwo1_',
13                 str(prob), '_', str(int(time.time())),
14                 '.log']), 'w')
15 print(entropy, file=f)
16
17 Max_Block = 20
18 N = round(2 ** (
19     entropy * Max_Block)) * 200
20 print(N, file=f)
21
22 x = list()
23 for i in range(1, N + 1):
24     x.append(math.trunc(random.randint(0, k) / k))
25
26 t = x.count(1) / N
27 print(t, file=f)
28
29 binarySeries = ''.join([str(tempX) for tempX in x])
30
31 S = 5000
32 M = N - Max_Block - S + 2
33 print(M, file=f)
34
35 R = [list() for _ in range(0, S)]
```

```
''' Szukamy pierwszego powrotu n-bloku od s-tego miejsca '''
for i in range(0, S):
    for b in range(0, Max_Block):
        ReturnIndex = binarySeries.find(binarySeries[i:i + b + 1],
                                                              i + 1)
        if ReturnIndex != -1:
            R[i].append(ReturnIndex - i)
        else:
            break
            #R[i].append(N)

AveragesOfR = [0 for _ in range(0, Max_Block)]
for i in range(0, Max_Block):
    tempAve = list()
    for j in range(0, S):
        if len(R[j]) > i:
            tempAve.append(R[j][i])
    AveragesOfR[i] = sum(tempAve) / len(tempAve)

g1 = {i: abs(math.log(AveragesOfR[i - 1], 2.)) / i
      for i in range(1, len(AveragesOfR))}
print(g1, file=f)

plt.plot(np.arange(1, len(AveragesOfR), 1), list(g1.values()),
                label='logarytm srednich', lw=3)
plt.axhline(entropy, lw=2, color='black', label=str(entropy))
plt.ylabel('entropia')
plt.xlabel('n')


AveLog = [0 for _ in range(0, Max_Block)]
for i in range(0, Max_Block):
    tempAveLog = list()
    for j in range(0, S):
        if len(R[j]) > i:
            tempAveLog.append(R[j][i])
    temp = [abs(math.log(tempAveLog[t], 2.)) / (i + 1)
            for t in range(0, len(tempAveLog))]
    if sum(temp) == 0 or len(temp) == 0:
        continue
    else:
        AveLog[i] = sum(temp) / len(temp)

g2 = {i: AveLog[i] for i in range(1, len(AveLog))}
print(g2, file=f)
plt.plot(np.arange(1, len(AveLog), 1), list(g2.values()),
                color="red",
        label='srednie logarytmu',
        lw=3)
plt.legend(loc='best')
plt.savefig(''.join(['AvReturnTime/AveTimes', 'Prawdopodobienstwo1_',
                str(prob), '_', str(int(time.time())),
                '.png']))
```

6.3. Analyze.py

```
# -*- coding: utf-8 -*-
import math
import re
import glob

import numpy as np
import matplotlib.pyplot as plt
```

```
8
9  pattern = re.compile('([^\s\w]|_)+')
10 Teksty = glob.glob('Liryka/*.txt')
11 # Teksty = glob.glob('Epika/*.txt')
12 AvgEntropy = list()
13 LogAvgEntropy = list()
14
15 for tekst in Teksty:
16     helper = re.search('\\\(.*)\.txt', tekst, flags=0)
17     toAnalyze = open(''.join(['Liryka/', helper.group(1), '.txt']),
18                                 'r', encoding='utf-8')
19
20     f = open(''.join(['TekstyLirykaAnaliza/', helper.group(1), '.log']),
21                 'w')
22     k = "".join(line for line in toAnalyze if not line.isspace())
23     k = pattern.sub('', k)
24
25     binary = ' '.join(format(ord(x), 'b') for x in k).replace(' ', '')
26     dlugoscTekstu = len(binary)
27     print(dlugoscTekstu, file=f)
28
29     prob = binary.count('1') / dlugoscTekstu
30     entropy = -(1. - prob) * math.log(1. - prob, 2.) \
31               - prob * math.log(prob, 2.)
32     print(prob, file=f)
33     print(entropy, file=f)
34     print(abs(math.log(dlugoscTekstu, 2. * entropy)), file=f)
35
36     Max_Block = math.ceil(abs(math.log(dlugoscTekstu, 2. * entropy)))
37     S = 5000
38     R = [list() for _ in range(0, S)]
39     AveragesOfR = [0 for _ in range(0, Max_Block)]
40
41     ''' Szukamy pierwszego powrotu n-bloku od s-tego miejsca '''
42     for i in range(0, S):
43         for b in range(0, Max_Block):
44             ReturnIndex = binary.find(binary[i:i + b + 1], i + 1)
45             if ReturnIndex != -1:
46                 R[i].append(ReturnIndex - i)
47             else:
48                 break
49
50     AveragesOfR = [0 for _ in range(0, Max_Block)]
51     for i in range(0, Max_Block):
52         tempAve = list()
53         for j in range(0, S):
54             if len(R[j]) > i:
55                 tempAve.append(R[j][i])
56         AveragesOfR[i] = sum(tempAve) / len(tempAve)
57
58     g1 = {i: abs(math.log(AveragesOfR[i - 1], 2.)) / i
59           for i in range(1, len(AveragesOfR))}
60     print(g1, file=f)
61
62     plt.plot(np.arange(1, len(AveragesOfR), 1), list(g1.values()),
63                          label='logarytm srednich', lw=2)
64
65     AveLog = [0 for _ in range(0, Max_Block)]
66     for i in range(0, Max_Block):
67         tempAveLog = list()
68         for j in range(0, S):
69             if len(R[j]) > i:
70                 tempAveLog.append(R[j][i])
71         temp = [abs(math.log(tempAveLog[t], 2.)) / (i + 1)
```

```
                        for t in range(0, len(tempAveLog))]
        if sum(temp) == 0 or len(temp) == 0:
            continue
        else:
            AveLog[i] = sum(temp) / len(temp)

    g2 = {i: AveLog[i] for i in range(1, len(AveLog))}
    print(g2, file=f)
    plt.plot(np.arange(1, len(AveLog), 1), list(g2.values()),
                label='srednia logarytmu', color="red", lw=2)
    plt.ylabel('entropia')
    plt.xlabel('n')
    plt.legend(loc='best')
    plt.savefig(''.join(['TekstyLirykaAnaliza/',
                helper.group(1), '.png']))
    plt.close('all')
    AvgEntropy.append(g2[len(g2)-1])
    LogAvgEntropy.append(g1[len(g1)-1])
    f.close()

newFile = open(''.join(['FinalResultForLiryka.log']), 'w')
print(np.arange(1, len(AvgEntropy), 1))
print(AvgEntropy, file=newFile)
print(LogAvgEntropy, file=newFile)
plt.plot(np.arange(0, len(AvgEntropy), 1), AvgEntropy,
                label='srednia logarytmu', color="red", lw=1)
plt.plot(np.arange(0, len(LogAvgEntropy), 1), LogAvgEntropy,
                label='logarytm sredniej', color="blue", lw=1)
plt.ylabel('entropia')
plt.xlabel('n')
plt.legend(loc='best')
plt.savefig(''.join(['FinalResultForLiryka.png']))
plt.close('all')
newFile.close()
```

6.4. AnalyzeV2.py

```
# -*- coding: utf-8 -*-
import math
import re
import glob

import numpy as np
import matplotlib.pyplot as plt

pattern = re.compile('([^\s\w]|_)+')
Teksty = glob.glob('Epika/*.txt')
AvgEntropy = list()
LogAvgEntropy = list()
polishAlphabet = [
    'a', 'ą', 'b', 'c', 'ć', 'd',
    'e', 'ę', 'f', 'g', 'h', 'i',
    'j', 'k', 'l', 'ł', 'm', 'n',
    'ń', 'o', 'ó', 'p', 'r', 's',
    'ś', 't', 'u', 'w', 'y', 'z',
    'ź', 'ż'
]

for tekst in Teksty:
    helper = re.search('\\\(.*)\.txt', tekst, flags=0)
    toAnalyze = open(''.join(['Epika/', helper.group(1), '.txt']),
                                    'r', encoding='utf-8')

```

```
27      f = open(''.join(['TestEpika32litery/', helper.group(1), '.log']), 'w')
28
29      k = "".join(line for line in toAnalyze if not line.isspace())
30      k = k.lower() #wielkość liter ma znaczenie
31      k = pattern.sub('', k)
32      dlugoscTekstu = len(k)
33      print(dlugoscTekstu, file=f)
34
35      prob = [k.count(str(t)) / dlugoscTekstu for t in polishAlphabet]
36      entropy = sum([0 if t == 0 else -t * math.log(t, len(polishAlphabet))
37                             for t in prob])
38      print(prob, file=f)
39      print(entropy, file=f)
40      print(abs(math.log(dlugoscTekstu, len(polishAlphabet) * entropy)), file=f)
41
42      Max_Block = math.ceil(abs(math.log(dlugoscTekstu,
43                                                 len(polishAlphabet) * entropy)))
44      S = 5000
45      R = [list() for _ in range(0, S)]
46      AveragesOfR = [0 for _ in range(0, Max_Block)]
47
48      for i in range(0, S):
49          for b in range(0, Max_Block):
50              ReturnIndex = k.find(k[i:i + b + 1], i + 1)
51              if ReturnIndex != -1:
52                  R[i].append(ReturnIndex - i)
53              else:
54                  break
55
56      AveragesOfR = [0 for _ in range(0, Max_Block)]
57      for i in range(0, Max_Block):
58          tempAve = list()
59          for j in range(0, S):
60              if len(R[j]) > i:
61                  tempAve.append(R[j][i])
62          if not tempAve:
63              AveragesOfR = AveragesOfR[0:i - 1]
64              break
65          else:
66              AveragesOfR[i] = sum(tempAve) / len(tempAve)
67
68      g1 = {i: abs(math.log(AveragesOfR[i - 1], len(polishAlphabet))) / i
69            for i in range(1, len(AveragesOfR))}
70      print(g1, file=f)
71
72      plt.plot(np.arange(1, len(AveragesOfR), 1),
73                  list(g1.values()), label='logarytm srednich', lw=2)
74
75      AveLog = [0 for _ in range(0, Max_Block)]
76      for i in range(0, Max_Block):
77          tempAveLog = list()
78          for j in range(0, S):
79              if len(R[j]) > i:
80                  tempAveLog.append(R[j][i])
81          temp = [abs(math.log(tempAveLog[t], len(polishAlphabet)))
82                          / (i + 1) for t in range(0, len(tempAveLog))]
83          if sum(temp) == 0 or len(temp) == 0:
84              AveLog = AveLog[0:i - 1]
85              continue
86          else:
87              AveLog[i] = sum(temp) / len(temp)
88
89      g2 = {i: AveLog[i] for i in range(1, len(AveLog))}
90      print(g2, file=f)
```

```
91      plt.plot(np.arange(1, len(AveLog), 1),
92                   list(g2.values()),
93               label='srednia logarytmu',
94               color="red",
95               lw=2)
96      plt.ylabel('entropia')
97      plt.xlabel('n')
98      plt.legend(loc='best')
99      plt.savefig(''.join(['TestEpika32litery/',
100                                          helper.group(1),
101                          '.png']))
102     plt.close('all')
103     AvgEntropy.append(g2[1 if len(g2) == 1 else len(g2) - 1])
104     LogAvgEntropy.append(g1[1 if len(g1) == 1 else len(g1) - 1])
105     f.close()
106
107 newFile = open(''.join(['TestEpika32litery.log']), 'w')
108 print(np.arange(1, len(AvgEntropy), 1))
109 print(AvgEntropy, file=newFile)
110 print(LogAvgEntropy, file=newFile)
111 plt.plot(np.arange(0, len(AvgEntropy), 1),
112          AvgEntropy,
113          label='srednia logarytmu',
114          color="red",
115          lw=1)
116 plt.plot(np.arange(0, len(LogAvgEntropy), 1),
117          LogAvgEntropy,
118          label='logarytm sredniej',
119          color="blue",
120          lw=1)
121 plt.ylabel('entropia')
122 plt.xlabel('n')
123 plt.legend(loc='best')
124 plt.savefig(''.join(['TestEpika32litery.png']))
125 plt.close('all')
126 newFile.close()
```

6.5. Crawler

Kod robota pobierającego teksty ze strony projektu wolne lektury - w tym przypadku dzieł z rodzaju liryki:

```
1  # -*- coding: utf-8 -*-
2  import requests
3  from bs4 import BeautifulSoup
4  import re
5
6
7  def liryka_spider(max_pages=2):
8      page = 1
9      while page < max_pages:
10         url = 'http://wolnelektury.pl/katalog/rodzaj/liryka/?page=' \
11               + str(page)
12         source_code = requests.get(url)
13         plain_text = source_code.text
14         soup = BeautifulSoup(plain_text, "html.parser")
15         href = list()
16         for elem in soup(text=re.compile(r'TXT')):
17             OnlyPolishTexts = \
18                 elem.parent.parent.parent.parent.parent.parent
19             if str(OnlyPolishTexts).find(u'Język:') == -1:
20                 href.append('https://wolnelektury.pl'
21                             + elem.parent.get('href'))
22         print(href)
```

```
23          page += 1
24          get_text(href)
25
26
27  def get_text(href_list):
28      for href in href_list:
29          source_code = requests.get(href)
30          LastSign = source_code.text.rfind('-----')
31          groups = re.search('txt/(.*)\.txt', href, flags=0)
32          f = open(''.join(['H:/Teksty/Liryka/', groups.group(1), '.txt']),
33                   'w',
34                   encoding='utf-8')
35          for t in range(0, LastSign):
36              f.write(source_code.text[t])
37          #f.write(source_code.text[0: LastSign]) # lepiej
38
39  liryka_spider(114)
```

Nie jest to kod w żaden sposób zoptymalizowany, gdyż strumień danych zapisywany jest do pliku po jednym znaku.

Spis treści

Literatura

[1] Manfred Einsiedler i Thomas Ward, *Ergodic Theory with a view towards Number Theory.* © Springer-Verlag London Limited 2011

[2] Manfred Einsiedler, Elon Lindenstrauss i Thomas Ward, *Entropy in dynamics.*Link do pracy: http://maths.dur.ac.uk/ tpcc68/entropy/welcome.html

[3] Geon Ho Choe, *Computational Ergodic Theory.* © Springer-Verlag Berlin Heidelberg 2005

[4] Patrick Billingsley, *Ergodic Theory and Information.* Wiley, New York, 1965.

[5] Karl E. Petersen, *Ergodic Theory.* Cambridge University Press, Cambridge, 1983.

[6] Kai-lai Chung, *A Note on the Ergodic Theorem of Information Theory.* The Annals of Mathematical Statistics, Volume 32, Number 2 (1961).

[7] Tomasz Downarowicz, *Entropy in Dynamical Systems.* Cambridge University Press, Cambridge, 2011.

[8] George Polya i Gabor Szego, *Problems and Theorems in Analysis I.* © Springer-Verlag Berlin Heidelberg 1998

[9] Dokumentacja pakietu numpy: http://docs.scipy.org/doc/

[10] Dokumentacja pakietu matplotlib: http://matplotlib.org/contents.html

[11] Dokumentacja modułu math: https://docs.python.org/3/library/math.html

[12] Dokumentacja modułu random: https://docs.python.org/3/library/random.html

[13] Dokumentacja modułu time: https://docs.python.org/3/library/time.html

[14] Dokumentacja modułu glob:https://docs.python.org/2/library/glob.html

Klasyfikacja AMS

94A17 Measures of information, entropy

37A35 Entropy and other invariants, isomorphism, classification

37M25 Computational methods for ergodic theory (approximation of invariant measures, computation of Lyapunov exponents,entropy)

37B40 Topological entropy