



**POLITECHNIKA KRAKOWSKA im. T.
Kościuszki**
Wydział Mechaniczny
**Instytut Technologii Maszyn i Automatyzacji
Produkcji**



Kierunek studiów : Automatyka i Robotyka
Specjalność : Sterowanie i Monitoring Maszyn i
Urządzeń

STUDIA STACJONARNE

PRACA DYPLOMOWA

MAGISTERSKA

Maciej Dąbrowski

SAMOADAPTACYJNY ALGORYTM EWOLUCYJNY

SELF-ADAPTIVE EVOLUTIONARY ALGORITHM

Promotor:
dr inż. **Stanisław Krenich**

Kraków, rok akad. 2019/2020

Autor pracy: **Maciej Dąbrowski**
Nr pracy:

OŚWIADCZENIE O SAMODZIELNYM WYKONANIU PRACY DYPLOMOWEJ

Oświadczam, że przedkładana przeze mnie praca dyplomowa magisterska/~~inżynierska~~^{*)} została napisana przeze mnie samodzielnie. Jednocześnie oświadczam, że ww. praca:

- 1) nie narusza praw autorskich w rozumieniu ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (Dz.U. z 2017 r. poz. 880 z późn. zm.) oraz dóbr osobistych chronionych prawem cywilnym, a także nie zawiera danych i informacji, które uzyskałem/~~am~~^{*)} w sposób niedozwolony,
- 2) nie była wcześniej podstawą żadnej innej procedury związanej z nadawaniem tytułów zawodowych, stopni lub tytułów naukowych.

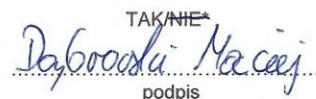
Jednocześnie wyrażam zgodę na:

- 1) poddanie mojej pracy kontroli za pomocą systemu antyplagiatowego oraz na umieszczenie tekstu pracy w bazie danych uczelni, w celu ochrony go przed nieuprawnionym wykorzystaniem. Oświadczam, że zostałem/~~am~~^{*)} poinformowany/~~a~~^{*)} i wyrażam zgodę, by system antyplagiatowy porównywał tekst mojej pracy z tekstem innych prac znajdujących się w bazie danych uczelni, z tekstami dostępnymi w zasobach światowego Internetu oraz z bazą porównawczą systemu antyplagiatowego,
- 2) to, aby moja praca pozostała w bazie danych uczelni przez okres, który uczelnia uzna za stosowny. Oświadczam, że zostałem poinformowany i wyrażam zgodę, że tekst mojej pracy stanie się elementem porównawczej bazy danych uczelni, która będzie wykorzystywana, w tym także udostępniana innym podmiotom, na zasadach określonych przez uczelnię, w celu dokonywania kontroli antyplagiatowej prac dyplomowych/doktorskich, a także innych tekstów, które powstaną w przyszłości.

Jednocześnie przyjmuję do wiadomości, że w przypadku stwierdzenia popełnienia przeze mnie czynu polegającego na przypisaniu sobie autorstwa istotnego fragmentu lub innych elementów cudzej pracy, lub ustalenia naukowego, właściwy organ stwierdzi nieważność postępowania w sprawie nadania mi tytułu zawodowego (art. 193 ustawy z dnia 27 lipca 2005 r. Prawo o szkolnictwie wyższym, Dz.U. z 2017 r., poz. 2183, z późn. zm.).


.....
podpis

- 3) Wyrażam zgodę na udostępnianie mojej pracy dyplomowej w Archiwum PK do celów naukowo-badawczych z poszanowaniem przepisów ustawy o prawie autorskim i prawach pokrewnych (Dz.U. z 2017 r. poz. 880 z późn. zm.).

TAK/NIE*

.....
podpis

* niepotrzebne skreślić

Uzgodniona ocena pracy:

.....
podpis promotora

.....
podpis recenzenta

SPIS TREŚCI

WYKAZ OZNACZEŃ	4
1. CEL I ZAKRES PRACY	5
2. WSTĘP TEORETYCZNY	6
2.1. Czym jest optymalizacja?.....	6
2.2. Matematyczny opis problemów optymalizacyjnych.....	6
2.3. Typy problemów optymalizacyjnych.....	8
2.4. Metody rozwiązywania problemów optymalizacyjnych.....	9
2.5. Algorytmy ewolucyjne.....	9
2.5.1. Podstawowe pojęcia algorytmów ewolucyjnych.....	10
2.5.2. Ogólna zasada działania algorytmów ewolucyjnych.....	11
2.5.3. Algorytmy genetyczne.....	15
2.5.4. Programowanie genetyczne.....	15
2.5.5. Programowanie ewolucyjne.....	16
2.5.6. Strategie ewolucyjne.....	16
2.5.7. Zastosowanie algorytmów ewolucyjnych.....	16
2.5.8. Samoadaptacja algorytmów ewolucyjnych.....	17
3. PROJEKT SAMOADAPTACYJNEGO ALGORYTMU EWOLUCYJNEGO	19
3.1. Projekt Algorytmu Ewolucyjnego Górnego Poziomu.....	19
3.2. Projekt Algorytmu Ewolucyjnego Dolnego Poziomu.....	22
4. APLIKACJA KOMPUTEROWA DO CELÓW BADAWCZYCH	24
5. EKSPERYMENT BADAWCZY	25
5.1. Problemy optymalizacyjne.....	25
5.1.1. Problem nr 1 – funkcja De Jonga F1.....	25
5.1.2. Problem nr 2 – funkcja De Jonga F3.....	26
5.1.3. Problem nr 3 – sprężyna naciskowa.....	27
5.2. Przebieg eksperymentu.....	29
5.3. Wyniki eksperymentu.....	32
5.3.1. Analiza wyników eksperymentu nr 1.....	33
5.3.1. Analiza wyników eksperymentu nr 2.....	45
5.3.2. Analiza wyników eksperymentu nr 3.....	47
5.3.3. Podsumowanie.....	59
5.4. Ocena Samoadaptacyjnego Algorytmu Ewolucyjnego.....	61
6. WNIOSKI	62
LITERATURA	63
SUMMARY	64

WYKAZ OZNACZEŃ

x_1, x_2, \dots – zmienne decyzyjne

X – wektor zmiennych decyzyjnych

g_1, g_2, \dots – ograniczenia nierównościowe

h_1, h_2, \dots – ograniczenia równościowe

f – funkcja celu

p – funkcja kary

AE – Algorytm Ewolucyjny

SAE – Samoadaptacyjny Algorytm Ewolucyjny

$AE GP$ – Algorytm Ewolucyjny Górnego Poziomu

$AEDP$ – Algorytm Ewolucyjny Dolnego Poziomu

1. Cel i zakres pracy

Celem badawczym pracy było określenie skuteczności działania samoadaptacyjnych algorytmów ewolucyjnych w jednokryterialnej optymalizacji dyskretniej oraz ciągłej nieliniowej.

Celem użytecznym pracy było stworzenie modelu samoadaptacyjnego algorytmu ewolucyjnego, opracowanie programu komputerowego pozwalającego na obserwację przebiegu działania oraz porównania skuteczności samoadaptacyjnego algorytmu ewolucyjnego na tle klasycznych algorytmów ewolucyjnych oraz algorytmu losowego.

Praca podzielona jest na pięć części, które dotyczą kolejnych etapów wykonywania projektu:

- Wstęp teoretyczny zawierający informacje z tematyki metod optymalizacyjnych oraz algorytmów ewolucyjnych.
- Opracowanie oraz przedstawienie modelu algorytmu samoadaptacyjnego ewolucyjnego, który został użyty w kolejnych etapach pracy.
- Opracowanie oraz przedstawienie stworzonej aplikacji, która została przygotowana w ramach pracy badawczej.
- Opis przebiegu eksperymentu badawczego mającego na celu określić skuteczność samoadaptacyjnych algorytmów ewolucyjnych. Do elementów tej części pracy należą między innymi:
 - Przedstawienie problemów optymalizacyjnych użytych w eksperymencie.
 - Opis przebiegu całego eksperymentu.
 - Przedstawienie i analiza otrzymanych wyników.
 - Analiza wyników wszystkich algorytmów biorących udział w eksperymencie oraz przebiegu procesu optymalizacyjnego samoadaptacyjnych algorytmów ewolucyjnych.
- Sformułowanie wniosków oraz podsumowanie wartości merytorycznej pracy.

2. Wstęp Teoretyczny

2.1. Czym jest optymalizacja?

„Proces optymalizacji polega na przeszukiwaniu przestrzeni potencjalnych rozwiązań danego problemu w celu znalezienia najlepszego rozwiązania [1].” Określenie czy odnalezione rozwiązanie jest tym najlepszym (optymalnym) nie zawsze jest proste i możliwe do wykonania w krótkim czasie, dlatego proces poszukiwania i porównywania rozwiązań powinien mieć określony warunek zakończenia procesu optymalizacyjnego. Warunek ten nazywany jest warunkiem stopu i określa on kiedy dalsze poszukiwanie ma być zaprzestane. Oto kilka przykładowych warunków stopu:

- Zostało odnalezione rozwiązanie optymalne – ten warunek stopu może zostać zastosowany tylko w przypadku użycia metody optymalizacyjnej gwarantującej odnalezienie rozwiązania optymalnego. Dla wielu problemów optymalizacyjnych nie jest możliwe jego zastosowanie.
- Został osiągnięty założony cel – zatrzymanie procesu optymalizacji następuje, gdy odnalezione rozwiązanie jest satysfakcjonujące.
- Lepsze rozwiązanie nie może zostać odnalezione – warunek zatrzymania aktywowany jest, gdy nowo odnalezione rozwiązania od jakiegoś czasu nie wykazują poprawy.
- Nastąpiło przekroczenie ograniczenia czasowego dla procesu optymalizacji – proces optymalizacji jest zatrzymywany, gdy limit czasu poświęcony na cały proces został wyczerpany [2].

W procesie optymalizacji oprócz określenia warunku stopu, kluczowe jest również określenie kryterium jakościowego, dzięki któremu będziemy w stanie określić czy dane rozwiązanie jest dobre lub złe. Kryterium takim może być np. koszt wyprodukowania, wytrzymałość produktu, ilość wyprodukowanych odpadów, lub inne czynniki. W większości prac oraz odkryć naukowych w dziedzinie optymalizacji rozważa się tylko jedno kryterium optymalizacji, jednak większość problemów w świecie rzeczywistym rozważanych jest w więcej niż jednym kryterium. Dla przykładu, większość firm produkcyjnych optymalizuje wymiary swoich produktów, tak aby zminimalizować koszty produkcji i jednocześnie zmaksymalizować wytrzymałość wytwarzanego produktu. Ze względu na liczbę uwzględnionych kryterium, wyróżniamy:

- optymalizację jednokryterialną
- optymalizację wielokryterialną [3]

2.2. Matematyczny opis problemów optymalizacyjnych

Aby usprawnić proces optymalizacji i zastosować aparat matematyczny do poszukiwania lub wyznaczania rozwiązań optymalnych, w pierwszej kolejności musimy ustalić matematyczny opis problemów optymalizacyjnych. Poniżej przedstawiono najważniejsze pojęcia metod optymalizacji wraz z ich matematycznym opisem używanym w dalszej części pracy:

- **Zmienne decyzyjne** – są to zmienne, których wielkości należy wyznaczyć w ramach zadania optymalizacyjnego. Oznaczmy je jako:

$$x_1, x_2, \dots, x_n$$

- **Wektor zmiennych decyzyjnych** – jest to uporządkowany zbiór zmiennych decyzyjnych. Oznaczmy go jako:

$$X$$

Zauważmy, że:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix}$$

- **Rozwiązanie** – uporządkowany zbiór wartości zmiennych decyzyjnych. Oznaczmy je jako:

$$\bar{X}$$

Zauważmy, że:

$$\bar{X} = \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \dots \\ \bar{x}_n \end{bmatrix}$$

gdzie:

$$\bar{x}_i \in \mathbb{R}, \text{ dla każdego } i \in \mathbb{Z} \text{ oraz } 1 \leq i \leq n$$

- **Przestrzeń rozwiązań** – przeszukiwana przestrzeń rozwiązań, na której opisana jest funkcja celu. Może być to pewien zbiór skończony, zbiór przeliczalny, przestrzeń euklidesowa \mathbb{R}^n lub jej podzbiór. Oznaczmy ją jako:

$$\mathbb{X}$$

- **Przestrzeń możliwych rozwiązań** – podzbiór przestrzeni wszystkich rozwiązań. Oznaczmy ją jako:

$$\mathbb{X}_D$$

Zauważmy, że:

$$\mathbb{X}_D \in \mathbb{X}$$

- **Warunki ograniczające** – zwane są też funkcjami ograniczeń, określają ograniczenia narzucone na zmienne decyzyjne. Wyróżniamy *ograniczenia równościowe* oraz *ograniczenia nierównościowe*.

Oznaczmy *ograniczenia nierównościowe* jako funkcje:

$$g_1, g_2, \dots, g_m$$

Natomiast *ograniczenia równościowe* jako funkcje:

$$h_1, h_2, \dots, h_k$$

Zauważmy, że:

$$\mathbb{X}_D = \{x \in \mathbb{X} : g_i(x) = 0 \text{ dla } i \in \{1, 2, \dots, m\} \text{ oraz } h_i(x) \leq 0 \text{ dla } i \in \{1, 2, \dots, k\}\}$$

- **Funkcja celu** – zwana też jest funkcją kryterialną. Określa ona kryterium jakościowe, które umożliwia ocenę konkretnych rozwiązań zadania optymalizacyjnego. W ramach zadania optymalizacyjnego szukamy takiego rozwiązania, dla którego funkcja ta osiąga swoje minimum lub maksimum. Oznaczmy ją jako:

$$f$$

Zauważmy, że:

$$f: \mathbb{X} \rightarrow \mathbb{R}$$

- **Funkcja kary** – funkcja kary ma za zadanie stanowić „barierę” nie dopuszczającą do przekraczania ograniczeń i może być składową funkcji celu. Oznaczmy ją jako:

$$p$$

Zauważmy, że:

$$p(x) = 0 \text{ dla } x \in \mathbb{X}_D$$

$$p(x) > 0 \text{ dla } x \notin \mathbb{X}_D$$

2.3. Typy problemów optymalizacyjnych

Problemy optymalizacyjne można dzielić ze względu na wiele czynników. Jednym z nich może być poszukiwana wartość funkcji celu i tak wyróżniamy:

- Zadania minimalizacji – poszukiwane jest rozwiązanie, dla którego funkcja celu osiąga wartość minimalną:

$$f(X_{opt}) \leq f(X) \text{ dla każdego } X \in \mathbb{X}$$

- Zadania maksymalizacji – poszukiwane jest rozwiązanie, dla którego funkcja celu osiąga wartość maksymalną:

$$f(X_{opt}) \geq f(X) \text{ dla każdego } X \in \mathbb{X}$$

Nie trudno zauważyć, że każde zadanie maksymalizacji można sprowadzić do zadania minimalizacji i na odwrót, gdyż:

$$\max_{X \in \mathbb{X}} f(X) = - \min_{X \in \mathbb{X}} -f(X)$$

Jeśli weźmiemy pod uwagę przestrzeń rozwiązań, możemy dokonać podziału problemów optymalizacyjnych na:

- Problemy optymalizacji ciągłej – w tego typu problemach poszukiwanie wszystkich zmiennych decyzyjnych odbywa się w zbiorze liczb rzeczywistych:

$$\mathbb{X} = \mathbb{R}^n$$

Ze względu na charakter funkcji celu możemy dokonać dodatkowego podziału na:

- Problemy optymalizacji ciągłej liniowej – funkcja celu wyrażona jest w postaci liniowej funkcji zmiennych decyzyjnych:

$$f(X) = a_0 + a_1x_1 + a_2x_2 + \dots + a_nx_n$$

- Problemy optymalizacji ciągłej nieliniowej – funkcja celu wyrażona jest w postaci nieliniowej funkcji zmiennych decyzyjnych.

- Problemy optymalizacji dyskretnej – w tego typu problemach poszukiwanie wszystkich zmiennych decyzyjnych odbywa się w zbiorze liczb całkowitych:

$$\mathbb{X} = \mathbb{Z}^n$$

- Problemy optymalizacji dyskretno-ciągłej – są to problemy mieszane, w których część zmiennych decyzyjnych poszukiwanych jest w zbiorze liczb rzeczywistych, natomiast pozostałe z nich w zbiorze liczb całkowitych.

2.4. Metody rozwiązywania problemów optymalizacyjnych

Metody optymalizacji jako dziedzina nauki wyróżnia mnóstwo różnych sposobów rozwiązywania problemów optymalizacyjnych i wybór najlepszej metody rozwiązywania konkretnego zadania optymalizacyjnego można uznać jako problemem sam w sobie. Naturalnym i intuicyjnym rozwiązaniem tej sytuacji wydaje się podejście, w którym najpierw staramy się odnaleźć technikę (o ile takowa istnieje), która gwarantuje odnalezienie rozwiązania optymalnego – mówimy wtedy o **dokładnych metodach** rozwiązywania problemów optymalizacyjnych. Jeśli z jakiś przyczyn użycie **metody dokładnej** nie jest możliwe, wtedy powinniśmy zastosować jedną z **metod przybliżonych**, które co prawda nie zagwarantują nam odnalezienia rozwiązania optymalnego, jednak pozwolą na efektywne przeszukiwanie przestrzeni rozwiązań. Przykładami **metod dokładnych** są:

- Metoda SIMPLEX – może zostać zastosowana wyłącznie do wyznaczania rozwiązania optymalnego dla problemów optymalizacji ciągłej liniowej.
- Efektywne algorytmy dedykowane dla danego problemu – ze względu na to, że nie wszystkie problemy posiadają dedykowane algorytmy, ta metoda nie zawsze może być zastosowana.
- Przeszukiwanie całego zbioru rozwiązań dopuszczalnych – ze względu na ograniczenia czasowe (przeszukiwanie całego zbioru rozwiązań dopuszczalnych może trwać miliony lat lub nigdy się nie skończyć, jeśli przestrzeń rozwiązań jest nieskończenie wielka).

Natomiast, przykładami **metod przybliżonych** są:

- Metoda Monte-Carlo
- Metody gradientowe – stosowane do efektywnego znajdowania lokalnego optimum
- Metoda symulowanego wyżarzania.
- Algorytmy oparte na inteligencji roju – należą do nich między innymi algorytmy pszczołe oraz mrówkowe.
- Algorytmy ewolucyjne – algorytmy wzorowane na biologicznej ewolucji.

2.5. Algorytmy ewolucyjne

„Algorytm ewolucyjny jest to termin określający przybliżony algorytm optymalizacyjny, w którym stosowane są mechanizmy selekcji, reprodukcji i mutacji inspirowane przez biologiczny proces ewolucji. W biologicznym procesie ewolucji na daną populację osobników działa presja środowiska powodując naturalną selekcję. Tylko najlepiej przystosowane osobniki mają szansę na przetrwanie i zapoczątkowanie nowych coraz to lepszych populacji. W algorytmie ewolucyjnym problem, który mamy rozwiązać, gra rolę środowiska, w którym żyje populacja osobników. Każdy osobnik reprezentuje potencjalne (możliwe) rozwiązanie problemu. Podobnie jak w procesie biologicznym, algorytm ewolucyjny tworzy stopniowo coraz to lepsze rozwiązania. Stąd wynika, że algorytm ewolucyjny może służyć do rozwiązywania problemów optymalizacyjnych (...).

Działanie algorytmu ewolucyjnego można opisać następująco: algorytm ewolucyjny rozpoczyna proces przeszukiwania od utworzenia populacji potencjalnych rozwiązań nazywanych osobnikami, które są reprezentowane przez chromosomy zawierające genetyczną informację o osobnikach. W każdym ewolucyjnym kroku, nazywanym generacją, chromosomy

są dekodowane i ocenianie zgodnie z pewnym z góry przyjętym kryterium jakości nazywanym przystosowaniem (funkcją przystosowania może być na przykład funkcja celu), a następnie przeprowadzana jest selekcja w celu eliminacji osobników ocenionych jako najgorsze. Osobniki wykazujące wysokie przystosowanie podlegają mutacji oraz rekombinacji przeprowadzanej przy pomocy operatora krzyżowania. Sama selekcja nie wprowadza żadnego nowego osobnika do populacji, tj. nie znajduje nowych punktów w przestrzeni poszukiwań, natomiast takie punkty wprowadzane są przez krzyżowanie i mutację. Dzięki krzyżowaniu ewolucyjny proces może się przesunąć w kierunku obiecujących obszarów w przestrzeni poszukiwań. Mutacja zapobiega zbieżności do lokalnego optimum. W wyniku działania operatora krzyżowania i mutacji tworzone są nowe rozwiązania, z których następnie budowana jest populacja następnej generacji. Warunkiem zakończenia algorytmu może być na przykład pewna określona liczba generacji albo osiągnięcie zadawalającego poziomu przystosowania [1].”

W obecnych czasach wyróżnia się cztery podstawowe typy algorytmów ewolucyjnych:

- Algorytmy genetyczne
- Programowanie genetyczne
- Programowanie ewolucyjne
- Strategie ewolucyjne [3, 4]

2.5.1. Podstawowe pojęcia algorytmów ewolucyjnych

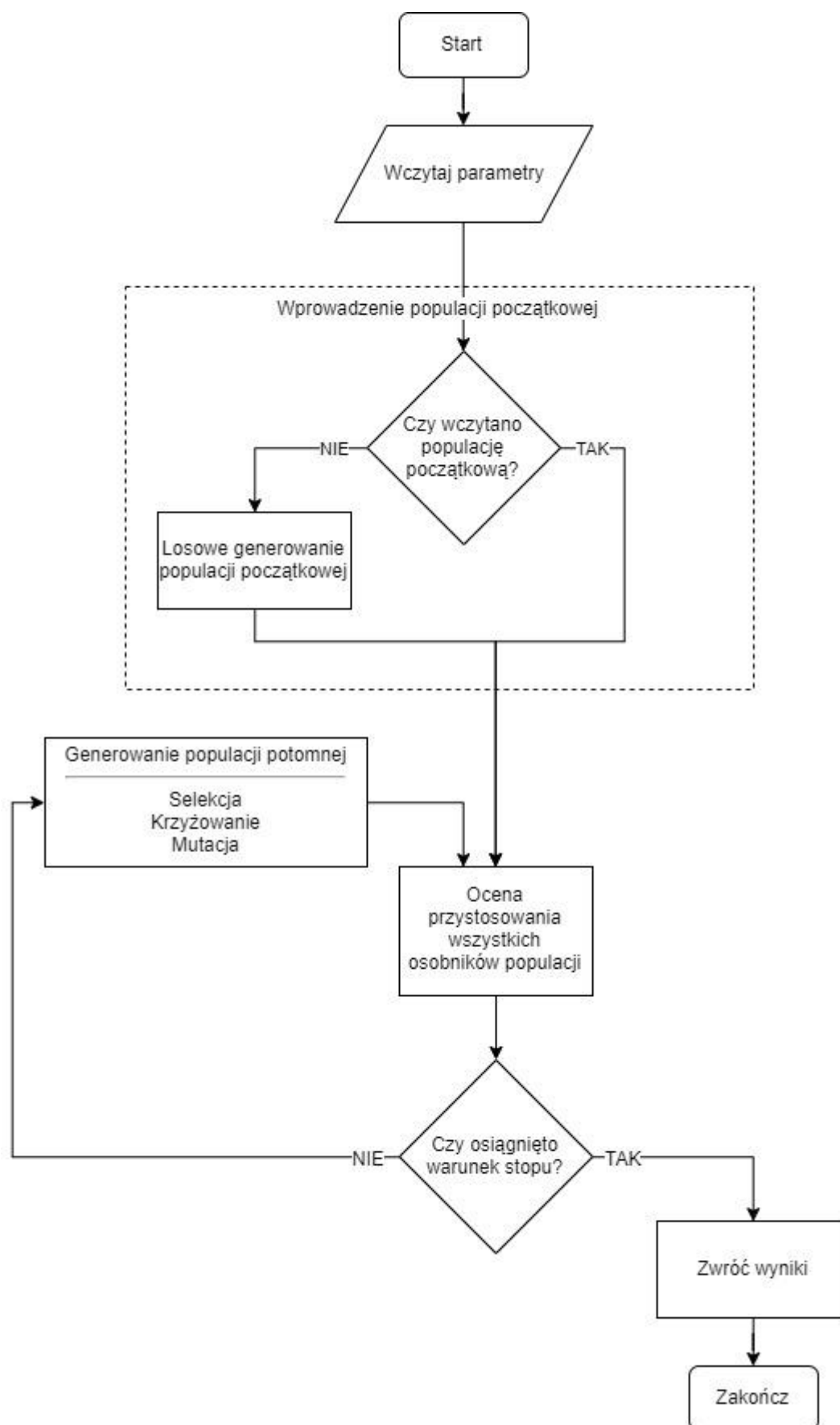
Algorytmy ewolucyjne oraz genetyczne zaczerpnęły wiele pojęć wprost z genetyki. Do najczęściej używanych terminów należą:

- *Populacja* – zbiór osobników o określonej liczebności, który jest w danej chwili rozważany (np. w danej iteracji działania algorytmu).
- *Osobnik* – nazywane też *organizmami*, są to parametry zadania zakodowane w postaci chromosomów.
- *Chromosomy* – zwane też *łańcuchami* są to uporządkowane ciągi genów. Długość chromosomów jest uzależniona od tego jak zostało sformułowane zadanie optymalizacyjne.
- *Gen* – pojedyncza cecha, czyli najmniejszy element genotypu. Jest to podstawowa jednostka, która jest przekazywana podczas dziedziczenia.
- *Genotyp* – zespół chromosomów danego osobnika. W algorytmach genetycznych najczęściej przyjmuje się, że genotyp składa się z jednego chromosomu, co oznacza że pojedynczy chromosom traktowany jest jako osobnik populacji.
- *Fenotyp* – rozwiązanie ze zdekodowaną strukturą (punkt w przestrzeni poszukiwań).
- *Allel* – wartość danego genu lub cechy.
- *Locus* – pozycja danego genu w chromosomie.
- *Funkcja przystosowania* – wartość funkcji przystosowania osiąga wartość wyższą dla osobników reprezentujących lepsze rozwiązania, a niższą dla osobników reprezentujących rozwiązania gorsze. Może być wyrażona przy użyciu funkcji celu [5].

2.5.2. Ogólna zasada działania algorytmów ewolucyjnych

Zasada działania klasycznego algorytmu ewolucyjnego przypomina przeprowadzenie symulacji w środowisku naturalnym. Schemat blokowy obrazujący kolejne kroki wykonywane w trakcie jego działania zostały przedstawione na *Rysunek 1*. Wyróżnić możemy kilka kluczowych etapów działania tego algorytmu:

- **Wczytanie parametrów** – określenie parametrów algorytmu ewolucyjnego można przyrównać do modelowania warunków środowiskowych, w których odbywać się będzie symulacja.
- **Wprowadzenie populacji początkowej** – określenie warunków początkowych symulacji poprzez wprowadzenie rozwiązań do zerowej iteracji algorytmu. Może się to odbywać losowo lub poprzez selekcję osobników pewnej znanej już populacji.
- **Ocena przystosowania wszystkich osobników obecnej populacji** – wykonywana jest poprzez określenie wartości funkcji celu (przystosowania) dla wszystkich obecnie rozważanych rozwiązań (organizmów).
- **Generowanie populacji potomnej** – wykorzystanie funkcji selekcji, krzyżowania oraz mutacji (określone są przez warunki środowiskowe) do wygenerowania nowej grupy rozwiązań.
- **Zwrócenie wyników** – określenie i wyróżnienia najlepszego rozwiązania.



Rysunek 1: Schemat działania algorytmu ewolucyjnego [4]

2.5.2.1. Selekcja

Selekcja jest pierwszym z etapów generowania populacji potomnej i polega na wybraniu tych osobników, które będą brały udział w tworzeniu potomków (będą rodzicami) następnego pokolenia. Wybór ten odbywa się w zgodzie z zasadą selekcji naturalnej, to znaczy że osobniki najlepiej przystosowane mają największą szansę na zostanie rodzicami. Istnieje wiele technik selekcji, a najbardziej popularnymi są:

- **Selekcja proporcjonalna** – metoda selekcji, w której rodzice wybierani są z prawdopodobieństwem proporcjonalnym do wartości ich funkcji przystosowania. Metoda ta nazywana jest często **metodą ruletki**, gdyż zasada jej działania przypomina losowanie z użyciem koła ruletki o szerokości pól odpowiadającej wartości funkcji przystosowania kolejnych osobników. Prawdopodobieństwo wyboru i-tego osobnika przy zastosowaniu tej reguły wynosi:

$$P_i = f(\bar{X}_i) / \sum_{j=1}^n f(\bar{X}_j)$$

gdzie:

f – funkcja przystosowania

n – liczba osobników danej populacji [6]

- **Selekcja rangowa** – przypomina selekcję proporcjonalną, z tą różnicą, że prawdopodobieństwo wyboru zależy od rangi tj. liczby porządkowej osobnika przy zastosowaniu sortowania według wartości funkcji przystosowania. Prawdopodobieństwo wyboru i-tego osobnika wynosi:

$$P_i = \frac{2-s}{n} + \frac{2 \cdot (i-1) \cdot (s-1)}{n \cdot (n-1)}$$

przy założeniu, że:

$$\begin{aligned} f(\bar{X}_i) &\leq f(\bar{X}_{i+1}) \\ 1 &< s \leq 2 \end{aligned}$$

gdzie:

f – funkcja przystosowania

n – liczba osobników danej populacji

s – współczynnik presji społeczeństwa (im większą ma wartość, tym silniej promowane są lepiej przystosowane osobniki) [7]

- **Uniwersalna selekcja stochastyczna** – każdy z osobników ma taką samą szansę na zostanie wybranym, co oznacza, że prawdopodobieństwo wyboru i-tego osobnika wynosi:

$$P_i = \frac{1}{n}$$

gdzie:

n – liczba osobników danej populacji [6]

- **Pojedyncza selekcja turniejowa** – metoda selekcji polegająca wymieszaniu i podziale populacji na małe grupy, najczęściej złożone z czterech osobników. Z każdej grupy wybieranych jest dwóch osobników o najwyższej wartości funkcji przystosowania [6].

- **Podwójna selekcja turniejowa** – metoda selekcji polegająca na losowym wyborze grupy k osobników z populacji, a następnie wyznaczeniu najlepiej dostosowanego osobnika z tej grupy, który zostaje pierwszym rodzicem. W celu wyboru kolejnych rodziców przyszłego potomstwa, cała procedura jest powtarzana [6].

2.5.2.2. Krzyżowanie

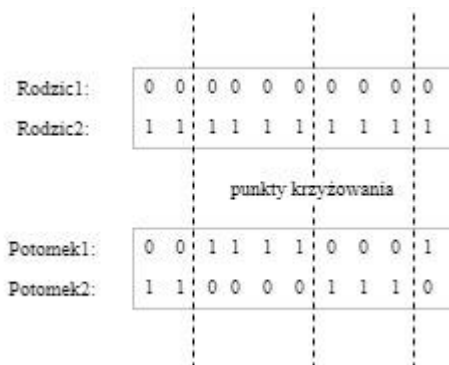
Krzyżowanie jest to operacja polegająca na wymianie materiału genetycznego między parą rodziców, w celu otrzymania pary potomstwa. Do najbardziej znanych operacji krzyżowania należą:

- **Krzyżowanie jednopunktowe** – polega na rozcięciu łańcuchów w jednym losowo wybranym punkcie, a następnie przekazaniu wymieszanych genów do potomstwa tj. pierwszy potomek otrzymuje geny od pierwszego rodzica znajdujące się przed punktem krzyżowania i geny od drugiego rodzica znajdujące się za punktem przecięcia. Drugi potomek otrzymuje pozostałe geny [6]. *Rysunek 2* przedstawia przykład krzyżowania jednopunktowego.



Rysunek 2: Krzyżowanie jednopunktowe

- **Krzyżowanie wielopunktowe** – zasada działania jest analogiczna jak przy krzyżowaniu jednopunktowym z tą różnicą, że łańcuch przecięty jest w kilku punktach, co prowadzi do większego wymieszania genów. *Rysunek 3* przedstawia przykład krzyżowania wielopunktowego dla 3 punktów krzyżowania [6].



Rysunek 3: Krzyżowanie wielopunktowe

- **Krzyżowanie jednostajne** – ta metoda krzyżowania polega na każdorazowym losowaniu, czy dany gen pierwszego rodzica trafi do pierwszego czy drugiego potomka (gen drugiego rodzica trafia do niewylosowanego potomka). Pozwala to uniknąć

stosowania punktów przecięć, które z reguły przenoszą sąsiadujące geny do tego samego potomka [6].

- **Krzyżowanie adaptacyjne** – krzyżowania adaptacyjne wykorzystuje dodatkowy łańcuch binarny jako wzorzec wskazujący, które geny pierwszego rodzica trafią do którego potomka. *Rysunek 4* przedstawia przykładowe krzyżowanie adaptacyjne z użyciem wzorca [6].

Rodzic1:	0 0 0 0 0 0 0 0 0 0
Rodzic2:	1 1 1 1 1 1 1 1 1 1
Wzorzec:	0 0 1 0 0 1 1 1 0 0
Potomek1:	0 0 1 0 0 1 1 1 0 0
Potomek2:	1 1 0 1 1 0 0 0 1 1

Rysunek 4: Krzyżowanie adaptacyjne

2.5.2.3. Mutacja

Mutacja ma na celu wprowadzenie drobnych modyfikacji genów, dzięki czemu możliwe jest przeszukiwanie nowych obszarów przestrzeni rozwiązań. Do najpopularniejszych typów mutacji należą:

- **Mutacja punktowa** – polega na zmianie jednego losowo wybranego genu.
- **Mutacja wielopunktowa** – polega na zmianie kilku losowo wybranych genów.
- **Mutacja probabilistyczna** – polega na zmianie każdego genu z góry zadany prawdopodobieństwem [6].

2.5.3. Algorytmy genetyczne

„Algorytmy genetyczne stanowią najlepiej znaną klasę algorytmów ewolucyjnych. Algorytmy te zostały zapoczątkowane przez J. Hollanda w 1960 r. Tradycyjnie chromosomy w algorytmach genetycznych są ciągami binarnymi o ustalonej długości (chromosomy mogą być także kodowane za pomocą ciągów liczb całkowitych lub rzeczywistych). Chromosomy są oceniane w każdej generacji. Rozmiar populacji jest stały. Parametrami, które muszą być określone są: wielkość populacji, prawdopodobieństwo mutacji oraz warunek zakończenia działania algorytmu. Z góry musi być również określona funkcja przystosowania [1].”

2.5.4. Programowanie genetyczne

„Programowanie genetyczne wykorzystuje zasady genetyki i darwinowskiej naturalnej selekcji do tworzenia programów komputerowych. Programowanie genetyczne jest w dużym stopniu podobne do algorytmów genetycznych. Podstawowa różnicą między programowaniem genetycznym i algorytmem genetycznym leży w reprezentacji rozwiązania. Podczas gdy algorytm genetyczny tworzy ciąg liczb, który reprezentuje rozwiązanie problemu, w programowaniu genetycznym osobniki są programami o strukturze drzew, a operatory genetyczne są stosowane do gałęzi i węzłów w tych drzewach [1].”

2.5.5. Programowanie ewolucyjne

„Programowanie ewolucyjne skupia się głównie na problemach optymalizacji z ciągłymi parametrami. W programowaniu ewolucyjnym każdy osobnik rodzicielski w populacji generuje potomka na drodze mutacji. Prawdopodobieństwo mutacji ma na ogół rozkład równomierny. Po ocenie potomków pewien wariant stochastycznej selekcji turniejowej wybiera pewną ilość najlepszych osobników spośród zespołu rodziców i potomków. Najlepszy osobnik jest zawsze przechowywany, co zapewnia, że jeżeli optimum zostanie osiągnięte, nie może on zostać zgubiony. Programowanie ewolucyjne jest algorytmem stosującym wyłącznie mechanizmy selekcji i mutacji (bez krzyżowania) [1].”

2.5.6. Strategie ewolucyjne

„Istnieją dwie główne strategie ewolucyjne (μ, β) ES i $(\mu + \beta)$ ES, gdzie μ rodziców produkuje β potomków wykorzystując operator krzyżowania i mutacji. W (μ, β) ES najlepszych β potomków przeżywa i zastępuje rodziców. W rezultacie rodzice nie są obecni w następnej populacji. Przeciwnie, $(\mu + \beta)$ ES dopuszcza przeżycie zarówno potomków jak i rodziców. W $(\mu + \beta)$ ES stosowana jest strategia elitarna (najlepszy osobnik zawsze jest kopiowany do następnej populacji). W obu wymienionych strategiach przeprowadzana jest zarówno operacja krzyżowania jak i mutacji [1].”

2.5.7. Zastosowanie algorytmów ewolucyjnych

„Algorytmy ewolucyjne mogą być użyte do rozwiązywania wszelakich problemów, zarówno ciągłych jak i dyskretnych oraz dyskretno-ciągłych. Jako, że są one jedną z metod przybliżonych, najczęściej stosuje się je do rozwiązywania problemów nieliniowych, nieciągłych, źle uwarunkowanych lub trudnych do matematycznego sformułowania.

Algorytmy genetyczne stosuje się jako doskonałe narzędzie do poprawienia efektywności innych metod optymalizacyjnych poprzez wskazanie dobrych punktów startowych do tych metod [5].”

„Ważnym obszarem zastosowań algorytmów ewolucyjnych jest harmonogramowanie i planowanie procesów przemysłowych. Problemy harmonogramowania, gdzie należy przydzielić zasoby do kolekcji zadań, są zwykle trudne do rozwiązania z powodu występowania różnorodnych ograniczeń oraz złożonych struktur produktów. Stosowane tutaj techniki programowania matematycznego pozwalają zwykle na uzyskanie rozwiązań tylko dla problemów o małych rozmiarach (...).

Kolejnym obszarem zastosowań algorytmów ewolucyjnych jest przemysł chemiczny. Zakłady chemiczne zawierają urządzenia takie jak pompy, destylatory i reaktory chemiczne, które tworzą złożoną sieć wzajemnie powiązanych strumieni materiału, ogrzewania i informacji, przeznaczonych do przeprowadzania procesu chemicznego, podczas którego surowy materiał jest przekształcany w wymagany produkt. Optymalizacja w zakładach chemicznych polega na modyfikacji struktury parametrów operacyjnych tak, aby znaleźć globalne optimum w celu wykonania pewnego zadania chemicznego. Klasyczne podejścia do rozwiązywania problemów chemicznych nie pozwalają na otrzymanie rozwiązań w zadowalającym czasie. Techniki gradientowe zbiegają do lokalnych optimum i nie są

odpowiednie dla występujących tu problemów nieróżniczkowalnych. Stąd też wynika zainteresowanie stosowaniem technik opartych na algorytmach ewolucyjnych w tym klasycznych algorytmów genetycznych (...).

Innym przykładem związanym z zastosowaniem przemysłowym może być wykorzystanie algorytmu genetycznego do kontroli fermentacji piwa. Algorytm genetyczny jest tutaj użyty do dopasowania profilu temperatury mieszanki w ustalonym okresie czasu.

Algorytmy ewolucyjne znalazły również dość szerokie zastosowanie w medycynie. Większość medycznych decyzji może być sformułowana jako poszukiwanie w pewnej odpowiedniej przestrzeni. Na przykład radiolog planując serię naświetlań poszukuje najlepszego sposobu leczenia w przestrzeni wszystkich możliwych sposobów. Przestrzenie poszukiwań w medycynie są zwykle bardzo duże i złożone. Decyzje są oparte na testach klinicznych, które dostarczają ogromnych ilości danych. W oparciu o te dane trzeba ostatecznie podjąć jedną decyzję (np. patolog musi stwierdzić czy nowotwór jest łagodny czy złośliwy na podstawie cech komórek, czyli wśród wszystkich możliwych cech komórek poszukuje się zbioru cech, który pozwala postawić właściwą diagnozę). Algorytmy ewolucyjne są stosowane w medycynie do wykonywania zadań, które mogą być podzielone na trzy grupy: (a) eksploracja danych, głównie w celu diagnozowania i prognozowania, (b) obrazowanie i przetwarzanie sygnałów, (c) planowanie i harmonogramowanie. (a) Eksploracja danych (odkrywanie wiedzy) jest procesem znajdowania wzorów, trendów i regularności poprzez „przesiewanie” dużych ilości danych. W medycznej eksploracji danych algorytmy ewolucyjne są zwykle wykorzystywane w celu znajdowania wartości parametrów ustawianych przez projektanta tak, aby przeszukiwane dane były interpretowane w sposób optymalny (np. znajdowanie wag w sieciach neuronowych). (b) Wiele danych medycznych jest wyrażanych za pomocą obrazów lub innych sygnałów. Algorytmy ewolucyjne znajdują tu zastosowanie do poprawiania działania algorytmów przetwarzających sygnały (np. filtrów lub kompresorów) przez znalezienie ich optymalnych parametrów. Mogą też zostać wykorzystane do bezpośredniego wyprowadzenia użytecznej informacji z dostarczonych danych. Algorytmy ewolucyjne szczególnie dobrze nadają się do rozwiązywania problemów planowania i harmonogramowania (c). Przykładem może tu być problem harmonogramowania dla pacjenta, poddawanego różnym medycznym procedurom i potrzebującego konsultacji różnych specjalistów, w celu optymalizacji zarówno czasu oczekiwania pacjenta jak również wykorzystania aparatury [1].”

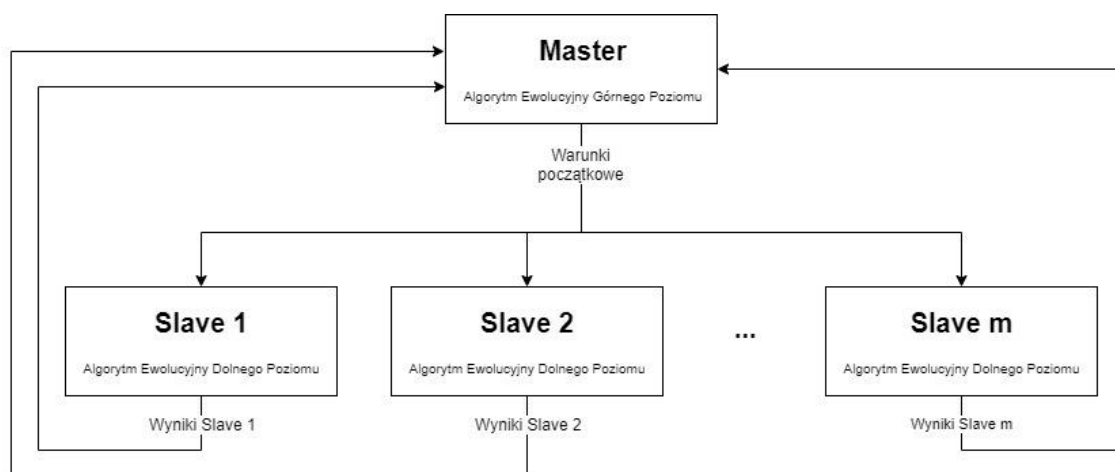
2.5.8. Samoadaptacja algorytmów ewolucyjnych

Skuteczność algorytmów ewolucyjnych w dużej mierze zależy od ich konfiguracji (wybór funkcji selekcji, krzyżowania, mutacji, rozmiaru populacji oraz parametrów powiązanych takich jak współczynnik mutacji). Nieefektywne dobranie parametrów algorytmów ewolucyjnych może prowadzić do gwałtownego obniżenia skuteczności ich działania. Za przykład może posłużyć dobór współczynnika mutacji, którego zbyt niska wartość może prowadzić do braku postępu na pewnym etapie procesu optymalizacji (utknięcia w optimum lokalnym ze względu na brak możliwości przeszukiwania rozwiązań sąsiadujących przestrzeni rozwiązań), natomiast obranie zbyt wysokiej wartości tego współczynnika spowoduje, że działania algorytmu ewolucyjnego przypominać będą algorytm losowy.

Niestety nie istnieje jedna uniwersalna metoda wyboru współczynników algorytmów ewolucyjnych, gdyż zależy ona od charakterystyki rozwiązywanego problemu. Ponadto można podejrzewać, że optymalne parametry algorytmów ewolucyjnych zmieniają się z kolejnymi iteracjami rozwiązywania problemu optymalizacyjnego. Biorąc pod uwagę wszystkie te aspekty, pokusić się można o konkluzję, że dobór parametrów algorytmu ewolucyjnego jest problemem optymalizacyjnym samym w sobie. [8, 9]

Koncepcja samoadaptacyjnych algorytmów ewolucyjnych polega na dwustopniowej optymalizacji zarówno rozwiązywanego problemu, ale również nastaw algorytmów ewolucyjnych. *Rysunek 5: Schemat samoadaptacyjnego algorytmu ewolucyjnego* ukazuje dwupoziomową optymalizację, w której możemy wyróżnić:

- Algorytm Ewolucyjny Górnego Poziomu (Master) – nadrzędny algorytm, którego zadaniem jest optymalizacja nastaw AEDP w trakcie rozwiązywania problemu optymalizacyjnego.
- Algorytmy Ewolucyjne Dolnego Poziomu (Slave) – algorytmy będące osobnikami AEGP, których funkcja dostosowania opiera się na skuteczności rozwiązywania problemu optymalizacyjnego (promowane są te AEDP, które wyszukują lepsze rozwiązania, albo prowadzą do szybszej ich poprawy).



Rysunek 5: Schemat samoadaptacyjnego algorytmu ewolucyjnego [10]

3. Projekt Samoadaptacyjnego Algorytmu Ewolucyjnego

Zaprojektowanie Samoadaptacyjnego Algorytmu Ewolucyjnego zostało szczegółowo opisane w podrozdziałach i obejmowało:

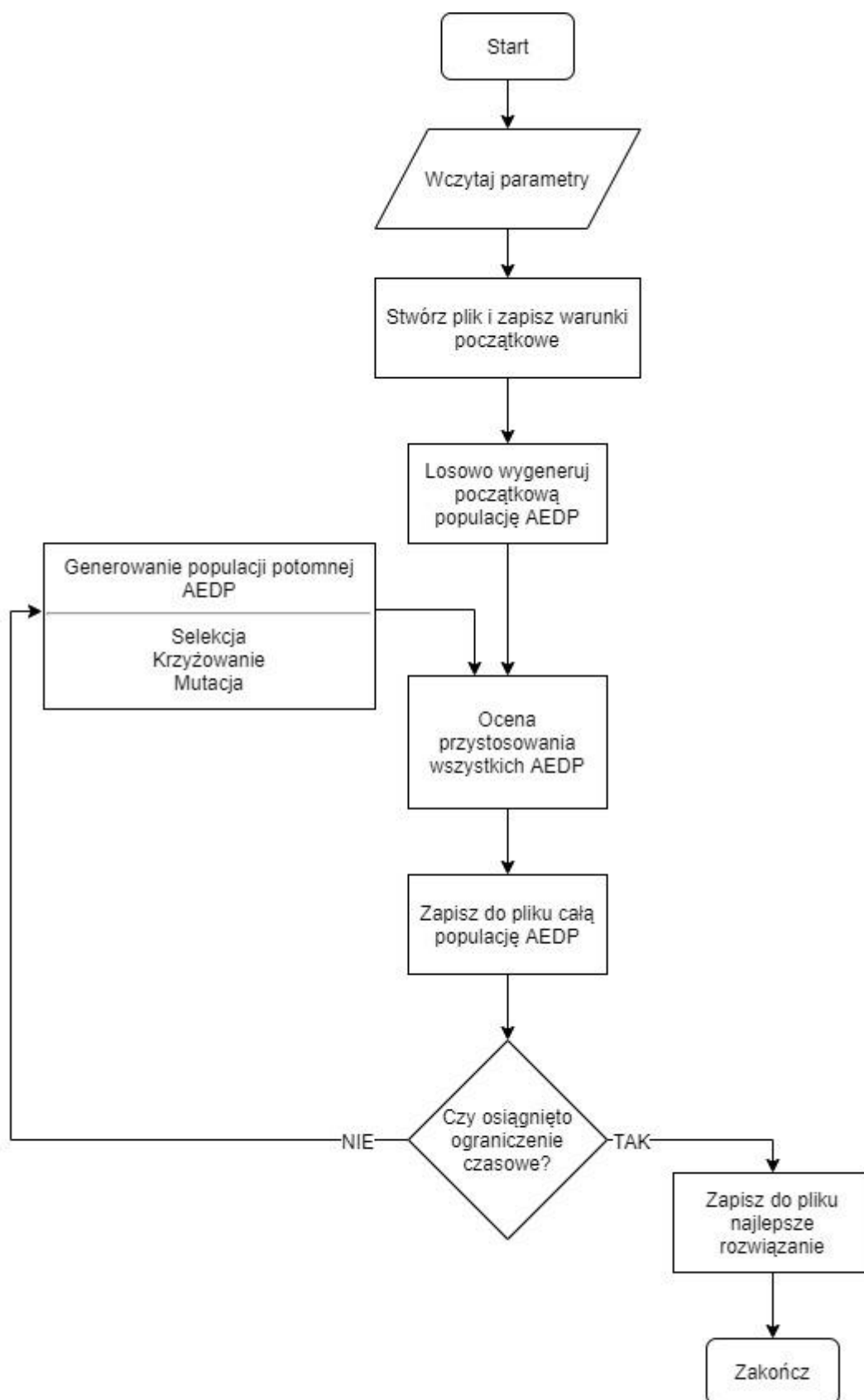
- Stworzeniu modelu Algorytmu Ewolucyjnego Górnego Poziomu
- Stworzenie modelu Algorytmu Ewolucyjnego Dolnego Poziomu

3.1. Projekt Algorytmu Ewolucyjnego Górnego Poziomu

Rysunek 6: Schemat blokowy Algorytmu Ewolucyjnego Górnego Poziomu przedstawia opracowany schemat blokowy Algorytmu Ewolucyjnego Górnego Poziomu, który jako część Samoadaptacyjnego Algorytmu Ewolucyjnego wykonuje następujące zadania w kolejnych krokach algorytmu:

- *Start* – rozpoczęcie SAE
- *Wczytanie parametrów* – polega na pobraniu wszelkich parametrów dotyczących przebiegu optymalizacji oraz warunków zadania i obejmuje:
 - Odebranie parametrów AEGP:
 - Rozmiar populacji AEGP (liczba AEDP)
 - Planowana liczba iteracji procesu optymalizacji
 - Ograniczenie czasowe całego procesu optymalizacji
 - Metoda selekcji stosowana przez AEGP (np. turniejowa)
 - Metoda krzyżowania stosowana przez AEGP (np. adaptacyjna)
 - Metoda mutacji stosowana przez AEGP (np. probabilistyczna)
 - Wartości parametrów towarzyszących (np. prawdopodobieństwo mutacji)
 - Odebranie parametrów AEDP:
 - Dopuszczalny rozmiar populacji AEDP (minimalna i maksymalna wielkość populacji rozwiązań problemu)
 - Zbiór dopuszczalnych funkcji selekcji AEDP (np. turniejowa, rangowa i stochastyczna)
 - Zbiór dopuszczalnych funkcji krzyżowania AEDP (np. jednopunktowe, wielopunktowe i adaptacyjne)
 - Zbiór dopuszczalnych funkcji mutacji AEDP (np. jednopunktowe, wielopunktowe i probabilistyczne)
 - Dopuszczalne wartości parametrów towarzyszących takich jak prawdopodobieństwo mutacji
 - Kryterium oceny AEDP zdefiniowane w postaci funkcji przystosowania
 - Odebranie parametrów zadania:
 - Liczba zmiennych decyzyjnych
 - Typy zmiennych decyzyjnych (np. całkowitoliczbowe, zmiennoprzecinkowe)
 - Ograniczenia nałożone na zmienne decyzyjne
 - Kryterium oceny rozwiązania w postaci funkcji przystosowania

- *Stworzenie pliku do logowania i zapisanie warunków początkowych* – w ramach tego kroku następuje utworzenie pliku tekstowego, do którego zapisywany będzie przebieg optymalizacji z perspektywy AEGP.
- *Wygenerowanie losowej populacji początkowej AEDP* – w tym kroku w sposób losowy zostaje utworzona populacja początkowa AEDP zgodnie z dopuszczalnymi wartościami parametrów, które zostały wczytane na początku procesu optymalizacji.
- *Ocena przystosowania wszystkich AEDP* – następuje ocena przystosowania całej populacji AEDP zgodnie z kryterium zdefiniowanym na początku zadania. Zauważmy, że w celu dokonania oceny przystosowania AEDP konieczne jest przeprowadzenie procesu optymalizacji w ramach AEDP, gdyż oceniana jest efektywność poszukiwania rozwiązań optymalizowanego problemu.
- *Zapisanie do pliku populacji AEDP* – polega na zapisaniu do wcześniej utworzonego pliku kluczowych informacji z przebiegu obecnej iteracji AEGP.
- *Sprawdzenie ograniczenia czasowego* – decyzja sprawdzająca czy osiągnięto warunek stopu w postaci ograniczenia czasowego zadanego na początku procesu optymalizacji.
- *Generowanie populacji potomnej AEDP* – w tym kroku następuje utworzenie populacji potomnej AEDP zgodnie z zasadami Selekcji, Krzyżowania oraz Mutacji zdefiniowanymi na początku procesu optymalizacji.
- *Zapisanie do pliku najlepszego rozwiązania* – polega na wyróżnieniu najlepszego rozwiązania znalezione przez SAE.
- *Zakończenie* – zakończenie całego procesu optymalizacji. [10]

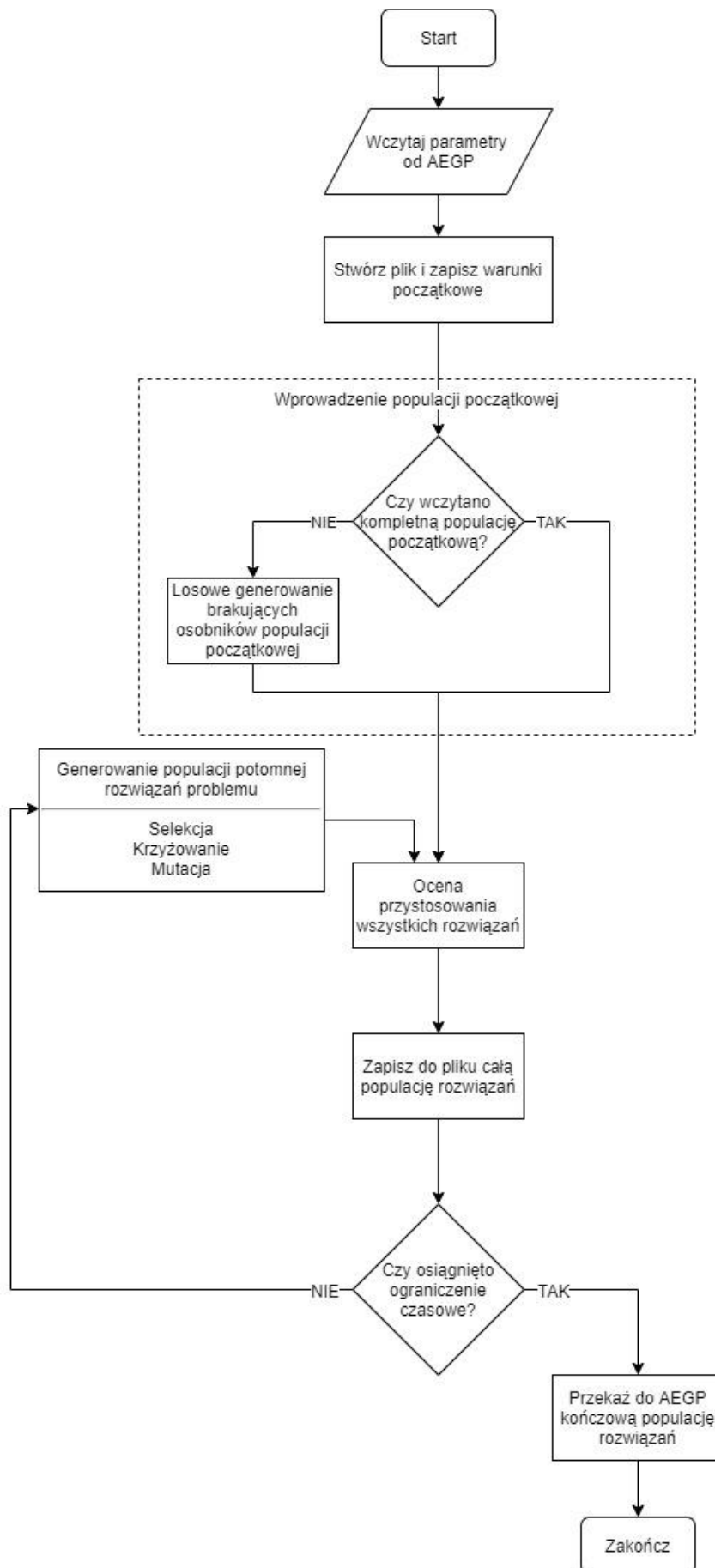


Rysunek 6: Schemat blokowy Algorytmu Ewolucyjnego Górnego Poziomu

3.2. Projekt Algorytmu Ewolucyjnego Dolnego Poziomu

Rysunek 7: Schemat blokowy Algorytmu Ewolucyjnego Dolnego Poziomu przedstawia opracowany schemat blokowy Algorytmu Ewolucyjnego Dolnego Poziomu, który wykonuje następujące kroki:

- *Start* – utworzenie AEDP
- *Wczytanie parametrów od AEGP* – następuje odebranie parametrów AEDP określonych przez AEGP, a w szczególności:
 - Odebranie parametrów AEDP:
 - Rozmiar populacji
 - Metoda Selekcji
 - Metoda Krzyżowania
 - Metoda Mutacji
 - Parametry towarzyszące (np. prawdopodobieństwo mutacji, wzorzec krzyżowania)
 - Odebranie całości lub części populacji początkowej rozwiązań. Może wystąpić jeden z trzech przypadków
 - W pierwszej iteracji otrzymany zbiór jest zawsze pusty, gdyż żadne rozwiązania nie zostały przekazane od rodziców (rodzice nie istnieją).
 - W drugiej lub kolejnej iteracji otrzymany zbiór może być mniejszy od rozmiaru populacji.
 - W drugiej lub kolejnej iteracji otrzymany zbiór może być równy rozmiarowi populacji.
- *Stworzenie pliku i zapisanie warunków początkowych* – w ramach tego kroku następuje utworzenie pliku tekstowego, do którego zapisywany jest przebieg optymalizacji z perspektywy AEDP.
- *Wprowadzenie populacji początkowej* – polega na uzupełnieniu populacji rozwiązań w przypadku, gdy zbiór rozwiązań otrzymany od AEGP jest mniejszy od rozmiaru populacji wewnątrz AEDP. W takim przypadku w sposób losowy generowane są nowe osobniki.
- *Ocenia przystosowania wszystkich rozwiązań* – wykonywana jest ocena przystosowania obecnej populacji rozwiązań.
- *Zapisanie do pliku populacji rozwiązań* – w tym kroku do pliku zapisywane są wszystkie istotne informacje na temat populacji rozwiązań obecnych w tej iteracji AEDP.
- *Sprawdzenie ograniczenia czasowego* – decyzja sprawdzająca czy osiągnięto warunek stopu w postaci ograniczenia czasowego dla AEDP.
- *Generacja populacji potomnej* – w tym kroku następuje utworzenie populacji potomnej rozwiązań problemów zgodnie z technikami selekcji, krzyżowania i mutacji przypisanymi do AEDP.
- *Przekazanie końcowej populacji rozwiązań do AEGP* – przekazanie wszystkich parametrów oraz populacji końcowej do AEGP w celu dokonania oceny tego osobnika AEDP oraz wzięcia udziału w tworzeniu nowej populacji AEDP.
- *Zakończenie* – zakończenie optymalizacji w ramach tego AEDP. [10]



Rysunek 7: Schemat blokowy Algorytmu Ewolucyjnego Dolnego Poziomu

4. Aplikacja komputerowa do celów badawczych

W ramach pracy badawczej została stworzona aplikacja komputerowa pozwalająca na przeprowadzenie optymalizacji przy użyciu wybranego algorytmu optymalizacyjnego. Została ona napisana w języku programowania Python jako paczka (więcej informacji na temat paczek znajduje się na stronie internetowej <https://packaging.python.org/tutorials/packaging-projects/>) oraz opublikowana na stronie internetowej <https://github.com/mdabrowski1990/optimization>, gdzie jest dalej rozwijana jako projekt open-source.

Do uruchomienia aplikacji wymagane są:

- Zainstalowany interpreter języka Python w wersji co najmniej 3.6 (najnowsza wersja może zostać bezpiecznie pobrana ze strony internetowej <https://www.python.org/downloads/>)
- Zainstalowane biblioteki języka Python w wersjach wyspecyfikowanych w pliku „requirements.txt” lub nowszych. Można to wykonać przy użyciu komendy:
pip install -r requirements.txt

Aplikacja składa się z 4 głównych modułów:

- logging – moduł zawierający definicje konfigurowalnych rejestratorów do zapisywania przebiegu procesu optymalizacyjnego.
- optimization_algorithms – moduł zawierający definicje konfigurowalnych algorytmów optymalizacyjnych. W pierwotnej wersji (dołączonej do tej pracy) zawiera on definicje algorytmów ewolucyjnego (EvolutionaryAlgorithm), samoadaptacyjnego algorytmu ewolucyjnego (SEA) oraz algorytmu losowego (RandomAlgorithm).
- optimization_problem – moduł umożliwiający definicje problemów optymalizacyjnych przy użyciu 3 podstawowych typów zmiennych decyzyjnych:
 - IntegerVariable – całkowitoliczbowa zmienna decyzyjna o określonych granicach (wartości minimalnej i maksymalnej).
 - FloatVariable – zmiennoprzecinkowa zmienna decyzyjna o określonych granicach (wartości minimalnej i maksymalnej).
 - ChoiceVariable – zmienna decyzyjna o dowolnym zbiorze dopuszczalnych wartości (definiowana jest przez przekazanie zbioru posiadającego wszystkie możliwe wartości jakie mogą zostać przyjęte przez tę zmienną decyzyjną).
- utilities – dodatkowy moduł zawierający funkcje współdzielone przez różne elementy aplikacji takie jak generatory liczb (pseudo)losowych.

5. Eksperyment badawczy

W tej części pracy przedstawiony zostanie eksperyment badawczy, który miał na celu ocenę Samoadaptacyjnego Algorytmu Ewolucyjnego i porównania jego skuteczności na tle klasycznych Algorytmów Ewolucyjnych i Algorytmu Losowego. W tym celu został użyty program komputerowy w języku programowania Python, która został opisany w poprzednim rozdziale tej pracy. Dodatkowo zostały przygotowane 3 skrypty („experiment_1.py”, „experiment_2.py” oraz „experiment_3.py”), których wykonanie pozwala na przeprowadzenie kolejnych eksperymentów porównawczych. Wszelkie modele, konfiguracje, założenia, przebieg eksperymentu i jego wyniki oraz ich analiza zostały przedstawione w kolejnych podrozdziałach.

5.1. Problemy optymalizacyjne

W eksperymencie zostały wykorzystane 3 problemy optymalizacyjne. Dwa z problemów są to dwie z pięciu funkcji zaproponowanych przez De Jonga w 1975 do badania algorytmów genetycznych [11]. Trzeci problem jest dyskretnym problemem polegającym na optymalizacji objętości sprężyny naciskowej będącym typowym zadaniem technicznym.

5.1.1. Problem nr 1 – funkcja De Jonga F1

Funkcja De Jonga F1 jest to pierwsza z funkcji zaproponowanych przez De Jonga do badania algorytmów genetycznych. Opis problemu:

- Zmienne decyzyjne:
 - x_1, x_2, x_3 – ciągłe (zmiennoprzecinkowe) zmienne decyzyjne zdefiniowana w dziedzinie:

$$-5.12 \leq x_i \leq 5.12, \quad \text{dla } i \in \{1, 2, 3\}$$

- Wektor zmiennych decyzyjnych:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

- Funkcja celu wyrażona jest przy pomocy wzoru:

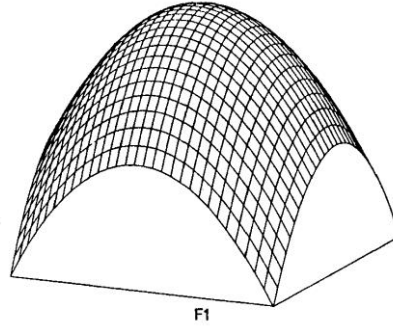
$$f(X) = \sum_{i=1}^3 x_i^2$$

- Typ optymalizacji: poszukiwanie wartości minimalnej funkcji celu.
- Optymalne rozwiązanie:

$$X_{OPT} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$f(X_{OPT}) = 0$$

- Graficzne prezentacja funkcji (*Rysunek 8: Graficzna prezentacja funkcji De Jonga F1*):



Rysunek 8: Graficzna prezentacja funkcji De Jonga F1 [11]

5.1.2. Problem nr 2 – funkcja De Jonga F3

Funkcja De Jonga F3 jest to trzecia z funkcji zaproponowanych przez De Jonga do badania algorytmów genetycznych. Opis problemu:

- Zmienne decyzyjne:
 - x_1, x_2, x_3, x_4, x_5 – ciągłe (zmiennoprzecinkowe) zmienne decyzyjne zdefiniowana w dziedzinie:

$$-5.12 \leq x_i \leq 5.12, \quad \text{dla } i \in \{1, 2, 3, 4, 5\}$$

- Wektor zmiennych decyzyjnych:

$$X = \begin{bmatrix} x_1 \\ \vdots \\ x_5 \end{bmatrix}$$

- Funkcja celu wyrażona jest przy pomocy wzoru:

$$f(X) = \sum_{i=1}^5 [x_i]$$

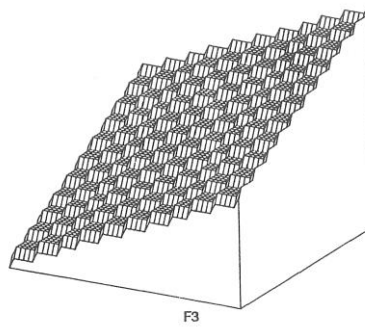
- Typ optymalizacji: poszukiwanie wartości minimalnej funkcji celu.
- Optymalne rozwiązanie:

$$-5.12 \leq x_{iOPT} < -5, \quad \text{dla } i \in \{1, 2, 3, 4, 5\}$$

$$X_{OPT} = \begin{bmatrix} x_{1OPT} \\ \vdots \\ x_{5OPT} \end{bmatrix}$$

$$f(X_{OPT}) = -30$$

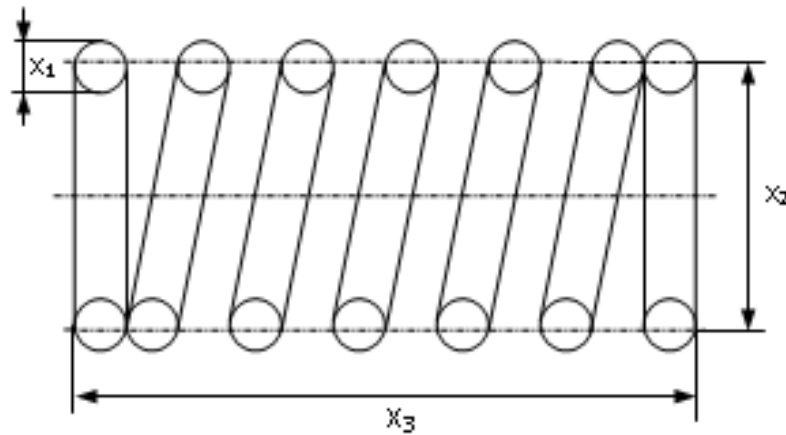
- Graficzne prezentacja funkcji (Rysunek 9: Graficzna prezentacja funkcji De Jonga F3):



Rysunek 9: Graficzna prezentacja funkcji De Jonga F3 [11]

5.1.3. Problem nr 3 – sprężyna naciskowa

Problem ten polega na minimalizacji objętości sprężyny naciskowej przy uwzględnieniu wymagań geometrycznych i wytrzymałościowych. *Rysunek 10* przedstawia poglądowy schemat sprężyny, której wymiary są optymalizowane.



Rysunek 10: Schemat sprężyny naciskowej

Opis problemu:

- Zmienne decyzyjne:

- x_1 – średnica drutu sprężyny [mm]

$$x_1 \in \{4, 4.5, 5, \dots, 12\}$$

- x_2 – średnica nawinięcia sprężyny [mm]

$$x_2 \in \{30, 31, 32, \dots, 60\}$$

- x_3 – wysokość sprężyny [mm]

$$x_3 \in \{100, 102, 104, \dots, 300\}$$

- x_4 – ilość zwojów czynnych

$$x_4 \in \{5, 5.5, 6, \dots, 14\}$$

- Dane wejściowe:

- Siła ściskająca:

$$P = 1850 \text{ [N]}$$

- Współczynnik sprężystości poprzecznej:

$$G = 81400 \left[\frac{\text{N}}{\text{mm}^2} \right]$$

- Ugięcie sprężyny:

$$\delta = 90 [mm]$$

- Maksymalna średnica zewnętrzna sprężyny:

$$D_{max} = 60 [mm]$$

- Dopuszczalny zakres zmiany sztywności:

$$\Delta c = 0.1$$

- Dopuszczalne naprężenie skręcające:

$$\tau_{dop} = 605 \left[\frac{N}{mm^2} \right]$$

- Współczynnik dopuszczalnego przekroczenia naprężeń skręcających wg. PN-85/M.-80701-3:

$$S_f = 1.12$$

- Współczynnik poprawkowy według PN-85/M.-80701-3:

$$k = 1 + \frac{5}{4 \cdot w} + \frac{7}{8 \cdot w^2} + \frac{1}{w^3}, \quad \text{gdzie } w = \frac{x_2}{x_1}$$

- Współczynnik prześwitu aproksymowany z wykresu PN-85/M.-80701-3:

$$\alpha = -0.00002 \cdot w^3 + 0.002 \cdot w^2 + 0.0018 \cdot w + 0.0627$$

- Dopuszczalny wskaźnik sprężystości sprężyny aproksymowany z wykresu PN-85/M.-80701-3:

$$\eta_{dop} = -0.0122 \cdot \lambda^6 + 0.2562 \cdot \lambda^5 - 1.9775 \cdot \lambda^4 + 6.9448 \cdot \lambda^3 - 12.807 \cdot \lambda^2 + 9.8974 \cdot \lambda + 52.444$$

gdzie:

$$\lambda = \frac{x_3}{x_2}$$

- Wektor zmiennych decyzyjnych:

$$X = \begin{bmatrix} x_1 \\ \vdots \\ x_4 \end{bmatrix}$$

- Funkcja celu to objętość sprężyny, która wyrażona jest wzorem:

$$f(X) = \frac{\pi^2}{2} x_1^2 \sqrt{x_2^2 \cdot x_4^2 + x_3^2} + \frac{\pi}{4} x_1^2 \cdot x_2$$

- Typ optymalizacji: poszukiwanie wartości minimalnej funkcji celu (objętości sprężyny).

- Ograniczenia:

- Warunek na naprężenia skręcające:

$$g_1(X) = S_f \cdot \tau_{dop} - \frac{8 \cdot k \cdot P \cdot x_2}{\pi \cdot x_1^3} \geq 0$$

- Warunek na sztywność sprężyny:

$$g_2(X) = \Delta c \cdot \frac{P}{\delta} - \left| \frac{P}{\delta} - \frac{G \cdot x_1^4}{8 \cdot x_2^3 \cdot x_4} \right| \geq 0$$

- Warunek na prześwit międzyzwojowy:

$$g_3(X) = x_3 - \frac{8 \cdot P \cdot x_2^3 \cdot x_4}{G \cdot x_1^4} - x_1 \cdot x_4 \cdot (1 + \alpha) \geq 0$$

- Warunek na wyboczenie:

$$g_4(X) = \eta_{dop} - \frac{800 \cdot P \cdot x_2^3 \cdot x_4}{G \cdot x_1^4 \cdot x_3} \geq 0$$

- Warunki geometryczne:

$$g_5(X) = x_2 - x_1 \geq 0$$

$$g_6(X) = 7 \cdot x_2 - x_3 \geq 0$$

- Współczynnik kary: 10^{10}

5.2. Przebieg eksperymentu

Eksperyment polegał na przeprowadzeniu procesu optymalizacji dla trzech problemów (dwie funkcje De Jonga i sprężyna naciskowa) przedstawionych w poprzednim podrozdziale (5.1) z użyciem algorytmu losowego, klasycznego algorytmu ewolucyjnego i samoadaptacyjnego algorytmu ewolucyjnego. Jako, że konfiguracje algorytmu ewolucyjnego jak i samoadaptacyjnego algorytmu ewolucyjnego mogą być zrealizowana na wiele różnych sposobów, w ramach eksperymentu zostały zaprojektowane po trzy konfiguracje dla obu typów tych algorytmów. W celu lepszego porównywania AE z SAE konfiguracje tych algorytmów są możliwie identyczne. Konfiguracja SAE dla problemu nr 3 nieco różni się od poprzednich dwóch ze względu na rozmiar problemu nr 3 (jest on dużo mniejszy od poprzednich dwóch).

- Konfiguracje AE:

- AE 1:

- Rozmiar populacji: 20
- Typ selekcji: Stochastyczna
- Typ krzyżowania: Jednopunktowe
- Typ mutacji: Probabilistyczna
- Współczynnik mutacji: 5%
- Elitaryzm: Tak

- AE 2:

- Rozmiar populacji: 40
- Typ selekcji: Rankingowa ze współczynnikiem 1.5
- Typ krzyżowania: Jednostajne
- Typ mutacji: Wielopunktowa (2 punkty mutacji)
- Współczynnik mutacji: 1%
- Elitaryzm: Nie

- AE 3:

- Rozmiar populacji: 60
- Typ selekcji: Proporcjonalna ze współczynnikiem 10
- Typ krzyżowania: Wielopunktowe (2 punkty krzyżowania)
- Typ mutacji: Jednopunktowa
- Współczynnik mutacji: 2%
- Elitaryzm: Nie

- Konfiguracje SAE dla eksperymentów nr 1 i 2:

- Konfiguracja AEDP (wspólna dla SAE1, SAE2 oraz SAE3):
 - Rozmiar populacji: od 10 do 100
 - Współczynnik mutacji: od 1% do 20%
 - Dopuszczalne wartości elitaryzmu: „Tak” oraz „Nie”
 - Dopuszczalne funkcje selekcji:
 - Stochastyczna
 - Turniejowa
 - Rozmiar grupy: od 2 do 6
 - Proporcjonalna
 - Współczynnik skalowania: od 1. do 100.
 - Rankingowa
 - Współczynnik presji: od 1. do 2.
 - Dopuszczalne funkcje krzyżowania:
 - Jednopunktowe
 - Wielopunktowe
 - Liczba punktów krzyżowania: od 1 do max. możliwej
 - Jednostajne
 - Adaptacyjne
 - Wzorzec krzyżowania, który również ulega ewolucji.
 - Dopuszczalne funkcje mutacji:
 - Jednopunktowa
 - Wielopunktowa
 - Liczba punktów mutacji: od 1 do max. możliwej
 - Probabilistyczna
- Konfiguracja AEGP dla SAE1:
 - Rozmiar populacji: 20
 - Max liczba iteracji: 20
 - Typ selekcji: Stochastyczna
 - Typ krzyżowania: Jednopunktowe
 - Typ mutacji: Probabilistyczna
 - Współczynnik mutacji: 5%
 - Ocena AEDP: Najlepsze 3 rozwiązania
- Konfiguracja AEGP dla SAE2:
 - Rozmiar populacji: 20
 - Max liczba iteracji: 20
 - Typ selekcji: Rankingowa ze współczynnikiem 1.5
 - Typ krzyżowania: Jednostajne
 - Typ mutacji: Wielopunktowa (2 punkty mutacji)
 - Współczynnik mutacji: 1%
 - Ocena AEDP: Najlepsze 10% rozwiązań
- Konfiguracja AEGP dla SAE3:
 - Rozmiar populacji: 20
 - Max liczba iteracji: 20

- Typ selekcji: Proporcjonalna ze współczynnikiem 10
 - Typ krzyżowania: Wielopunktowe (2 punkty krzyżowania)
 - Typ mutacji: Jednopunktowa
 - Współczynnik mutacji: 2%
 - Ocena AEDP: Najlepsze 10% rozwiązań
- Konfiguracje SAE dla eksperymentu nr 3:
 - Konfiguracja AEDP (wspólna dla SAE1, SAE2 oraz SAE3):
 - Rozmiar populacji: od 10 do 100
 - Współczynnik mutacji: od 1% do 20%
 - Dopuszczalne wartości elitaryzmu: „Tak” oraz „Nie”
 - Dopuszczalne funkcje selekcji:
 - Stochastyczna
 - Turniejowa
 - Rozmiar grupy: od 2 do 6
 - Proporcjonalna
 - Współczynnik skalowania: od 1. do 100.
 - Rankingowa
 - Współczynnik presji: od 1. do 2.
 - Dopuszczalne funkcje krzyżowania:
 - Jednopunktowe
 - Wielopunktowe
 - Liczba punktów krzyżowania: od 1 do max. możliwej
 - Jednostajne
 - Adaptacyjne
 - Wzorzec krzyżowania, który również ulega ewolucji.
 - Dopuszczalne funkcje mutacji:
 - Jednopunktowa
 - Wielopunktowa
 - Liczba punktów mutacji: od 1 do max. możliwej
 - Probabilistyczna
 - Konfiguracja AEGP dla SAE1:
 - Rozmiar populacji: **10**
 - Max liczba iteracji: **10**
 - Typ selekcji: Stochastyczna
 - Typ krzyżowania: Jednopunktowe
 - Typ mutacji: Probabilistyczna
 - Współczynnik mutacji: 5%
 - Ocena AEDP: Najlepsze 3 rozwiązania
 - Konfiguracja AEGP dla SAE2:
 - Rozmiar populacji: **10**
 - Max liczba iteracji: **10**
 - Typ selekcji: Rankingowa ze współczynnikiem 1.5

- Typ krzyżowania: Jednostajne
- Typ mutacji: Wielopunktowa (2 punkty mutacji)
- Współczynnik mutacji: 1%
- Ocena AEDP: Najlepsze 10% rozwiązań
- Konfiguracja AEGP dla SAE3:
 - Rozmiar populacji: **10**
 - Max liczba iteracji: **10**
 - Typ selekcji: Proporcjonalna ze współczynnikiem 10
 - Typ krzyżowania: Wielopunktowe (2 punkty krzyżowania)
 - Typ mutacji: Jednopunktowa
 - Współczynnik mutacji: 2%
 - Ocena AEDP: Najlepsze 10% rozwiązań

Następnie konieczne było określenie warunków zakończenia procesu optymalizacji wspólnego dla każdego z algorytmów:

- Warunek stopu dla Problemu nr 1:
 - Ograniczenie czasowe: 60s
- Warunek stopu dla Problemu nr 2:
 - Znalezienie rozwiązania optymalnego: -30 (wartość funkcji celu)
 - Ograniczenie czasowe: 300s
- Warunek stopu dla Problemy nr 3:
 - Ograniczenie czasowe: 20s

Ostatnim etapem było przygotowanie skryptów („experiment_1.py”, „experiment_2.py” oraz „experiment_3.py”) z uwzględnieniem wszystkich wcześniej wymienionych ustawień i założeń oraz przeprowadzenie procesu optymalizacji dla każdego (wcześniej skonfigurowanego) algorytmu optymalizacyjnego w możliwie identycznych warunkach. Otrzymane wyniki i wnioski przedstawione są w kolejnych częściach pracy.

5.3. Wyniki eksperymentu

Otrzymane wyniki kolejnych eksperymentów zostały dołączone do pracy w formie elektronicznej i umieszczone w folderach o nazwach:

- „Eksperyment 1 - funkcja De Jonga F1”
- „Eksperyment 2 - funkcja De Jonga F3”
- „Eksperyment 3 - sprężyna naciskowa”

Wyniki każdego z eksperymentów będą analizowane w kolejnych podrozdziałach tej pracy pod następującymi względami:

- Porównanie najlepszego odnalezionego rozwiązania przez każdy ze skonfigurowanych algorytmów optymalizacyjnych.
- Porównanie postępu procesu optymalizacji dla każdego ze skonfigurowanych algorytmów optymalizacyjnych.
- Analiza zmienności nastaw AEDP w kolejnych iteracjach każdego z SAE.

5.3.1. Analiza wyników eksperymentu nr 1

Eksperyment nr 1 polegał na optymalizacji funkcji De Jonga F1 (szczegółowe informacje na temat modelu tego problemu optymalizacyjnego znajdują się w rozdziale 5.1.1) z ograniczeniem czasowym wynoszącym 60 sekund. Optymalizacja wykonana była przy użyciu algorytmu losowego, trzech klasycznych algorytmów ewolucyjnych i trzech samoadaptacyjnych algorytmów ewolucyjnych (szczegóły konfiguracji algorytmów znajdują się w rozdziale 5.2). Otrzymane wyniki, które są analizowane w tym rozdziale znajdują się w folderze „Eksperyment 1 - funkcja De Jonga F1”, który został dołączony (w formie elektronicznej) do tej pracy. Aby odtworzyć warunki eksperymentu należy uruchomić skrypt „experiment_1.py” przy użyciu interpretera języka Python (szczegółowe informacje na temat konfiguracji aplikacji można znaleźć w rozdziale 4).

5.3.1.1. Analiza najlepszych rozwiązań

Tabela 1 zawiera porównanie najlepszych rozwiązań odnalezionych przez algorytmy optymalizacyjne, które były częścią eksperymentu nr 1. W kolumnie „Ranking” została umieszczona liczba porządkowa sortująca badane algorytmy według jakości najlepszych rozwiązań odnalezionych przez te algorytmy.

Tabela 1: Porównanie najlepszych rozwiązań otrzymanych w eksperymencie nr 1

Algorytm Optymalizacyjny	Najlepsze znalezione rozwiązanie	Wartość funkcji celu najlepszego rozwiązania	Ranking
Algorytm Losowy	$x_1 \approx 0.0264123256$ $x_2 \approx -0.0047953586$ $x_3 \approx -0.0104181029$	~ 0.0008291433	5
SAE1	$x_1 \approx 0.0007537832$ $x_2 \approx -0.0007922868$ $x_3 \approx -0.0003762829$	~ 0.0000013375	1
SAE2	$x_1 \approx 0.0047213786$ $x_2 \approx 0.0028585184$ $x_3 \approx 0.0049803331$	~ 0.0000552663	4
SAE3	$x_1 \approx 0.0011868117$ $x_2 \approx -0.0006206727$ $x_3 \approx -0.0014549595$	~ 0.0000039107	2
AE1	$x_1 \approx -0.0001954616$ $x_2 \approx 0.0025317983$ $x_3 \approx -0.0023157406$	~ 0.0000118109	3
AE2	$x_1 \approx 0.0148121493$ $x_2 \approx -0.0325604128$ $x_3 \approx -0.1357641157$	~ 0.0197114754	7
AE3	$x_1 \approx 0.0216264446$ $x_2 \approx 0.0553694411$	~ 0.0035450134	6

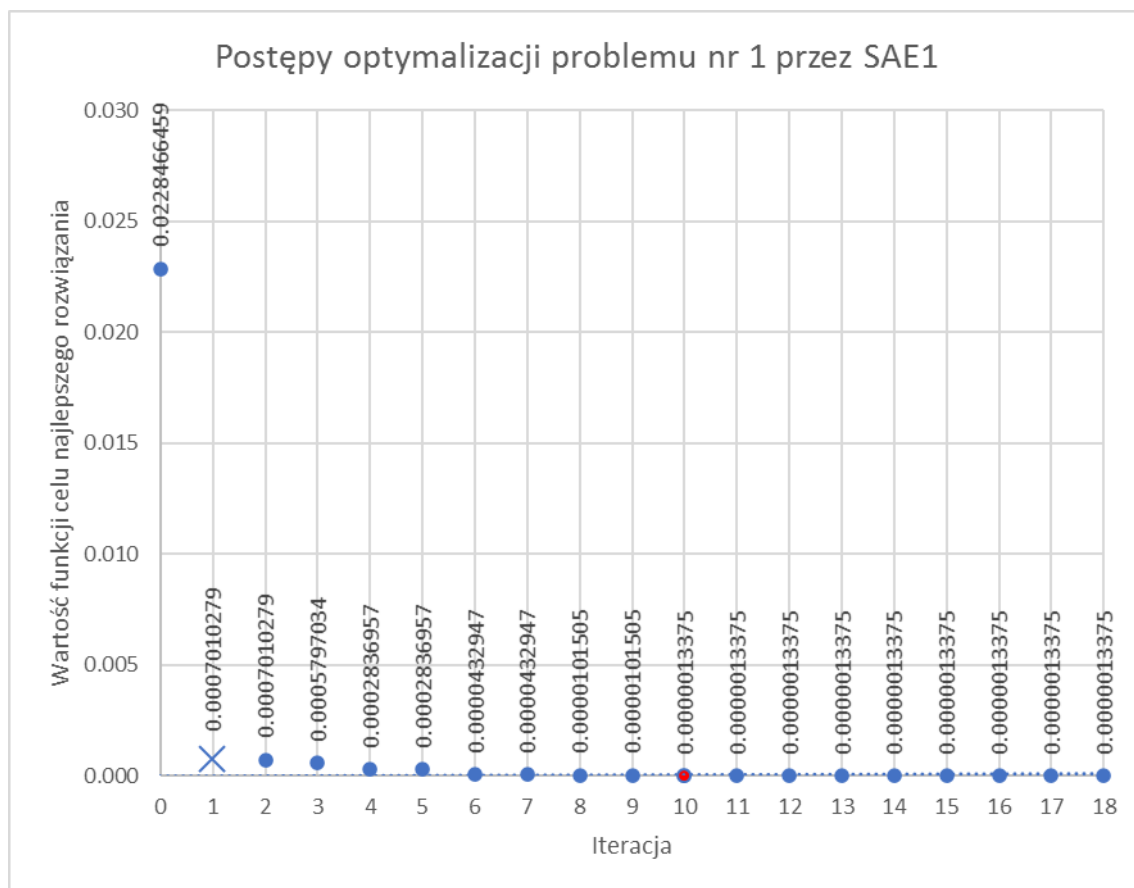
	$x_3 \approx 0.0033963672$		
--	----------------------------	--	--

Wyniki przedstawione w tabeli powyżej pokazują, iż SAE okazały się najskuteczniejsze w poszukiwaniu najlepszych rozwiązań podczas tego eksperymenty i znalezione przez nie rozwiązania zajęły kolejno 1, 4 i 2 miejsca w rankingu.

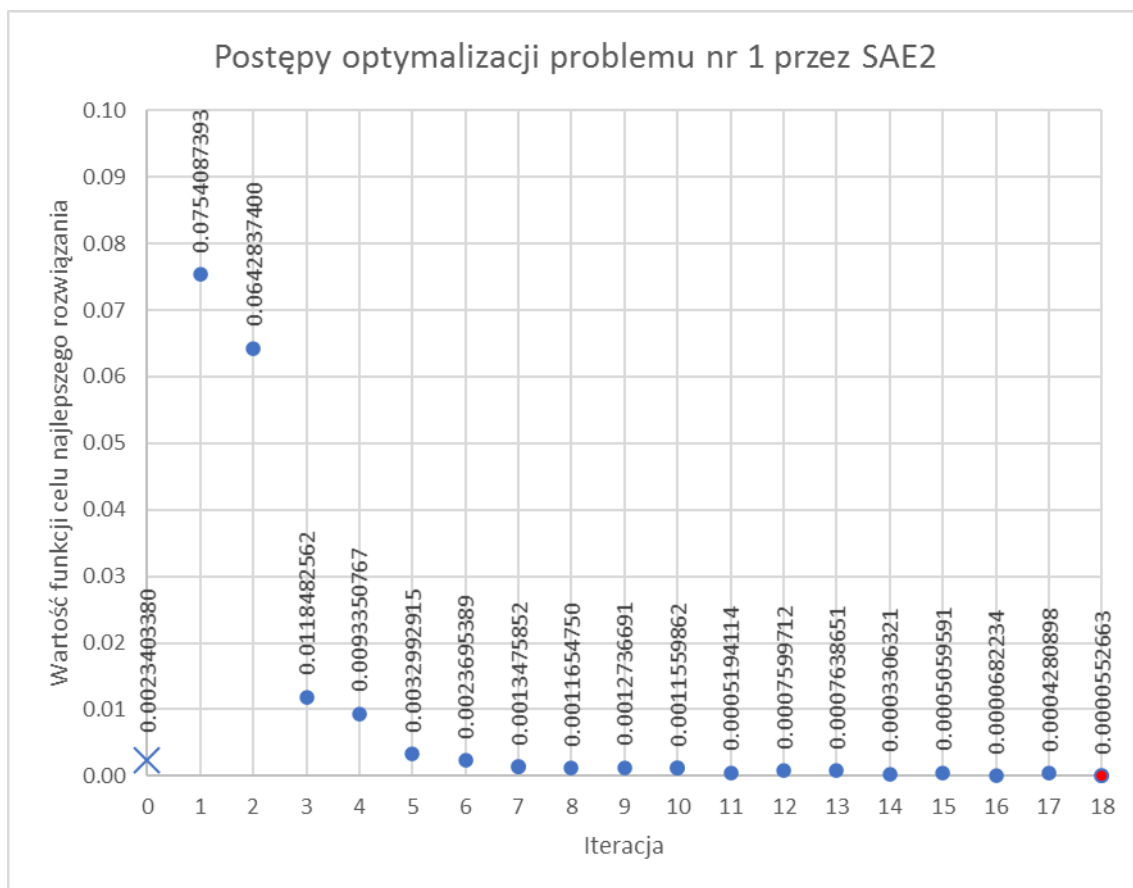
Zauważmy również, iż Algorytm Losowy okazał się skuteczniejszy od dwóch algorytmów ewolucyjnych (AE2 i AE3), co może sugerować iż konfiguracje zaproponowane dla tych algorytmów okazały się nieskuteczne dla tego typu problemu.

5.3.1.2. Analiza postępu poszukiwań

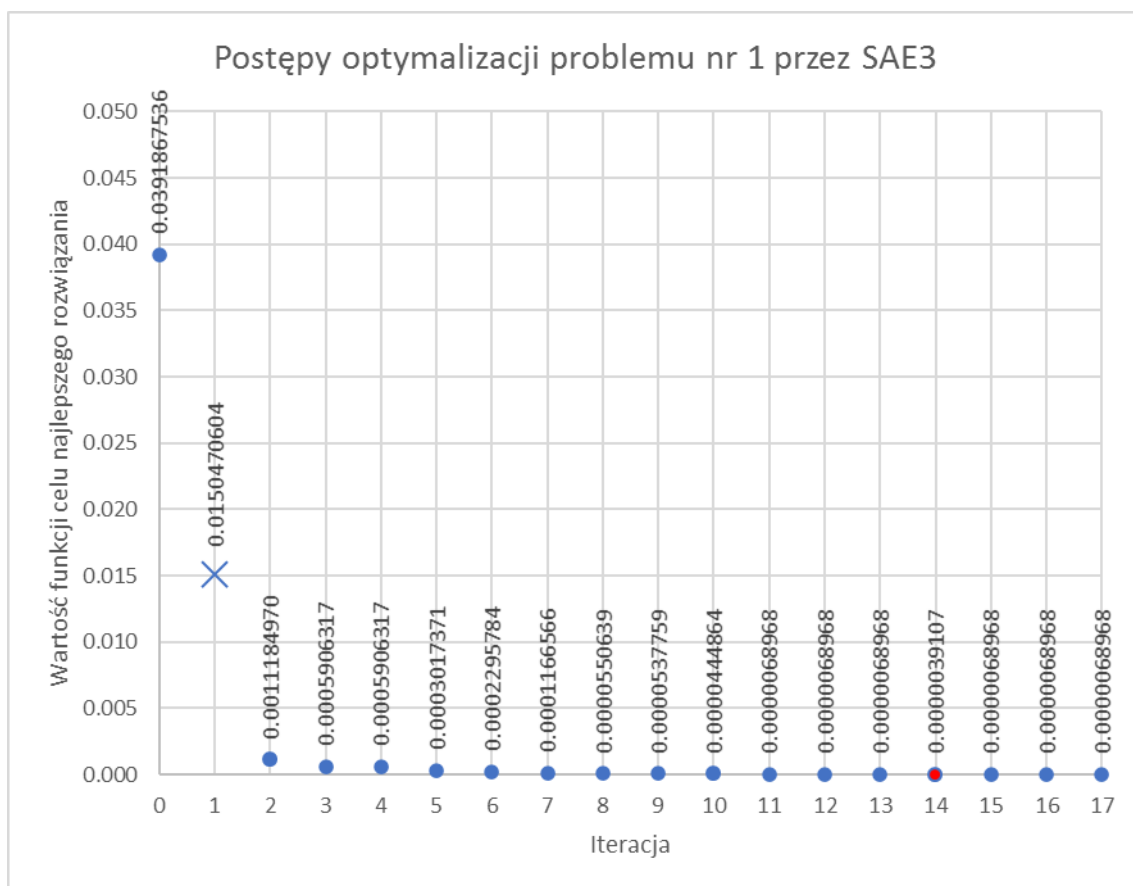
Na wykresach *Rysunek 11*, *Rysunek 12*, *Rysunek 13*, *Rysunek 14*, *Rysunek 15*, *Rysunek 16* oraz *Rysunek 17* przedstawione są postępy poszukiwaniach rozwiązań problemu nr 1 przez kolejne algorytmy optymalizacyjne biorące udział w eksperymencie badawczym. Ze względu na znacząco odbiegającą liczbę iteracji wykonanych przez różne algorytmy optymalizacyjne (dla SAE od 18 do 19, dla AE od 2034 do 6308, dla Algorytmu Losowego 1355) wynikającą z różnej złożoności obliczeniowej różnych algorytmów oraz różnych rozmiarów populacji, wszystkie wykresy zostały podzielone na około 20 etapów o identycznej długości pod względem czasu. Ze względu na różny rząd wielkości wartości funkcji celu rozwiązań odnalezionych przez różne algorytmy optymalizacyjne, na każdym wykresie została zastosowana inna skala osi Y. Aby ułatwić porównanie, różnych wykresów, na każdym wykresie znajduje się punkt w kształcie litery „x” określający iterację, w której po raz pierwszy zostało odnalezione rozwiązanie o wartości funkcji celu niższej od poziomu referencyjnego wynoszącego 0,02. Najlepsze rozwiązanie odnalezione przez dany algorytm optymalizacyjny oznaczone jest czerwonym punktem na danym wykresie.



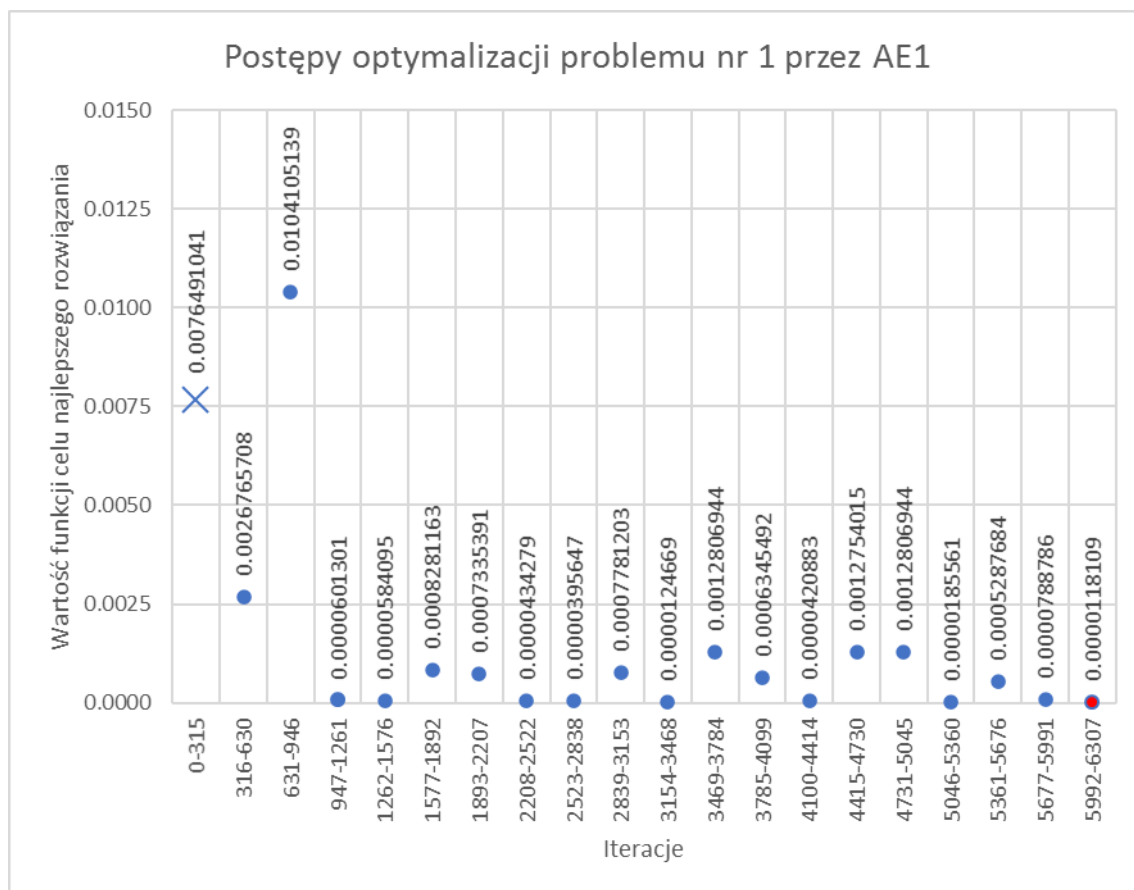
Rysunek 11: Postępy optymalizacji problemu nr 1 przez SAE1



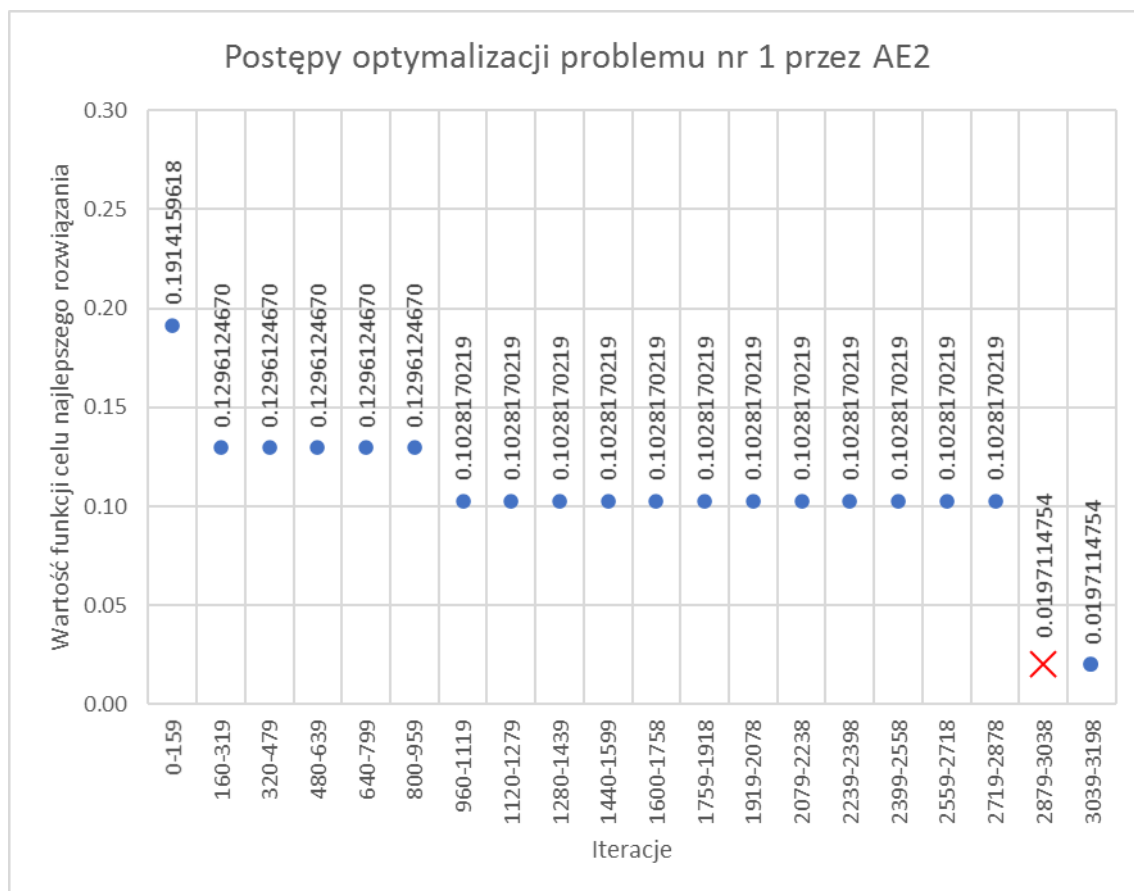
Rysunek 12: Postępy optymalizacji problemu nr 1 przez SAE2



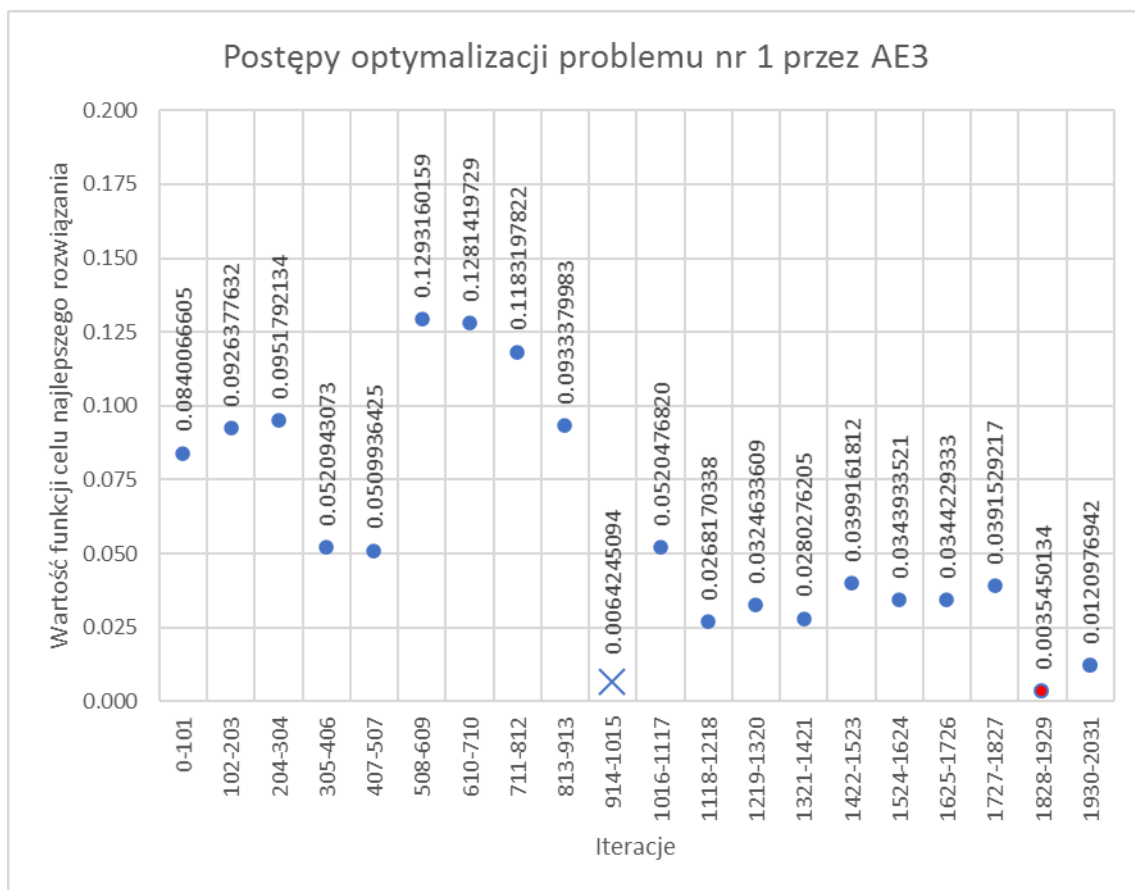
Rysunek 13: Postępy optymalizacji problemu nr 1 przez SAE3



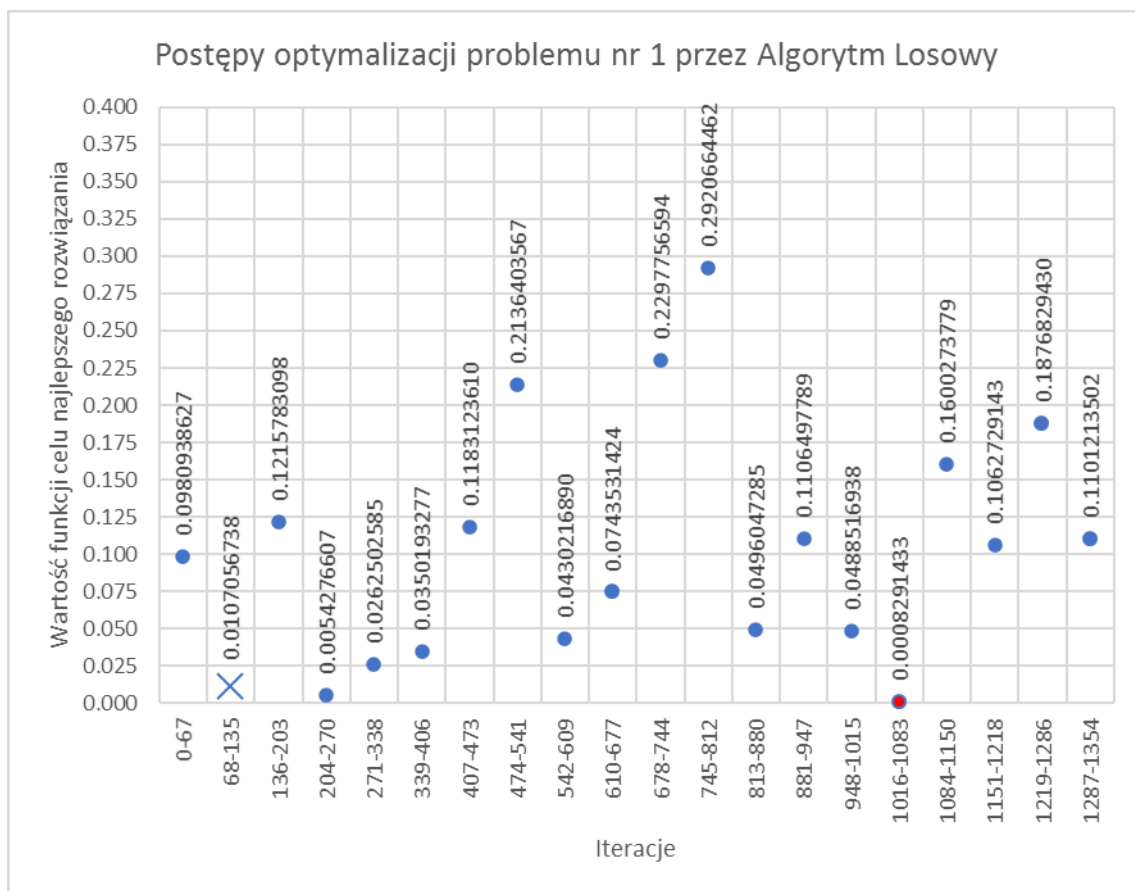
Rysunek 14: Postępy optymalizacji problemu nr 1 przez AE1



Rysunek 15: Postępy optymalizacji problemu nr 1 przez AE2



Rysunek 16: Postępy optymalizacji problemu nr 1 przez AE3



Rysunek 17: Postępy optymalizacji problemu nr 1 przez Algorytm Losowy

Tabela 2 zawiera porównanie etapów optymalizacji osiągnięcia wartości referencyjnej oraz etapu, w którym zostało odnalezione najlepsze rozwiązanie przez kolejne algorytmy optymalizacyjne biorące udział w eksperymencie badawczym.

Tabela 2: Porównanie etapów optymalizacji problemu nr 1

Algorytm Optymalizacyjny	Etap, w którym osiągnięto wartość referencyjną	Etap, w którym odnaleziono najlepsze rozwiązanie
Algorytm Losowy	2 z 20	16 z 20
SAE1	2 z 19	11 z 19
SAE2	1 z 19	19 z 19
SAE3	2 z 18	15 z 18
AE1	1 z 20	20 z 20
AE2	19 z 20	19 z 20
AE3	10 z 20	19 z 20

Na wykresach Rysunek 11, Rysunek 12 oraz Rysunek 13 przedstawiających optymalizację dokonywaną przez SAE można zauważyć iż kilka pierwszych (od 1 do 3) iteracji SAE przebiegało dynamicznie z dużą różnicą między wartościami najlepszych rozwiązań odnalezionymi w tych iteracjach. Zauważmy, że każdy z SAE w tym czasie odnalazł rozwiązanie lepsze od wartości referencyjnej. Po tej początkowej fazie, we wszystkich SAE następował etap stabilizacji i konsekwentnej, lecz powolnej poprawy w poszukiwaniu kolejnych rozwiązań. Wartym odnotowania jest również fakt, iż SAE odnajdywało najlepsze rozwiązania w późnych etapach optymalizacji co może świadczyć iż algorytmy te mają potencjał do ciągłego odnajdywania coraz to lepszych rozwiązań.

Dla porównania optymalizacja w wykonaniu klasycznych AE charakteryzowała się konsekwentną, acz powolną poprawą odnajdywanych rozwiązań (patrz Rysunek 15), bądź bardziej burzliwą (o większej wariancji), acz skuteczniejszą optymalizacją (patrz Rysunek 14 oraz Rysunek 16), która bardziej przypomina turbulentną optymalizację w wykonaniu algorytmu losowego (patrz Rysunek 17).

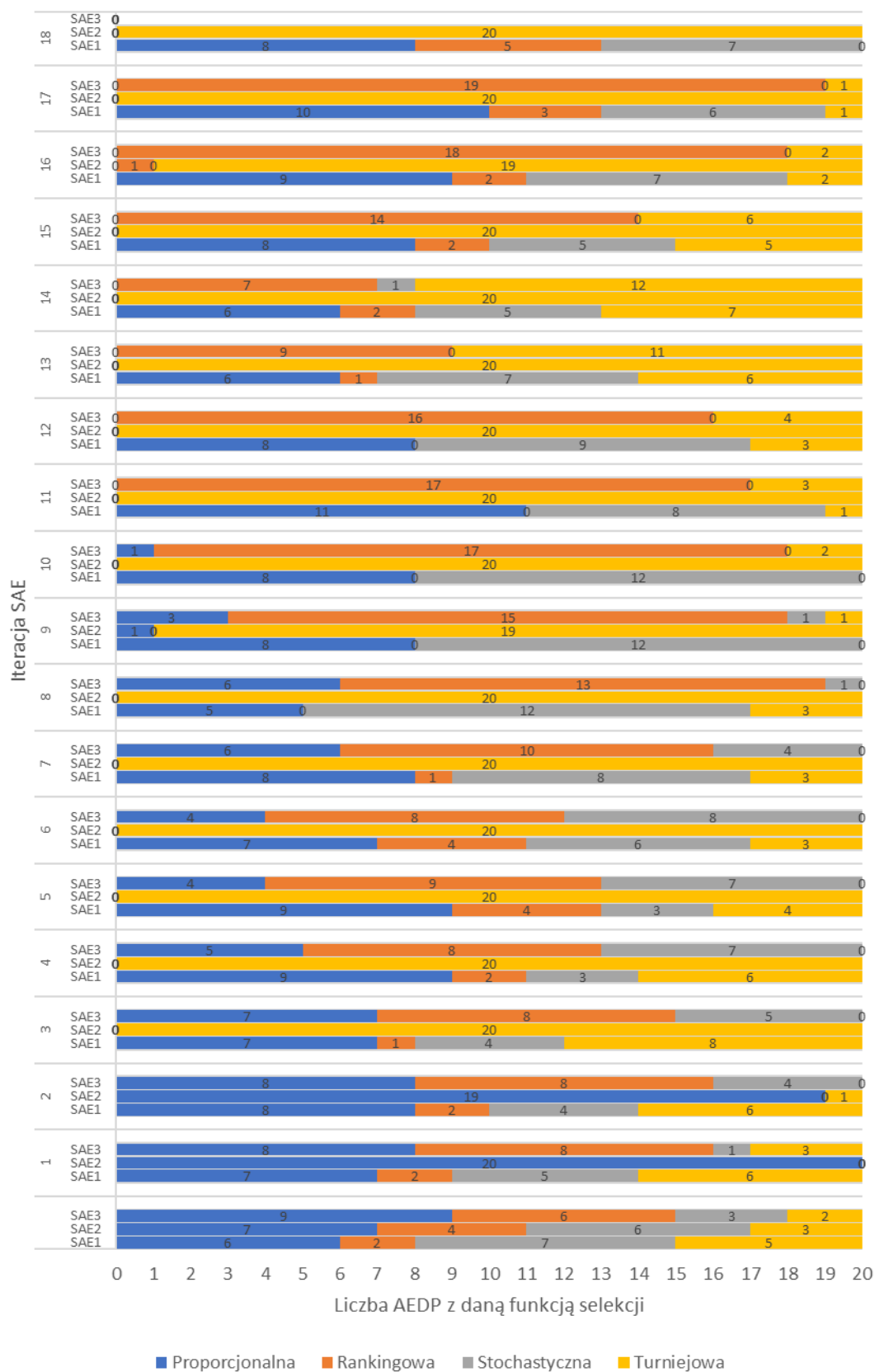
Biorąc pod uwagę charakterystykę SAE oraz dane przedstawione na wykresach powyżej można podejrzewać, iż skuteczność SAE dla tego problemu mogła wynikać z osiągnięcia równowagi między turbulentnym i losowym przeszukiwaniem przestrzeni rozwiązań w poszukiwaniu nowych „silnych genów”, a konsekwentnym procesem wymiany genów między najsilniejszymi osobnikami w celu odnalezienia potomstwa lepiej przystosowanego od swoich poprzedników.

5.3.1.3. Analiza optymalizacji nastaw AEDP w SAE

W tej części pracy zostanie porównana ewolucja konfiguracji AEDP w kolejnych iteracjach SAE podczas procesu optymalizacji problemu nr 1, a w szczególności:

- zmiany dystrybucji funkcji selekcji w populacji AEDP
- zmiany dystrybucji funkcji krzyżowania w populacji AEDP
- zmiany dystrybucji funkcji mutacji w populacji AEDP

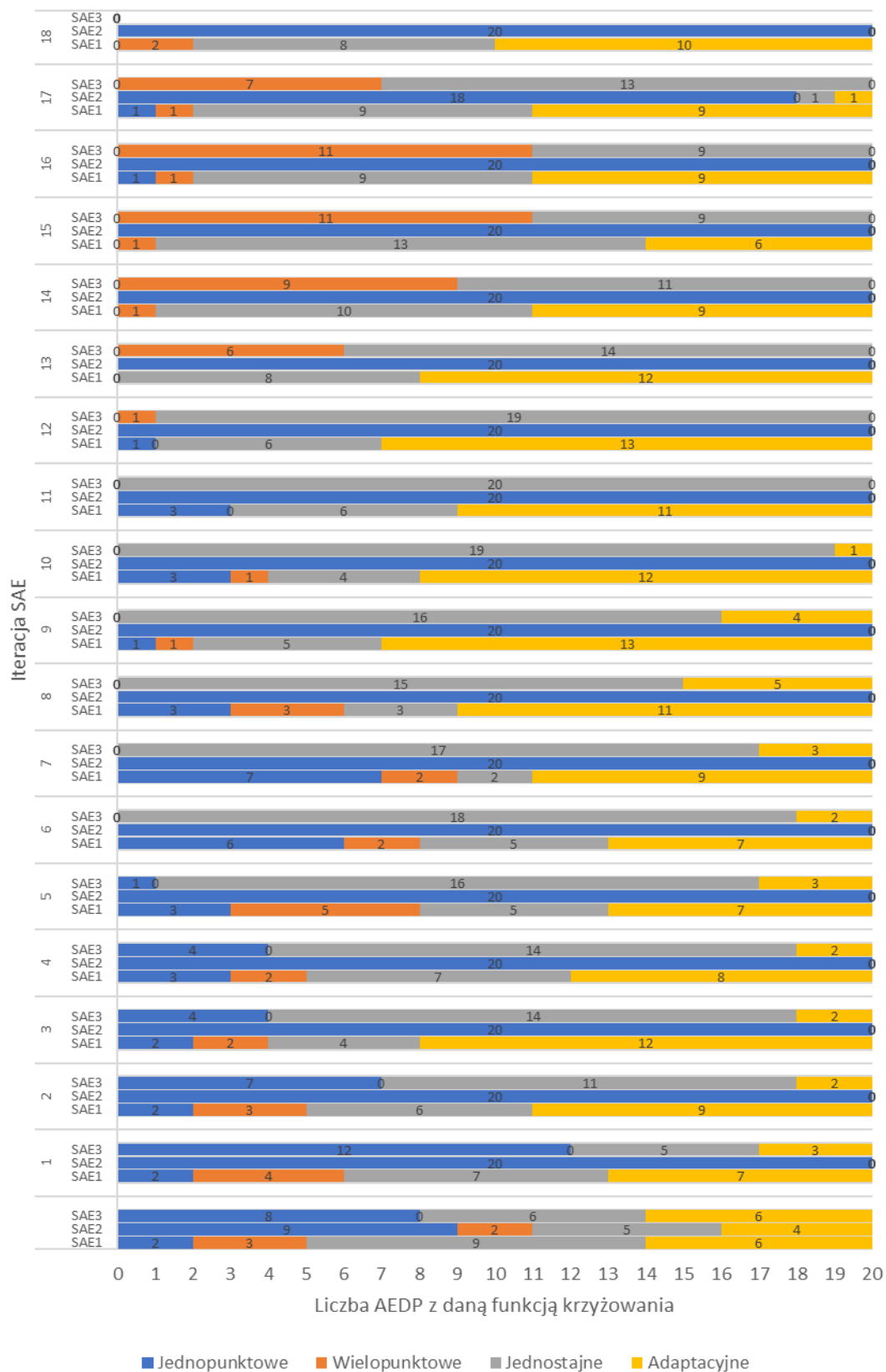
Dystrybucja funkcji selekcji AEDP w kolejnych iteracjach SAE dla problemu nr 1



Rysunek 18 przedstawia dystrybucję różnych wartości funkcji selekcji AEDP w kolejnych iteracjach SAE (SAE1, SAE2 oraz SAE3) podczas procesu optymalizacji problemu nr 1. Z wykresu możemy odczytać, że:

- Zmiany dystrybucji wartości funkcji selekcji AEDP w ramach SAE1 przebiegały w sposób zrównoważony i wszystkie cztery wartości były obecne przez większość czasu procesu optymalizacyjnego za wyjątkiem iteracji od 8 do 12. Funkcje selekcji proporcjonalna (zasada ruletki) oraz stochastyczna były najczęstszymi wartościami występującymi przez większość procesu optymalizacyjnego.
- Zmiany dystrybucji wartości funkcji selekcji AEDP w ramach SAE2 w początkowej fazie przebiegały w bardzo dynamiczny sposób. Selekcja proporcjonalna wyparła inne typy selekcji w iteracji nr 1, by zostać wypartą w iteracji nr 3 przez selekcję turniejową. Selekcja turniejowa utrzymała swoją dominację do końca procesu optymalizacji.
- Zmiany dystrybucji wartości funkcji selekcji AEDP w ramach SAE3 przebiegały w sposób płynny zgodnie z kilkoma widocznymi trendami:
 - Malejący udział selekcji proporcjonalnej, która jako cecha wymarła w 11 iteracji SAE3.
 - Rosnący udział selekcji stochastycznej do iteracji nr 6. Następnie silny trend malejący trwający aż do całkowitego wymarcia tej cechy w populacji AEDP w iteracji nr 10.
 - Szybkie wymarcie selekcji turniejowej w iteracji nr 3, by odrodzić się w iteracji nr 9, zdominować tę cechę populacji w iteracjach nr 13 oraz 14. Następnie wyraźnie widoczny jest trend zanikania tej cechy wśród AEDP w kolejnych iteracjach SAE.
 - Rosnący udział selekcji rankingowej w kolejnych iteracjach SAE3.
- Dostrzec można również kilka podobnych cech podczas ewolucji nastaw AEDP wszystkich trzech SAE:
 - Znaczny udział selekcji proporcjonalnej w początkowej fazie procesu optymalizacyjnego
 - Przebudzenie selekcji turniejowej w końcowej fazie procesu optymalizacyjnego (około iteracji 13 oraz 14).

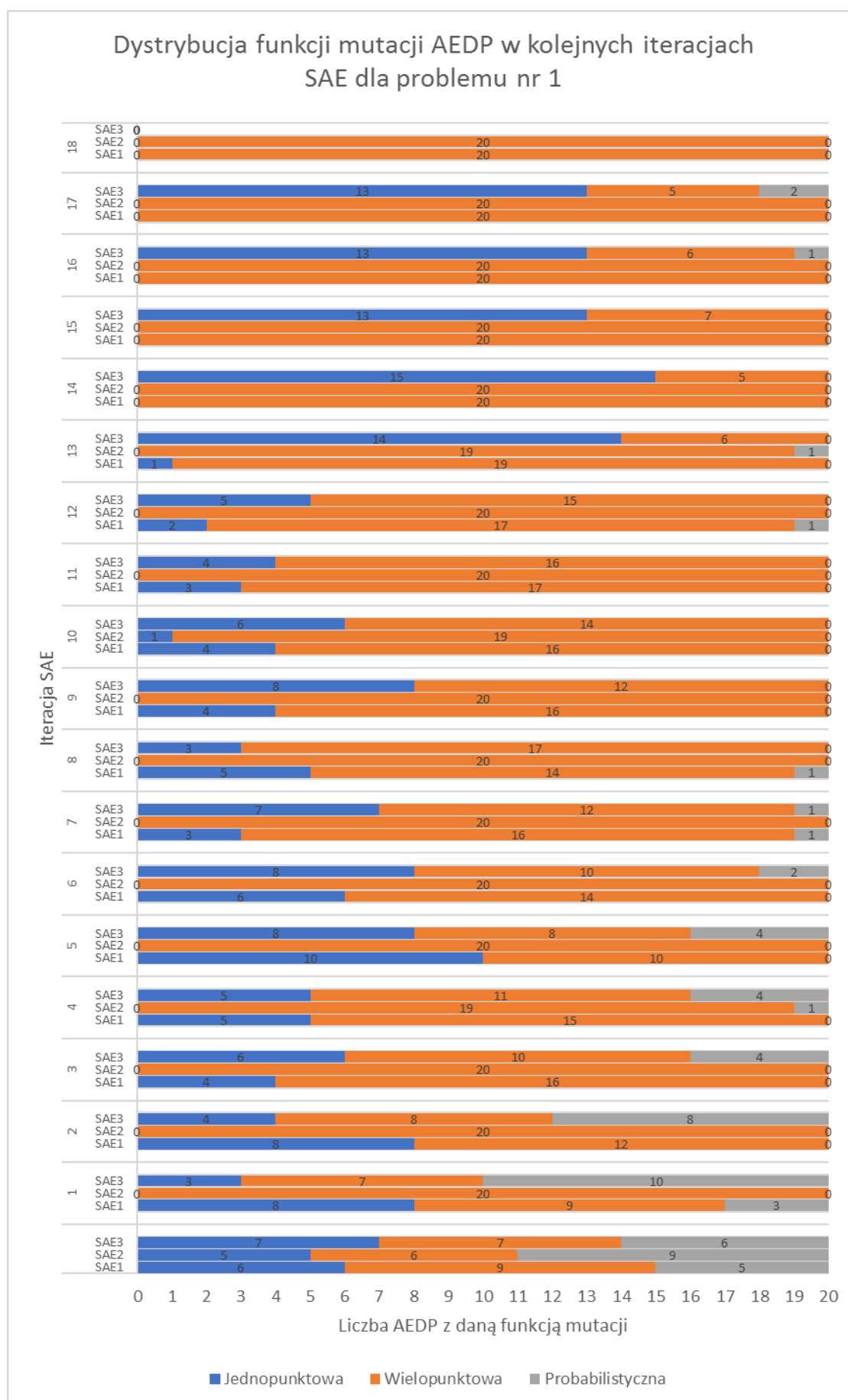
Dystrybucja funkcji krzyżowania AEDP w kolejnych iteracjach SAE dla problemu nr 1



Rysunek 19: Wykres dystrybucji funkcji krzyżowania AEDP w kolejnych iteracjach SAE dla problemu nr 1

Rysunek 19 przedstawia dystrybucję różnych wartości funkcji krzyżowania AEDP w kolejnych iteracjach SAE (SAE1, SAE2 oraz SAE3) podczas eksperymentu optymalizacji problemu nr 1. Z wykresu możemy odczytać, że:

- Zmiany dystrybucji wartości funkcji krzyżowania AEDP w SAE1 przebiegało w sposób zrównoważony z przewagą krzyżowania adaptacyjnego oraz jednostajnego nad innymi wartościami tej cechy.
- Zmiany dystrybucji wartości funkcji krzyżowania AEDP w SAE2 w początkowej fazie przebiegało w bardzo dynamiczny sposób. W iteracji nr 1 krzyżowanie jednopunktowe całkowicie zdominowało tę cechę i utrzymało swoją dominację do końca procesu optymalizacyjnego.
- Przez cały proces optymalizacji SAE3 były aktywne od 1 do 3 różnych wartości funkcji selekcji w AEDP. Jedyną wartością aktywną przez cały proces optymalizacji było krzyżowanie jednostajne, które jednocześnie było wartością dominującą przez większą część procesu optymalizacji za wyjątkiem iteracji nr 0, 1, 15 i 16.



Rysunek 20: Wykres dystrybucji funkcji mutacji AEDP w kolejnych iteracjach SAE dla problemu nr 1

Rysunek 20 przedstawia dystrybucję różnych wartości funkcji mutacji AEDP w kolejnych iteracjach SAE (SAE1, SAE2 oraz SAE3) podczas eksperymentu optymalizacji problemu nr 1. Z wykresu możemy odczytać, że:

- Podczas optymalizacji SAE1, mutacja wielopunktowa systematycznie stawała się coraz to liczniejszą cechą wśród AEDP, aż do całkowitego wyparcia innych wartości w iteracji nr 14. Mutacja probabilistyczna pierwsza została wyparta jako cecha populacji AEDP, bo już w iteracji nr 9. Następna była mutacja jednopunktowa, która została wyparta w iteracji nr 14.
- Zmiany wartości funkcji mutacji AEDP podczas optymalizacji SAE2 ponownie nastąpiły w sposób skokowy. Podobna sytuacja miała miejsce dla funkcji selekcji oraz krzyżowania. Mutacja wielopunktowa wyparła pozostałe wartości w iteracji nr 1 i utrzymała dominujący wpływ do aż końca procesu optymalizacji.
- Optymalizacja funkcji mutacji AEDP w SAE3 przebiegała w sposób zrównoważony z przewagą mutacji wielopunktowej w środkowej fazie procesu optymalizacyjnego (iteracje od 7 do 12), by ustąpić miejscach mutacji jednopunktowej w końcowej fazie (iteracje od 13 do 17).
- Wspólne cechy optymalizacji wartości funkcji mutacji AEDP dla wszystkich SAE biorących udział w procesie optymalizacji to:
 - Mutacja probabilistyczna odegrała tylko sporadyczny udział w procesie optymalizacyjnym i została wyparta przez inne wartości.
 - Mutacja wielopunktowa była najczęściej występującą wartością funkcji mutacji AEDP.

5.3.1. Analiza wyników eksperymentu nr 2

Eksperyment nr 2 polegał na optymalizacji funkcji De Jonga F3 (szczegółowe informacje na temat modelu tego problemu optymalizacyjnego znajdują się w rozdziale 5.1.2) z ograniczeniem czasowym wynoszącym 300 sekund oraz ustawionym warunkiem zakończenia optymalizacji po znalezieniu rozwiązania optymalnego. Optymalizacja wykonana była przy użyciu algorytmu losowego, trzech klasycznych algorytmów ewolucyjnych i trzech samoadaptacyjnych algorytmów ewolucyjnych (szczegóły konfiguracji algorytmów znajdują się w rozdziale 5.2). Otrzymane wyniki zostały załączone do pracy w formie elektronicznej, w folderze „Eksperyment 2 - funkcja De Jonga F2”. Aby odtworzyć eksperyment należy uruchomić skrypt „experiment_2.py” przy użyciu interpretera języka Python (szczegółowe informacje na temat konfiguracji aplikacji można znaleźć w rozdziale 4).

Analiza postępu oraz optymalizacji AEDP nie będzie wykonana dla tego eksperymentu, gdyż czas trwania procesu optymalizacyjnego miał różne wartości dla różnych algorytmów i nie jest możliwe efektywne porównanie osiągnięć różnych algorytmów w jednostce iteracji. Dodatkowo wszystkie trzy SAE biorące udział w eksperymencie odbyły tylko jedną iterację i tym samym ewolucji nastaw AEDP nie miała miejsca.

5.3.1.1. Analiza najlepszych rozwiązań

Tabela 3 zawiera porównanie najlepszych rozwiązań odnalezionych przez algorytmy optymalizacyjne, które były częścią eksperymentu nr 2. W kolumnie „Ranking” została umieszczona liczba porządkowa sortująca badane algorytmy według jakości najlepszych rozwiązań odnalezionych przez te algorytmy.

Tabela 3: Porównanie najlepszych rozwiązań otrzymanych w eksperymencie nr 2

Algorytm Optymalizacyjny	Najlepsze znalezione rozwiązanie	Wartość funkcji celu najlepszego rozwiązania	Ranking
Algorytm Losowy	$x_1 \approx -4.893$ $x_2 \approx -4.547$ $x_3 \approx -5.070$ $x_4 \approx -4.788$ $x_5 \approx -5.089$	-27	7
SAE1	$x_1 \approx -5.073$ $x_2 \approx -5.101$ $x_3 \approx -5.049$ $x_4 \approx -5.077$ $x_5 \approx -5.003$	-30 Wynik osiągnięto po około 15 sekundach.	2
SAE2	$x_1 \approx -5.002$ $x_2 \approx -5.015$ $x_3 \approx -5.098$ $x_4 \approx -5.035$ $x_5 \approx -5.118$	-30 Wynik osiągnięto po około 15 sekundach.	2
SAE3	$x_1 \approx -5.027$ $x_2 \approx -5.120$ $x_3 \approx -5.038$ $x_4 \approx -5.012$ $x_5 \approx -5.015$	-30 Wynik osiągnięto po około 15 sekundach.	2
AE1	$x_1 \approx -5.031$ $x_2 \approx -5.118$ $x_3 \approx -5.060$ $x_4 \approx -5.085$ $x_5 \approx -5.047$	-30 Wynik osiągnięto po około 4 sekundach.	1
AE2	$x_1 \approx -5.115$ $x_2 \approx -5.083$ $x_3 \approx -5.043$ $x_4 \approx -5.086$ $x_5 \approx -5.022$	-30 Wynik osiągnięto po około 28 sekundach.	6
AE3	$x_1 \approx -5.016$ $x_2 \approx -5.023$ $x_3 \approx -5.035$ $x_4 \approx -5.083$ $x_5 \approx -5.072$	-30 Wynik osiągnięto po około 18 sekundach.	5

Podczas tego eksperymentu wszystkie algorytmy z wyjątkiem algorytmu losowego odnalazły rozwiązanie optymalne (w takim przypadku pod uwagę brany był również czas jaki algorytmy potrzebowały, aby odnaleźć optymalne rozwiązanie). Wyniki zamieszczone w tabeli powyżej pokazują, iż dobrze przystosowany do tego problemu klasyczny algorytm ewolucyjny AE1 okazał się najskuteczniejszy. Następne pod względem efektywności znalazły się trzy samoadaptacyjne algorytmy ewolucyjne, które w podobnym czasie odnalazły rozwiązanie optymalne.

5.3.2. Analiza wyników eksperymentu nr 3

Eksperyment nr 3 polegał na optymalizacji wymiarów sprężyny naciskowej (szczegółowe informacje na temat modelu tego problemu optymalizacyjnego znajdują się w rozdziale 5.1.3) z ograniczeniem czasowym wynoszącym 20 sekund. Optymalizacja wykonana była przy użyciu algorytmu losowego, trzech klasycznych algorytmów ewolucyjnych i trzech samoadaptacyjnych algorytmów ewolucyjnych (szczegóły konfiguracji algorytmów znajdują się w rozdziale 5.2). Otrzymane wyniki, które są analizowane w tym rozdziale znajdują się w folderze „Eksperyment 3 – sprężyna naciskowa”, który został dołączony (w formie elektronicznej) do tej pracy. Aby, odtworzyć eksperyment należy uruchomić skrypt „experiment_3.py” przy użyciu interpretera języka Python (szczegółowe informacje na temat konfiguracji aplikacji można znaleźć w rozdziale 4).

5.3.2.1. Analiza najlepszych rozwiązań

Tabela 4 zawiera porównanie najlepszych rozwiązań odnalezionych przez algorytmy optymalizacyjne, które były częścią eksperymentu nr 3. W kolumnie „Ranking” została umieszczona liczba porządkowa sortująca badane algorytmy według jakości najlepszych rozwiązań odnalezionych przez te algorytmy.

Tabela 4: Porównanie najlepszych rozwiązań otrzymanych w eksperymencie nr 3

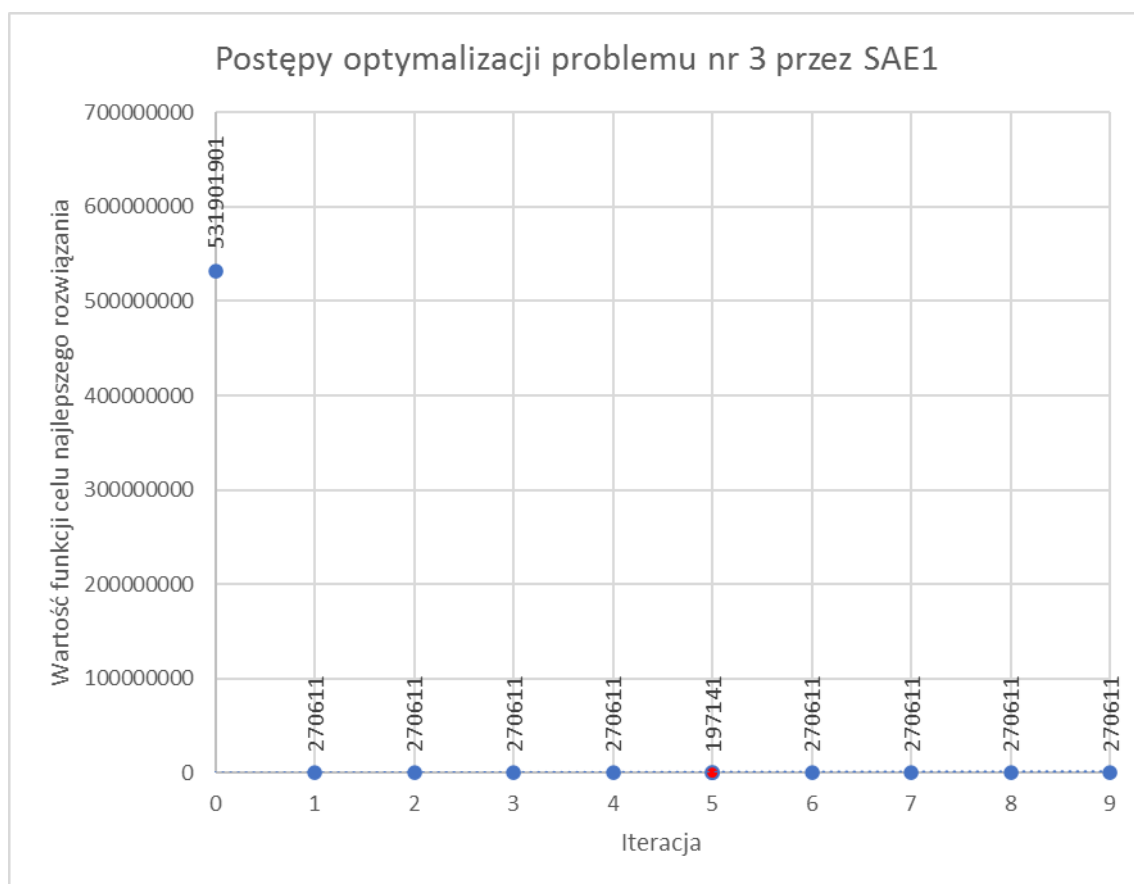
Algorytm Optymalizacyjny	Najlepsze znalezione rozwiązanie	Wartość funkcji celu najlepszego rozwiązania	Ranking
Algorytm Losowy	$x_1 = 8.5$ $x_2 = 60$ $x_3 = 222$ $x_4 = 12$	~ 272038.670	6
SAE1	$x_1 = 8$ $x_2 = 60$ $x_3 = 230$ $x_4 = 9.5$	~ 197140.586	3
SAE2	$x_1 = 8$ $x_2 = 60$ $x_3 = 232$ $x_4 = 9.5$	~ 197377.830	4
SAE3	$x_1 = 8$ $x_2 = 60$ $x_3 = 214$ $x_4 = 9.5$	~ 195306.803	1
AE1	$x_1 = 8$ $x_2 = 52$ $x_3 = 256$ $x_4 = 11$	~ 76778842685.454	7
AE2	$x_1 = 8.5$ $x_2 = 60$ $x_3 = 210$ $x_4 = 12$	~ 270809.295	5
AE3	$x_1 = 8$ $x_2 = 60$ $x_3 = 214$ $x_4 = 9.5$	~ 195306.803	1

Wyniki przedstawione w tabeli powyżej pokazują, iż SAE ponownie okazały się skuteczniejsze od AE zajmując kolejno 3, 4 i 1 miejsce w rankingu.

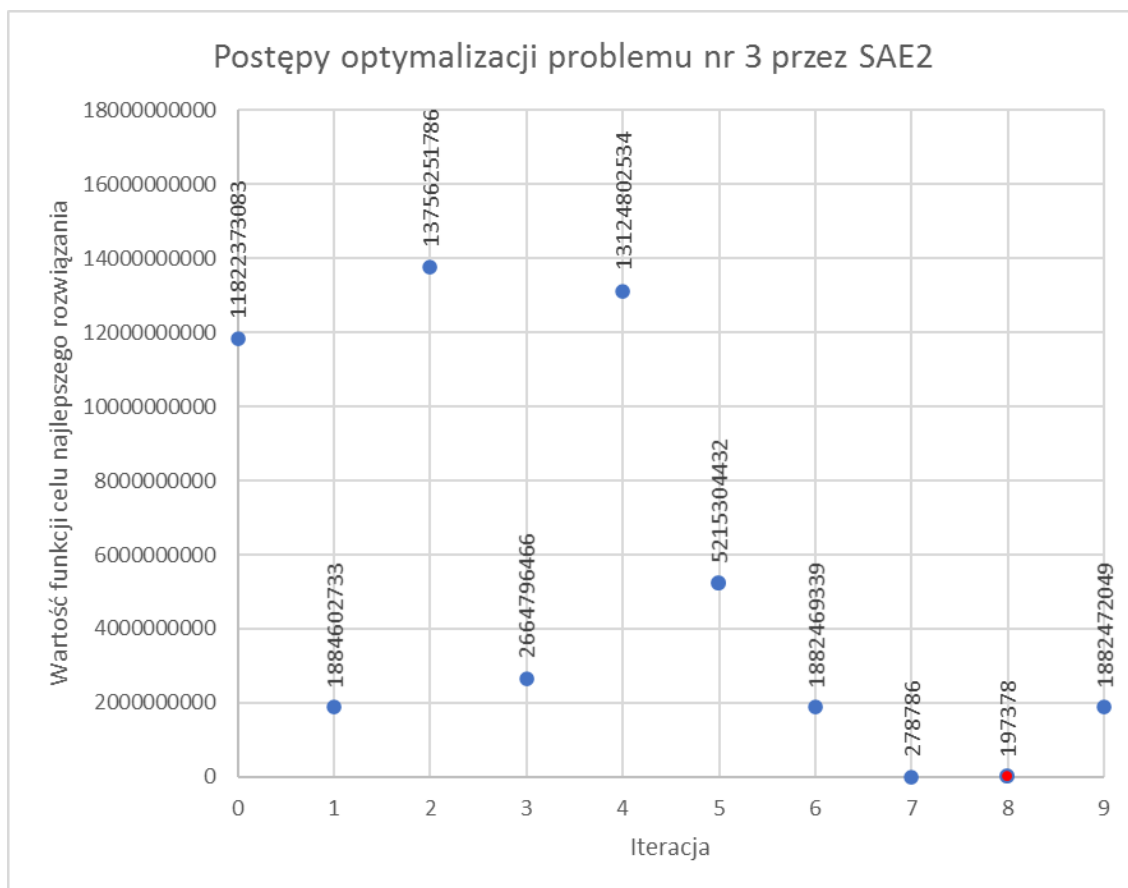
Wartym odnotowania jest to iż AE1 zwrócił najgorszy wynik spośród wszystkich porównywanych algorytmów. Przyczyną takiego stanu rzeczy prawdopodobnie była konfiguracja algorytmu, która okazała się nieskuteczna dla tego problemu. Drugim najgorszym algorytmem okazał się algorytm losowy.

5.3.2.2. Analiza postępu poszukiwań

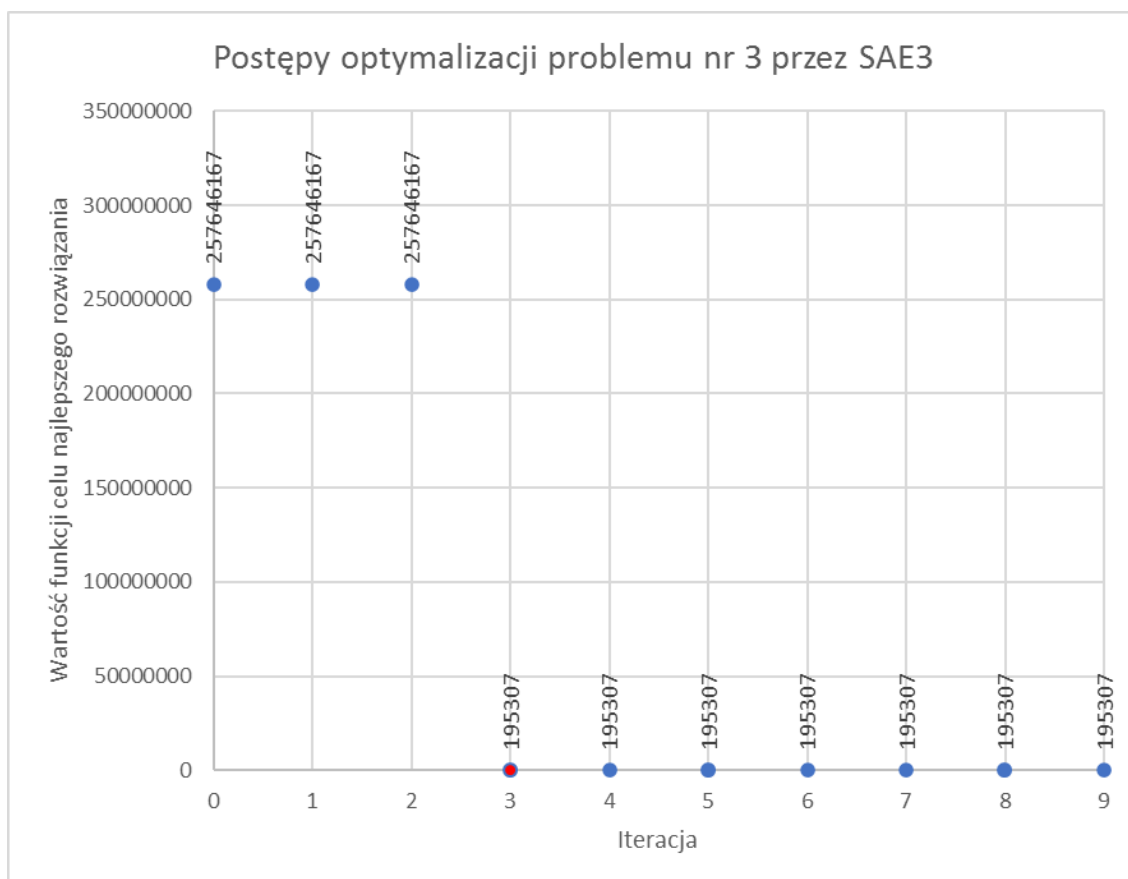
Na wykresach *Rysunek 21*, *Rysunek 22*, *Rysunek 23*, *Rysunek 24*, *Rysunek 25*, *Rysunek 26* oraz *Rysunek 27* przedstawione są postępy poszukiwaniach rozwiązań problemu nr 3 przez kolejne algorytmy optymalizacyjne biorące udział w eksperymencie badawczym. Ze względu na znacząco odbiegającą liczbę iteracji wykonanych przez różne algorytmy optymalizacyjne (dla SAE 10, dla AE od 660 do 1803, dla Algorytmu Losowego 428) wynikającą z różnej złożoności obliczeniowej różnych algorytmów, wszystkie wykresy zostały podzielone 10 etapów o identycznej długości pod względem czasu. Najlepsze rozwiązanie odnalezione przez dany algorytm optymalizacyjny oznaczone jest czerwonym punktem na wykresie.



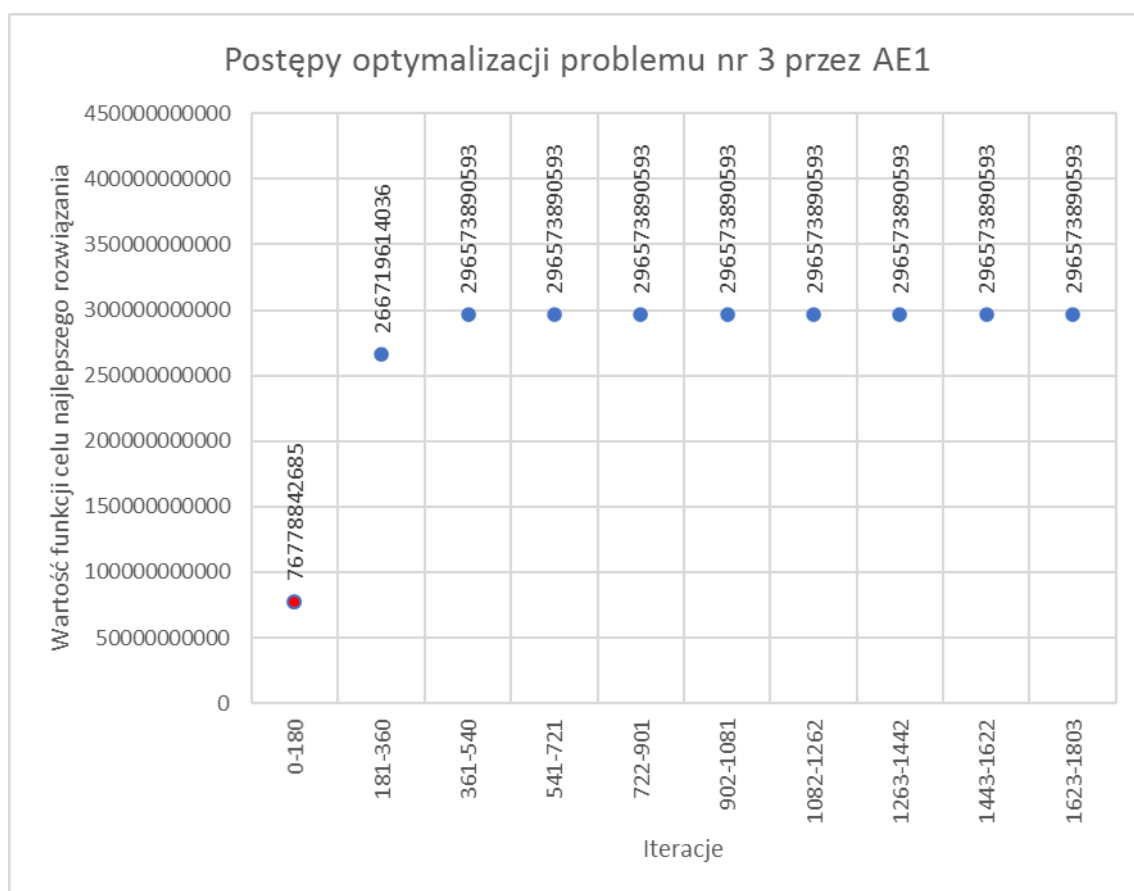
Rysunek 21: Postępy optymalizacji problemu nr 3 przez SAE1



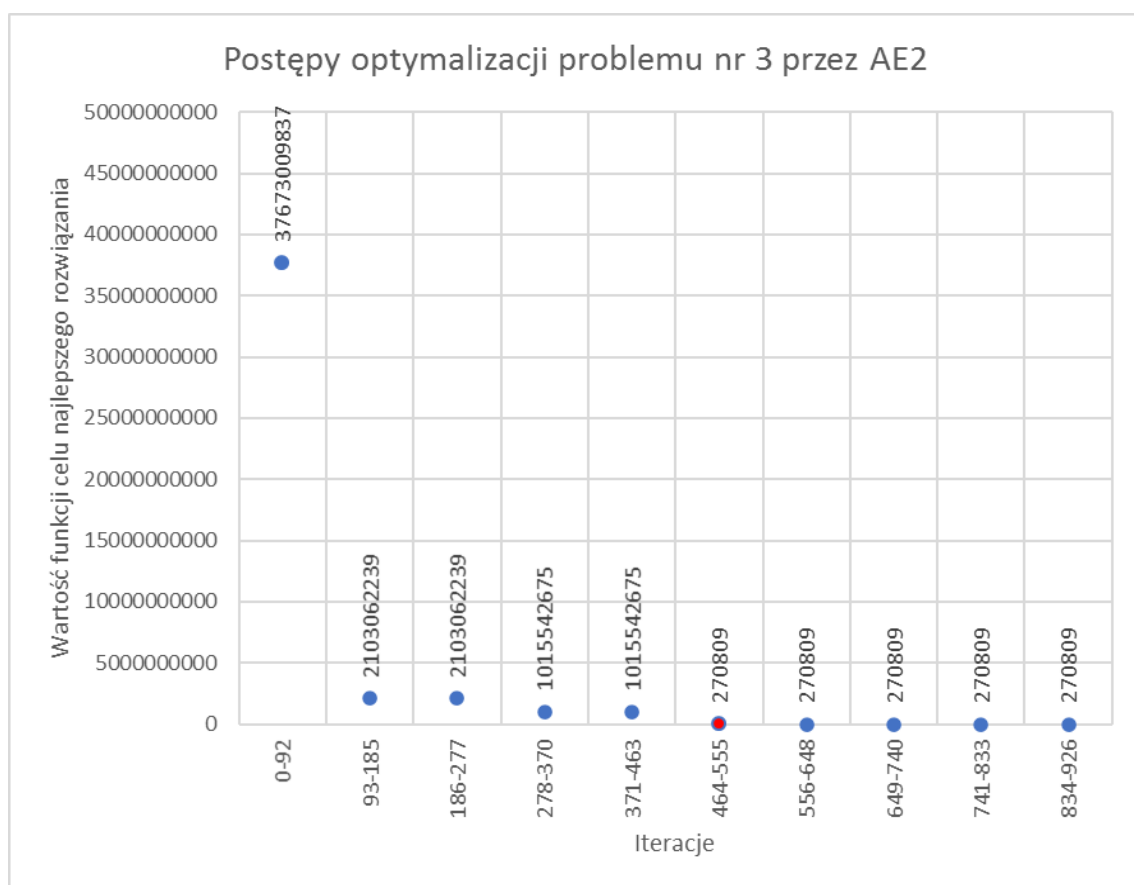
Rysunek 22: Postępy optymalizacji problemu nr 3 przez SAE2



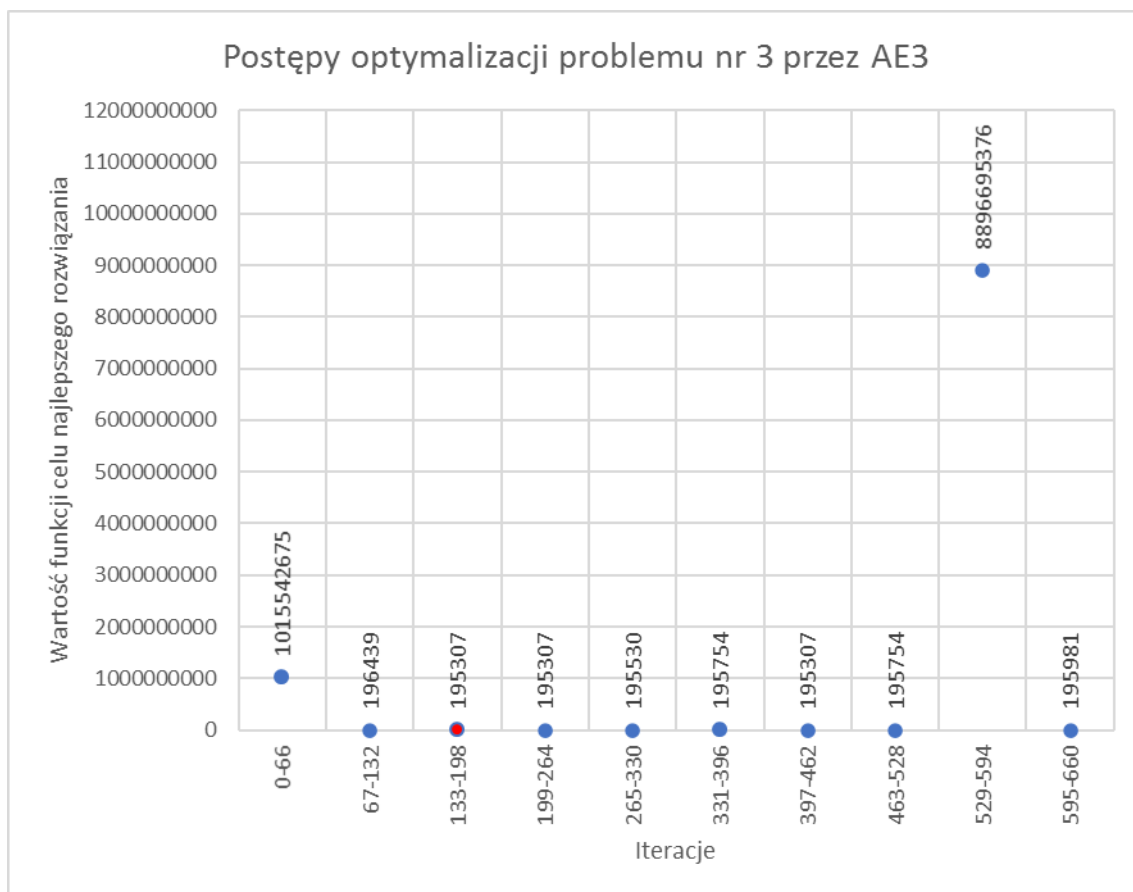
Rysunek 23: Postępy optymalizacji problemu nr 3 przez SAE3



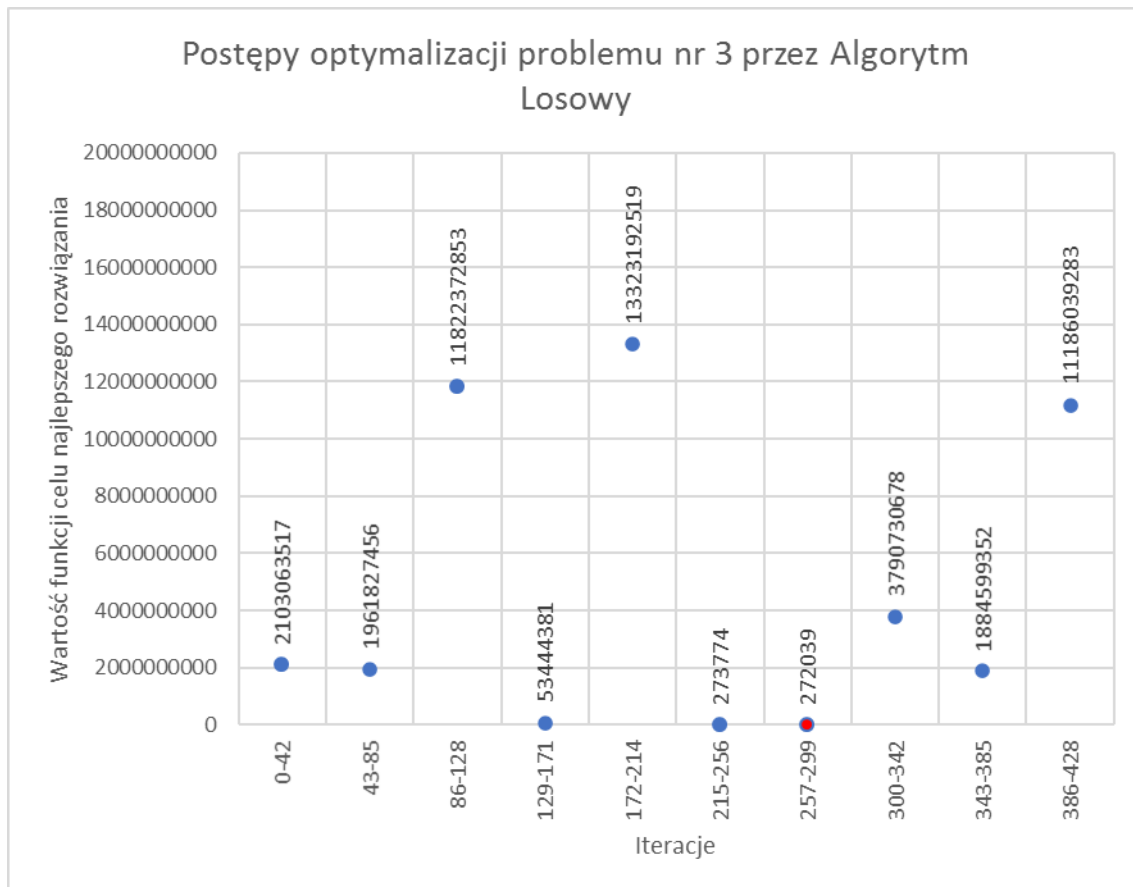
Rysunek 24: Postępy optymalizacji problemu nr 3 przez AE1



Rysunek 25: Postępy optymalizacji problemu nr 3 przez AE2



Rysunek 26: Postępy optymalizacji problemu nr 3 przez AE3



Rysunek 27: Postępy optymalizacji problemu nr 3 przez Algorytm Losowy

Tabela 5 zawiera porównanie etapów optymalizacji osiągnięcia najlepszego rozwiązania przez kolejne algorytmy optymalizacyjne biorące udział w eksperymencie badawczym.

Tabela 5: Porównanie etapów optymalizacji problemu nr 3

Algorytm Optymalizacyjny	Etap, w którym odnaleziono najlepsze rozwiązanie
<i>Algorytm Losowy</i>	7 z 10
<i>SAE1</i>	6 z 10
<i>SAE2</i>	8 z 10
<i>SAE3</i>	3 z 10
<i>AE1</i>	1 z 10
<i>AE2</i>	6 z 10
<i>AE3</i>	3 z 10

Wykresy (*Rysunek 21* oraz *Rysunek 23*) przedstawiające poprawę najlepszych rozwiązań znajdowanych przez SAE1 oraz SAE3 przypominają analogiczne wykresy sporządzone dla problemu nr 1 (*Rysunek 11* oraz *Rysunek 13*). Podobnie jak na tamtych wykresach, po początkowym okresie trwającym 1-3 iteracje, dostrzec można okres stabilizacji i poszukiwania lepszego rozwiązania. W przypadku SAE1 udało się w iteracji nr 5 odnaleźć rozwiązanie lepsze od tego odnalezionego w iteracji nr 1. Natomiast SAE3, nie potrafiło poprawić wyniki z iteracji nr 3.

W przypadku SAE2 (patrz *Rysunek 22*) proces optymalizacji przebiegał bardziej dynamicznie, wykazując większą wariancję. Przypomina on bardziej poszukiwania w wykonaniu Algorytmu Losowego niż innych SAE.

Poszukiwania w wykonaniu AE1 przedstawione na *Rysunek 24* pokazują, że algorytm ten znalazł najlepsze rozwiązanie w początkowej fazie, po której odnajdywane były coraz to gorsze rozwiązania. Sugeruje to, iż algorytm ten był totalnie nieprzystosowany do tego typu problemu, gdyż nie wykazywał on żadnej poprawy w kolejnych iteracjach.

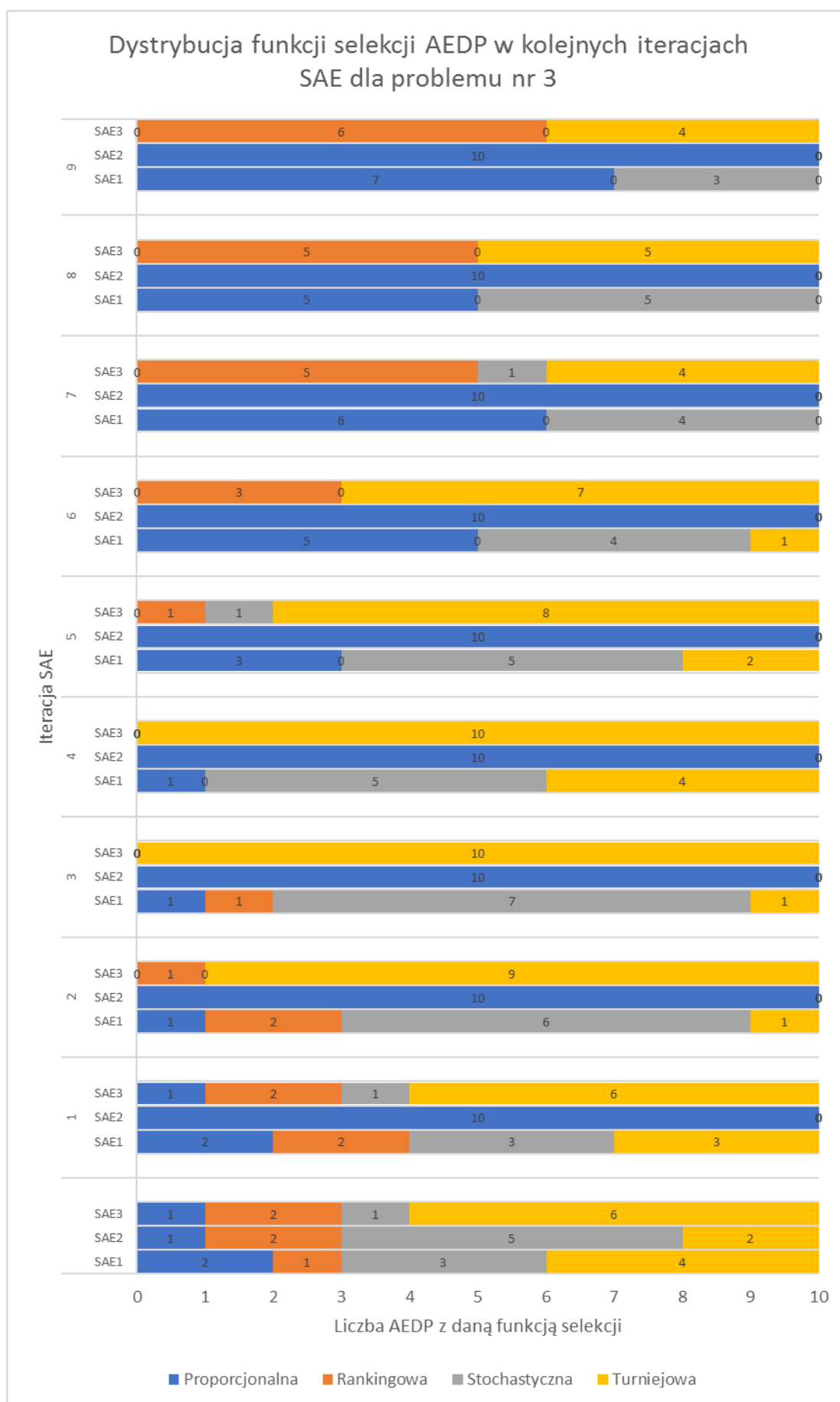
AE2 oraz AE3 wykazały się systematycznością w poszukiwaniach coraz to lepszych rozwiązań znajdując dobre rozwiązania w późniejszych iteracjach. Jedyne wyjątek od tej reguły stanowi końcowy etap AE3 (iteracje 529-594).

Wyniki odnajdywane przez Algorytm Losowy wykazywały się wysoką wariancją i dużym rozrzutem jakości najlepszych rozwiązań.

5.3.2.3. Analiza optymalizacji nastaw AEDP w SAE

W tej części pracy zostanie porównana ewolucja konfiguracji AEDP w kolejnych iteracjach SAE podczas procesu optymalizacji problemu nr 3, a w szczególności:

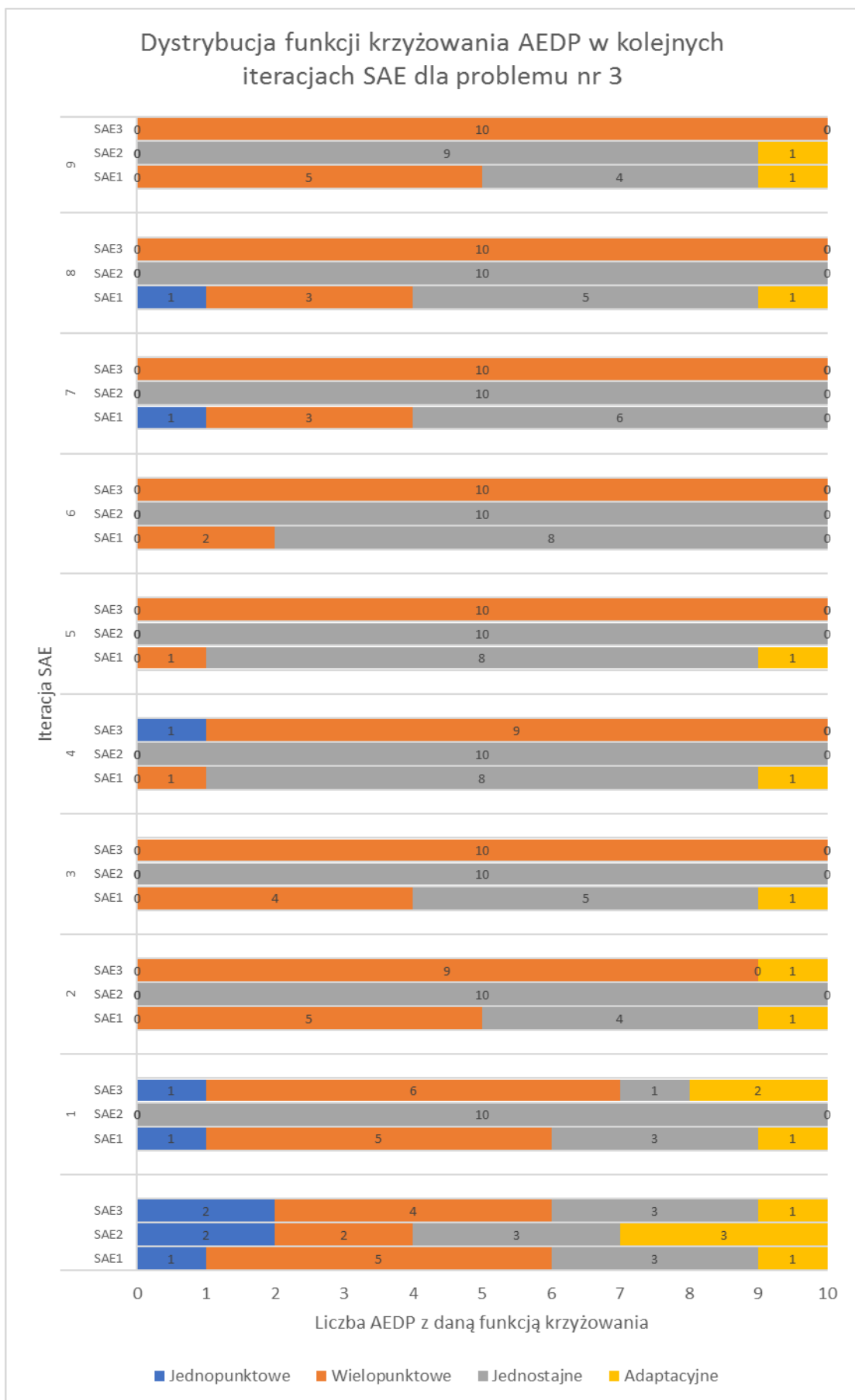
- zmiany dystrybucji funkcji selekcji w populacji AEDP
- zmiany dystrybucji funkcji krzyżowania w populacji AEDP
- zmiany dystrybucji funkcji mutacji w populacji AEDP



Rysunek 28: Wykres dystrybucji funkcji selekcji AEDP w kolejnych iteracjach SAE dla problemu nr 3

Rysunek 28 przedstawia dystrybucję różnych wartości funkcji selekcji AEDP w kolejnych iteracjach SAE (SAE1, SAE2 oraz SAE3) podczas eksperymentu optymalizacji problemu nr 3. Z wykresu możemy odczytać, że:

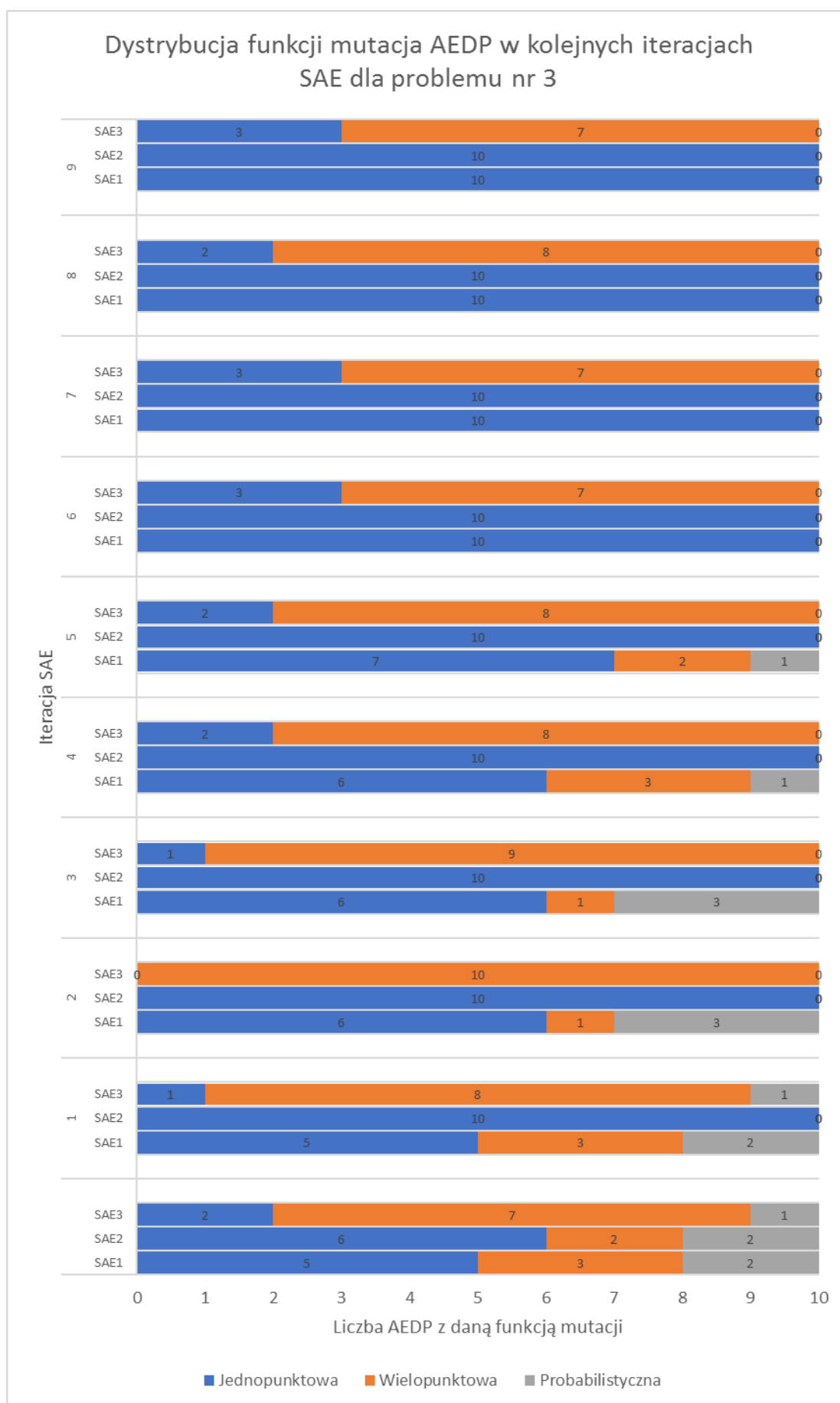
- Optymalizacja funkcji selekcji AEDP w SAE1 przebiegała w sposób zrównoważony. Selekcja stochastyczna była najczęstszą cechą AEDP w środkowej fazie (od iteracji nr 2 do iteracji nr 5), natomiast selekcja proporcjonalna w końcowej fazie procesu optymalizacyjnego (od iteracji nr 6 do końca procesu optymalizacji) SAE1. Selekcja turniejowa powolnie wymierała jako cecha AEDP, by zniknąć w iteracji nr 7. Selekcja rankingowa wymarła dużo wcześniej bo już w iteracji nr 4.
- Zmiany wartości funkcji selekcji AEDP podczas optymalizacji SAE2 następowały w sposób dynamiczny. Selekcja proporcjonalna całkowicie zdominowała tę cechę już w iteracji nr 1. Ten stan utrzymał się do końca optymalizacji SAE2.
- Optymalizacja funkcji selekcji AEDP w SAE3 przebiegała z największym udziałem selekcji turniejowej oraz rankingowej. Selekcja proporcjonalna i stochastyczna miały sporadyczny udział (maksymalnie 1 AEDP posiadał daną wartość w każdej z iteracji SAE3).



Rysunek 29: Wykres dystrybucji funkcji krzyżowania AEDP w kolejnych iteracjach SAE dla problemu nr 3

Rysunek 29 przedstawia dystrybucję różnych wartości funkcji krzyżowania AEDP w kolejnych iteracjach SAE (SAE1, SAE2 oraz SAE3) podczas eksperymentu optymalizacji problemu nr 3. Z wykresu możemy odczytać, że:

- Optymalizacja funkcji krzyżowania AEDP w SAE1 przebiegała w sposób zrównoważony. Krzyżowanie jednostajne i wielopunktowe były najczęstszymi cechami AEDP tej cechy. Pozostałe wartości (krzyżowanie jednopunktowe oraz adaptacyjne) miały sporadyczny udział (maksymalnie 1 AEDP posiadał daną wartość w każdej z iteracji SAE3).
- Zmiany wartości funkcji krzyżowania AEDP podczas optymalizacji SAE2 ponownie postępowały w sposób dynamiczny, tak jak miało to miejsca w przypadku ewolucji funkcji selekcji. Krzyżowanie jednostajne osiągnęło całkowitą dominację w iteracji nr 1 i utrzymało swoją przewagę do końca procesu optymalizacyjnego SAE2.
- Optymalizacja funkcji krzyżowania AEDP w SAE3 przebiegała z największym udziałem krzyżowania wielopunktowego, które od iteracji nr 2 do końca optymalizacji SAE3 było wartością obecną u co najmniej 90% AEDP.
- Wspólne cechy optymalizacji wartości funkcji krzyżowania AEDP dla wszystkich SAE biorących udział w eksperymencie to:
 - sporadyczny udział krzyżowania adaptacyjnego
 - sporadyczny udział krzyżowania jednopunktowego
 - dominujący wpływ krzyżowania wielopunktowego, jednostajnego lub obu



Rysunek 30: Wykres dystrybucji funkcji mutacji AEDP w kolejnych iteracjach SAE dla problemu nr 3

Rysunek 30 przedstawia dystrybucję różnych wartości funkcji mutacji AEDP w kolejnych iteracjach SAE (SAE1, SAE2 oraz SAE3) podczas eksperymentu optymalizacji problemu nr 3. Z wykresu możemy odczytać, że:

- Funkcja mutacji AEDP w SAE1 została zdominowana przez mutację jednopunktową, która była najczęściej występującą wartością przez cały proces optymalizacji.
- Zmiany wartości funkcji mutacji AEDP podczas optymalizacji SAE2 ponownie postępowały w sposób dynamiczny. Mutacja jednopunktowa wyparła pozostałe wartości w iteracji nr 1 i ten stan rzeczy utrzymał się do końca procesu optymalizacyjnego.
- Optymalizacja funkcji mutacji AEDP w SAE3 przebiegała z głównym udziałem mutacji wielopunktowej, która to była najczęściej występującą wartością przez cały proces optymalizacyjny.
- Wspólne cechy optymalizacji wartości funkcji krzyżowania AEDP dla wszystkich SAE biorących udział w eksperymencie to:
 - sporadyczny udział mutacji probabilistycznej
 - mutacja jednopunktowa była obecna przez większość procesu optymalizacyjnego wszystkich SAE z dominującym wpływem dla dwóch z nich (SAE1 oraz SAE2)

5.3.3. Podsumowanie

5.3.3.1. Porównanie najlepszych rozwiązań

Tabela 6 zawiera porównanie najlepszych rozwiązań odnalezionych przez wszystkie algorytmy biorące udział w eksperymencie.

Tabela 6: Porównanie najlepszych rozwiązań otrzymanych we wszystkich trzech eksperymentach

Algorytm Optymalizacyjny	Najlepsze rozwiązanie problemu nr 1	Najlepsze rozwiązanie problemu nr 2	Najlepsze rozwiązanie problemu nr 3
Algorytm Losowy	~ 0.0008291433	-27	~ 272038.670
SAE1	~ 0.0000013375	- 30 (w 15 s)	~ 197140.586
SAE2	~ 0.0000552663	-30 (w 15 s)	~ 197377.830
SAE3	~ 0.0000039107	-30 (w 15 s)	~ 195306.803
AE1	~ 0.0000118109	-30 (w 4 s)	~ 76778842685.454
AE2	~ 0.0197114754	-30 (w 28 s)	~ 270809.295
AE3	~ 0.0035450134	-30 (w 18 s)	~ 195306.803

Tabela 7 zawiera porównanie miejsc rankingowych osiągniętych przez badane algorytmy w kolejnych eksperymentach (kolumny „Ranking w eksp. Nr 1”, „Ranking w eksp. nr 2”, „Ranking w eksp. nr 3”), a także wartość średnią (kolumna „Średnie miejsce”) i odchylenie standardowe (kolumna „Odchylenie standardowe miejsca”) obliczone dla tych wartości.

W kolumnie „Ranking” została umieszczona liczba porządkowa sortująca algorytmy od najmniejszej do największej wartości średniej miejsca.

Tabela 7: Porównanie skuteczności badanych algorytmów we wszystkich trzech eksperymentach

<i>Algorytm Optymalizacyjny</i>	Ranking w eksp. nr 1	Ranking w eksp. nr 2	Ranking w eksp. nr 3	Średnie miejsce	Odchylenie standardowe miejsca	Ranking
<i>Algorytm Losowy</i>	5	7	6	6	~ 0,82	7
<i>SAE1</i>	1	2	3	2	~ 0,82	2
<i>SAE2</i>	4	2	4	~ 3,33	~ 0,94	3
<i>SAE3</i>	2	2	1	1,67	~ 0,47	1
<i>AE1</i>	3	1	7	~ 3,67	~ 2,49	4
<i>AE2</i>	7	6	5	~ 6	~ 0,82	6
<i>AE3</i>	6	5	1	4	~ 2,16	5

Na podstawie wyników zawartych w Tabela 6 oraz Tabela 7 można stwierdzić, iż podczas badania zaproponowane Samoadaptacyjne Algorytmy Ewolucyjne okazały się statystycznie zdecydowanie skuteczniejsze od zaproponowanych klasycznych Algorytmów Ewolucyjnych oraz Algorytmu Losowego. Algorytm Losowy zgodnie z oczekiwaniami okazał się najmniej efektywnym algorytmem i był częścią eksperymentu wyłącznie ze względów referencyjnych. Należy zauważyć, iż klasyczne AE dla niektórych problemów osiągały lepsze rezultaty od SAE (np. AE1 dla problemu nr 2, AE3 dla problemu nr 3), co sugeruje jakoby dobrze skonfigurowany AE pod dany problem optymalizacyjny może osiągać lepsze rezultaty od SAE.

Kolejnym aspektem jaki należy wziąć pod uwagę przy ocenie algorytmów jest wartość wariancji, która świadczy o uniwersalności i podobnej efektywności algorytmu dla różnych rodzajów problemów optymalizacyjnych. Na podstawie otrzymanych wyników można podejrzewać, iż SAE mają dużą większą zdolność adaptacji do różnego rodzaju problemu i nie wykazują takiej wariancji w efektywności (w zależności od bazowej konfiguracji) jak ma to miejsce w przypadku klasycznych AE.

5.3.3.2. Porównanie optymalizacji nastaw AEDP w SAE

Na podstawie danych zebranych następujących częściach tej pracy 5.3.1.3 Analiza optymalizacji nastaw AEDP w SAE oraz 5.3.2.3 Analiza optymalizacji nastaw AEDP w SAE, możemy wnioskować, iż:

- Wszystkie dostępne funkcje selekcji (proporcjonalna, rankingowa, stochastyczna oraz turniejowa) miały znaczący udział w procesach optymalizacyjnych różnych SAE i cieszyły się umiarkowaną lub dużą popularnością wśród AEDP w zależności od postawionego problemu i konfiguracji SAE. Warto zauważyć, iż podczas optymalizacji problemu nr 1 największy udział przypadł selekcja turniejowej, natomiast przy optymalizacji problemu nr 3 selekcji jednopunktowa.
- Wszystkie dostępne funkcje krzyżowania (jednopunktowe, wielopunktowe, jednostajne oraz adaptacyjne) miały znaczący udział w procesach

optymalizacyjnych różnych SAE i cieszyły się umiarkowaną lub dużą popularnością wśród AEDP w zależności od postawionego problemu i konfiguracji SAE. Warto zauważyć, iż podczas optymalizacji problemu nr 1 największy udział przypadł krzyżowaniu jednopunktowemu i jednostajnemu, natomiast przy optymalizacji problemu nr 3 krzyżowaniu wielopunktowemu i jednostajnemu.

- Mutacje wielopunktowa i jednopunktowa zdominowały nastawy AEDP w SAE podczas optymalizacji kolejno problemów nr 1 oraz 3. Mutacja probabilistyczna miała sporadyczny udział w procesie optymalizacji została wyparta jako cecha we wszystkich obserwowanych procesach optymalizacji. Mogło to wynikać z jej niskiej efektywności (np. spowodowanej większą złożonością obliczeniową od pozostałych typów mutacji) lub niskim przystosowaniem do postawionych problemów.
- Zmiany nastaw AEDP w SAE2 zachodziły bardzo dynamicznie zarówno podczas optymalizacji problemu nr 1 jak i problemu nr 3. Biorąc pod uwagę, iż jakość rozwiązań odnalezionych przez SAE1 oraz SAE3 była nie gorsza od rozwiązań znalezionych przez SAE2 dla wszystkich optymalizowanych problemów można podejrzewać iż ten algorytm został dużo gorzej skonfigurowany od swoich sąsiadów. Można też podejrzewać, iż różnorodność nastaw AEDP wzmacnia działanie SAE.

5.4. Ocena Samoadaptacyjnego Algorytmu Ewolucyjnego

Samoadaptacyjne Algorytmy Ewolucyjne we wszystkich przeprowadzonych eksperymentach średnio wypadły lepiej od swoich odpowiedników w postaci klasycznych Algorytmów Ewolucyjnych. Tym samym można wnioskować, iż z dużym prawdopodobieństwem SAE okażą się skuteczniejsze od AE dla problemów, dla których optymalne nastawy AE nie są znane (tak jak miało to miejsce w przedstawionym eksperymencie badawczym), gdyż SAE wykazują dużą mniejszą wariację w działaniu niż ma to miejsce dla klasycznych AE. Dodatkowo SAE powinny okazać się skuteczniejsze w długotrwałej optymalizacji, gdyż mają one zdolność do dostosowywania sposobu przeszukiwania przestrzeni rozwiązań poprzez promowanie skuteczniejszych technik przeszukiwania na wszystkich etapach procesu optymalizacyjnego.

Wartym odnotowania jest fakt, iż skuteczność Samoadaptacyjnych Algorytmów Ewolucyjnych opierała się w dużej mierze na różnorodności AEDP, a nie ukierunkowanemu dążeniu do osiągnięcia jednolitej populacji AEDP o identycznych nastawach. Na tej podstawie można podejrzewać, iż SAE pomimo większej złożoności obliczeniowej od klasycznych AE, mają potencjał aby osiągać lepsze wyniki poprzez uzyskania efektu synergii między AEDP skonfigurowanymi na różne sposoby. Dodatkowo jednolity sposób optymalizacji we wszystkich etapach procesu optymalizacyjnego (tak jak ma to miejsce w AE) dla wielu problemów może okazać się mniej skuteczny niż adaptacyjne dostosowywanie procesu optymalizacyjnego w trakcie jego trwania.

6. Wnioski

Cele badawczy oraz użyteczny pracy zostały osiągnięte poprzez:

- Zaprojektowanie modelu Samoadaptacyjnego Algorytmu Ewolucyjnego, który szczegółowo został opisany w 3 części pracy *Projekt Samoadaptacyjnego Algorytmu Ewolucyjnego*.
- Opracowanie aplikacji komputerowej, opisanej 24 w części pracy *Aplikacja komputerowa do celów badawczych*.
- Porównanie działania Samoadaptacyjnego Algorytmu Ewolucyjnego na tle klasycznych Algorytmów Ewolucyjnych oraz Algorytmu Losowe zostało szczegółowo opisane w 5 części pracy *Eksperyment badawczy*.

W ramach pracy wykonano trzy eksperymenty badawcze, których celem była ocena efektywności SAE na tle innych algorytmów. Przebieg eksperymentów opisany został w części pracy 5.2 *Przebieg eksperymentu*. Na podstawie otrzymanych wyników została dokonana analiza (szczegóły znajdują się w częściach 5.3.1 Analiza wyników eksperymentu nr 1, 5.3.1 Analiza wyników eksperymentu nr 2 oraz 5.3.2 Analiza wyników eksperymentu nr 3), na podstawie które można wyciągnąć następujące wnioski na temat Samoadaptacyjnych Algorytmów Ewolucyjnych:

- SAE okazywały się statystycznie efektywniejsze od klasycznych AE.
- Skuteczność SAE wykazywała się mniejszą zależnością od konfiguracji (ustawień) niż miało to miejsce w przypadku klasycznych AE.
- SAE jest ciągłą adaptacją procesu optymalizacyjnego poprzez osiągnięcie synergii między różnymi konfiguracjami AEDP, co sugeruje iż SAE powinny wypadać dużo lepiej od AE w długotrwałych procesach optymalizacyjnych a także mieć większy potencjał do przeszukiwania przestrzeni rozwiązań poza lokalnym optimum.

Przeprowadzone wstępne badania nad Samoadaptacyjnymi Algorytmami Ewolucyjnymi są bardzo obiecujące i pokazują, iż algorytmy te mają dużą szansę na zastąpienie i wyparcie klasycznych Algorytmów Ewolucyjnych. W większości przeprowadzonych eksperymentów SAE okazywały się skuteczniejsze od AE. Największymi zaletami SAE są ciągła adaptacja procesu optymalizacyjnego oraz osiągnięcie synergii między różnymi konfiguracjami AEDP, dzięki czemu SAE powinny wypadać dużo lepiej od AE w długotrwałych procesach optymalizacyjnych a także mieć większą zdolność do przeszukiwania przestrzeni rozwiązań poza lokalnym optimum. Dodatkowo skuteczność SAE nie jest tak mocno uzależniona od konfiguracji jak ma to miejsce w przypadku AE, dlatego konfiguracje SAE mają szansę na okazanie się powszechnie przenaszalnymi dla różnych rodzajów problemów optymalizacyjnych.

Należy zaznaczyć, że wszelkie badania przeprowadzone w ramach tej pracy miały charakter powierzchniowy i do potwierdzenia wszystkich stawianych tez potrzebne są dogłębne i szczegółowe badania.

Literatura

- [1] Ewa Figielska: *Algorytmy ewolucyjne i ich zastosowania*, Zeszyty naukowe, 2006, s. 81-89.
- [2] K. Zielinski, D. Peters, R. Laur: *Stopping Criteria for Single-Objective Optimization*, W *Proceedings of the Third International Conference on Computational Intelligence, Robotics and Autonomous Systems*, Singapore, 2005.
- [3] K. Deb: *Multi-objective optimization using evolutionary algorithms*, 2001, s. 1-3 i 81-84.
- [4] D. Dasgupta, Z. Michalewicz: *Evolutionary Algorithms – An Overview*, Rozdział w *Evolutionary Algorithms in Engineering Applications*, Springer, Berlin, 1997, s. 3-28.
- [5] Radosław Winiczenko: *Algorytmy genetyczne i ich zastosowania*, Postępy Techniki Przetwórstwa Spożywczego, nr 1, 2008, s. 107-110.
- [6] S. T. Wierzchoń: *Sztuczne systemy immunologiczne. Teoria i zastosowania*, Rozdział 3, Akademicka Oficyna Wydawnicza EXIT, Warszawa 2001.
- [7] A.E. Eiben, J.E. Smith: *Introduction to Evolutionary Computing. Second edition*, 2003, s. 81-82.
- [8] Kenneth De Jong: *Parameter Setting in EAs: a 30 year perspective*, Rozdział w *Evolutionary Algorithms*, Springer, Berlin, 2007, s. 1-18.
- [9] Silja Meyer-Nieberg, Hans-Georg Beyer: *Self-Adaptation in Evolutionary Algorithms. Parameter Setting in Evolutionary Algorithms. Studies in Computational Intelligence Vol. 54*, Springer 2007, s. 47-75.
- [10] Stanisław Krenich: *Self-adaptive Evolutionary Algorithm for Single Criteria Optimization*, Rozdział w *EVEOLVE 2011 – A Bridge between Probability, Set Oriented Numerics and Evolutionary Computetation*. ISBN 978-2-87971-106-5 ISSN 2222-9434. Paper 15, 2011, s. 56-60.
- [11] D. E. Goldberg: *Algorytmy genetyczne i ich zastosowania*, WNT 1995, s. 38-39 i 122-136.

Summary

In this thesis there is described a study about Self-adaptive Evolutionary Algorithms (acronym SEA) that are new type of Evolutionary Algorithm (acronym EA) that are evolving and adapting during the optimization process. The aim of this research was to assess efficiency of SEA in discrete and nonlinear single-objective optimization. In order to achieve this goals, there was created a model of Self-adaptive Evolutionary Algorithm and computer program for performing and observing optimization with Self-adaptive Evolutionary Algorithm, classic Evolutionary Algorithm and Random Algorithm. Then, this computer program was used to execute optimization processes for a few example problems with these algorithms. Basing on received results, there was made an analysis of SAE activity and a comparison of algorithms efficiency. Final part of this study contains conclusions that were formed basing on previously described results.

One of major steps in this study was preparation of the Self-adaptive Evolutionary Algorithm model. In this thesis, it was proposed to consider SEA as composition of two types of Evolutionary Algorithms with different purposes:

- Evolutionary Algorithm Master (acronym EAM) – This Evolutionary Algorithm is unique in SEA and it remains unchanged during SEA optimization process. Its duty is to monitor Evolutionary Algorithm Slaves and optimize theirs settings during SEA optimization. In other words in it works as the server that is meant to improve and adjust Evolutionary Algorithm Slaves in following iterations of SEA so the algorithm continue to effectively search for better solutions. Having that in mind, EAM are responsible for self-adaptation of the SEA.
- Evolutionary Algorithm Slave (acronym EAS) – There are many Evolutionary Algorithms of this type in SEA as they are created and adjusted by EAM in following iteration of SEA. EAS are meant to search for possibly the best solution of numeric problem that is essence of the optimization process. At the same time, they are assessed by EAM and basing on the result of this assessment, their settings are evolving in order to improve all EAS efficiency.

Following major part of this was creating a design and developing computer program that was used during the research. This program was releases as Python package on webpage <https://github.com/mdabrowski1990/optimization> as open-source project so it can be used in future studies.

Crucial part is of this research is chapter 5 in which comparison experiments are performed and its results are presented and analysed. In the comparison, there were used 3 optimization problems:

- De Jong function F1 – defined and described in chapter 5.1.1
- De Jong function F3 – defined and described in chapter 5.1.2
- Compression spring dimensions – defined and described in chapter 5.1.3

All the results that were received in experiments are attached to this thesis.

Last but no least is a chapter in which summary and conclusions are formed. Basing on the results of performed experiments, Self-adaptive Evolutionary Algorithm looked like the most consistent and efficient algorithm. This leads to conclusion that SEA generally achieves better

results than classic Evolutionary Algorithm when both algorithms settings are not perfectly configured for given type of the solving problem as it had place in the performed experiments. It is also worth mentioning that SEA were generally achieving the best results when Evolutionary Algorithm Master were forming Evolutionary Algorithm Slave population with diversity rather than homogeneous population. This may mean that SEA might have potential to achieve better results than perfectly configured AE due to achieving synergy between different AES types and additionally it has ability to change during different optimization phases (it is possible that other way of performing search is optimal during early phases of optimization process than during latter). To prove all these hypothesis further experiments and deeper analysis must be performed.