Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie

Wydział Informatyki, Elektroniki i Telekomunikacji

KATEDRA INFORMATYKI

PRACA MAGISTERSKA

ALEKSANDER SURMAN

EWOLUCYJNIE GENEROWANA POEZJA

PROMOTOR:
dr inż. Aleksander Smywiński-Pohl

Kraków 2017

OŚWIADCZENIE AUTORA PRACY

UPRZEDZONY O ODPOWIEDZIALNOŚCI KARNEJ NA PODSTAWIE ART. 115 UST. 1 I 2 USTAWY Z DNIA 4 LUTEGO 1994 R. O PRAWIE AUTORSKIM I PRAWACH POKREWNYCH (T.J. DZ.U. Z 2006 R. NR 90, POZ. 631 Z PÓŹN. ZM.): "KTO PRZYWŁASZCZA SOBIE AUTORSTWO ALBO WPROWADZA W BŁĄD CO DO AUTORSTWA CAŁOŚCI LUB CZĘŚCI CUDZEGO UTWORU ALBO ARTYSTYCZNEGO WYKONANIA, PODLEGA GRZYWNIE, KARZE OGRANICZENIA WOLNOŚCI ALBO POZBAWIENIA WOLNOŚCI DO LAT 3. TEJ SAMEJ KARZE PODLEGA, KTO ROZPOWSZECHNIA BEZ PODANIA NAZWISKA LUB PSEUDONIMU TWÓRCY CUDZY UTWÓR W WERSJI ORYGINALNEJ ALBO W POSTACI OPRACOWANIA, ARTYSTYCZNE WYKONANIE ALBO PUBLICZNIE ZNIEKSZTAŁCA TAKI UTWÓR, ARTYSTYCZNE WYKONANIE, FONOGRAM, WIDEOGRAM LUB NADANIE.", A TAKŻE UPRZEDZONY O ODPOWIEDZIALNOŚCI DYSCYPLINARNEJ NA PODSTAWIE ART. 211 UST. 1 USTAWY Z DNIA 27 LIPCA 2005 R. PRAWO O SZKOLNICTWIE WYŻSZYM (T.J. DZ. U. Z 2012 R. POZ. 572, Z PÓŹN. ZM.) "ZA NARUSZENIE PRZEPISÓW OBOWIĄZUJĄCYCH W UCZELNI ORAZ ZA CZYNY UCHYBIAJĄCE GODNOŚCI STUDENTA STUDENT PONOSI ODPOWIEDZIALNOŚĆ DYSCYPLINARNĄ PRZED KOMISJĄ DYSCYPLINARNĄ ALBO PRZED SĄDEM KOLEŻEŃSKIM SAMORZĄDU STUDENCKIEGO, ZWANYM DALEJ "SĄDEM KOLEŻEŃSKIM", OŚWIADCZAM, ŻE NINIEJSZĄ PRACĘ DYPLOMOWĄ WYKONAŁEM(-AM) OSOBIŚCIE, SAMODZIELNIE I ŻE NIE KORZYSTAŁEM(-AM) ZE ŹRÓDEŁ INNYCH NIŻ WYMIENIONE W PRACY.

.....

PODPIS

Serdecznie dziękuję dr inż. Aleksandrowi Smywińskiemu-Pohlowi, mgr inż. Łukaszowi Faberowi oraz mgr inż. Krzysztofowi Wróblowi za ich wkład w badania. Dodatkowo, chciałbym podziękować zespołowi PL-Grid za udostępnienie infrastruktury potrzebnej do obliczeń oraz rodzinie i bliskim za okazane wsparcie podczas pisania tej pracy.

Spis treści

1. Wprowadzenie

Przetwarzanie języka naturalnego (ang. *natural language processing* - NLP), zwane czasem inżynierią lingwistyczną, jest dziedziną nauki, stanowiącą połączenie sztucznej inteligencji i lingwistyki komputerowej. Zajmuje się ona automatyzacją tworzenia i przetwarzania tekstów w języku naturalnym, w formie mówionej i pisanej.

NLP jest obecnie tematem wielu badań naukowych, które prowadzą do odnajdywania coraz to nowszych praktycznych zastosowań inżynierii lingwistycznej. Powstają aplikacje, które ułatwiają osobom niepełnosprawnym dostęp do informacji, na przykład poprzez generowanie instrukcji werbalnych dla osób niewidomych[1]. Tworzone są programy do nauki języków obcych, korzystające z dorobku tej dziedziny wiedzy[2]. Prowadzone są również poszukiwania alternatywnych rozwiązań dla tradycyjnych infolinii[3], które mimo ciągłego usprawniania moją swoje liczne ograniczenia, np. długi czas oczekiwania na połączenie czy ograniczony czas pracy konsultantów.

Lepsze rozumienie języka naturalnego przez komputer pozwala także na udoskonalenie autokorekt w różnego rodzaju edytorach tekstu. Bardzo ciekawym przykładem jest również analiza sentymentu[4] wpisów na portalach społecznościowych, pozwalająca na wydobycie wiedzy o aktualnych nastrojach społecznych. Wymienione zastosowania to oczywiście tylko kilka przykładów z bogatego dorobku prac nad przetwarzaniem języka naturalnego.

W niniejszej pracy skupiono się na poddziedzinie NLP, jaką jest generowanie języka naturalnego. Tworzenie przy pomocy komputera takiego tekstu, który jest łatwy do zrozumienia dla człowieka oraz poprawny gramatycznie, jest nie lada wyzwaniem i stanowi podstawę interakcji człowiek - komputer. Szczególną trudność stanowi wygenerowanie tekstu, który poza podstawowymi wymaganiami poprawności językowej, spełniać będzie również inne kryteria, takie jak rymiczność czy posiadanie określonej formy. Takim zadaniem jest z pewnością automatyczne generowanie poezji, czym postanowiono zająć się w niniejszej pracy.

Języki różnią się pomiędzy sobą zasadami gramatyki oraz stopniem skomplikowania tych zasad. Zatem problem generowania tekstu jest różny w przypadku poszczególnych języków. Z tego też powodu dotychczasowe badania w tej dziedzinie skupiały się głównie na językach o najmniej złożonej gramatyce, takich jak język angielski czy hiszpański. Język polski bez wątpienia jest jednym z trudniejszych języków, mając jednak na względzie możliwości zastosowania NLP w przyszłości, w poniższej pracy postanowiono zbadać problem generowania poezji w języku polskim.

1.1. Cel pracy

Celem niniejszej pracy jest opracowanie i implementacja algorytmu ewolucyjnego do generowania poezji w języku polskim. Z założenia generowany tekst powinien być poprawny gramatycznie oraz wykazywać cechy charakterystyczne dla poezji, w tym przypadku skupiono się na rymie wiersza, a także na określonej liczbie sylab w wersie.

1.2. Zawartość pracy

Pracę podzielono w następujący sposób:

- rozdział 2 zawiera wstęp merytoryczny,
- rozdział 3 opisuje proponowane rozwiązania,
- rozdział 4 przedstawia uzyskane wyniki,
- rozdział 5 prezentuje badawcze plany na przyszłość,
- rozdział 6 jest podsumowaniem pracy.

2. Wstęp merytoryczny

Rozdział ten został podzielony na trzy główne części. Pierwsza jest przeglądem dotychczasowej literatury związanej z problemem generowania poezji. Nawiązując do pracy Manurunga[5] przedstawiono cztery zdefiniowane przez niego podejścia do tego problemu. Jest to: mieszanka słów (ang. *word salad*), systemy oparte na szablonie i gramatyce (ang. *template and grammar-based*), świadome formy (ang. *form-aware*) oraz generatory poezji (ang. *poetry generation*). Dokonano ich przeglądu, podając ich charakterystyki oraz przykładowe wyniki dla wybranych z danej kategorii systemów.

Następnie zaprezentowano podejście ewolucyjne, którego zastosowanie dla problemu generowania tekstów literackich w języku polskim stanowi zasadniczy przedmiot dociekań w niniejszej pracy. Zostało ono wykorzystane jako sposób przeszukiwania przestrzeni możliwych rozwiązań, w tym przypadku wierszy.

Na koniec opisany został N-gramowy model językowy. W niniejszej pracy posłużono się nim w celu określania prawdopodobieństwa wystąpienia zdania (wersu wiersza) w języku polskim, co za tym idzie przybliżonego prawdopodobieństwa poprawności gramatycznej wygenerowanego zdania.

2.1. Dotychczasowe podejścia do problemu generowania poezji

W swojej pracy doktorskiej Manurung[5] przedstawił listę warunków, które musi spełniać dzieło literackie, aby można było określić je mianem poezji. Są to:

- **gramatyczność (ang. *grammaticality*)** - tekst jest poprawny gramatycznie,
- **zawartość treści (ang. *meaningfulness*)** - tekst zawiera pewien przekaz,
- **poetyckość (ang. *poeticness*)** - tekst wykazuje cechy charakterystyczne dla poezji.

Manurung opisał w niej także sposób klasyfikowania generatorów tworzących poezję. Każda grupa, wraz z przykładowymi systemami, została pokrótce scharakteryzowana poniżej. Uporządkowano je w kolejności od najprostszych i najmniej poprawnych, do najbardziej złożonych i cechujących się największą poprawnością.

2.1.1. Mieszanka słów

W systemach tej grupy tekst powstaje poprzez łączenie losowych słów w dowolnej formie. Nie spełniają one jednak żadnego z kryteriów postawionych przez Manurunga. Poniżej zaprezentowano utwór, który wygenerowany został przez należący do tej grupy system *LYRIC 3205*[6]:

judy gotta want upon someone.
wanna sadly will go about.

sammy gotta want the thief him but the
every reason. real distance carry.

2.1.2. Systemy oparte na szablonie i gramatyce

Systemy te tworzy się w następujący sposób: w zadanym szablonie umieszcza się luki, w które kolejno wprowadzane są słowa. Słowa te dobierane są ze słownika w taki sposób, aby uzupełniając luki, nie stracić poprawności gramatycznej. Takie systemy spełniają jedynie warunek gramatyczności. Za przykład tak zbudowanego generatora może posłużyć *Masterman*[7]. Poniżej zaprezentowano przykładowy szablon i wynik tego systemu:

All [1] in the [2]	*All green in the leaves*
I [3] [4] [5] in the [6]	*I smell dark pools in the trees*
[7] the [8] has [9]	*Crash the moon has fled*

2.1.3. Świadome formy

W tej grupie systemów słowa dobierane są tak, aby pasowały do zdefiniowanej formy poetyckiej np. haiku[1]. Dodatkowo generowany tekst ma być poprawny gramatycznie, a zatem w systemach tych spełnione są warunki gramatyczności i poetyckości. Przykładami takich systemów są: *WASP*[8], *POEVOLVE*[9], czy komercyjny *Ray Kurzweil's Cybernetic Poet*[10]. Wynik działania ostatniego z nich zaprezentowano poniżej:

Scattered sandals
a call back to myself,
so hollow I would echo

Crazy moon child
Hide from your coffin
To spite your doom.

You broke my soul
the juice of eternity,
the spirit of my lips

[1] Japońska forma poetycka, gdzie wers składa się z trzech części znaczeniowych po 5, 7 i 5 sylab.

2.1.4. Generatory poezji

Dla tych systemów generowany tekst musi spełniać wszystkie trzy warunki zaproponowane przez Manurunga: gramatyczności, poetyckości oraz zawierania treści. Generatory: *ASPERA*[11], *COLIBRI*[12] oraz *MCGONAGALL*[5] można zaliczyć do tej kategorii.

ASPERA i COLIBRI

ASPERA i *COLIBRI* są to dwa bardzo podobne systemy, służące do generowania formalnej poezji w języku hiszpańskim. W obu przypadkach jako wejście podawana jest treść, jaką ma przekazywać generowany tekst oraz informacja, w jakiej formie ma być on tworzony.

Oba algorytmy wykorzystują metodę wnioskowania na podstawie przypadków (ang. *case-based reasoning*), która polega na generowaniu nowego rozwiązania poprzez wyszukanie w bazie rozwiązania dla podobnego przypadku. Podejście to oparte jest na czterech cyklach: **odzyskanie**, **ponowne użycie**, **poprawienie** oraz **zachowanie**. Dla obu tych algorytmów poszczególne kroki wyglądają następująco:

Odzyskanie (ang. *retrieve step*) - dla każdego zdania z podanej na wejściu treści wybierany jest najbardziej pasujący wers z korpusu wersów.

Ponowne użycie (ang. *reuse step*) - z wybranego wersu tworzony jest szablon, który wypełniany jest słowami.

Poprawienie (ang. *revise step*) - wypełniony szablon prezentowany jest użytkownikowi w celu weryfikacji i ewentualnego poprawienia.

Zachowanie (ang. *retain step*) - zaakceptowany wers dodawany jest do bazy wersów do ponownego użycia.

MCGONAGALL

W swojej pracy Manurung[5] sformułował problem generowania poezji, jako problem przeszukiwania przestrzeni stanów. W podejściu tym przeszukiwaną przestrzenią jest dowolny możliwy tekst z całą jego reprezentacją, od semantycznej do fonetycznej. Ruch wykonać można na jakimkolwiek poziomie jego reprezentacji. Tak, jak w wyżej opisanych algorytmach *ASPERA* i *COLIBRI*, jako wejście podawana jest treść, jaką ma przekazywać generowany tekst. Tutaj jednak wykorzystano płaską reprezentację semantyczną, przykładowo:

john(j), marry(m), love(l, j, m)

oznacza:

John loves Marry

Algorytm MCGONAGALL do przeszukiwania przestrzeni rozwiązań wykorzystuje podejście ewolucyjne (omówione w rozdziale 2.2). Początkowa populacja tworzona jest na podstawie danych podanych na wejściu. Funkcja przystosowania wyliczana jest z kolei na podstawie różnych aspektów, takich jak

forma fonetyczna czy semantyka. Mutacja wystąpić może na dowolnym poziomie reprezentacji, z dbałością o zachowanie spójności tekstu. Poniżej znajduje się przykładowy wynik algorytmu *MCGONAGALL*[5]:

They play. An expense is a waist.
A lion, he dwells in a dish.
He dwells in a skin.
A sensitive child,
he dwells in a child with a fish.

2.2. Podejście ewolucyjne

W niniejszej pracy postanowiono wykorzystać podejście genetyczne, będące heurystycznym algorytmem przeszukującym przestrzeń rozwiązań, należące do klasy algorytmów ewolucyjnych. W podejściu genetycznym symulowana jest pewna populacja osobników, której celem jest jak najlepsze dopasowanie się do środowiska. Środowisko to zdefiniowane zostaje na podstawie zadanego problemu. Każdy osobnik reprezentuje pewne rozwiązanie problemu, a parametry tego rozwiązania są zakodowane przy pomocy poszczególnych genów danego osobnika.

Dla zadanego problemu definiuje się także funkcję przystosowania, określającą stopień przydatności danego rozwiązania (osobnika). W pierwszej kolejności losowo inicjalizowana jest populacja składająca się z określonej liczby osobników. Następnie symulowany jest rozwój tej populacji, czyli tworzenie potomstwa, mutacje oraz obumieranie nieprzystosowanych osobników.

Przebieg algorytmu genetycznego

W podejściu ewolucyjnym można spotkać się z różnymi algorytmami postępowania. Poniżej zaprezentowano ten, który wykorzystano dla potrzeb niniejszej pracy.

1. Wygenerowanie początkowej populacji, składającej się z określonej ilości osobników.
2. Wyliczenie wartości funkcji przystosowania dla każdego osobnika w populacji.
3. Rozszerzenie populacji o nowe osobniki, powstałe wskutek krzyżowania się osobników z dotychczasowej populacji.
4. Mutacja osobników.
5. Selekcja osobników - nowa populacja składa się z takiej samej ilości osobników, jak populacja początkowa.
6. Powrót do punktu 3., o ile nie zaistniał określony warunek stopu.

Dokładny opis, przedstawiający realizację wymienionych kroków, zaprezentowany zostanie w następnym rozdziale.

2.3. N-gramowy model językowy

W funkcji przystosowania opisanej szczegółowo w punkcie 3.2, jako część składową odpowiadającą za poprawność gramatyczną wiersza, wykorzystany został N-gramowy model językowy, opisany przez Martina Jurafsky'ego[13]. Model ten wykorzystuje się w celu określenia prawdopodobieństwa wystąpienia danego ciągu wyrazów w języku.

N-gram jest to zbiór N występujących po sobie słów w pojedynczym zdaniu. Przykładowo, dla zdania ''Ala ma kota.'' możliwe N-gramy to:

1-gramy (unigramy) - *Ala*, *ma*, *kota*
2-gramy (bigramy) - *Ala ma*, *ma kota*
3-gramy (trigramy) - *Ala ma kota*

Stworzenie takiego modelu językowego polega na zliczaniu N-gramów, występujących w zdaniach korpusu, tj. zbiorach tekstów. W celu jak najwierniejszego oddania właściwości danego języka, należy dobrać odpowiednio duży korpus. Teksty w nim zawarte powinny mieć także zróżnicowane źródła pochodzenia. Przykładowo, model nauczony jedynie na tekstach pochodzących z prac naukowych, stanowić będzie reprezentację cech żargonu naukowego, nie całego języka.

Wykorzystując metodę Maximum Likelihood Estimate (*MLE*), w celu przybliżenia prawdopodobieństwa wystąpienia określonego N-gramu w korpusie (języku), wystarczy policzyć stosunek wystąpień tego N-gramu, do ilości wszystkich N-gramów tej wielkości. Zatem możemy określić prawdopodobieństwo wystąpienia słowa (unigramu) w w języku, jako liczbę wystąpień w w stosunku do wszystkich możliwych unigramów:

$$P(w) = \frac{C(w)}{\|U\|} \tag{2.1}$$

Gdzie $C(w)$ oznacza ilość wystąpień unigramu w w korpusie, a $\|U\|$ ilość wszystkich unigramów.

Analogicznie możemy zdefiniować prawdopodobieństwo warunkowe wystąpienia słowa w_n po N-gramie (ciągu słów) $w_1, ..., w_{n-1}$ za pomocą wzoru:

$$P(w_n|w_1,\ ...,\ w_{n-1}) = \frac{C(w_1,\ ...,\ w_{n-1},\ w_n)}{C(w_1,\ ...,\ w_{n-1})} \tag{2.2}$$

Gdzie $C(w_1,\ ...,\ w_n)$ oznacza ilość wystąpień N-gramu, składającego się z wyrazów $w_1,\ ...,\ w_n$.

Stosując regułę łańcuchową możemy określić prawdopodobieństwo wystąpienia N-gramu $w_1, ..., w_n$, jako iloczyn prawdopodobieństw wystąpienia kolejnych słów, pod warunkiem, że wcześniejsze już wystąpiły:

$$P(w_1,\ ...,\ w_n) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)...P(w_n|w_1,\ ...,\ w_{n-1}) \tag{2.3}$$

Przechowywanie informacji o wszystkich występujących w korpusie N-gramach (dowolnej długości), wymagałoby zbyt dużej ilości pamięci i byłoby wysoce nieefektywne podczas procesu przeszukiwania. Dlatego też w N-gramowym modelu językowym, posłużono się pewnym przybliżeniem. Założono bowiem, że aby oszacować wartość prawdopodobieństwa wystąpienia następnego słowa, wystarczy znajomość jedynie $N-1$ słów go poprzedzających. Skutkiem takiego podejścia jest ograniczenie modelu językowego do uwzględniania maksymalnie N-elementowych N-gramów, co przekłada się na jego rozmiar.

Przykładowo, w modelu unigramowym prawdopodobieństwo wystąpienia ciągu wyrazów $w_1, ..., w_n$ określa się wzorem:

$$P(w_1,\ ...,\ w_n) = P(w_1)P(w_2)P(w_3)...P(w_n) \tag{2.4}$$

W modelu bigramowym:

$$P(w_1,\ ...,\ w_n) = P(w_1)P(w_2|w_1)P(w_3|w_2)...P(w_n|w_{n-1}) \tag{2.5}$$

Z kolei w modelu trigramowym:

$$P(w_1,\ ...,\ w_n) = P(w_1)P(w_2|w_1)P(w_3|w_1,w_2)...P(w_n|w_{n-2},w_{n-1}) \tag{2.6}$$

W celu zobrazowania działania wyżej opisanego modelu na konkretnym przykładzie, załóżmy że korpus składa się z następujących zdań:

Ala ma kota.
Ala ma psa.
Kasia ma papugę.

Model bigramowy wykorzystujący powyższy korpus, zdaniu "*Ala ma kota.*" przypisałby następujące prawdopodobieństwo:

$$P(\text{“}Ala\text{“}|<s>)P(\text{“}ma\text{“}|\text{“}Ala\text{“})P(\text{“}kota\text{“}|\text{“}ma\text{“}) = \frac{2}{3}\cdot 1\cdot\frac{1}{3} = \frac{2}{9} \tag{2.7}$$

Gdzie $<s>$ w oznacza początek zdania.

Natomiast dla zdania "*Kasia ma kota.*" prawdopodobieństwo to wynosiło będzie:

$$P(\text{“}Kasia\text{“}|<s>)P(\text{“}ma\text{“}|\text{“}Kasia\text{“})P(\text{“}kota\text{“}|\text{“}ma\text{“}) = \frac{1}{3}\cdot 1\cdot\frac{1}{3} = \frac{1}{9} \tag{2.8}$$

Dla zdania "*Basia ma kota.*" będzie to z kolei:

$$= P(\text{“}Basia\text{“}|<s>)P(\text{“}ma\text{“}|\text{“}Basia\text{“})P(\text{“}kota\text{“}|\text{“}ma\text{“}) = 0\cdot 1\cdot\frac{1}{3} = 0 \tag{2.9}$$

W niniejszej pracy posłużono się **modelem trigramowym**. Aby możliwe było porównywanie zdań o różnej długości, zamiast prawdopodobieństwa wystąpienia zdania - które zależne jest również od jego długości - posłużono się miarą *perplexity*. Jest to odwrotność pierwiastka n-tego stopnia z uzyskanego prawdopodobieństwa, gdzie n oznacza ilość wyrazów.

$$PP(w_1, \ ..., \ w_n) = \frac{1}{\sqrt[n]{P(w_1, \ ..., \ w_n)}} \tag{2.10}$$

3. Proponowane rozwiązanie

W niniejszej pracy postanowiono zaimplementować generator tworzący wiersze w języku polskim i wykorzystujący do tego celu podejście genetyczne. Jednak, z powodu dużego stopnia skomplikowania gramatyki języka polskiego, postanowiono pominąć w badaniach kryterium sensowności Manurunga[5], a zamiast tego skupiono się głównie na poprawności gramatycznej. Co za tym idzie, nie uwzględniono w generowanych utworach literackich interpunkcji, która także może wpłynąć na sens zdania. Takie działanie interpunkcji prezentuje poniższy przykład:

Rozstrzelać, nie wolno ułaskawić!
Rozstrzelać nie wolno, ułaskawić!

Dodatkowo, generowany tekst powinien posiadać także aspekt poetycki. Jako formę docelową generowanego tekstu wybrano zatem **czterowersowy trzynastozgłoskowiec z rymami parzystymi**[1].

3.1. Reprezentacja osobników

Jednym z kluczowych elementów algorytmu genetycznego jest zdefiniowanie reprezentacji pojedynczego osobnika, w tym przypadku wiersza. Przetestowane zostały dwa podejścia, w których gen odpowiada pojedynczemu słowu lub całemu wersowi.

Przykładowy wiersz, powstały z czterech pierwszych wersów Inwokacji *Pana Tadeusza* Adama Mickiewicza[14], brzmi:

Litwo! Ojczyzno moja! ty jesteś jak zdrowie:
Ile cię trzeba cenić, ten tylko się dowie,

Kto cię stracił. Dziś piękność twą w całej ozdobie
Widzę i opisuję, bo tęsknię po tobie. (...)

[1] Rymy występują w następujących po sobie wersach - układ AABB.

Dla genu odpowiadającemu pojedynczemu słowu, powyższy wiersz reprezentowany byłby następująco ("\n" oznacza koniec wersu):

[„Litwo!", „Ojczyzno", „moja!", „ty", „jesteś", „jak", „zdrowie:\n",
„Ile", „cię", „trzeba", „cenić,", „ten", „tylko", „się", „dowie,\n",

„Kto", „cię", „stracił.", „Dziś", „piękność", „twą", „w", „całej", „ozdobie\n",
„Widzę", „i", „opisuję,", „bo", „tęsknię", „po", „tobie.\n"]

Z kolei dla genu odpowiadającemu całemu wersowi, otrzymano by taką reprezentację:

[„Litwo! Ojczyzno moja! ty jesteś jak zdrowie:\n",
„Ile cię trzeba cenić, ten tylko się dowie,\n",

„Kto cię stracił. Dziś piękność twą w całej ozdobie\n",
„Widzę i opisuję, bo tęsknię po tobie.\n"]

3.2. Funkcja przystosowania

Funkcja przystosowania osobników (F) składa się z trzech opisanych poniżej części, a wartość funkcji stanowi odwrotność ich iloczynu.

$$F = \frac{1}{\mathrm{F}_S \cdot F_R \cdot F_G} \tag{3.1}$$

W celu ułatwienia interpretacji wyników poszczególne składowe F_S, F_R i F_G poddano normalizacji, dzięki czemu zawierają się one w przedziale $[1, \infty)$. W konsekwencji, wartość funkcji przystosowania F uzyskuje wyniki z przedziału $(0, 1]$, gdzie 1 oznacza największy stopień przystosowania osobnika.

3.2.1. Część sylabiczna (F_S)

Składowa sylabiczna określa w jakim stopniu spełniony jest warunek posiadania przez wiersz określonej liczby sylab w wersach. W celu określania liczby sylab w wersie, należy policzyć wszystkie samogłoski, pamiętając o tym, że "i" przed samogłoską traktowane jest jedynie jako zmiękczenie, a nie samodzielna głoska. Następnie, dla każdego wersu obliczana jest wartość bezwzględna różnicy pomiędzy ilością sylab, a referencyjną ich liczbą, w tym przypadku wartość ta ustalona została na 13. Wartością składowej sylabicznej jest średnia arytmetyczna uzyskanych różnic:

$$F_S = \frac{\sum_{i=1}^{N} \sum_{j=1}^{n_i} \left| V(w_{i,j}) - T \right|}{\mathrm{N}} \tag{3.2}$$

gdzie N oznacza ilość wersów, n_i oznacza ilość wyrazów w i-tym wersie, $V(w)$ oznacza liczbę samogłosek w wyrazie w, a T oznacza referencyjną liczbę sylab.

3.2.2. Część rymiczna (F_R)

Składowa rymiczna jest to miara rymowania się wiersza. Dla uproszczenia uznajemy, że dwa słowa się rymują, jeśli posiadają takie same końcówki, licząc od przedostatniej samogłoski. W przypadku kiedy słowo posiada tylko jedną samogłoskę, za końcówkę uznajemy całe słowo. Składowa rymiczna wyliczana jest na podstawie średniej arytmetycznej odległości edycyjnych pomiędzy końcówkami ostatnich wyrazów w następujących po sobie wersach, w tym przypadku pierwszego i drugiego oraz trzeciego i czwartego.

$$F_R = \frac{\sum_{i=1}^{\frac{N}{2}} lev(K_{2i}, K_{2i+1})}{\frac{N}{2}} \tag{3.3}$$

Gdzie N oznacza ilość wersów, $lev(a, b)$ odległość edycyjną (Levenshteina) pomiędzy wyrazami a i b, a K_i oznacza końcówkę dla i-tego wersu.

Przykładowo dla słów m**a-tka** i kwi**a-tka** odległość ta wynosi 0 (słowa się rymują), a dla m**a-tka** i są-si**a-dka** odległość wynosi 1, ponieważ nie uwzględnia się tu głosek podobnie brzmiących.

3.2.3. Część gramatyczna (F_G)

Do oceny poprawności gramatycznej postanowiono wykorzystać N-gramowy model językowy, opisany w podrozdziale 2.3. Składowa gramatyczna odpowiada średniej arytmetycznej z prawdopodobieństw wszystkich wersów w wierszu. Wyrazić można ją wzorem:

$$F_G = \frac{\sum_{i=1}^{N} PP(w_{i,1},\ ...,\ w_{i,n_i})}{N} = \frac{\sum_{n=1}^{N} \sqrt[n]{P(w_{i,1},\ ...,\ w_{i,n_i})}}{N} \tag{3.4}$$

Gdzie N oznacza ilość wersów, a $w_{i,1},\ ...,\ w_{i,n_i}$ słowa w i-tym wersie.

Na potrzeby pracy przetestowane zostały dwa modele. Pierwszym testowanym modelem był model językowy zbudowany na podstawie ręcznie znakowanego milionowego podkorpusu Narodowego Korpusu Języka Polskiego (NKJP)[15], gdzie zastosowano podejście opisane w pracy A. Smywińskiego-Pohla i B. Ziółki[16] - słowa zostały zastąpione odpowiadającymi im tagami morfosyntaktycznymi.

Tagi morfosyntaktyczne określają cechy gramatyczne, takie jak kategorie i klasy gramatyczne. Przykładowo, w zdaniu “Ania idzie do szkoły“, słowu Ania odpowiadać będzie zbiór znaczników *subst:nom:sg:f*, który oznacza, że słowo “Ania“ jest rzeczownikiem (*subst*) rodzaju żeńskiego (*f*), występującym tu w liczbie pojedynczej (*sg*) i odmienionym w pierwszym przypadku - mianowniku (*nom*). Pełny opis znaczników wraz z wyjaśnieniami można znaleźć w pracy [17].

Drugim testowanym modelem był model autorstwa Aleksandra Smywińskiego-Pohla, składający się z form tekstowych. Model ten, w przeciwieństwie do poprzedniego, zbudowano na podstawie bardzo dużej ilości tekstów - korpusu tekstów zgromadzonych przez Zespół Przetwarzania Sygnałów z Katedry

Elektroniki AGH[18].

Do stworzenia pierwszego modelu i wyznaczania prawdopodobieństw na podstawie obu modeli zostało wykorzystane narzędzie *The SRI Language Modeling Toolkit*[19]. Natomiast do tagowania słów wykorzystany został tager udostępniony przez Krzysztofa Wróbla[20], wykorzystujący Morphological Analysis Converter and Aggregator (MACA)[21].

3.3. Selekcja osobników

Dla potrzeb niniejszej pracy przetestowane zostały dwie następujące metody selekcji:

Strategia elitarna - polegająca na wybraniu do nowej populacji tylko osobników najlepiej przystosowanych (z największą wartością funkcji przystosowania).

Przykładowo, jeśli w populacji mamy dwa osobniki o wartościach funkcji przystosowania odpowiednio: 1 oraz 2, to do nowej populacji trafi osobnik bardziej przystosowany tj. drugi.

Metoda koła ruletki – polega na losowym wyborze osobników do nowej populacji z prawdopodobieństwem proporcjonalnym do wartości funkcji przystosowania. Ta metoda odpowiada wydzieleniu wycinka koła ruletki dla każdego osobnika, a wielkość wycinka koła jest proporcjonalna do wartości funkcji przystosowania osobnika.

Przykładowo, jeśli w populacji mamy dwa osobniki o wartościach funkcji przystosowania odpowiednio: 1 oraz 2, to prawdopodobieństwo, że do nowej populacji trafi pierwszy wynosi $\frac{1}{3}$, natomiast dla drugiego jest to - $\frac{2}{3}$.

3.4. Krzyżowanie osobników

Podczas krzyżowania dwójki osobników, zwanych rodzicami, powstaje dwójka nowych osobników - zwanych potomstwem - będących połączeniem cech obu rodziców. W niniejszej pracy postanowiono wykorzystać krzyżowanie jednopunktowe, polegające na wyborze jednego punktu krzyżowania dla obu rodziców. Jeden z nowo powstałych osobników posiada początkowe cechy (geny) pierwszego rodzica do jego punktu krzyżowania, a następnie geny drugiego, począwszy od jego punktu krzyżowania. Natomiast drugi zawiera początkowe cechy drugiego i końcowe pierwszego.

Przykład krzyżowania dwóch osobników zaprezentowano poniżej. Za gen w tym przypadku przyjęto pojedyncze słowo. Tak, jak poprzednio, przykładowe wiersze powstały na podstawie Inwokacji *Pana Tadeusza* Adama Mickiewicza:

Pierwszy rodzic

[„Litwo!”, „Ojczyzno”, „moja!” || „ty”, „jesteś”, „jak”, „zdrowie:\n”,
„Ile”, „cię”, „trzeba”, „cenić,”, „ten”, „tylko”, „się”, „dowie,\n”,

„Kto”, „cię”, „stracił.”, „Dziś”, „piękność”, „twą”, „w”, „całej”, „ozdobie\n”,
„Widzę”, „i”, „opisuję,”, „bo”, „tęsknię”, „po”, „tobie.\n“]

Drugi rodzic

[„Panno”, „święta,” || „co”, „Jasnej”, „bronisz”, „Częstochowy\n”,
„I”, „w”, „Ostrej”, „świecisz”, „Bramie!”, „Ty,”, „co”, „gród”, „zamkowy\n”,

„Nowogródzki”, „ochraniasz”, „z”, „jego”, „wiernym”, „ludem!\n”,
„Jak”, „mnie”, „dziecko”, „do”, „zdrowia”, „powróciłaś”, „cudem\n”]

Gdzie || oznacza wylosowany punkt krzyżowania. Powstaną dwa nowe osobniki, zaprezentowane poniżej:

Pierwszy potomek

[„Litwo!”, „Ojczyzno”, „moja!”, „co”, „Jasnej”, „bronisz”, „Częstochowy\n”,
„I”, „w”, „Ostrej”, „świecisz”, „Bramie!”, „Ty,”, „co”, „gród”, „zamkowy\n”,

„Nowogródzki”, „ochraniasz”, „z”, „jego”, „wiernym”, „ludem!\n”,
„Jak”, „mnie”, „dziecko”, „do”, „zdrowia”, „powróciłaś”, „cudem\n”]

Drugi potomek

[„Panno”, „święta,”, „ty”, „jesteś”, „jak”, „zdrowie:\n”,
„Ile”, „cię”, „trzeba”, „cenić,”, „ten”, „tylko”, „się”, „dowie,\n”,

„Kto”, „cię”, „stracił.”, „Dziś”, „piękność”, „twą”, „w”, „całej”, „ozdobie\n”,
„Widzę”, „i”, „opisuję,”, „bo”, „tęsknię”, „po”, „tobie.\n“]

3.5. Mutowanie osobników

Mutacja, analogicznie jak w genetyce, oznacza zmianę pewnych genów osobnika. W przypadku wierszy może przybrać ona jedną z dwóch postaci. Dla genu reprezentowanego przez słowo będzie to odpowiednio zmiana pewnych słów na inne, niekoniecznie w takiej samej liczbie. W przypadku gdy gen reprezentowany jest przez cały wers, zostanie on zastąpiony nowym wersem.

Przykład mutacji dla reprezentacji genu, jako słowa, przedstawiono poniżej. Słowa podlegające mutacji pogrubiono.

Osobnik

*[„**Litwo!**”, „Ojczyzno”, „moja!”, „ty”, „jesteś”, „jak”, „zdrowie:\n”,*
*„Ile”, „cię”, „trzeba”, „cenić,”, „ten”, „tylko”, „się”, „**dowie**,\n”,*

*„Kto”, „cię”, „stracił.”, „**Dziś**”, „**piękność**”, „twą”, „w”, „całej”, „ozdobie\n”,*
„Widzę”, „i”, „opisuję,”, „bo”, „tęsknię”, „po”, „tobie.\n“]

Zmutowany osobnik

*[„**Polsko!**”, „Ojczyzno”, „moja!”, „ty”, „jesteś”, „jak”, „zdrowie:\n”,*
*„Ile”, „cię”, „trzeba”, „cenić,”, „ten”, „tylko”, „się”, „**zowie**,\n”,*

*„Kto”, „cię”, „stracił.”, „**Cudowność**”, „twą”, „w”, „całej”, „ozdobie\n”,*
„Widzę”, „i”, „opisuję,”, „bo”, „tęsknię”, „po”, „tobie.\n“]

3.6. Modyfikacje algorytmu

Na potrzeby niniejszej pracy postanowiono dodatkowo przetestować modyfikację opisanego wcześniej postępowania. Modyfikacja ta zakłada nałożenie dwóch dodatkowych wymagań na generowane osobniki. Wymagania te są następujące:

1. Każdy wiersz musi posiadać trzynaście sylab w wersie.
2. Każdy wiersz musi posiadać rymy.

Warunki te muszą zostać spełnione podczas każdej z wykonywanych operacji, a więc dotyczy to także operacji mutacji oraz krzyżowania.

4. Wyniki

Rozdział ten prezentuje uzyskane wyniki. Badania przeprowadzono w kilku etapach. Na początku przeprowadzono testy przydatności obu modeli językowych przy wyznaczaniu składowej gramatycznej funkcji przystosowania.

Dalej przetestowano osiem konfiguracji podstawowych. Dla każdej z konfiguracji utworzono osobną sekcję, która przedstawia uzyskane przy zastosowanych parametrach wyniki. Uzyskane wyniki zadecydowały o wybraniu dwóch, najlepszych spośród nich, na których wykonano dalsze obliczenia. Je także, jako ostatni etap badań, poddano modyfikacji by sprawdzić, czy pozwoli ona jeszcze poprawić uzyskane wyniki.

4.1. Funkcja przystosowania

Utwór Adama Mickiewicza pt. *Pan Tadeusz* przyjęto jako wzorcowy dla wybranej w niniejszej pracy formy wiersza. Dlatego też wykorzystano go w celu oceny funkcji przystosowania, porównując wyniki uzyskane przy pomocy dwóch testowanych modeli językowych. Pierwszy z nich wykorzystuje tagi morfosyntaktyczne, drugi natomiast formy tekstowe.

Na podstawie trzydziestu sześciu pierwszych wersów stworzonych zostało dziewięć osobników (wierszy), a następnie poddane zostały one ocenie za pomocą funkcji oceny. Poniżej zaprezentowano otrzymane wyniki:

$lp.$	F_S	F_R	F_G	F
1	1.000	1.000	1.002	0.998
2	1.000	1.000	1.005	0.995
3	1.000	1.000	1.002	0.998
4	1.000	1.500	1.002	0.666
5	1.000	1.500	1.002	0.665
6	1.000	1.000	1.002	0.998
7	1.000	1.000	1.003	0.997
8	1.000	1.000	1.005	0.995
9	1.000	1.000	1.003	0.997
śr.	**1.000**	**1.111**	**1.003**	**0.923**

Tabela 4.1. Wyniki poszczególnych składowych funkcji przystosowania przy wykorzystaniu modelu z formami tekstowymi.

$lp.$	F_S	F_R	F_G	F
1	1.000	1.000	1.192	0.839
2	1.000	1.000	1.708	0.585
3	1.000	1.000	1.175	0.851
4	1.000	1.500	1.256	0.531
5	1.000	1.500	4.344	0.153
6	1.000	1.000	1.119	0.894
7	1.000	1.000	1.218	0.821
8	1.000	1.000	1.145	0.874
9	1.000	1.000	1.339	0.747
śr.	**1.000**	**1.111**	**1.611**	**0.699**

Tabela 4.2. Wyniki poszczególnych składowych funkcji przystosowania przy wykorzystaniu modelu z tagami morfosyntaktycznymi.

$lp.$	F_S	F_R	F_G	F
1	1.000	1.000	1.194	0.670
2	1.000	1.000	1.717	0.582
3	1.000	1.000	1.178	0.566
4	1.000	1.500	1.258	0.530
5	1.000	1.500	4.351	0.092
6	1.000	1.000	1.120	0.714
7	1.000	1.000	1.222	0.546
8	1.000	1.000	1.150	0.579
9	1.000	1.000	1.343	0.331
śr.	**1.000**	**1.111**	**1.615**	**0.512**

Tabela 4.3. Wyniki poszczególnych składowych funkcji przystosowania przy wykorzystaniu obu modeli równocześnie.

Otrzymane wyniki potwierdzono podczas testów przeprowadzonych na innych utworach literackich. W badaniach postanowiono wykorzystać wyłącznie model językowy z formami tekstowymi, ze względu na jego największą skuteczność.

Uzyskanie niskich wyników przy wykorzystaniu modelu językowego stworzonego na tagach morfosyntaktycznych, wynika najprawdopodobniej z małej ilości danych, wykorzystanych do jego uczenia. Niestety, tylko ręcznie znakowany milionowy podkorpus NKJP posiada informację o tagach morfosyntaktycznych słów występujących w tekstach języka polskiego. Wykorzystanie automatycznego tagera mogłoby z kolei spowodować nakładanie się kolejnych błędów.

4.2. Początkowy wybór parametrów

W celu określenia parametrów pozwalających na uzyskanie najwyższych wyników, postanowiono przeprowadzić próbne obliczenia dla różnych konfiguracji. Testowany był wpływ wyboru typu reprezentacji osobnika (patrz pkt 3.1), metody selekcji (patrz pkt 3.3) oraz prawdopodobieństwa mutacji (patrz pkt 3.5).

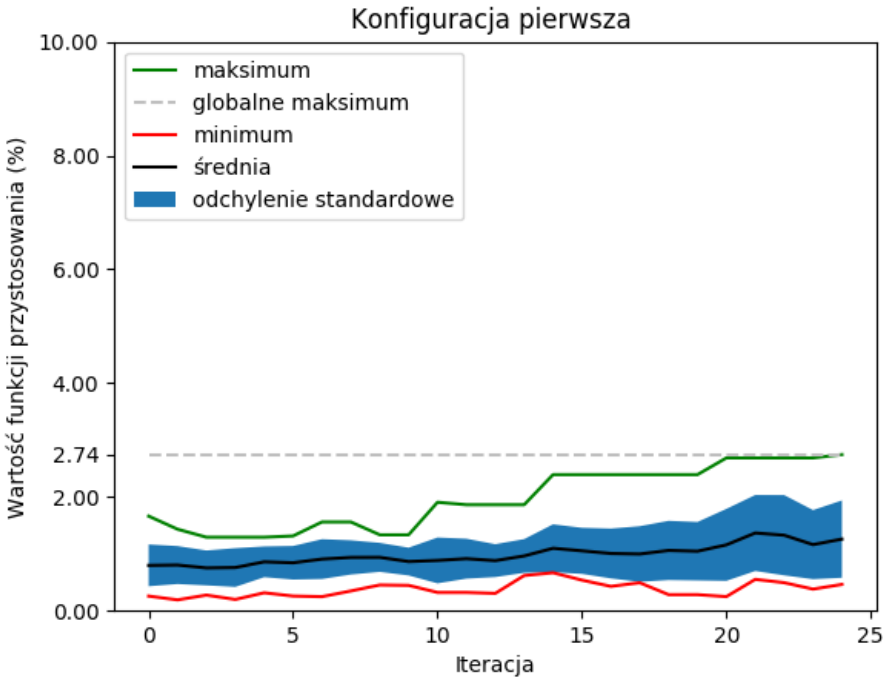
Ze względu na fakt, że operowanie obszernym modelem językowym wymaga bardzo dużej mocy obliczeniowej, w przypadku mutacji postanowiono ograniczyć się do dwóch wartości prawdopodobieństwa. Określono je w następujący sposób: małe prawdopodobieństwo - 10% oraz duże prawdopodobieństwo - 50%. Dodatkowo każda konfiguracja testowana była jedynie przez 25 iteracji, z początkową populacją o rozmiarze 10 osobników.

4.2.1. Konfiguracja pierwsza

Pierwsza próba miała za zadanie przetestować następujący układ parametrów:

1. Gen odpowiadał pojedynczemu słowu.
2. Selekcja osobników odbywała się metodą koła ruletki.
3. Prawdopodobieństwo mutacji wynosiło 10%.

Zmiany wartości funkcji przystosowania osobników na przestrzeni kolejnych iteracji i przy zastosowaniu wymienionych parametrów przedstawiono na wykresie:



Rysunek 4.1. Testowe wyniki dla konfiguracji pierwszej.

Dla pierwszej konfiguracji najlepszym wygenerowanym wierszem, który otrzymał ocenę 2.74%, był utwór zaprezentowany poniżej:

przyczyna polecić lekkoatletyczny drop
niewystarczalność biurokratyczny modlitwa

porfir agawa umożliwić kolanowy
prawy czujność wiatrowy faszerować

Stworzony dla uzyskanych wyników wykres zmian wartości funkcji przystosowania wydaje się obiecujący. Jednak dokładna analiza wygenerowanej populacji wykazała, że najlepsze osobniki po ostatniej iteracji są niemal identyczne. Różni je zaledwie kilka wyrazów. Przyczyną takiego stanu rzeczy jest szybkie wyłonienie się lidera, posiadającego najwyższą wartość przystosowania. Pozostałe osobniki

danej populacji w sposób naturalny do niego dążą.

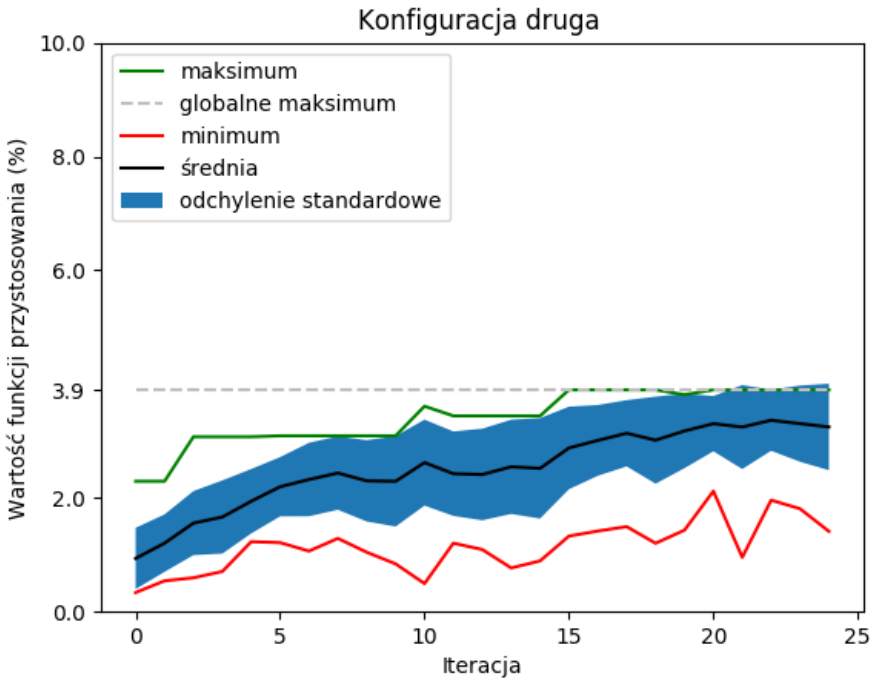
Zbyt małe prawdopodobieństwo mutacji nie pozwoliło z kolei na uzyskanie istotnych zmian w osobnikach, które umożliwiłyby powstanie nowych wierszy. Pomimo użycia selekcji metodą koła ruletki nie było możliwości pozbycia się lidera, ponieważ posiadał on już dużo swoich prawie identycznych kopii. Skutkuje to utykaniem algorytmu w lokalnym maksimum. Istnieje oczywiście prawdopodobieństwo, że po pewnej ilości iteracji dojdzie do znaczącej modyfikacji najlepszych generowanych utworów - widać to przy iteracji 19. - pozwalających algorytmowi na wydostanie się z maksimum, jednak algorytm szybko utknie w kolejnym maksimum. Taka konfiguracja zatem nie nadaje się do szybkiego i efektywnego przeszukiwania przestrzeni rozwiązań.

4.2.2. Konfiguracja druga

Druga konfiguracja posiadała następujące parametry:

1. Gen odpowiadał pojedynczemu słowu.
2. Selekcja osobników odbywała się metodą elitarną.
3. Prawdopodobieństwo mutacji wynosiło 10%.

Dla drugiej konfiguracji zmiany wartości funkcji przystosowania osobników na przestrzeni kolejnych iteracji, prezentowały się następująco:



Rysunek 4.2. Testowe wyniki dla konfiguracji drugiej.

Najlepszym wygenerowanym wierszem, uzyskującym ocenę 3.90%, był utwór zaprezentowany poniżej:

głośnik szarzyzna sowity instynktowny
prywatyzować wykład blich nadal główka

flażeolet igła zaś zaś halucynogen
wyczerpująco wyczerpująco ponoć

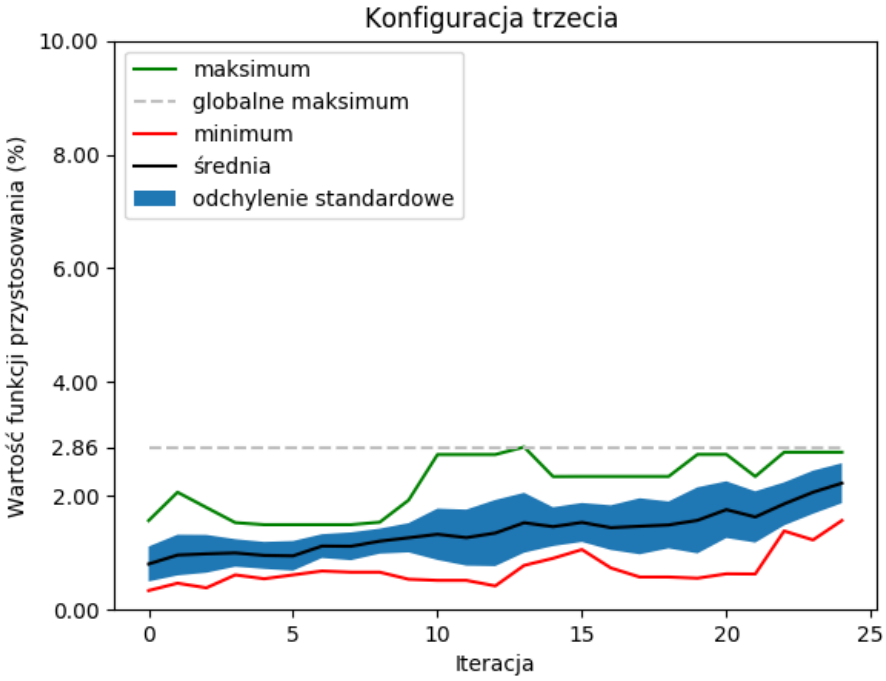
W przypadku konfiguracji drugiej widać, że średnia wartość funkcji przystosowania wyraźnie zmierza w kierunku maksimum. Dodatkowo, dokładna analiza wygenerowanej populacji wykazała, że tak samo jak w przypadku konfiguracji pierwszej, algorytm utyka w lokalnym maksimum. Poza wymienionymi już wcześniej powodami takiej sytuacji, warto wspomnieć o wyborze dla tej konfiguracji selekcji metodą elitarną, która spotęgowała ten efekt.

4.2.3. Konfiguracja trzecia

Trzecia konfiguracja posiadała następujące parametry:

1. Gen odpowiadał całemu wersowi.
2. Selekcja osobników odbywała się metodą koła ruletki.
3. Prawdopodobieństwo mutacji wynosiło 10%.

Dla trzeciej konfiguracji zmiany wartości funkcji przystosowania osobników na przestrzeni kolejnych iteracji, prezentowały się następująco:



Rysunek 4.3. Testowe wyniki dla konfiguracji trzeciej.

Najlepszym wygenerowanym wierszem, uzyskującym ocenę 2.86%, był utwór zaprezentowany poniżej:

doholować kraksa poślubiać brutalność
wydzwaniać przegląd plebejusz efemeryczność

niezawodny lustro puszka tęgi konsumpcja
żleb wyczuć egzegeza przypadek sierocy

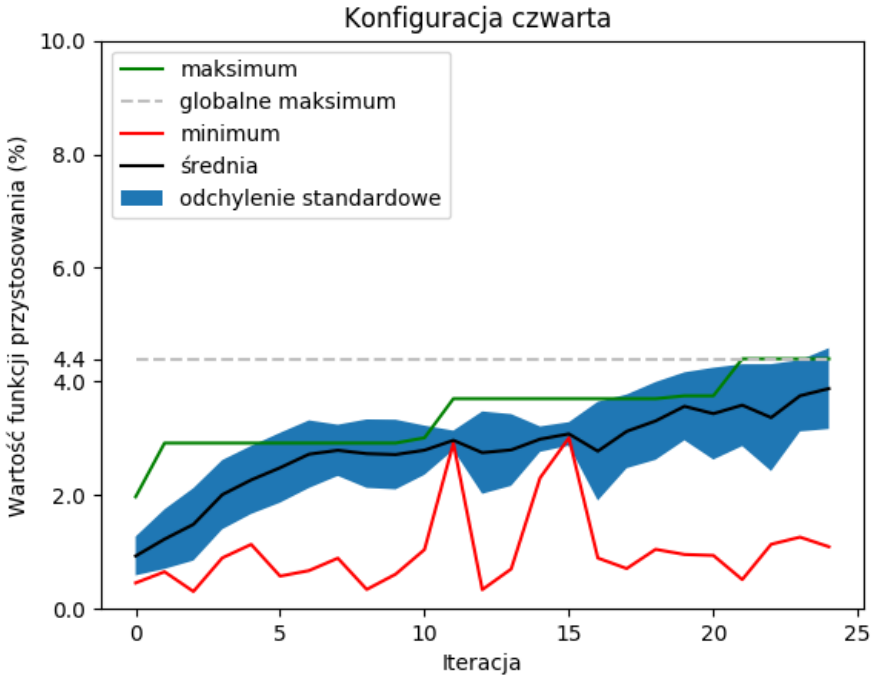
W przypadku konfiguracji trzeciej napotkano problem, który pojawił się już przy pierwszym obliczeniu. Obie te konfiguracje różni jedynie stopień reprezentacji genu. Pomimo, że algorytm w konfiguracji trzeciej operował na całych wersach, a nie na słowach (jak w konfiguracji pierwszej), zmiany nadal nie były dostatecznie znaczące by uniknąć utykania w lokalnym maksimum.

4.2.4. Konfiguracja czwarta

Czwarta konfiguracja posiadała następujące parametry:

1. Gen odpowiadał całemu wersowi.
2. Selekcja osobników odbywała się metodą elitarną.
3. Prawdopodobieństwo mutacji wynosiło 10%.

Dla czwartej konfiguracji zmiany wartości funkcji przystosowania osobników na przestrzeni kolejnych iteracji, prezentowały się następująco:



Rysunek 4.4. Testowe wyniki dla konfiguracji czwartej.

Najlepszym wygenerowanym wierszem, uzyskującym ocenę 4.40%, był utwór zaprezentowany poniżej:

ekranowy odkopać prać zaobserwować
morderczy maklerski nieważki foliowy ja

hen naprędce prezenter błękitnooki skiba
uporać przekaźnikowy jezuita

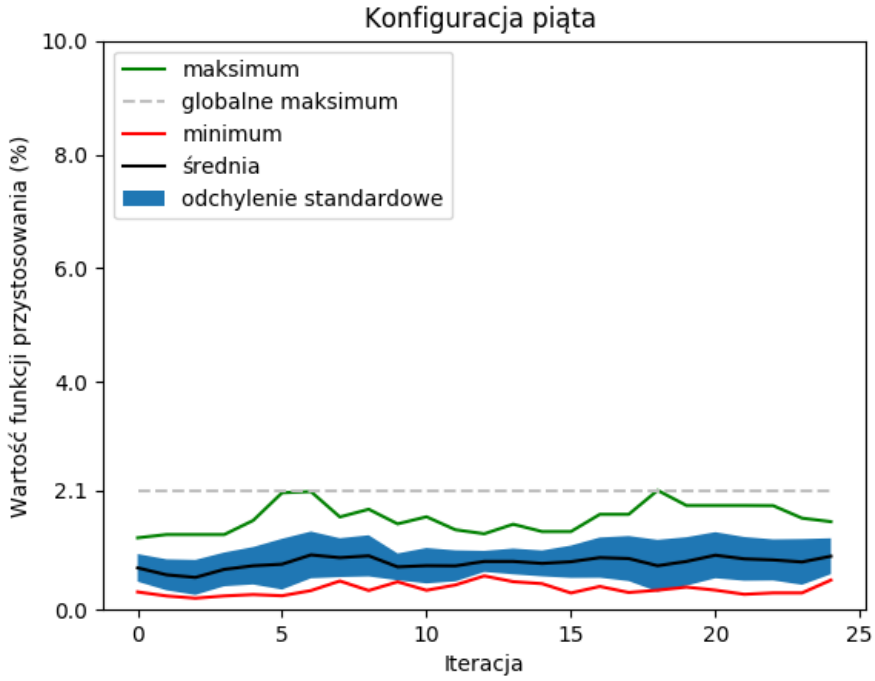
W przypadku konfiguracji czwartej wyjątkowo dobrze widać, że średnia wartość funkcji przystosowania zmierza w kierunku maksimum. Dodatkowo przy iteracji 11. i 15. algorytm znacząco utknął w lokalnym maksimum. Powody uzyskanych wyników są takie same, jak dla konfiguracji trzeciej. Dodatkowo, dla konfiguracji czwartej również zastosowano elitarną metodę selekcji, która jest akceleratorem tego procesu.

4.2.5. Konfiguracja piąta

Piąta konfiguracja posiadała następujące parametry:

1. Gen odpowiadał pojedynczemu słowu.
2. Selekcja osobników odbywała się metodą koła ruletki.
3. Prawdopodobieństwo mutacji wynosiło 50%.

Dla piątej konfiguracji zmiany wartości funkcji przystosowania osobników na przestrzeni kolejnych iteracji, prezentowały się następująco:



Rysunek 4.5. Testowe wyniki dla konfiguracji piątej.

Najlepszym wygenerowanym wierszem, uzyskującym ocenę 2.10%, był utwór zaprezentowany poniżej:

unieruchomienie czarownica kluczyk
namiernik szlifować spadzisty

teść melancholia rekwizytornia zatem
szewc kostiumowy czternastolatek

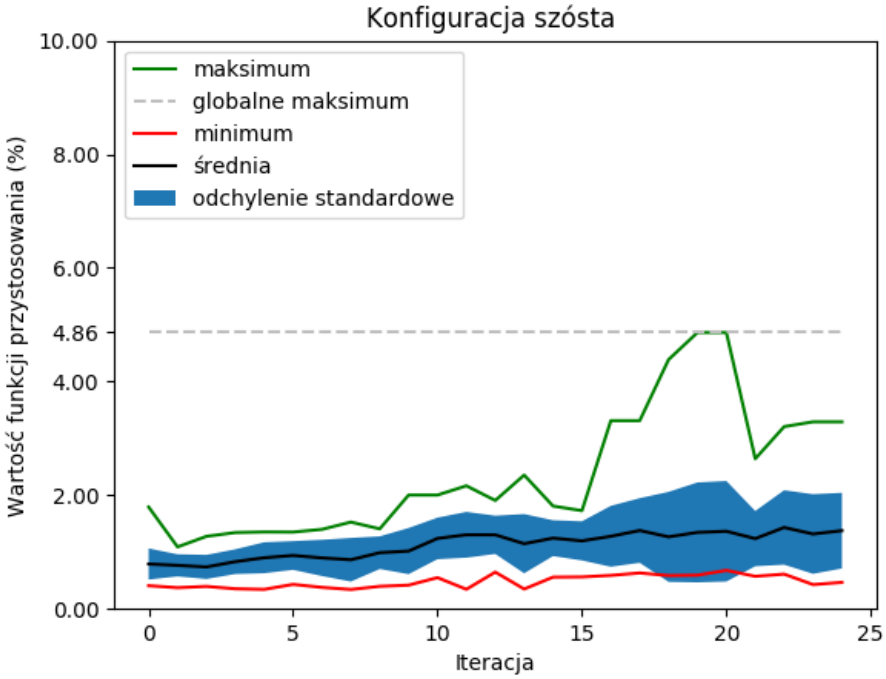
Konfigurację piątą cechowało odpowiednie zróżnicowanie osobników. Nie wyłonił się wyraźny lider. Nie zaobserwowano także upodabniania się do siebie wygenerowanych tekstów literackich. Dodatkowo, 50% szansa mutacji wniosła pewną pożądaną losowość w nowych populacjach. Konfigurację piątą wybrano, jako jedną z dwóch najbardziej obiecujących, do dalszych badań.

4.2.6. Konfiguracja szósta

Szósta konfiguracja posiadała następujące parametry:

1. Gen odpowiadał pojedynczemu słowu.
2. Selekcja osobników odbywała się metodą elitarną.
3. Prawdopodobieństwo mutacji wynosiło 50%.

Dla szóstej konfiguracji zmiany wartości funkcji przystosowania osobników na przestrzeni kolejnych iteracji, prezentowały się następująco:



Rysunek 4.6. Testowe wyniki dla konfiguracji szóstej.

Najlepszym wygenerowanym wierszem, uzyskującym ocenę 4.87%, był utwór zaprezentowany poniżej:

przekonać architekt bębenkowy zagubić
rektor nadtlenek zafascynowanie łasić

rumsztyk przystaniowy objaśnić objaśnić gnać
naplotkować podsumowanie historiograf

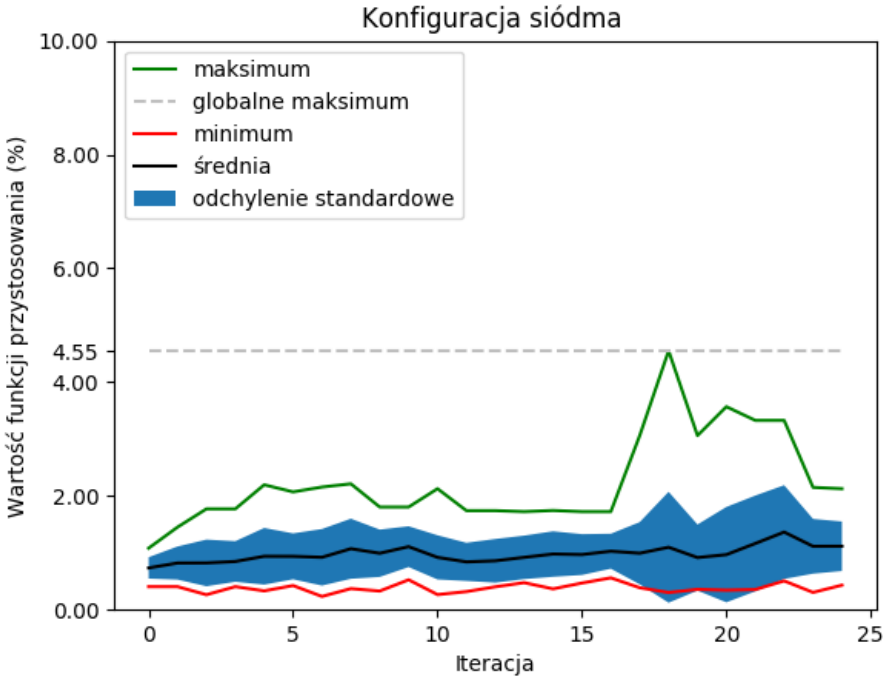
Przyjęty dla tej konfiguracji parametr, elitarna strategia doboru osobników, po raz kolejny przyczynił się do uniemożliwienia różnicowania się osobników wewnątrz populacji. Oznaczało to dalsze utykanie algorytmu w lokalnym maksimum. Zastosowanie większego prawdopodobieństwa mutacji (50%) zmniejszyło jednak skalę tego problemu w porównaniu do konfiguracji z małym (10%) prawdopodobieństwem mutacji.

4.2.7. Konfiguracja siódma

Siódma konfiguracja posiadała następujące parametry:

1. Gen odpowiadał całemu wersowi.
2. Selekcja osobników odbywała się metodą koła ruletki.
3. Prawdopodobieństwo mutacji wynosiło 50%.

Dla siódmej konfiguracji zmiany wartości funkcji przystosowania osobników na przestrzeni kolejnych iteracji, prezentowały się następująco:



Rysunek 4.7. Testowe wyniki dla konfiguracji siódmej.

Najlepszym wygenerowanym wierszem, uzyskującym ocenę 4.55%, był utwór zaprezentowany poniżej:

zużyć zalotnik rokrocznie zakamuflować
kuni bulwar omam zdyskwalifikować

prowincja zdegradowanie przykucnięcie profanować
alfabet tępak wideoklip zaprenumerować

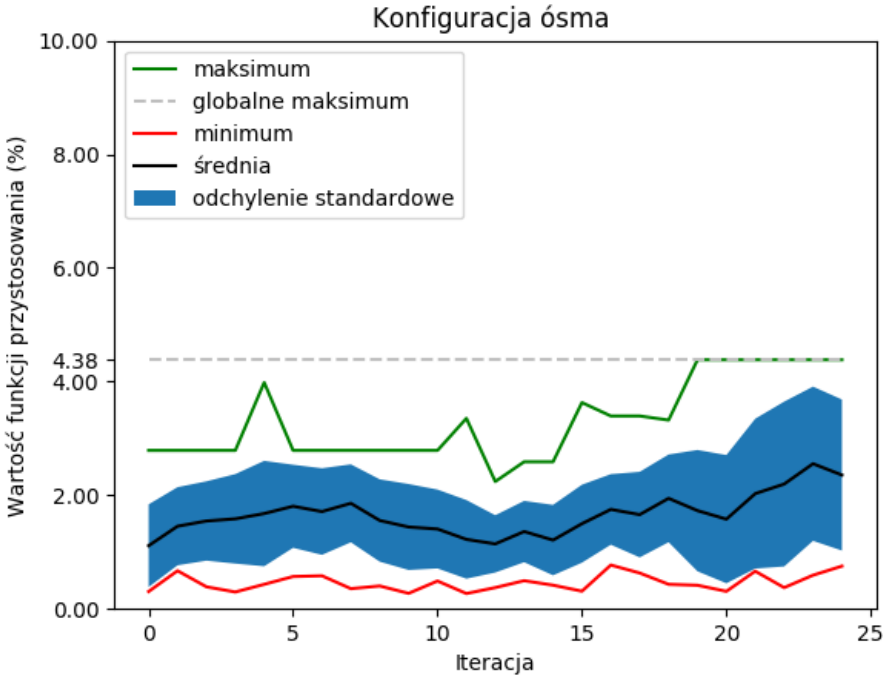
Podobnie jak konfiguracja piąta, konfiguracja siódma osiągnęła bardzo obiecujące wyniki. Jedyną znaczącą różnicą pomiędzy nimi był sposób reprezentacji genów. Dzięki temu uzyskano większą różnorodność osobników w wygenerowanej populacji (co widać na wykresie zmian odchylenia standardowego w porównaniu do Rys. 4.5). W tym przypadku selekcja metodą koła ruletki skutecznie wyeliminowała zbyt odstających liderów (Patrz. Rys. 4.7, okolice 17 iteracji). Tę konfigurację wybrano także do kolejnych badań.

4.2.8. Konfiguracja ósma

Ósma konfiguracja posiadała następujące parametry:

1. Gen odpowiadał całemu wersowi.
2. Selekcja osobników odbywała się metodą elitarną.
3. Prawdopodobieństwo mutacji wynosiło 50%.

Dla ósmej konfiguracji zmiany wartości funkcji przystosowania osobników na przestrzeni kolejnych iteracji, prezentowały się następująco:



Rysunek 4.8. Testowe wyniki dla konfiguracji ósmej.

Najlepszym wygenerowanym wierszem, uzyskującym ocenę 4.38%, był utwór zaprezentowany poniżej:

teza pochwalać boleśnie charakterystyka
wędrowiec solennie komunistyczny jego

żartować autoportret fosa trudny
ciemność butla ceremoniować nieodzownie

Po analizie wyników uzyskanych dla tak określonych parametrów okazało się, że algorytm nadal wykazuje skłonność do utykania w lokalnym maksimum (tak samo, jak w przypadku konfiguracji szóstej). 50% prawdopodobieństwo mutacji jest jednak czynnikiem osłabiającym tą zależność, ponieważ operowanie na całych wersach zwiększa zróżnicowanie powstałych w ten sposób tekstów.

4.3. Kontynuacja najbardziej obiecujących konfiguracji

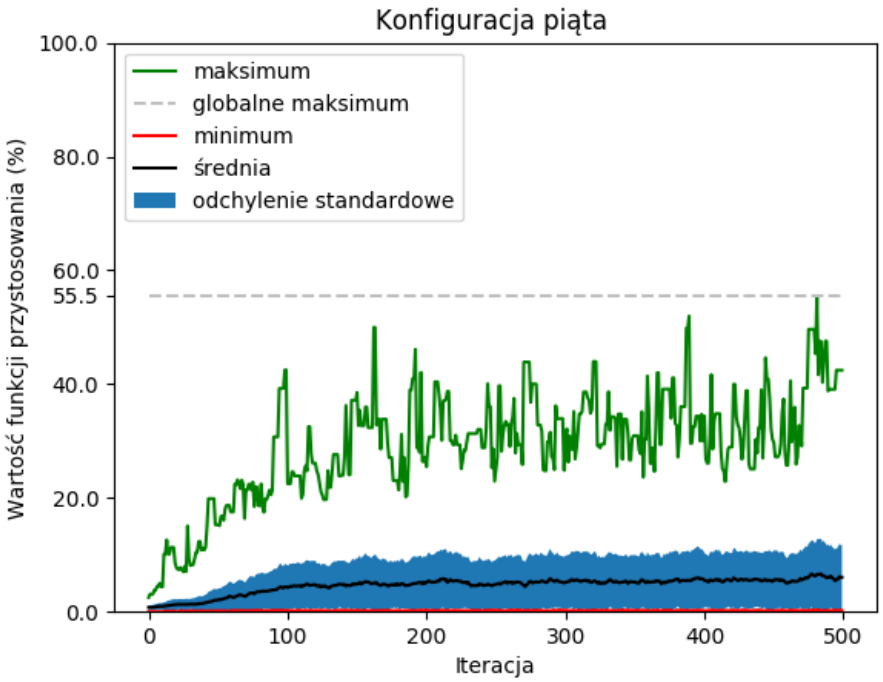
Po zapoznaniu się z wynikami przeprowadzonych testów postanowiono kontynuować obliczenia jedynie na dwóch z ośmiu, zaprezentowanych w niniejszej pracy konfiguracji. U pozostałych widoczna była tendencja do tworzenia się populacji, składających się z niemal identycznych osobników.

4.3.1. Konfiguracja piąta

Poza podstawowymi parametrami tej konfiguracji, tj:

1. Gen odpowiadał pojedynczemu słowu.
2. Selekcja osobników odbywała się metodą koła ruletki.
3. Prawdopodobieństwo mutacji wynosiło 50%.

w próbie tej zwiększono liczbę osobników w początkowej populacji do 500 oraz wykonano 500 iteracji. Po takiej modyfikacji zmiany wartości funkcji przystosowania osobników na przestrzeni kolejnych iteracji, prezentowały się następująco:



Rysunek 4.9. Wyniki dla konfiguracji piątej.

Poniżej zaprezentowano wiersze wraz z ich oceną, powstałe z dziesięciu najbardziej przystosowanych osobników. Uporządkowano je wraz z malejącą wartością funkcji przystosowania. Wiersze identyczne pominięto.

Ocena 55.53%

wielokropek przywiązać wymusić wskazanie
stale nieskończoność skrzętny oddziaływanie

ogień idiota ogień nieoczekiwanie
znajomy czarodziej każdy zastosowanie

Ocena 47.55%

ujęcie bardzo zmowa wymusić wskazanie
bezpośrednio nieskończoność oddziaływanie

ciąg idiota ogień nieoczekiwanie
podbródek zdemaskowanie zastosowanie

Ocena 51.96%

drenarski hamburger śmierć stosunek wskazanie
podróż porządek nożownik oddziaływanie

powstanie szyna razy ogień zaniechanie
koń każdy wór wór beztroska zastosowanie

Ocena 47.51%

dlaczego wykonawca wymusić wskazanie
stale stokroć cenzurka oddziaływanie

ogień idiota ogień nieoczekiwanie
znajomy czarodziej każdy zastosowanie

Ocena 49.96%

pakować wykonawca wymusić wskazanie
szerokość smak smak szerokość zapisywanie

pierworodny specyficznie zażenowanie
och znów otulić abstrakcja zastosowanie

Ocena 46.10%

kubek przekonstruować wymusić wskazanie
gwarantować nekromanta oddziaływanie

do umieszczać klerykalizm zażenowanie
zamyślenie uzasadniać zastosowanie

Ocena 49.83%

drenarski hamburger śmierć stosunek wskazanie
podróż porządek nożownik oddziaływanie

powstanie szyna razy ogień zaniechanie
koń każdy wór ułaskawić zastosowanie

Ocena 45.36%

wielokropek przywiązać wymusić wskazanie
awersja bezpośrednio oddziaływanie

ogień idiota ogień nieoczekiwanie
znajomy czarodziej każdy zastosowanie

Ocena 49.63%

totalnie smutno skuter wymusić wskazanie
stale awersja bezpośrednio oddziaływanie

ogień idiota ogień nieoczekiwanie
znajomy czarodziej każdy zastosowanie

Ocena 44.64%

totalnie wykonawca wymusić wskazanie
stale nieskończoność przejąć oddziaływanie

projekt zagospodarowywać zaniechanie
metalurgiczny niedługo zastosowanie

Biorąc pod uwagę stosunkowo małą liczbę iteracji oraz niedużą ilość osobników w populacji - wartości te wymuszone były wysoką złożonością obliczeniową algorytmu - uzyskane przy pomocy tej konfiguracji wyniki są wysoce zadowalające. Wiersze są różnorodne, chociaż posiadają pewne części wspólne.

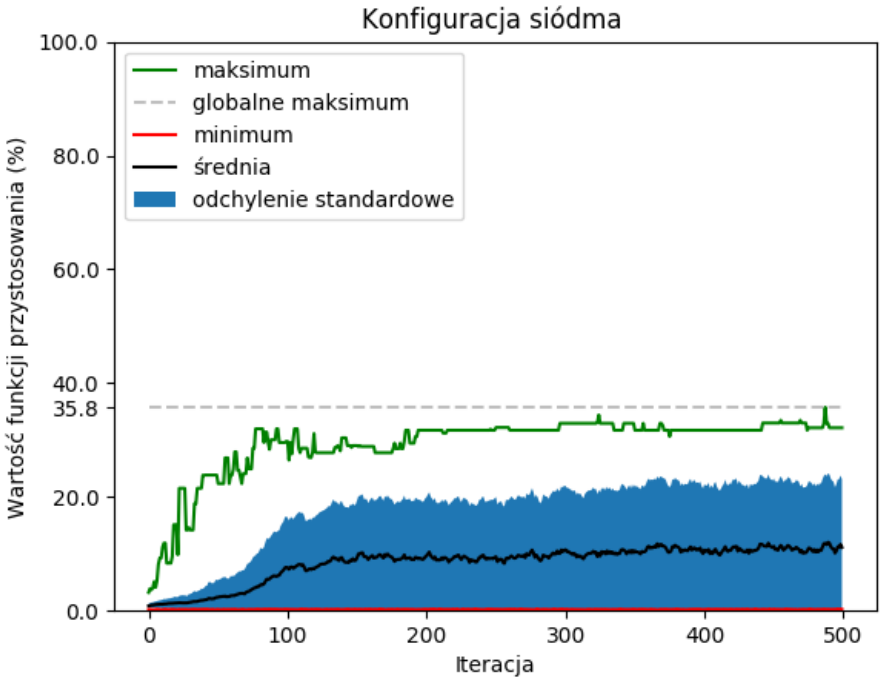
Ciekawym spostrzeżeniem jest fakt, że ostatnie słowa poszczególnych wersów dla większości z wygenerowanych utworów są identyczne. Oznacza to posiadanie przez nie pewnego wspólnego przodka. Ewolucja przebiegała tutaj jednak prawidłowo, dzięki czemu osobniki te są odpowiednio zróżnicowane. Na wykresie można też zauważyć, że selekcja metodą ruletki spełniała swoje zadanie i skutecznie eliminowała liderów, przez co wykres maksymalnych wartości funkcji przystosowania jest bardzo nieregularny.

4.3.2. Konfiguracja siódma

Poza podstawowymi parametrami tej konfiguracji, tj:

1. Gen odpowiadał całemu wersowi.
2. Selekcja osobników odbywała się metodą koła ruletki.
3. Prawdopodobieństwo mutacji wynosiło 50%.

w próbie tej zwiększono także ilość osobników w początkowej populacji do 500 oraz wykonano 500 iteracji. Po takiej modyfikacji zmiany wartości funkcji przystosowania osobników na przestrzeni kolejnych iteracji, prezentowały się następująco:



Rysunek 4.10. Wyniki dla konfiguracji siódmej.

Poniżej zaprezentowano wiersze wraz z ich oceną, powstałe z dziesięciu najbardziej przystosowanych osobników. Uporządkowano je wraz z malejącą wartością funkcji przystosowania. Wiersze identyczne pominięto.

Ocena 35.77%
ren czubek zawędrować ukryć panowanie
prawodawstwo mianowicie bóg umniejszanie

przez fortyfikacja zalegalizowanie
bluzka zwykły istny zatopienie chlapanie

Ocena 34.41%
ren czubek zawędrować ukryć panowanie
sabotaż nielitościwy nieubłaganie

przez fortyfikacja zalegalizowanie
niewątpliwie widnieć straganiarz pokonanie

Ocena 33.46%
głośność podstawowy zapewne rozdzielanie
miał kultywowanie gratka symulowanie

przez fortyfikacja zalegalizowanie
bluzka zwykły istny zatopienie chlapanie

Ocena 33.08%
ren czubek zawędrować ukryć panowanie
niepotrzebnie pierwodruk sklasyfikowanie

przez fortyfikacja zalegalizowanie
niewątpliwie widnieć straganiarz pokonanie

Ocena 32.99%
głośność podstawowy zapewne rozdzielanie
niepotrzebnie pierwodruk sklasyfikowanie

przez fortyfikacja zalegalizowanie
bluzka zwykły istny zatopienie chlapanie

Ocena 32.93%
ren czubek zawędrować ukryć panowanie
sabotaż nielitościwy nieubłaganie

przez fortyfikacja zalegalizowanie
bluzka zwykły istny zatopienie chlapanie

Ocena 32.90%
ren czubek zawędrować ukryć panowanie
prawodawstwo mianowicie bóg umniejszanie

przez fortyfikacja zalegalizowanie
dekarz skryba zirytować wypracowanie

Ocena 32.56%
włącznie partnerstwo rozpylić ar określanie
niepotrzebnie pierwodruk sklasyfikowanie

przez fortyfikacja zalegalizowanie
flamandzki otoczenie żal definiowanie

Ocena 32.20%
ren czubek zawędrować ukryć panowanie
niepotrzebnie pierwodruk sklasyfikowanie

przez fortyfikacja zalegalizowanie
wędrować pedagogiczny cienko rozdanie

Ocena 32.15%
ren czubek zawędrować ukryć panowanie
miał kultywowanie gratka symulowanie

przez fortyfikacja zalegalizowanie
bluzka zwykły istny zatopienie chlapanie

Konfiguracja siódma uzyskała znacznie gorsze wyniki, niż konfiguracja piąta. Pomimo większego odchylenia standardowego wartości funkcji przystosowania - oznaczającego większą różnorodność osobników - najlepsze z wygenerowanych wierszy posiadają więcej dłuższych części wspólnych. Może to być spowodowane faktem, że przy reprezentacji genu jako wersu, punkty krzyżowania występują wyłącznie na końcu wersów. Bardzo ogranicza to ich zakres i ilość, np. w przypadku niniejszej pracy

było ich zaledwie trzy. Dodatkowo z powodu ograniczenia możliwych punktów krzyżowania wzrasta prawdopodobieństwo powstania w populacji identycznych osobników.

Przykładowo, jeśli wiersz o wersach odpowiadających *ABCD* zostanie skrzyżowany ze swoim zmutowanym odpowiednikiem *AECD*, a punkt krzyżowania przypadnie po drugim lub trzecim wersie (co stanowi aż dwa, z trzech możliwych punktów krzyżowania), to jako potomstwo powstaną dokładnie takie same osobniki, tj. *ABCD* i *AECD*. W przypadku genu jako słowo, potencjalnych punktów krzyżowania jest znacznie więcej, a zatem prawdopodobieństwo zaistnienia przedstawionego tu problemu jest istotnie mniejsze.

Innym możliwym powodem uzyskiwania lepszych wyników dla genu, jako słowa, jest korzystniejszy przy tym modelu zakres zmian zachodzących w czasie mutacji. Zastępowanie całego wersu - nowym, jest zmianą inwazyjną i istotnie wpływającą na kształt wiersza. Możliwość mutacji pojedynczych słów daje większą elastyczność, a także pewną precyzję w doskonaleniu wygenerowanego utworu.

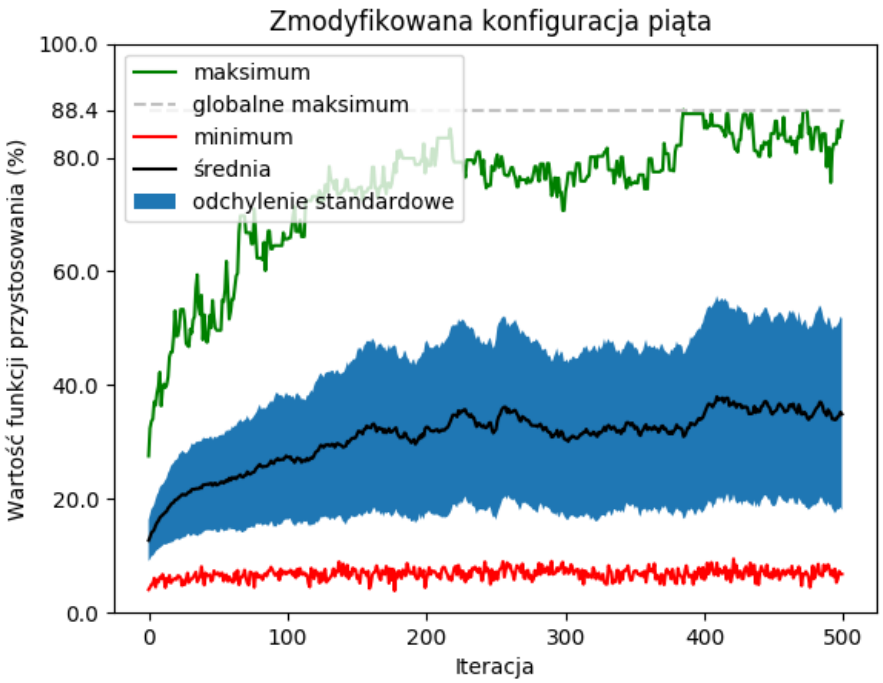
4.4. Wyniki uzyskane po modyfikacji

4.4.1. Zmodyfikowana konfiguracja piąta

Poza podstawowymi parametrami piątej konfiguracji, nałożono na osobniki wymóg spełnienia dwóch dodatkowych właściwości. Zmodyfikowana konfiguracja piąta wyglądała następująco:

1. Gen odpowiadał pojedynczemu słowu.
2. Selekcja osobników odbywała się metodą koła ruletki.
3. Prawdopodobieństwo mutacji wynosiło 50%.
4. Każdy wiersz musiał posiadać trzynaście sylab w wersie.
5. Każdy wiersz musiał posiadać rymy.

Dla tak zmodyfikowanej konfiguracji piątej wartości funkcji przystosowania osobników na przestrzeni kolejnych iteracji prezentowały się następująco:



Rysunek 4.11. Wyniki dla zmodyfikowanej konfiguracji piątej.

Poniżej zaprezentowano wiersze wraz z ich oceną, powstałe z dziesięciu najbardziej przystosowanych osobników. Uporządkowano je wraz z malejącą wartością funkcji przystosowania. Wiersze identyczne pominięto.

Ocena 88.37%

pycha zwierzyć że zapalnik jako troll ale
gazeta tylko omijać aby myśl ale

a żona marengo szufladkować ich długo
ogniskowa i ile wręcz na co niedługo

Ocena 88.10%

pycha zwierzyć że zawsze ku uziemić ale
gazeta tylko omijać aby wyjść ale

a żona zmienna oto rozmyślny śnieg długo
uwertura i moje wręcz na co niedługo

Ocena 87.69%

pycha zwierzyć że zapalnik jako troll ale
gazeta tylko psalmista aby myśl ale

a żona zmora oto frymarczyć ich długo
ogniskowa i ile wręcz na co niedługo

Ocena 87.65%

pycha zwierzyć że zapalnik jako troll ale
gazeta ciało imperium aby myśl ale

a żona drzewo oto frymarczyć ich długo
ogniskowa i dowódca na co niedługo

Ocena 87.57%

pycha zwierzyć że zapalnik jako troll ale
gazeta ciało imperium aby myśl ale

a przeredagować niepokojąco długo
ogniskowa i dowódca na co niedługo

Ocena 87.08%

niestety tucz że zapalnik jako troll ale
gazeta tylko folwarczny aby myśl ale

a żona drzewo lek niepokojąco długo
ogniskowa i dowódca na co niedługo

Ocena 86.83%

pycha zwierzyć że zapalnik jako troll ale
gazeta tylko psalmista aby myśl ale

a żona bochen oto frymarczyć ich długo
ogniskowa i ile wręcz na co niedługo

Ocena 86.68%

pycha zwierzyć że zapalnik jako troll ale
gazeta tylko psalmista aby myśl ale

a żona kiełkować przeźroczysty ich długo
ogniskowa i ile wręcz na co niedługo

Ocena 86.52%

zagrożenie tłumacz również pal zmagać ale
gazeta tylko omijać aby myśl ale

a żona marengo szufladkować ich długo
ogniskowa i ile wręcz na co niedługo

Ocena 86.45%

pycha zwierzyć że zapalnik jako troll ale
gazeta tylko wypróbowywać blask ale

a żona zmora oto frymarczyć ich długo
ogniskowa i ile wręcz na co niedługo

Zastosowana modyfikacja, tak jak przypuszczano, znacząco poprawiła uzyskane wyniki. Jest to spowodowane faktem nałożenia dodatkowych wymogów na osobniki, co spowodowało znaczne ograniczenie przeszukiwanej przestrzeni rozwiązań. Zauważyć można również, że tym razem wszystkie ostatnie słowa poszczególnych wersów wygenerowanych utworów są identyczne. Wskazuje to (podobnie jak wcześniej) na posiadanie przez nie pewnego wspólnego przodka, jednak w tym przypadku ewolucja

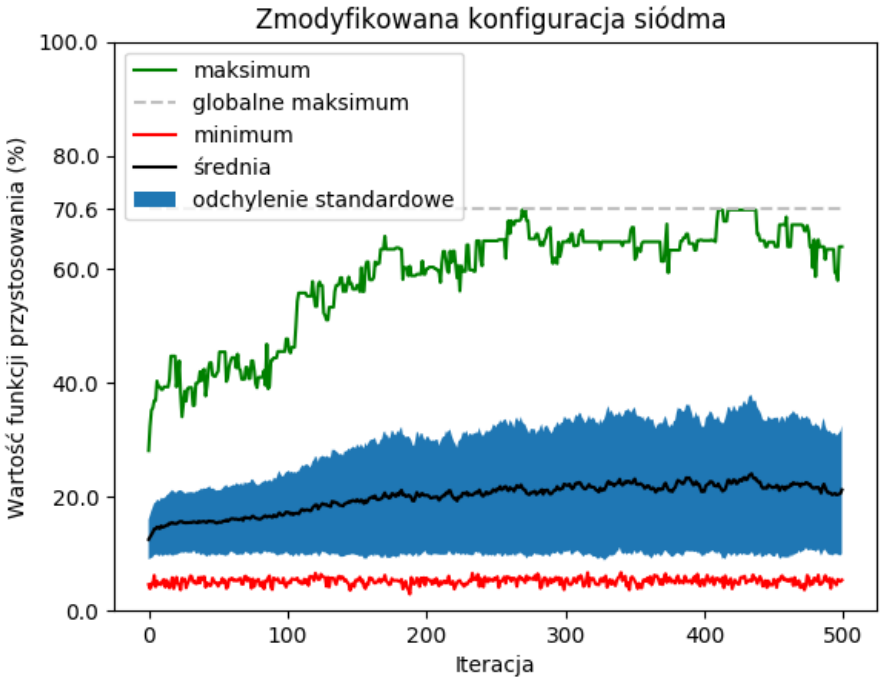
była zbyt powolna, aby powstały zróżnicowane osobniki. Wynika to najprawdopodobniej z faktu, że w przypadku zmodyfikowanej konfiguracji piątej przeszukiwana przestrzeń rozwiązań różniła się od tej w oryginalnej konfiguracji, a co za tym idzie, należy przeprowadzić ponownie proces dobierania parametrów.

4.4.2. Zmodyfikowana konfiguracja siódma

Poza podstawowymi parametrami siódmej konfiguracji, nałożono na osobniki wymóg spełnienia obu dodatkowych właściwości. Zmodyfikowana konfiguracja siódma, wyglądała następująco:

1. Gen odpowiadał całemu wersowi.
2. Selekcja osobników odbywała się metodą koła ruletki.
3. Prawdopodobieństwo mutacji wynosiło 50%.
4. Każdy wiersz musiał posiadać trzynaście sylab w wersie.
5. Każdy wiersz musiał posiadać rymy.

Dla tak zmodyfikowanej konfiguracji siódmej wartości funkcji przystosowania osobników na przestrzeni kolejnych iteracji prezentowały się następująco:



Rysunek 4.12. Wyniki dla zmodyfikowanej konfiguracji siódmej.

Poniżej zaprezentowano wiersze wraz z ich oceną, powstałe z dziesięciu najbardziej przystosowanych osobników. Uporządkowano je wraz z malejącą wartością funkcji przystosowania. Wiersze identyczne pominięto.

Ocena 70.60%

wrzaskliwy poprzestać towarzystwo gdyż wkrótce
pozwolić odrodzenie zbawienie pokrótce

porozumieć potrzebny wyniesienie tego
niemal oświecenie każdorazowy tego

Ocena 69.19%

szczęka królować komicznie polo gdzie wkrótce
pozwolić odrodzenie zbawienie pokrótce

neutralizowanie rzeczywiście tego
wynarodowienie drętwieć ciało którego

Ocena 70.49%

głośno ognik świąteczny prześliczny że wkrótce
pozwolić odrodzenie zbawienie pokrótce

porozumieć potrzebny wyniesienie tego
żarnowiec delikatny a oględny jego

Ocena 68.80%

głośno ognik świąteczny prześliczny że wkrótce
pozwolić odrodzenie zbawienie pokrótce

porozumieć potrzebny wyniesienie tego
niemal oświecenie każdorazowy tego

Ocena 70.44%

odmowa paskudny bezczelny łgarz raz wkrótce
pozwolić odrodzenie zbawienie pokrótce

neutralizowanie rzeczywiście tego
wynarodowienie drętwieć ciało którego

Ocena 68.55%

prasa uwolnić kuzyn towarzyszyć wkrótce
pozwolić odrodzenie zbawienie pokrótce

porozumieć potrzebny wyniesienie tego
żarnowiec delikatny a oględny jego

Ocena 69.78%

wrzaskliwy poprzestać towarzystwo gdyż wkrótce
pozwolić odrodzenie zbawienie pokrótce

transportowy labirynt zadąć dla którego
wynarodowienie drętwieć ciało którego

Ocena 68.45%

odmowa paskudny bezczelny łgarz raz wkrótce
pozwolić odrodzenie zbawienie pokrótce

bolesność poranny koronować i czego
wynarodowienie drętwieć ciało którego

Ocena 69.30%

głośno ognik świąteczny prześliczny że wkrótce
pozwolić odrodzenie zbawienie pokrótce

porozumieć potrzebny wyniesienie tego
wynarodowienie drętwieć ciało którego

Ocena 67.96%

nieadekwatność zatrzymywanie tam wkrótce
pozwolić odrodzenie zbawienie pokrótce

porozumieć potrzebny wyniesienie tego
wynarodowienie drętwieć ciało którego

Tak, jak w poprzednim przypadku, nałożenie dodatkowych wymagań na osobniki wyraźnie poprawiło uzyskane przy pomocy konfiguracji siódmej wyniki. Dodatkowo, po modyfikacji, wygenerowane wiersze stały się bardziej zróżnicowane. Jednakże, zmodyfikowanej konfiguracji siódmej nie udało się uzyskać tak wysokich wyników, jak zmodyfikowanej konfiguracji piątej. Przyczyną jest prawdopodobnie zbyt duży zakres zmian w populacji pomiędzy następującymi iteracjami.

5. Plany na przyszłość

Przeprowadzone badania pozwoliły uzyskać obiecujące wyniki. Warto kontynuować prace nad przetwarzaniem języka naturalnego, ponieważ wydaje się mieć ono wyjątkowo szeroki wachlarz zastosowań.

Elementem, który wymaga uwagi, jest bez wątpienia próba spełnienia warunku zawierania treści Manurunga. Do wygenerowania bardziej logicznych, spójnych znaczeniowo utworów literackich może doprowadzić próba zadania tematu wiersza. Obiecujące wydaje się być tu podejście *topic modeling*. Przy jego pomocy można podjąć próbę znalezienia tematu wygenerowanego wiersza. Następnie należałoby wybrać odpowiednią metrykę do porównywania tematów, po czym dokonać porównania jego zgodności z zadanym tematem. Odległość pomiędzy tymi dwoma tematami, zadanym i wygenerowanym, będzie kolejnym elementem funkcji przystosowania.

Sposób oceny poprawności gramatycznej jest kolejnym zagadnieniem, które warto rozwijać. Dobry model językowy, zbudowany na podstawie form tekstowych, cechuje się bardzo dużą ilością N-gramów, ponieważ występują w nich prawie wszystkie słowa i ich odmiany obecne w danym języku. W sposób oczywisty przekłada się to na jego rozmiar. Przeszukiwanie tak dużego zbioru jest zatem czasochłonne. Sprawdzanie poprawności gramatycznej pojedynczego wiersza dla potrzeb przeprowadzonego badania trwało ok. 1 sekundy. Posługując się wykorzystanym w pracy modelem, niemożliwe jest przeprowadzenie badań na dużej populacji przy dużej liczbie iteracji.

Model zbudowany na tagach morfosyntaktycznych będzie z pewnością efektywniejszy i pozwoli na szybsze uzyskiwanie wyników. Powodem tego jest stosunkowo mała ilość tagów morfosyntaktycznych, w porównaniu do wszystkich możliwych poprawnych form tekstowych, wstępujących w języku polskim. Mniejszy rozmiar modelu umożliwia szybsze obliczanie prawdopodobieństwa poprawności gramatycznej. Pozwoli to na przeprowadzanie badań z większą ilością osobników w populacji lub/oraz przez większą ilość iteracji i z pewnością podniesie jakość uzyskiwanych wyników.

6. Podsumowanie

Podczas przeprowadzania badań najlepsze wyniki uzyskano dla konfiguracji piątej. Konfiguracja ta, zawierała następujące parametry: gen odpowiadał pojedynczemu słowu, selekcja osobników odbywała się metodą koła ruletki oraz prawdopodobieństwo mutacji wynosiło 50%. Uzyskane przy takim układzie wyniki - wiersze, cechowały się powtarzającym się układem słów. Ostatnie słowa poszczególnych wersów dla większości z wygenerowanych utworów były identyczne. Oznacza to, że posiadają one wspólnego przodka, jednak ewolucja przebiegała prawidłowo, dzięki czemu osobniki te były odpowiednio zróżnicowane.

Podczas analizowania wyników zauważono też, że posługiwanie się wyłącznie wykresem zmian wartości funkcji przystosowania, w celu zdobycia informacji o uzyskanych wynikach, nie zawsze pozwala na uzyskanie miarodajnego obrazu efektywności danego rozwiązania. Koniecznym okazało się dokładniejsze zanalizowanie osobników w poszczególnych iteracjach.

Przykładem tego problemu może być konfiguracja 1. Na podstawie wykresu wnioskować można o dużym odchyleniu standardowym wyników dla poszczególnych osobników. Dokładna analiza wykazała jednak, że osobniki te są niemal identyczne - różni je zaledwie kilka wyrazów. Dodatkowo, w konfiguracjach z małym prawdopodobieństwem mutacji, zmiany nie były na tyle istotne, by zapobiec wyłonieniu się lidera, do którego zaczynały dążyć pozostałe wiersze.

Przedstawione powyżej sytuacje prowadziły algorytm do utykania w lokalnym maksimum. Dobrą ilustracją zaistniałego problemu jest tocząca się po pofałdowanej płaszczyźnie kula, posiadająca określoną energię. Jej celem jest odnalezienie najniższego punktu owej płaszczyzny. W pewnym momencie może natrafić ona na zagłębienie na tyle duże (jednocześnie nie będące poszukiwanym celem), że posiadana przez nią energia okaże się niewystarczająca, by się z niego uwolnić. Tocząca się kula symbolizuje tutaj algorytm przeszukujący przestrzeń rozwiązań. Energia odpowiada dynamice zmian. Większe szanse na występowanie znaczących różnic między kolejnymi populacjami, odpowiadają większej energii. Warto dodać jednak, że zmiany te nie mogą odbywać się również zbyt dynamicznie. Za duża energia kuli wprawiałyby ją w prędkość uniemożliwiającą dostanie się do części zagłębień, w tym do zagłębienia będącego potencjalnym celem.

Dodatkowo - przy podejściu wykorzystującym reprezentację pojedynczego genu osobnika, jako całego wersu - prawdopodobieństwo powstania w populacji identycznych osobników wzrasta. Potwierdzają to uzyskane wyniki. Drugie podejście - korzystające z reprezentacji genu, jako słowa - jest w mniejszym stopniu obarczone tym błędem.

Innym możliwym powodem uzyskiwania lepszych wyników dla genu, jako słowa, jest korzystniejszy przy tym modelu zakres zmian zachodzących w czasie mutacji. Zastępowanie całego wersu - nowym, jest zmianą inwazyjną i wpływającą w znaczący sposób na kształt wiersza. Możliwość mutacji pojedynczych słów daje większą elastyczność, a także pewną precyzję w doskonaleniu wygenerowanego utworu.

Przeprowadzone modyfikacje znacząco poprawiły uzyskane wyniki. Jest to skutek nałożenia na osobniki wymogu posiadania trzynastu sylab i rymów, co przełożyło się na znaczne ograniczenie przeszukiwanej przestrzeni. Jednakże, z powodu zmiany dostępnej przestrzeni rozwiązań, należałoby przeprowadzić ponownie proces dobierania parametrów algorytmu, aby znaleźć te najlepsze.

Kontynuowanie badań nad inżynierią lingwistyczną jest zajęciem posiadającym głęboki sens. Dzięki niej znaleźć można rozwiązania wielu problemów współczesnego człowieka. Jak wykazano we wprowadzeniu do niniejszej pracy, przetwarzanie języka naturalnego daje ogromne możliwości zastosowania. Wierzę, że praca ta ubogaca dotychczasowy dorobek badań nad tą dziedziną, pokazując kreatywne podejście do zagadnienia automatycznego generowania tekstu.

Bibliografia

[1] J. Chelin, L. Kosseim, T. Radhakrishnan *Using Natural Language Processing to Assist the Visually Handicapped in Writing Compositions.* Advances in Artificial Intelligence. AI 2006. Lecture Notes in Computer Science, vol 4013. Springer, Berlin, Heidelberg

[2] Y. Zhang J. Liu *Natural Language Processing for Foreign Languages Learning as Computer-based Learning Tools* Modern Applied Science, Vol. 3, No. 1, January 2009

[3] *Call center automation advances, but only as far as NLP can take it* http://searchcrm.techtarget.com/feature/Call-center-automation-advances-but-only-as-far-as-NLP-can-take-it(dostęp 10 września 2017)

[4] M. Khan, M. Durrani, A. Ali, I. Inayat S. Khalid, K. Khan *Sentiment analysis and the complex natural language* Complex Adaptive Systems Modeling, 2016

[5] H. Manurung. *An evolutionary algorithm approach to poetry generation.* PhD thesis, University of Edinburgh, 2004.

[6] M. V. van Mechelen *Computer poetry.* http://www.trinp.org/Poet/ComP/ComPoe.HTM (dostęp 10 września 2017)

[7] M. Masterman. *Computerized haiku.* Cybernetics, Art and Ideas, pages 175–183. New York Graphic Society Ltd., Greenwich, UK, 1971

[8] P. Gervás. *Wasp: Evaluation of different strategies for the automatic generation of spanish verse.* Proceedings of the AISB00 Symposium on Creative Cultural Aspects and Applications of AI Cognitive Science, Birmingham, UK, 2000.

[9] R. P. Levy. *A computational model of poetic creativity with neural network as measure of adaptive fitness.* In Proceedings of the ICCBR-01 Workshop on Creative Systems, 2001.

[10] R. Kurzweil, *Ray kurzweil's cybernetic poet.* http://www.kurzweilcyberart.com/poetry/rkcp_overview.php

[11] P. Gervás. *An expert system for the composition of formal spanish poetry.* Journal of Knowledge-Based Systems, 14:200–1, 2001.

[12] B. Díaz-Agudo, P. Gervas, and P. A. González-Calero. *Poetry generation in colibri.* In ECCBR '02: Proceedings of the 6th European Conference on Advances in Case-Based Reasoning, pages 73–102, London, UK, 2002. Springer-Verlag

[13] D. Jurafsky and J. Martin *Speech and language processing* An introduction to natural language processing, computational linguistics, and speech recognition (second edition) (Prentice Hall, Upper Saddle River, New Jersey, 2009).

[14] A. Mickiewicz *Pan Tadeusz, czyli Ostatni zajazd na Litwie* Paryż, Francja, 1834.

[15] Ł. Szałkiewicz and A. Przepiórkowski. *Narodowy Korpus Języka Polskiego. (Wydawnictwo Naukowe PWN, Warszawa, 2012)* Warszawa, ch. Anotacja morfoskładniowa, p. 59–96. eng. National Corpus of Polish, ch. Morphosyntactic annotation.

[16] A. Smywiński-Pohl, B. Ziółko *Application of Morphosyntactic and Class-based Language Models in Automatic Speech Recognition of Polish* International Journal on Artificial Intelligence Tools, Vol. 25, Issue 02, April 2016.

[17] *Zestaw znaczników morfosyntaktycznych* http://nkjp.pl/poliqarp/help/plse2.htmlx3-40002.2 (dostęp 10 września 2017)

[18] B. Ziołko, D. Skurzok. *„N-Grams Model For Polish" Department of Electronics AGH University of Science and Technology. Krakow, Poland.*
http://www.dsp.agh.edu.pl/$media/pl{:}resources{:}ngram-docu.pdf$

[19] A. Stolcke. *SRILM-an extensible language modeling toolkit.* In Proceedings of the International Conference on Spoken Language Processing 2 2002, p. 901–904.

[20] K. Wróbel *KRNNT: Polish Recurrent Neural Network Tagger* (w publikacji)

[21] A. Radziszewski and T. Śniatowski *Maca — a configurable tool to integrate Polish morphological data* Proceedings of the Second International Workshop on Free/Open-Source Rule-Based Machine Translation, Barcelona, Spain, 2011.