



Politechnika
Wrocławska

POLITECHNIKA WROCŁAWSKA
WYDZIAŁ INFORMATYKI I ZARZĄDZANIA
INSTYTUT INFORMATYKI STOSOWANEJ
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław

Rozprawa doktorska

**ROZMYTE KIEROWANIE EWOLUCJĄ OSOBNIKÓW
W ALGORYTMACH GENETYCZNYCH**

Krzysztof Pytel

Promotor: dr hab. inż. Jerzy Kisilewicz, prof. nadzw. Politechniki Wrocławskiej

Wrocław 2005

Spis treści

Spis treści.....	1
Spis ilustracji.....	3
Spis tabel.....	5
Terminologia stosowana w algorytmach genetycznych	7
Terminologia zbiorów rozmytych	9
Przyjęte w pracy oznaczenia dla algorytmów genetycznych	10
Przyjęte w pracy oznaczenia dla systemów wykorzystujących logikę rozmytą	14
1. Wstęp	16
1.1. Analiza wyników dotychczasowych prac	18
1.2. Wykorzystanie wiedzy nabytej we wcześniejszych pokoleniach.....	31
1.3. Cel pracy	32
1.4. Zakres pracy	34
1.5. Plan pracy	36
2. Zasada działania algorytmu genetycznego i sterownika rozmytego.....	38
2.1. Elementarny algorytm genetyczny.....	38
2.1.1. Wprowadzenie.....	38
2.1.2. Budowa elementarnego algorytmu genetycznego	38
2.2. Budowa sterownika rozmytego.....	40
2.3. Propozycje zmian w elementarnym algorytmie genetycznym	41
3. Teoretyczne podstawy działania zmodyfikowanego algorytmu genetycznego	42
3.1. Biologiczna inspiracja kierowania ewolucją osobników.....	42
3.2. Definicje związane z kierowaniem ewolucją osobników	43
3.3. Wpływ reprodukcji na oczekiwaną liczbę reprezentantów schematów	49
3.3.1. Wpływ współczynnika adaptacji prawdopodobieństwa selekcji	51
3.4. Wpływ krzyżowania osobników na oczekiwaną liczbę reprezentantów schematów	56
3.5. Wpływ mutacji osobników na oczekiwaną liczbę reprezentantów schematów	57
3.5.1. Wpływ współczynnika adaptacji prawdopodobieństwa mutacji.....	57
3.6. Dowód zmodyfikowanego twierdzenia o schematach	63
3.7. Dowód zbieżności zmodyfikowanego algorytmu genetycznego.	71
3.8. Analiza nakładu obliczeniowego adaptacji prawdopodobieństwa selekcji	74
3.8.1. Analiza nakładu obliczeniowego procedury $lr(\Delta F_{pop}, W_i)$	75
3.8.2. Analiza nakładu obliczeniowego procedury rozmyj(wartosc).....	76
3.8.3. Analiza nakładu obliczeniowego procedury oblicz_przeslanki().....	77
3.8.4. Analiza nakładu obliczeniowego procedury wyostrz().....	78
4. Konstrukcja zmodyfikowanego algorytmu genetycznego.....	80
4.1. Modyfikacja prawdopodobieństwa selekcji.....	80
4.1.1. Koncepcja sterownika rozmytego realizującego adaptację prawdopodobieństwa selekcji	82
4.2. Modyfikacja prawdopodobieństwa mutacji	85
4.2.1. Koncepcja sterownika rozmytego realizującego adaptację prawdopodobieństwa mutacji	86
4.3. Modyfikacja prawdopodobieństwa selekcji i mutacji	90
5. Symulacje działania zmodyfikowanych algorytmów genetycznych	92
5.1. Weryfikacja skuteczności zmodyfikowanego algorytmu genetycznego	98
5.1.1. Liczba osobników wybieranych do puli rodzicielskiej	99
5.1.2. Liczba mutacji genów w algorytmach genetycznych	102
5.1.3. Badanie efektywności zmodyfikowanych algorytmów genetycznych.....	104
5.1.4. Wnioski z eksperymentów.....	108
5.2. Testowanie algorytmu w rozwiązywaniu standardowych problemów optymalizacyjnych.....	108
5.2.1. Sprawdzenie działania algorytmu w rozwiązywaniu problemu komiwojażera.....	108
5.2.2. Sprawdzenie działania algorytmu na przykładzie optymalizacji funkcji	111
5.2.3. Sprawdzenie działania algorytmu na przykładzie rozwiązywania zero-jedynkowego problemu plecakowego.....	114

5.2.4. Wnioski z eksperymentów.....	118
5.3. Porównanie skuteczności zmodyfikowanego algorytmu genetycznego i innych metod optymalizacyjnych.....	118
5.3.1. Wnioski z eksperymentu	122
5.4. Działanie sterownika rozmytego przy różnych parametrach algorytmu genetycznego.....	122
5.4.1. Badanie wartości funkcji optymalizowanej	123
5.4.2. Badanie liczby pokoleń potrzebnych, aby algorytm osiągnął założoną wartość funkcji optymalizowanej	127
5.4.3. Wnioski z eksperymentów.....	128
5.5. Zdolność algorytmu do wyszukiwania rozwiązania optymalnego.....	128
5.5.1. Wnioski z eksperymentu	131
5.6. Zdolność algorytmu do wychodzenia z optimum lokalnego.....	131
5.6.4. Wnioski z eksperymentów.....	136
5.7. Sprawdzenie skuteczności zmodyfikowanego algorytmu genetycznego w rozwiązywaniu skomplikowanych zadań.....	137
6. Zastosowanie algorytmów genetycznych do optymalizacji sieci komputerowych.....	141
6.1. Zadanie minimalizacji średniego opóźnienia pakietu	141
7. Wnioski i podsumowanie pracy	146
8. Literatura.....	152

Spis ilustracji

Rys.2.1. Schemat blokowy elementarnego algorytmu genetycznego.....	39
Rys.2.2. Schemat blokowy układu regulacji	40
Rys.2.3. Schemat blokowy klasycznego sterownika rozmytego	40
Rys. 3.1. Schemat ideowy sterownika rozmytego wykorzystywanego do adaptacji prawdopodobieństwa selekcji.....	52
Rys. 3.2. Schemat ideowy sterownika rozmytego wykorzystywanego do adaptacji prawdopodobieństwa mutacji.....	58
Rys. 3.3. Wpływ sterownika rozmytego na oczekiwaną liczbę osobników wybieranych do puli rodzicielskiej w algorytmach z modyfikacją prawdopodobieństwa selekcji	67
Rys. 3.4. Wpływ sterownika rozmytego na oczekiwaną liczbę mutacji osobników w algorytmach z modyfikacją prawdopodobieństwa mutacji	71
Rys.4.1. Schemat blokowy zmodyfikowanego algorytmu genetycznego z adaptacją prawdopodobieństwa selekcji.....	81
Rys.4.2. Schemat sterownika rozmytego wykorzystywanego do modyfikowania prawdopodobieństwa selekcji	82
Rys.4.3. Funkcja przynależności $\Delta F_{pop} = F_{pop}^{(T)} - F_{pop}^{(T-1)}$ w sterowniku do adaptacji prawdopodobieństwa selekcji.....	84
Rys.4.4. Funkcja przynależności $W_i = F_i^{(T)} - F_{pop}^{(T)}$ w sterowniku do adaptacji prawdopodobieństwa selekcji	84
Rys.4.5. Funkcja przynależności $\mu(wps_i)$ w sterowniku do adaptacji prawdopodobieństwa selekcji	85
Rys.4.6. Zmodyfikowany algorytm genetyczny z adaptacją prawdopodobieństwa mutacji	86
Rys.4.7. Schemat sterownika rozmytego wykorzystywanego do modyfikowania prawdopodobieństwa mutacji.....	87
Rys.4.8. Funkcja przynależności $\Delta F_{max} = F_{pop}^{(T)} - F_{pop}^{(T-1)}$ w sterowniku do adaptacji prawdopodobieństwa mutacji.....	89
Rys.4.9. Funkcja przynależności $W_i = F_i^{(T)} - F_{pop}^{(T)}$ w sterowniku do adaptacji prawdopodobieństwa mutacji.....	89
Rys.4.10. Funkcja przynależności wpm_i w sterowniku do adaptacji prawdopodobieństwa mutacji ..	89
Rys. 4.11. Zmodyfikowany algorytm genetyczny z adaptacją prawdopodobieństwa selekcji i mutacji	91
Rys. 5.1. Przebieg funkcji testowej Q_7	97
Rys. 5.2. Różnica w liczbie osobników w algorytmie elementarnym i zmodyfikowanym zakwalifikowanych do puli rodzicielskiej w optymalizacji funkcji F1	100
Rys. 5.3. Różnica w liczbie osobników w algorytmie elementarnym i zmodyfikowanym zakwalifikowanych do puli rodzicielskiej w optymalizacji funkcji F2.....	101
Rys. 5.4. Różnica w liczbie osobników w algorytmie elementarnym i zmodyfikowanym zakwalifikowanych do puli rodzicielskiej w problemie komiwojażera	101
Rys. 5.5. Różnica w liczbie mutacji genów w algorytmie zmodyfikowanym i elementarnym w zadaniu optymalizacji funkcji F1	102
Rys. 5.6. Różnica w liczbie mutacji genów w algorytmie zmodyfikowanym i elementarnym w zadaniu optymalizacji funkcji F2.....	103
Rys. 5.7. Różnica w liczbie mutacji genów w algorytmie zmodyfikowanym i elementarnym w problemie komiwojażera	103
Rys. 5.8. Efektywność „on-line” algorytmów dla funkcji F1	105
Rys. 5.9. Efektywność „on-line” algorytmów dla funkcji F2.....	106
Rys. 5.10. Efektywność „on-line” algorytmów dla problemu komiwojażera.....	106
Rys. 5.11. Efektywność „off-line” algorytmów dla funkcji F1	106
Rys. 5.12. Efektywność „off-line” algorytmów dla funkcji F2.....	107

Rys. 5.13. Efektywność „off-line” algorytmów dla problemu komiwojażera.....	107
Rys. 5.14. Średnie czasy rozwiązywania problemu komiwojażera przez algorytmy	110
Rys. 5.15. Średnia liczba wywołań funkcji obliczającej przystosowanie potrzebnych do uzyskania założonej wartości funkcji optymalizowanej.....	110
Rys. 5.16. Średnie czasy działania poszczególnych algorytmów	113
Rys. 5.17. Średnia liczba wywołań funkcji obliczającej przystosowanie potrzebnych do uzyskania założonej wartości funkcji optymalizowanej dla poszczególnych algorytmów.....	113
Rys. 5.18. Średnie czasy rozwiązywania problemów plecakowych przez algorytmy	116
Rys. 5.19. Średnia liczba wywołań funkcji obliczającej przystosowanie potrzebnych do uzyskania założonej wartości funkcji optymalizowanej.....	117
Rys. 5.20. Średnie wartości funkcji optymalizowanej uzyskane przez różne metody optymalizacji dla zadania gr24	120
Rys. 5.21. Średnie wartości funkcji optymalizowanej uzyskane przez różne metody optymalizacji dla zadania fri26.....	121
Rys. 5.22. Średnie wartości funkcji optymalizowanej uzyskane przez różne metody optymalizacji dla zadania bays29	121
Rys. 5.23. Funkcja przystosowania najlepszego osobnika w elementarnym algorytmie genetycznym	124
Rys. 5.24. Funkcja przystosowania najlepszego osobnika w algorytmie z modyfikacją selekcji.....	125
Rys. 5.25. Funkcja przystosowania najlepszego osobnika w algorytmie z modyfikacją mutacji.....	125
Rys. 5.26. Funkcja przystosowania najlepszego osobnika w algorytmie genetycznym z modyfikacją selekcji i mutacji.....	126
Rys. 5.27. Liczba pokoleń potrzebna, aby algorytm osiągnął założoną wartość w zadaniu optymalizacji funkcji.....	127
Rys. 5.28. Czas działania potrzebny do osiągnięcia wartości optymalnej.....	130
Rys. 5.29. Liczba wywołań funkcji potrzebna do osiągnięcia wartości optymalnej	130
Rys. 5.30. Liczba pokoleń potrzebna, aby średnia wartość funkcji przystosowania osobników uzyskała założoną wartość	133
Rys. 5.31. Liczba pokoleń potrzebna, aby wartość funkcji przystosowania przynajmniej połowy osobników przekroczyła założoną wartość.....	134
Rys. 5.32. Liczba pokoleń, po których następowała zmiana wartości najlepszego osobnika.....	136
Rys. 5.33. Liczba wywołań funkcji obliczającej przystosowanie osobnika w zadaniu Q7	138
Rys. 6.1. Czasy działania algorytmów potrzebne do osiągnięcia założonej wartości funkcji optymalizowanej	144
Rys. 6.2. Średnia liczba wywołań funkcji obliczającej przystosowanie, potrzebna do uzyskania założonej wartości funkcji optymalizowanej.....	144

Spis tabel

Tabela 3.1. Przynależności ΔF_{pop} , W_i i y_{jk} do poszczególnych przedziałów w sterowniku wykorzystywanym do adaptacji prawdopodobieństwa selekcji.....	52
Tabela 3.2. Wartości rozmyte współczynników wps_i dla sterownika do adaptacji prawdopodobieństwa selekcji	53
Tabela 3.3. Przynależności ΔF , W_i i y_{jk} do poszczególnych przedziałów w sterowniku wykorzystywanym do adaptacji prawdopodobieństwa mutacji.....	58
Tabela 3.4. Baza reguł dla sterownika realizującego adaptację prawdopodobieństwa mutacji w sterowniku wykorzystywanym do adaptacji prawdopodobieństwa mutacji	59
Tabela 3.5. Pseudokod procedury select().....	74
Tabela 3.6. Pseudokod procedury $lr(\Delta F_{pop}, W_i)$	75
Tabela 3.7. Pseudokod procedury rozmyj(wartosc)	76
Tabela 3.8. Pseudokod procedury oblicz_przeslanki().....	77
Tabela 3.9. Pseudokod procedury wyostrz()	78
Tabela 4.1. Baza reguł dla sterownika realizującego adaptację prawdopodobieństwa selekcji.....	82
Tabela 4.2. Baza reguł dla sterownika realizującego adaptację prawdopodobieństwa mutacji.....	87
Tabela 5.1. Liczba osobników wybranych do puli rodzicielskiej przy danej liczbie pokoleń.....	100
Tabela 5.2. Liczba mutacji osobników w algorytmie z modyfikacją mutacji i algorytmie elementarnym	102
Tabela 5.3. Efektywność „on-line” algorytmów	105
Tabela 5.4. Efektywność „off-line” algorytmów.....	105
Tabela 5.5. Zadania testowe stosowane w eksperymentach z problemem komiwojażera.....	109
Tabela 5.6. Średnie czasy rozwiązania problemu komiwojażera przez poszczególne algorytmy	109
Tabela 5.7. Średnie liczby wywołań funkcji obliczającej przystosowanie potrzebnych do rozwiązania problemu komiwojażera	109
Tabela 5.8. Zadania testowe wykorzystywane w eksperymentach z optymalizacją funkcji.....	112
Tabela 5.9. Wyniki działania algorytmów rozwiązujących problem optymalizacji funkcji.....	112
Tabela 5.10. Wyniki działania algorytmów rozwiązujących problem optymalizacji funkcji.....	113
Tabela 5.11. Podstawowe informacje o zadaniach testowych dla problemu plecakowego	115
Tabela 5.12. Średnie czasy rozwiązywania problemów plecakowych przez algorytmy	116
Tabela 5.13. Średnie liczby wywołań funkcji obliczającej przystosowanie potrzebnych do rozwiązania problemu plecakowego dla poszczególnych algorytmów.....	116
Tabela 5.14. Parametry algorytmów w eksperymencie z problemem komiwojażera.....	119
Tabela 5.15. Wartości funkcji optymalizowanej uzyskane przez algorytmy genetyczne i inne metody optymalizacyjne dla zadania gr24	119
Tabela 5.16. Wartości funkcji optymalizowanej uzyskane przez algorytmy genetyczne i inne metody optymalizacyjne dla zadania fri26.....	120
Tabela 5.17. Wartości funkcji optymalizowanej uzyskane przez algorytmy genetyczne i inne metody optymalizacyjne dla zadania bays29	120
Tabela 5.18. Wartości funkcji optymalizowanej uzyskane przez algorytm elementarny w zadaniu optymalizacji funkcji M1	123
Tabela 5.19. Wartości funkcji optymalizowanej uzyskane przez algorytm z modyfikacją prawdopodobieństwa selekcji w zadaniu optymalizacji funkcji M1.....	123
Tabela 5.20. Wartości funkcji optymalizowanej uzyskane przez algorytm z modyfikacją prawdopodobieństwa mutacji w zadaniu optymalizacji funkcji M1	124
Tabela 5.21. Wartości funkcji optymalizowanej uzyskane przez algorytm z modyfikacją prawdopodobieństwa selekcji i mutacji w zadaniu optymalizacji funkcji M1	124
Tabela 5.22. Liczba pokoleń potrzebna do osiągnięcia założonej wartości funkcji przystosowania w zadaniu optymalizacji funkcji.....	127
Tabela 5.23. Czas działania i liczba pokoleń potrzebna do uzyskania rozwiązania optymalnego.....	129
Tabela 5.24. Parametry algorytmów stosowane do optymalizacji funkcji eksponencjalnych	132

Tabela 5.25. Średnia liczba pokoleń potrzebnych, aby algorytm opuścił lokalne optimum.....	132
Tabela 5.26. Średnia liczba pokoleń potrzebnych, aby algorytm opuścił lokalne optimum.....	134
Tabela 5.27. Średnia liczba pokoleń, po których następowała zmiana wartości najlepszego osobnika	135
Tabela 5.28. Liczba wywołań funkcji obliczającej przystosowanie osobnika w zadaniu Q7.....	138
Tabela 6.1. Charakterystyka optymalizowanych sieci	143
Tabela 6.2. Czas i liczba wywołań funkcji potrzebne do uzyskania założonej wartości funkcji optymalizowanej	144

Terminologia stosowana w algorytmach genetycznych

Do opisu działania algorytmów genetycznych używane będą pojęcia [Rut97b][Gwi98]:

- allel - to wartość danego genu, określana również jako wartość cechy lub wariant cechy,
- algorytm genetyczny jest to probabilistyczny algorytm przeszukiwania przestrzeni rozwiązań problemu optymalizacyjnego P_o ,
- chromosom (łańcuch, ciąg kodowy) - to uporządkowany ciąg genów, będący rozwiązaniem problemu optymalizacyjnego, taki że
$$ch_i = (ch_i^1, \dots, ch_i^l) \in D(P_o),$$
- długość definiująca schematu - jest to odległość między pierwszą a ostatnią pozycją ustaloną schematu,
- dominacja - punkt $P1$ dominuje $P2$ ($P2$ jest zdominowany przez $P1$), gdy punkt $P1$ nie jest pod żadnym względem gorszy od $P2$ i istnieje kryterium, pod względem którego $P1$ jest lepszy od $P2$,
- fenotyp - jest zestawem wartości odpowiadających danemu genotypowi,
- funkcja przystosowania $U(ch_i)$ - wartość normy z funkcji optymalizowanej dla chromosomu ch_i ,
- gen (cecha, znak, detektor) - stanowi pojedynczy element genotypu, w szczególności chromosomu,
- genotyp (struktura) - jest to zespół chromosomów danego osobnika,
- krzyżowanie - procedura tworzenia dwóch nowych osobników, na podstawie dwóch losowo wybranych z puli rodzicielskiej - polega ona na wymianie części kodu rodziców, a utworzone w wyniku krzyżowania osobniki są następnie poddawane mutacji, by w końcu znaleźć się w nowo tworzonej populacji,
- locus (pozycja) - wskazuje miejsce położenia danego genu w łańcuchu, czyli chromosomie,

$$U(ch_i) = \|F(ch_i)\| \quad \text{gdzie} \quad U(ch_i) \geq 0 \quad \forall ch_i \in P(t),$$

Norma $\|*\|$ definiowana jest heurystycznie i w najprostszym przypadku

$$\|F(ch_i)\| = F(ch_i),$$

- mutacja - przypadkowa dobrana losowo z niewielkim prawdopodobieństwem zmiana wartości elementu kodu osobnika (może ona doprowadzić do odkrycia nowych, użytecznych cech),

- operator krzyżowania K_k – jest to przekształcenie

$$K_k : D(P_o) \times D(P_o) \rightarrow D(P_o) \times D(P_o)$$

dokonyjące z określonym prawdopodobieństwem wymiany genów pomiędzy chromosomami ch_i i ch_j

$$K_k(ch_i^t, ch_j^t) = (ch_i^{t+1}, ch_j^{t+1}),$$

- operator mutacji M_k – jest to przekształcenie

$$M_k : D(P_o) \rightarrow D(P_o)$$

dokonyjące z określonym prawdopodobieństwem losowej zmiany genów chromosomu

$$M_k(ch_i^t) = ch_i^{t+1},$$

- osobnik - odpowiednio zakodowany w postaci chromosomów zestaw parametrów zadania (rozwiązanie, punkt przestrzeni poszukiwań) dla rozważanego problemu (może być wyrażony w postaci ciągu bitów, jak też ciągu elementów innego typu),
- populacja - zbiór osobników o określonej liczebności występujących w danym momencie działania algorytmu, taki że

$$P(t) = (ch_1, \dots, ch_N) \quad P(t) \subset D(P_o),$$

- problem optymalizacyjny P_o – jest to zadanie polegające na znalezieniu rozwiązania optymalnego dla danego zestawu kryteriów,
- rzęd schematu - liczba pozycji ustalonych w schemacie (nieoznaczonych za pomocą symbolu „*” - nieistotne), czyli jest to różnica pomiędzy długością chromosomu a liczbą symboli „*” w schemacie,
- schemat - zestaw cech (elementów kodu osobnika) charakteryzujący grupę osobników, zawierający pewną liczbę cech o ustalonych wartościach (allelach) oraz pewną liczbę cech o wartościach dowolnych (schematy mają bezpośredni wpływ na stopień przystosowania osobnika, co pozwala dzielić schematy na bardziej i mniej wartościowe),
- selekcja - wybór do puli rodzicielskiej zbioru osobników, które posłużą do wygenerowania kolejnej populacji (poprzez krzyżowanie oraz mutację),
- zbiór dopuszczalnych rozwiązań problemu optymalizacyjnego $D(P_o)$ jest to zbiór wszystkich możliwych rozwiązań problemu optymalizacyjnego spełniających określone w problemie warunki.

Terminologia zbiorów rozmytych

Człowiek do oceny zjawisk wykorzystuje powszechnie ocenę jakościową np. wzrost niski, średni, wysoki. Pojęcia niski, średni, wysoki są pojęciami subiektywnymi i rozmytymi.

Do opisu zbiorów rozmytych wykorzystywać będziemy następujące pojęcia [Pie99]:

- funkcja przynależności (membership function) – jest to funkcja, która przyporządkowuje stopień przynależności każdemu elementowi danej zmiennej,
- moc przestrzeni numerycznej – jest to liczba jej elementów, np. $\|X\| = n$,
- moc zbioru rozmytego – jest to liczba par $(\mu_A(x), x)$ występujących w zbiorze,
- pusty zbiór rozmyty – zbiór A , którego funkcja przynależności $\mu_A(x)$ posiada wartość zero dla wszystkich elementów przestrzeni rozważań, oznaczany jest znakiem $\emptyset$
 $\emptyset : \mu_{\emptyset}(x) = 0, \forall x \in X$,
- przestrzeń lingwistyczna zmiennej (linguistic term-set) – jest to zbiór wszystkich wartości lingwistycznych stosowanych do opisu danej zmiennej lingwistycznej (przestrzeń lingwistyczna będzie oznaczana dużą literą łacińską, np.
 $X_L = \{\text{niska, średnia, wysoka}\} = \{x_{L1}, x_{L2}, x_{L3}\}$),
- przestrzeń numeryczna zmiennej (universe of discourse) – jest to zbiór wszystkich wartości numerycznych zmiennej, jakie może ona przyjmować, oraz innych istotnych dla danego problemu. Przestrzeń numeryczna może być ciągła lub dyskretna. Przestrzeń numeryczna będzie oznaczana dużą literą łacińską np.
 $X = \{x\}$ – przestrzeń ciągła (nieskończeniowymiarowa),
 $X = \{x_1, x_2, x_3, \dots, x_n\}$ – przestrzeń dyskretna (skończeniowymiarowa),
- stopień przynależności (grade of membership) – jest to wartość z zakresu $\langle 0, 1 \rangle$ określająca, w jakim stopniu element należy do zbioru rozmytego,
- uniwersalny zbiór rozmyty – zbiór A , którego funkcja przynależności $\mu_A(x)$ posiada wartość jeden dla wszystkich elementów przestrzeni rozważań, oznaczany jest znakiem U
 $U : \mu_U(x) = 1, \forall x \in X$,
- wartość lingwistyczna (linguistic value) – jest to słowna ocena zmiennej lingwistycznej np. mała, duża, niska, wysoka,
- zbiór rozmyty (fuzzy set) – jest to zbiór par:
 $A = \{(\mu_A^*(x), x)\}, \forall x \in X$

gdzie μ_A jest funkcją przynależności zbioru rozmytego A , która każdemu elementowi $x \in X$ przypisuje stopień przynależności $\mu_A^*(x)$ do zbioru rozmytego A ,

- zmienna lingwistyczna (linguistic variable) – jest to ta wielkość, którą zamierzamy oceniać np. prędkość, temperatura, wzrost.

Przyjęte w pracy oznaczenia dla algorytmów genetycznych

- ch_i – chromosom,
- $eval(z)$ - wartość przystosowania osobnika,
- $Eval(P)$ - funkcja oceniająca dla populacji P ,
- f – funkcja optymalizowana,
- $F(ch_i)$ – funkcja przystosowania,
- $F(S)$ – funkcja przystosowania schematu S ,
- F_{skal} - skalowana funkcja przystosowania osobnika,
- $\bar{F}$ - średnia funkcja przystosowania populacji,
- G_m – liczba mutacji genów osobników, których funkcja przystosowania jest niższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w elementarnym algorytmie genetycznym,
- G_m' – liczba mutacji genów osobników, których funkcja przystosowania jest niższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w zmodyfikowanym algorytmie genetycznym,
- G_s – liczba osobników w danym pokoleniu elementarnego algorytmu genetycznego, których funkcja przystosowania jest niższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,
- G_s' – liczba osobników w danym pokoleniu zmodyfikowanego algorytmu genetycznego, których funkcja przystosowania jest niższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,
- i_{max} - maksymalna liczba pokoleń w algorytmie genetycznym,
- J_m – liczba mutacji genów osobników, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w elementarnym algorytmie genetycznym,

- J_m' – liczba mutacji genów osobników, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w zmodyfikowanym algorytmie genetycznym,
- J_s – liczba osobników w danym pokoleniu elementarnego algorytmu genetycznego, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,
- J_s' – liczba osobników w danym pokoleniu zmodyfikowanego algorytmu genetycznego, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,
- l – długość ciągu kodowego osobnika,
- l_k – punkt krzyżowania,
- L_m – liczba mutacji genów osobników, których funkcja przystosowania jest wyższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w elementarnym algorytmie genetycznym,
- L_m' – liczba mutacji genów osobników, których funkcja przystosowania jest wyższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w zmodyfikowanym algorytmie genetycznym,
- L_s – liczba osobników w danym pokoleniu elementarnego algorytmu genetycznego, których funkcja przystosowania jest wyższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,
- L_s' – liczba osobników w danym pokoleniu zmodyfikowanego algorytmu genetycznego, których funkcja przystosowania jest wyższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,
- $m(S,t)$ – liczba reprezentantów schematu S w chwili t w algorytmie elementarnym,
- $m'(S,t)$ – liczba reprezentantów schematu S w chwili t w algorytmie zmodyfikowanym,
- M – liczba mutacji genów wszystkich osobników w populacji w elementarnym algorytmie genetycznym,
- Max – ograniczenie górne funkcji przystosowania osobnika $eval$ w rozważanej dziedzinie,
- M' – liczba mutacji genów wszystkich osobników w populacji w zmodyfikowanym algorytmie genetycznym,
- N – liczebność populacji w elementarnym algorytmie genetycznym,
- N_m – liczba miast w problemie komiwojażera,

- N' - liczebność populacji w zmodyfikowanym algorytmie genetycznym,
- $o(S)$ - rząd schematu (liczba pozycji ustalonych w schemacie),
- p_c – prawdopodobieństwo krzyżowania,
- p_d - prawdopodobieństwo zniszczenia schematu w procesie krzyżowania osobników,
- p_{ds} - prawdopodobieństwo przeżycia schematu w procesie krzyżowania osobników,
- p_{dm} - prawdopodobieństwo przeżycia schematu w procesie mutacji osobników w algorytmie elementarnym,
- p'_{dm} - prawdopodobieństwo przeżycia schematu w procesie mutacji osobników w algorytmie zmodyfikowanym,
- p_e - częstotliwość stosowania operatora eksploatacji,
- p_m – prawdopodobieństwo mutacji,
- $p_m(ch_i)$ – prawdopodobieństwo mutacji chromosomu ch_i ,
- $p'_m(ch_i)$ – zmodyfikowane prawdopodobieństwo mutacji chromosomu ch_i ,
- p_s – prawdopodobieństwo selekcji,
- $p_s(ch_i)$ – prawdopodobieństwo selekcji chromosomu ch_i ,
- $p'_s(ch_i)$ – zmodyfikowane prawdopodobieństwo selekcji chromosomu ch_i ,
- $p(\Delta)$ - prawdopodobieństwo zajścia zdarzenia Δ ,
- P – populacja osobników,
- $P[i]$ – wartość i-tego przedmiotu,
- $P(t)$ – populacja osobników w chwili t ,
- P^* - jednoznaczny punkt stały dla wszystkich populacji algorytmu genetycznego,
- S – schemat,
- $W[i]$ – waga i-tego przedmiotu,
- Z – zbiór punktów przestrzeni metrycznej,
- $\langle Z, \delta \rangle$ - przestrzeń metryczna,
- δ - odległość punktów w przestrzeni metrycznej,
- $\delta(S)$ - długość definiująca schematu,
- ΔG_m - globalna różnica między liczbą mutacji genów osobników, których funkcja przystosowania jest niższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w algorytmie zmodyfikowanym a liczbą mutacji genów tych osobników algorytmie elementarnym,

- $(\Delta G_m)_i$ - różnica między liczbą mutacji genów osobników, których funkcja przystosowania jest niższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w algorytmie zmodyfikowanym a liczbą mutacji genów tych osobników algorytmie elementarnym, w pokoleniu i ,
- ΔG_s - globalna różnica między liczbą osobników, których funkcja przystosowania jest niższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w algorytmie zmodyfikowanym a liczbą tych osobników algorytmie elementarnym,
- $(\Delta G_s)_i$ - różnica między liczbą osobników, których funkcja przystosowania jest niższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w algorytmie zmodyfikowanym a liczbą tych osobników algorytmie elementarnym, w pokoleniu i ,
- ΔJ_m - globalna różnica między liczbą mutacji genów osobników, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w algorytmie zmodyfikowanym a liczbą mutacji genów tych osobników algorytmie elementarnym,
- $(\Delta J_m)_i$ - różnica między liczbą mutacji genów osobników, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w algorytmie zmodyfikowanym a liczbą mutacji genów tych osobników algorytmie elementarnym, w pokoleniu i ,
- ΔJ_s - globalna różnica między liczbą osobników, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w algorytmie zmodyfikowanym a liczbą tych osobników algorytmie elementarnym,
- $(\Delta J_s)_i$ - różnica między liczbą osobników, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w algorytmie zmodyfikowanym a liczbą tych osobników algorytmie elementarnym, w pokoleniu i ,
- ΔL_m - globalna różnica między liczbą mutacji genów osobników, których funkcja przystosowania jest wyższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w algorytmie zmodyfikowanym a liczbą mutacji genów tych osobników algorytmie elementarnym,
- $(\Delta L_m)_i$ - różnica między liczbą mutacji genów osobników, których funkcja przystosowania jest wyższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w algorytmie zmodyfikowanym a liczbą mutacji genów tych osobników algorytmie elementarnym, w pokoleniu i ,

- ΔL_s - globalna różnica między liczbą osobników, których funkcja przystosowania jest wyższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w algorytmie zmodyfikowanym a liczbą tych osobników algorytmie elementarnym,
- $(\Delta L_s)_i$ - różnica między liczbą osobników, których funkcja przystosowania jest wyższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, w algorytmie zmodyfikowanym a liczbą tych osobników w algorytmie elementarnym, w pokoleniu i ,
- $\eta_{\min}$ - presja selekcyjna,
- η_{mut}^G - względna skuteczność modyfikacji prawdopodobieństwa mutacji osobników źle przystosowanych,
- η_{mut} - względna skuteczność modyfikacji prawdopodobieństwa mutacji,
- η_{sel} - względna skuteczność modyfikacji prawdopodobieństwa selekcji,
- η_{mut}^L - względna skuteczność modyfikacji prawdopodobieństwa mutacji osobników dobrze przystosowanych,
- $v(ch_i)$ – wycinek koła ruletki odpowiadający chromosomowi ch_i .

Przyjęte w pracy oznaczenia dla systemów wykorzystujących logikę rozmytą

- A_i^k, B_j^k - zbiory rozmyte,
- C – pojemność plecaka,
- $C_{\max}$ - wartość stała drogi maksymalnej,
- $F_{\max}^{(T)}$ - funkcja przystosowania najlepszego osobnika w populacji (T),
- $F_{pop}^{(T)}$ - funkcja przystosowania populacji w chwili (T),
- M^k - moment zbioru B^k ,
- N_r – liczba reguł (liczba naturalna),
- $R^{(k)}$ - reguła rozmyta,
- S^k - powierzchnia zbioru B^k ,
- W_{os} - wskaźnik stopnia przystosowania osobnika,
- $wpm(S)$ - współczynnik adaptacji prawdopodobieństwa mutacji schematu S ,
- wpm_{os} - współczynnik adaptacji prawdopodobieństwa mutacji,
- W_{pop} - wskaźnik wieku społeczeństwa,

- $wps(S)$ - współczynnik adaptacji prawdopodobieństwa selekcji schematu S ,
- wps_{os} - współczynnik adaptacji prawdopodobieństwa selekcji osobnika,
- x - wartości numeryczne zmiennej wejściowej,
- x_L - wartość lingwistyczna zmiennej,
- X - przestrzeń numeryczna zmiennej wejściowej,
- X_L - przestrzeń lingwistyczna zmiennej,
- y - wartości numeryczne zmiennej wyjściowej,
- Y - przestrzeń numeryczna zmiennej wyjściowej,
- α^k - waga reguły R^k ,
- ΔF - jakość populacji,
- ΔF_{pop} - przyrost funkcji przystosowania populacji,
- Δ_{pop} - wskaźnik zmiany liczebności populacji,
- $\mu_{A^k}(x_i)$ - funkcja przynależności zmiennej x_i do zbioru rozmytego A^k ,
- $\mu_A^*(x)$ - stopień przynależności elementu x do zbioru A ,
- $\mu_{A_i^k}^*(x_i)$ - stopień przynależności x_i do zbioru A_i^k ,
- $\mu_{R^{(k)}}$ - stopień spełnienia przesłanki,
- $\mu(\Delta F_{pop} \in \Delta F_{pop^k})$ - stopień przynależności wartości ΔF_{pop} do zbioru wartości ΔF_{pop^k} .
- $\mathfrak{R}$ - zbiór liczb rzeczywistych.

1. Wstęp

Systemy sztucznej inteligencji często przetwarzają dane i informacje w sposób podobny, jak to robią organizmy biologiczne. Do najpopularniejszych technik sztucznej inteligencji zalicza się systemy wykorzystujące logikę rozmytą [Ful] [Joh] [Kel] [Pie99] [Sla] [Zad65], sieci neuronowe [Żur96] i algorytmy ewolucyjne [Ala95] [Gol98] [Hol75] [Lan96] [Mic99] [Rut97b] [Wei].

Algorytmy genetyczne stanowią pewną grupę algorytmów ewolucyjnych, swoje istnienie zawdzięczają obserwacji i próbie naśladowania naturalnych procesów zachodzących w świecie organizmów żywych - ewolucji i związanej z nią naturalnej selekcji, występującej w populacjach żywych osobników. Algorytmy te służą do poszukiwania wyników jedynie przybliżonych, jednak w większości przypadków jest to rozwiązanie satysfakcjonujące. Algorytmy genetyczne nie operują na pojedynczych rozwiązaniach, lecz na całych ich populacjach. Rozwiązania kodowane są w struktury zwane ciągami kodowymi. Algorytm genetyczny poszukuje coraz lepszych rozwiązań, generując kolejne ich pokolenia. Pierwsze pokolenie tworzy się losowo albo wybiera w inny sposób. Dla każdego kolejnego pokolenia wybiera się pewną liczbę osobników najlepiej przystosowanych (proces selekcji lub reprodukcji). Osobniki te są kojarzone w pary (losowo), po czym następuje wymiana (wymieszanie się) materiału genetycznego rodziców, w wyniku czego otrzymujemy dwa nowe ciągi kodowe odpowiadające genotypowi dzieci. Ten fragment algorytmu nazywa się procesem krzyżowania (cross-over). Podczas przepisywania informacji genetycznej istnieje jednak ryzyko mutacji. Z pewnym prawdopodobieństwem (najczęściej niewielkim) w niektóre miejsca ciągu kodowego (geny) zostaną wstawione losowe wartości (allele). Nowo powstałe ciągi kodowe reprezentują kolejną populację. Algorytm kończy się po uzyskaniu satysfakcjonującego wyniku albo po z góry ustalonej liczbie pokoleń. Rozwiązaniem jest osobnik posiadający największą wartość funkcji przystosowania.

W otaczającym nas świecie w życiu codziennym spotkać możemy się z wieloma typami informacji. Wiedza może być przekazywana w sposób ilościowy i jakościowy. Informacje są przez nas przetwarzane w różny sposób. Do przetwarzania informacji precyzyjnych wykorzystać możemy matematykę konwencjonalną. Przykładem informacji precyzyjnych może być np.:

- temperatura wynosi 20 °C,
- wysokość 2 m,
- ciśnienie 2 MPa.

W wielu sytuacjach spotykamy się z informacjami nieprecyzyjnymi takimi jak:

- wysoka temperatura,
- niski wzrost,
- średnie ciśnienie,
- duża wysokość.

W oparciu o konwencjonalną matematykę informacji takich przetwarzać nie można, w związku z tym informacje te są nieprzydatne, a więc wiedza jest tracona. Spotyka się jednak sytuacje, w których przekazanie informacji w sposób precyzyjny jest bardzo trudne lub wręcz niemożliwe, a cała wiedza, jaką dysponujemy na temat zjawiska, składa się z informacji nieprecyzyjnych. W takich sytuacjach wykorzystać można teorię zbiorów rozmytych.

Teoria zbiorów rozmytych została zaproponowana w 1965 roku przez L.A. Zadeha [Zad65]. Głównym jej celem jest reprezentacja i przetwarzanie informacji nieprecyzyjnych, zawartych w niektórych określeniach języka naturalnego. Tego typu informacje pojawiają się często w praktyce, a zastosowanie zwykłej logiki boolowskiej do ich modelowania nie odzwierciedla postrzegania i rozumowania człowieka. Opis jakościowy jest mniej precyzyjny i zależny od osoby opisującej. Nieprecyzyjność ta nie wynika z braku wiedzy o wartości pewnej wielkości, ale właśnie z subiektywnej oceny osoby lub grupy osób. Subiektywne określanie pewnych właściwości prowadzi do rozmycia granic zbioru obiektów, do których ta właściwość się odnosi. Jest to teoria w szczególności przydatna w przypadku systemów, w których czynnik ludzki odgrywa zasadniczą rolę, takich jak medycyna, ekonomia, socjologia, teoria podejmowania decyzji, ale także w wielu dziedzinach techniki. Sposób rozumowania i interpretowania pewnych wielkości często wynika z wiedzy nabytej w sposób doświadczalny, np. w procesie regulacji wielkość przeregulowania sygnału wyjściowego rzędu 15% dla jednego projektanta może być niedopuszczalna, jednak inny uzna tę wielkość za odpowiednią. Na podejmowane decyzje wpływa specyfika procesu, jednak istnieje również dowolność interpretacji przesłanek.

W układach sterowania często spotyka się sytuacje, w których zachowanie obiektu sterowania bardzo trudno jest opisać przy pomocy modelu matematycznego. W przypadku, gdy dysponujemy wiedzą eksperta na temat reguł funkcjonowania obiektu, możemy zaprojektować układ wykorzystujący teorię zbiorów rozmytych. W większości wypadków ekspert przekazując nam wiedzę o obiekcie używa pojęć właściwych dla ludzkiego sposobu wnioskowania (dysponuje wiedzą jakościową „jak?” na temat obiektu, a nie wiedzą ilościową „ile?”). Zbudowany w oparciu o taką wiedzę system wykorzystuje zbiór rozmytych reguł zapisanych w postaci IF – THEN. Wiedza eksperta potrzebna jest w fazie projektowania.

Projektowanie polega na ustaleniu zbioru reguł rozmytych oraz funkcji przynależności zmiennych lingwistycznych na podstawie wiedzy eksperta oraz ocenie stopnia odwzorowania rzeczywistego przebiegu procesu.

Populację osobników w algorytmie genetycznym potraktować możemy jako obiekt podlegający ewolucji. Pożądana w takim przypadku byłaby możliwość wpływania na kierunki ewolucji populacji. Ponieważ na ogół wiemy, w jakim kierunku powinna zmierzać ewolucja populacji, natomiast trudno jest przedstawić matematyczny model tych zmian, zasadnym wydaje się zastosowanie systemu wykorzystującego logikę rozmytą do wyznaczania kierunku ewolucji algorytmu genetycznego.

1.1. Analiza wyników dotychczasowych prac

Zalety algorytmów genetycznych powodują, że badania nad ich wykorzystaniem prowadzone są od wielu lat przez wiele ośrodków naukowych na całym świecie. Podstawowe założenia funkcjonowania algorytmów genetycznych przedstawił J. Holland [Hol75]. Zainteresowało go zjawisko ewolucji w przyrodzie, w którym w wyniku zmian w chromosomach następuje przystosowanie do warunków otoczenia. Holland uważał, że algorytm naśladujący ewolucję, operujący na zbiorze osobników za pomocą reprodukcji, krzyżowania i mutacji będzie w stanie znaleźć optymalne rozwiązanie trudnych problemów. Holland [Hol75] opracował teorię schematów, według której „Krótkie, niskiego rzędu i oceniane powyżej średniej schematy uzyskują wykładniczo rosnącą liczbę łańcuchów w kolejnych pokoleniach”. Podstawową zaletą, ale i jednocześnie wadą algorytmu genetycznego jest fakt, że algorytm nie dysponuje wiedzą na temat rozwiązywanego problemu. Jedynym wskaźnikiem służącym do oceny osobników jest funkcja przystosowania obliczana dla każdego osobnika.

Szczególny wkład w rozwój algorytmów genetycznych w okresie początkowym wniósł między innymi Bagley J.D. [Bag67]. Wykorzystał on algorytm genetyczny w grach strategicznych. Zaprojektował algorytm genetyczny do gry w 6 pionków (zastosowane operacje reprodukcji, krzyżowania i mutacji, reprezentacja diploidalna, mechanizm dominowania oraz inwersja). Jako pierwszy przedstawił on pomysł samoregulacji algorytmu genetycznego poprzez kodowanie prawdopodobieństwa krzyżowania i mutacji w chromosomach, nie podał jednak wyników eksperymentalnych dotyczących tego mechanizmu. Rosenberg R.S. [Ros67] przeprowadził eksperymenty z symulowaną ewolucją populacji organizmów jednokomórkowych, w których wykorzystał reprezentację diploidalną oraz mechanizm krzyżowania adaptacyjnego. Cavicchio D.J. [Cav70] wykorzystał algorytm

genetyczny do doboru detektorów dla zadania rozpoznawania postaci (wprowadził oprócz reprodukcji i mutacji mechanizm preselekcji – osobnik o dobrym przystosowaniu zastępował swojego „rodzica”). Weinberg R. [Wei70] zaproponował użycie wielowarstwowego algorytmu do symulacji komputerowej żywej komórki. Algorytm genetyczny nieadaptacyjny (wyższego poziomu) obliczał użyteczność adaptacyjnego algorytmu genetycznego (niższego poziomu) na podstawie oceny populacji, która wyewoluowała pod nadzorem algorytmu adaptacyjnego i kieruje ewolucją populacji adaptacyjnych algorytmów. Hollstein R.B. [Hol71] zaproponował wykorzystanie algorytmów genetycznych do optymalizacji funkcji dwóch zmiennych. Przeprowadził badania różnych metod selekcji i typów preferencji przy kojarzeniu partnerów. Frantz D.R. [Fra72] badał efekt pozycyjnej nieliniowości (epistazy) w optymalizacji za pomocą algorytmu genetycznego. Wprowadził operację dopełnienia częściowego, polegającą na zastąpieniu w wybranych osobnikach około jednej trzeciej bitów wartościami przeciwnymi, w celu zachowania różnorodności populacji oraz operację krzyżowania wielopunktowego. Bosworth J., Foo N., Zeigler B.P. [Bos72] prowadzili badania algorytmów genetycznych wykorzystujących kodowanie preferujące wzajemną odpowiedniość między genami a parametrami, niezależnie od liczby alleli przypadających na jeden gen (wykorzystanie dużych alfabetów). De Jong K.A. [Jon75] wykorzystał algorytmy genetyczne w optymalizacji funkcji. Zaproponował funkcje testowe z zakresu minimalizacji funkcji oraz mierniki efektywności algorytmów („on-line” i „ off-line”).

Spośród literatury na temat algorytmów genetycznych największy wpływ na niniejszą pracę wywarły publikacje:

- Goldberg D.E., [Gol98] *Algorytmy genetyczne i ich zastosowania* – w pracy tej przedstawiono w krótki sposób podstawowe informacje i twierdzenia dotyczące teorii algorytmów genetycznych, podstawy matematyczne ich działania oraz niektóre zastosowania. W rozdziale czwartym autor przedstawia „zarys historii pierwszych algorytmów genetycznych, łącznie z przeglądem aktualnych zastosowań”,
- Michalewicz Z., [Mic99] *Algorytmy genetyczne + struktury danych = programy ewolucyjne* – w pracy tej oprócz podstawowych informacji dotyczących algorytmów genetycznych, autor porusza problemy wykorzystania ich w optymalizacji numerycznej (rozdział 6), zadaniach z ograniczeniami (rozdział 4), rozwiązywaniu problemu komiwojażera (rozdział 10),
- De Jong K.A., Spears W.M., [Jon89] *Using Genetic Algorithms to Solve NP-Complete Problems* – w artykule tym autorzy opisują sposób postępowania podczas rozwiązywania

problemów NP-zupełnych na przykładzie problemu spełnialności formuł logicznych. Autorzy zaproponowali strategię pozwalającą na uniknięcie kłopotów związanych z doбором reprezentacji chromosomów i funkcji celu, poprzez transformację rozwiązywanego problemu NP-zupełnego do podobnego problemu, dla którego istnieje efektywna reprezentacja. Strategia ta zaprezentowana została na przykładzie rozwiązywania problemu cyklu Hamiltona,

- Blickle T., Thiele L., [Bli95] *A comparison of selection schemes used in genetic algorithms* – w artykule tym autorzy przeprowadzają analizę matematyczną dla:
 - o selekcji turniejowej (Tournament Selection),
 - o selekcji wygaszającej (Truncatiuon Selection),
 - o selekcji rankingowej liniowej (Linear Ranking Selection),
 - o selekcji rankingowej wykładniczej (Exponential Ranking Selection),
 - o selekcji proporcjonalnej (Proportional Selection),

W pracy do opisu selekcji wykorzystywane są następujące parametry:

- o średnia funkcja przystosowania (Average Fitness),
 - o odchylenie funkcji przystosowania (Fitness Variance),
 - o współczynnik reprodukcji (Reproduction Rate),
 - o utrata materiału genetycznego (Loos of Diversity),
 - o intensywność selekcji (Selection Intensity),
 - o różnorodność selekcji (Selection Variance),
- Herrera F., Lozano M., [Her00b] *Two-Loop Real-Coded Genetic Algorithms with Adaptive Control of Mutation Step Sizes* – w artykule autorzy proponują wykorzystanie dwuwarstwowego algorytmu genetycznego. Algorytm składa się z dwóch pętli:
 - o pętla wewnętrzna – dokonuje przeszukiwania przestrzeni przy wykorzystaniu operatorów genetycznych w celu znalezienia rozwiązania o odpowiednim poziomie dokładności,
 - o pętla zewnętrzna – wprowadza nowe urozmaicenie populacji w przypadku, gdy pętla wewnętrzna osiągnie punkt, w którym nie występuje postęp (lokalne optimum) – pomaga w znalezieniu lepszego rozwiązania zwiększając niezawodność metody. Pętla zewnętrzna uruchamia iteracyjnie pętlę wewnętrzną oraz pozwala na powtórne uruchomienie procesu, poprzez ponowną inicjalizację populacji przez zmutowanie wszystkich genów.

Metody sterowania procesem mutacji podzielone zostały na 3 grupy:

- deterministyczne – zmiana parametrów algorytmu następuje według z góry ustalonej zależności, np. w miarę upływu czasu prawdopodobieństwo mutacji maleje, co sprzyja przeszukiwaniu lokalnego otoczenia,
- adaptacyjne – prawdopodobieństwo mutacji zmieniane jest w zależności od wyników osiąganych przez algorytm zgodnie z daną heurystyką, np. jeżeli nie nastąpiła poprawa to zwiększ prawdopodobieństwo mutacji, w przeciwnym przypadku zmniejsz prawdopodobieństwo mutacji,
- autoadaptacyjne – parametry algorytmu zakodowane są w genach i podlegają zmianom w procesie ewolucji.

Podstawy teorii zbiorów rozmytych stworzone zostały przez Lotfi Zadeha [Zad65] w roku 1965. Teoria ta umożliwiła opis rzeczywistych systemów za pomocą pojęć rozmytych. Systemy wykorzystujące logikę rozmytą znajdują zastosowanie w sytuacjach, gdy tradycyjne metody sterowania nie są w stanie sprostać wymaganiom użytkownika i rozwiązać w sposób zadowalający wielu praktycznych problemów technicznych. Do chwili obecnej systemy wykorzystujące logikę rozmytą wykorzystane zostały między innymi przez:

- Assilian S. [Ass74] – regulacja systemu wytwarzania pary w elektrowni,
- Mamdani E.H. [Mam77] – układ automatycznego sterowania obrotowego pieca do produkcji cementu,
- koncern Hitachi [Abe91] – układ sterowania metra Sendai,
- von Altrock C. [Alt92] – układ sterowania samochodu,
- Bien Z. [Bie92] – układ sterowania boilerem wodnym,
- Arita S. [Ari90] – układ sterowania ciśnienia krwi,
- Arita S. [Ari91] – diagnostyka nowotworowa.

W pracy tej pomocne były doświadczenia uzyskane przez różnych naukowców pracujących nad problemami logiki rozmytej opisane w publikacjach:

- Piegat A., [Pie99] *Modelowanie i sterowanie rozmyte* – w pracy tej autor przedstawia „podstawowe informacje z zakresu teorii zbiorów rozmytych oraz prezentuje najbardziej znane metody modelowania i sterowania rozmytego”,
- Driankov D., Hellendoorn H., Reinfrank M., [Dri96] *Wprowadzenie do sterowania rozmytego* – „to książka stanowiąca wprowadzenie do teorii sterowania”, która zawiera

„odpowiednią część teorii i koncentrowała się bardziej na zasadach sterowania rozmytego, niż na konkretnych zastosowaniach lub narzędziach”,

- Dvorak A., *Computational Properties of Fuzzy Logic Deduction* [Dvo97] – autor opisuje metody umożliwiające przyspieszenie obliczania wyników wnioskowania rozmytego. Autor dokonuje podziału na wnioskowanie lokalne (Local Inference) - operacje dokonywane na liczbach ostrych (crisp numbers) lub wartościach typu singleton oraz wnioskowanie globalne (Global Inference) – dla operacji na zbiorach rozmytych,
- Nauck D., Klawonn F., Kruse R., [Nau92] *Fuzzy Sets, Fuzzy Controllers, and Neural Networks* – w artykule tym autorzy przedstawiają wprowadzenie do teorii zbiorów rozmytych, pokazują sposób formalnego zapisu wyrażen lingwistycznych oraz metody wykonywania operacji na tych wyrażeniach (część pierwsza pracy). W drugiej części zaprezentowany został sterownik rozmyty, wykorzystujący zbiory rozmyte do reprezentowania wartości wejściowych i wyjściowych obiektu fizycznego oraz opisano bazę reguł rozmytych typu IF - THEN. Trzecia część dotyczy połączenia sterowników rozmytych z sieciami neuronowymi,
- Kosko B., Mitaim S., [Kos96] *What is the Best Shape for a Fuzzy Set in Function Approximation?* – autorzy poddają analizie różne typy funkcji przynależności zbiorów rozmytych w celu ustalenia wpływu typu funkcji na dokładność sterowania. Autorzy sugerują, aby podczas projektowania sterowników w pierwszej kolejności zastosować funkcje trójkątne lub Gaussa, w celu zlokalizowania wartości modalnej i szerokości, a następnie wykorzystać funkcję sinusoidalną.

Algorytmy genetyczne wykorzystane zostały do optymalizacji bazy reguł sterowników rozmytych i funkcji przynależności. Z kolei do optymalizacji parametrów algorytmów genetycznych zastosowano systemy wykorzystujące logikę rozmytą. W ten sposób w naturalny sposób powstały rozmyte algorytmy genetyczne (fuzzy genetic algorithms).

W pracy Herrera F., Lozano M., [Her98a] – *Fuzzy Genetic Algorithms: Issues and Models* zebrano podstawowe wiadomości dotyczące połączenia logiki rozmytej i algorytmów genetycznych oraz wyniki dotychczasowych prac w tej dziedzinie. W artykule rozważane są 4 możliwe przypadki połączenia algorytmów genetycznych i logiki rozmytej:

- reprezentacja (Representation) – klasyczne kodowanie binarne stosowane w algorytmach genetycznych może być zastąpione rozmytą reprezentacją (geny o wartości między 0 i 1),
- operatory krzyżowania (Fuzzy Connetcive-Based Crossover Operators) – logika rozmyta i trójkątne funkcje przynależności mogą być przydatne do tworzenia operatorów

krzyżowania, pozwalających na utrzymanie odpowiedniego stopnia zróżnicowania populacji, zapobiegającego przedwczesnej zbieżności algorytmów,

- kryteria zatrzymania (Stopping criteria) – logika rozmyta może być wykorzystana do podejmowania decyzji o osiągnięciu przez algorytm wartości optymalnej z wymaganą przez użytkownika dokładnością,
- adaptacja parametrów algorytmu genetycznego (Adaptation of GA control parameters) – sterownik rozmyty może być wykorzystany do dynamicznego obliczania parametrów na podstawie doświadczenia i wiedzy ekspertów, w celu zapewnienia odpowiedniej relacji pomiędzy eksploracją i eksploatacją zapobiegającej przedwczesnej zbieżności algorytmu.

Poniżej przedstawiona zostanie krótka charakterystyka przykładowych systemów łączących logikę rozmytą i algorytmy genetyczne.

Operatory krzyżowania (FCB-crossovers) bazujące na logice rozmytej pozwalają na dobranie odpowiedniej relacji pomiędzy eksploatacją i eksploracją przestrzeni poszukiwań. Opis tych operatorów będzie składał się z 2 części:

- funkcji do tworzenia kombinacji alleli,
- wykorzystania tych funkcji do definiowania operatorów krzyżowania pomiędzy chromosomami.

Do tworzenia kombinacji alleli wykorzystywane są:

- T-norma (oznaczana T),
- T-conorma (oznaczana G),
- funkcja uśredniająca (oznaczana P).

T-norma, T-conorma i funkcja uśredniająca wykorzystane zostały do uzyskania funkcji F, S i M zdefiniowanych w obszarze $<0,1>$. F została połączona z T-normą, S z T-conormą, zaś M z funkcją uśredniającą. Zastosowania operatorów F i S powoduje eksplorację, operatora M – eksploatację. Zastosowanie operatorów krzyżowania może odbywać się w oparciu o różne strategie, spośród których za najlepszą uznano następującą: dla każdej pary chromosomów tworzy się 4 potomków poprzez zastosowanie operatorów F, S i dwukrotnie M, następnie dwa najlepsze osobniki zastępują rodziców.

W oparciu o rezultaty zastosowania dynamicznych operatorów krzyżowania (dynamic FCB-crossover operators) powstała idea pozwalająca na eksplorację na początku i eksploatację pod koniec działania algorytmu, dzięki zastosowaniu parametryzowanych operatorów krzyżowania.

Miękkie operatory genetyczne (Soft Genetic Operators) bazują na idei trójkątnych rozkładów prawdopodobieństwa (triangular probability distributions). W miękkim modalnym operatorze krzyżowania (Soft Modal Crossover Operator) prawdopodobieństwo, że gen o identyfikatorze i potomka Z będzie miał wartość z_i , ustalone jest na podstawie rozkładu $p(z_i) \in \{\phi(x_i), \phi(y_i)\}$, gdzie $\phi(x_i)$ i $\phi(y_i)$ są rozkładami prawdopodobieństwa (przy założeniu $x_i \leq y_i$).

Z powodu możliwości wystąpienia przedwczesnej zbieżności, algorytmy genetyczne nie gwarantują uzyskania wartości optymalnej. Jeżeli rozwiązanie optymalne nie jest znane, trudno jest dokładnie określić moment zakończenia algorytmu. Rozmyte kryterium zatrzymania (fuzzy stop criterium mechanism – FSCM) jest mechanizmem do oceny momentu zatrzymania algorytmu. FSCM bazuje na osiągnięciu przez algorytm zdefiniowanego przez użytkownika poziomu dla danego problemu. W tym celu zawiera on proces przewidywania oparty na statystyce, w celu oszacowania wartości rozwiązania optymalnego, które porównywane jest z aktualnym rozwiązaniem. Jeżeli ustalony przez użytkownika procentowy błąd został osiągnięty, algorytm zostaje zatrzymany i zwracana jest wartość rozwiązania. Proces przewidywania uruchamiany jest po określonej liczbie (np. 40) generacji i wykorzystuje aktualne wyniki algorytmu.

Parametry algorytmów genetycznych takie jak prawdopodobieństwo krzyżowania (p_c), prawdopodobieństwo mutacji (p_m) i liczba osobników w populacji (N) stanowią kluczowe elementy, określające zdolność algorytmu do eksploracji/eksploatacji [Her98a]. Adaptacyjne algorytmy genetyczne (Adaptive GAs – AGAs) pozwalają w sposób dynamiczny dostosowywać wybrane parametry algorytmu podczas jego pracy. Bazujące na logice rozmytej reguły umożliwiają odkrycie i reprezentację szerokiego zakresu adaptacyjnych strategii, stosowanych w algorytmach genetycznych. Wiedza ta może być wykorzystana do sterowania przebiegiem pracy algorytmów. Istnieją różne strategie sterowania przebiegiem algorytmu genetycznego.

W pracach [Xu93] [Xu94] sterownik rozmyty zastosowany został w celu rozwiązania dwóch problemów występujących w klasycznych algorytmach:

- wolna prędkość przeszukiwania,
- przedwczesna zbieżność.

Zadaniem sterownika jest:

- dobrać parametry algorytmu przed uruchomieniem,
- dostosowywać parametry w trybie on-line w celu adaptacji do nowej sytuacji,

- asystować użytkownikowi w dostępie, projektowaniu, implementacji i wykonywaniu algorytmu dla zadanego problemu.

W eksperymentach opisanych w pracach [Xu93] [Xu94] zastosowano dwa sterowniki, kontrolujące prawdopodobieństwo krzyżowania i mutacji na podstawie dwóch wielkości wejściowych: liczebności populacji i liczby pokoleń.

Z bazy reguł sterownika stosowanego w tej strategii wynika:

- dla dużej wielkości populacji i dużej liczby pokoleń, wartość prawdopodobieństwa krzyżowania p_c powinna być duża, co oznacza że osobnik o wysokiej wartości funkcji przystosowania z dużym prawdopodobieństwem będzie mógł przekazać swój materiał genetyczny do następnego pokolenia,
- dla dużej wielkości populacji i dużej liczby pokoleń, wartość prawdopodobieństwa mutacji p_m powinna być bardzo mała, co oznacza że niewielka liczba osobników będzie poddawana procesowi mutacji, pozostałe zaś będą mogły przekazać swój niezmienny materiał genetyczny do następnego pokolenia.

Herrera F. i Lozano M., w swojej pracy [Her96a] zaproponowali strukturę adaptacyjnego algorytmu o nazwie AGRAF, wykorzystującego kodowania rzeczywiste i bazującego na sterowniku rozmytym. W programie AGRAF zastosowano dwa różne operatory krzyżowania: jeden o własnościach eksploracji, drugi o własnościach eksploatacji. Parametr oznaczany jako p_e definiuje częstotliwość stosowania operatora eksploatacji. Jego wartość wpływa na relacje pomiędzy eksploracją i eksploatacją poprzez zastosowanie operatorów krzyżowania. Jeżeli p_e jest niskie, AGRAF powinien zapewnić większą różnorodność genów - realizuje on eksplorację. Jeżeli p_e jest wysokie, aktualna różnorodność genów może być wykorzystana do poszukiwania najlepszego osobnika – realizowana jest eksploatacja. W programie AGRAF wykorzystano mechanizm selekcji rankingowej, w którym presja selekcyjna jest zdefiniowana za pomocą parametru $\eta_{\min} \in (0,1)$. Jeżeli $\eta_{\min}$ jest niski presja selekcyjna jest wysoka, w przeciwnym wypadku jest niska. Parametry p_e i $\eta_{\min}$ są dostosowywane przy wykorzystaniu dwóch sterowników rozmytych.

W pracy Arnone, Dell'Orto & Tettamanzi [Arn94] autorzy wprowadzili pojęcie rozmytego zarządzania (Fuzzy Government), stwierdzili oni, że algorytm genetyczny podczas pracy wymaga nadzoru człowieka w celu:

- stwierdzenia faktu wykrycia rozwiązania,
- dostosowywania parametrów algorytmów,

- śledzenia ewolucji i zapobiegania niewłaściwym zachowaniom np. przedwczesnej zbieżności.

Autorzy używają pojęcia rozmytego zarządzania dla zbioru reguł i procedur wykorzystywanych do kontrolowania ewolucji algorytmu genetycznego. Badania wykonane przez autorów pokazały, że rezultaty osiągane przez algorytmy wykorzystujące rozmyte zarządzanie, są niezauważalnie różne od rezultatów osiąganych przez te same algorytmy z bezpośrednim nadzorem człowieka.

Na podstawie badań eksperymentalnych wiadomo, że jeżeli wartość funkcji przystosowania jest wysoka (pod koniec działania algorytmu), preferowane są niskie wartości prawdopodobieństwa krzyżowania i wysokie prawdopodobieństwa mutacji. Jednocześnie, jeżeli funkcja przystosowania zatrzyma się na pewnej wartości przez dłuższy okres, oznaczać to może utknięcie algorytmu w lokalnym optimum. Algorytm powinien skoncentrować się bardziej na eksploracji niż na eksploatacji. To znaczy, że należy zwiększyć prawdopodobieństwo mutacji a zmniejszyć prawdopodobieństwo krzyżowania. W oparciu o te wiadomości Shi, Eberhart i Chen [Shi99] zbudowali system rozmyty dostosowujący prawdopodobieństwo krzyżowania i mutacji na podstawie danych wejściowych:

- funkcji przystosowania najlepszego osobnika (best fitness – BF),
- liczby pokoleń, w których funkcja przystosowania najlepszego osobnika nie ulega zmianie (number of generation for unchanged best fitness – UN),
- różnica funkcji przystosowania (variance of fitness – VF).

Badania wykonane przez autorów pokazały, że dzięki zastosowaniu systemu z logiką rozmytą do adaptacji prawdopodobieństwa mutacji i krzyżowania, znacząco mniej pokoleń potrzeba, w porównaniu z algorytmem o stałych parametrach, do osiągnięcia tych samych wyników.

Voget [Vog96] w swojej pracy zaprezentował algorytm genetyczny do optymalizacji wielokryterialnej wspomagany logiką rozmytą, w którym w każdym pokoleniu sterownik rozmyty decyduje o wyborze transformacji funkcji kosztów do jednowymiarowej funkcji przystosowania. Bardziej kompleksowa metoda nazywana rozmytą redukcją algorytmów genetycznych (fuzzy reduction GA) zaproponowana została w pracy Voget & Kolonko [Vog98]. Autorzy startowali od jawnego zdefiniowania pożądanego celu ewolucji populacji. Następnie zdefiniowali miarę odchylenia populacji od stanu docelowego, będącą wielkością wejściową dla sterownika rozmytego. Ustalono zestaw akcji (różnych mechanizmów selekcji bazujących na rozwiązaniach stosowanych dotychczas w optymalizacji wielokryterialnej), możliwych do wykonania w celu zmniejszenia odchylenia. Sterownik rozmyty określał stopień aktywacji poszczególnych akcji. O wyborze podejmowanej akcji decydowała wartość

stopnia aktywacji. Metodę zastosowano do optymalizacji kosztów/zysków inwestowania w sieć kolejową. Wyniki uzyskane przez autorów pokazały, że metoda pozwala uniknąć wielu skrótów empirycznych występujących w innych algorytmach genetycznych optymalizacji wielokryterialnej.

W pracy Streifel, Marks II, Reed & Choi [Str99] zaprezentowano algorytm nazywany GAP służący do adaptacyjnego kontrolowania parametrów kodowania algorytmów genetycznych (Dynamic Fuzzy Control of GA Parameter Coding). Algorytm wykorzystuje pośrednie zależności między łańcuchami genetycznymi i przestrzenią poszukiwań parametrów. Algorytm (GAP) wykonuje standardowe poszukiwanie genetyczne, dopóki populacja łańcuchów nie osiągnie zbieżności. Zbieżność jest mierzona poprzez ocenę średniej liczby bitów różnych pomiędzy wszystkimi łańcuchami genetycznymi. Każdy łańcuch jest porównywany ze wszystkimi innymi łańcuchami i mierzona jest liczba różnych bitów. Jeżeli średnia liczba bitów różnych przekroczy wartość progową, algorytm jest zbieżny. Po ustaleniu zbieżności łańcucha genetycznego ustalany jest nowy zakres i offset za pomocą sterownika rozmytego na podstawie sygnału wejściowego, będącego odległością między środkiem bieżącego zakresu i najlepszym spośród dotychczasowych rozwiązań.

Autorzy proponują różne reguły heurystyczne:

- Jeżeli najlepsze rozwiązanie jest blisko środka zakresu, oznacza to, że można zmniejszyć zakres. Najlepsze rozwiązanie w środku zakresu oznacza, że dostosowanie poprzedniego zakresu było prawidłowe i rzeczywiste rozwiązanie jest blisko środka zakresu,
- Jeżeli najlepsze rozwiązanie jest blisko granicy zakresu, oznacza to, że przestrzeń poszukiwań powinna być dopasowana tak, aby obejmować więcej przestrzeni w pobliżu najlepszego rozwiązania. Należy zwiększyć zakres i zmienić położenie zakresu.

Po dokonaniu zmian, algorytm jest uruchamiany ponownie z nowymi wartościami zakresu i offsetu. Badanie algorytmu autorzy przeprowadzili na przykładzie identyfikowania parametrów hamulca hydraulicznego. Wskazują one, że uzyskane wyniki są bardziej realistyczne niż przy zastosowaniu innych metod z kodowaniem dynamicznym oraz dają dokładniejsze rezultaty przy mniejszej liczbie pokoleń.

W rozmytych algorytmach kulturowych (Fuzzy Cultural Algorithms) wykorzystuje się doświadczenie osobników wybranych z populacji, za pomocą funkcji akceptacji do generowania wiedzy, do rozwiązywania problemów znajdujących się w przestrzeni wiary. Wiedza ta może być postrzegana jako zbiór drogowskazów kontrolujących ewolucję składników populacji za pomocą funkcji oddziaływania. Funkcja oddziaływania może wykorzystać wiedzę z przestrzeni wiary do modyfikowania dowolnego aspektu składników

populacji. W pracy Reynoldsa i Chunga [Rey97] prezentowany jest system, w którym sterownik rozmyty reguluje znaczenie informacji przesyłanych do przestrzeni wiary. W szczególnym przypadku sterownik rozmyty określa liczbę osobników mających wpływ na bieżące wyniki. Wejściem sterownika jest współczynnik sukcesów osobników (liczba sukcesów do ogólnej liczby mutacji) i numer bieżącego pokolenia. Podstawową intuicyjną zasadą wykorzystywaną do budowy reguł rozmytych jest:

- jeżeli pokolenie jest wczesne i współczynnik sukcesu jest niski, wtedy akceptuj średnią liczbę (ok. 30% najlepszych) osobników z populacji,
- jeżeli bieżące pokolenie jest wczesne i współczynnik sukcesu jest wysoki – akceptuj dużą liczbę (ok. 40%) osobników,
- jeżeli bieżące pokolenie jest bliskie końca i współczynnik sukcesu jest niski – akceptuj najmniejszą liczbę (ok. 20%) osobników,
- jeżeli bieżące pokolenie jest bliskie końca i współczynnik sukcesu jest wysoki – akceptuj średnią liczbę osobników.

W wyniku badań autorzy stwierdzili znaczącą poprawę współczynnika sukcesu oraz zmniejszenie czasu procesora w porównaniu z wersją programu niekorzystającą z logiki rozmytej.

W pracy Zeng i Rabenasolo [Zen97] zaprezentowany został algorytm genetyczny, w którym prawdopodobieństwo krzyżowania, miejsce krzyżowania i prawdopodobieństwo mutacji definiowane są na podstawie przykładowych danych, za pomocą sterownika rozmytego biorącego pod uwagę wartość funkcji przystosowania próbek oraz odległość między nimi. W regułach sterownika zaimplementowano zasady:

- utrzymywać różnorodność populacji - dwa odległe osobniki mają większą szansę na wybór do krzyżowania,
- zwiększać poszukiwania w optymalnym regionie - dwa bliskie osobniki o dużej wartości funkcji przystosowania mają większą szansę na wybór do krzyżowania,
- zapobiegać zbieżności do lokalnego optimum - operator krzyżowania jest zwiększany jeżeli różnorodność wartości funkcji przystosowania jest mała,
- stabilizować optymalną populację - operator krzyżowania jest zmniejszany jeżeli wartość funkcji przystosowania wybranej próbki jest wystarczająco bliska maksymalnej funkcji przystosowania bieżącej populacji.

W podobny sposób zbudowany został sterownik rozmyty do adaptacji prawdopodobieństwa mutacji, lecz operator mutacji jest zwiększany, jeżeli populacja zmierza do optimum lokalnego, albo zmniejszany, gdy występuje silne zróżnicowanie populacji lub uzyskano

globalne optimum. Badania przeprowadzone przez autorów pokazały, że populacja osobników jest bardziej zróżnicowana a algorytm wykazuje lepszą zbieżność do globalnego optimum w porównaniu z algorytmem prostym.

W pracy Lee i Takagi [Lee93] [Lee94] autorzy zaproponowali proces automatycznego uczenia sterownika, w celu uzyskania bazy reguł rozmytych. Mechanizm jest podobny do zaproponowanego przez Greffentette [Gre86] algorytmu sterowanego z metapoziomu. Bazuje on na algorytmie, w którym do chromosomów zakodowano bazę reguł wraz z odpowiadającymi bazami danych. Wartość funkcji przystosowania chromosomów obliczana jest przy użyciu mierników *on-line* i *off-line* [Jon75]. Wartości mierników *on-line* i *off-line* otrzymuje się z algorytmu wspomaganego logiką rozmytą, wykorzystującego bazę reguł zakodowaną w chromosomach. Baza reguł uzyskana jest na podstawie 5 funkcji testowych De Jong'a. Po zakończeniu przez algorytm z metapoziomu 1000 pokoleń zwracana jest najlepsza baza danych uzyskana w wyniku ewolucji. Metodę wykorzystano do uzyskania bazy reguł i bazy danych dla problemu stabilizacji odwróconego wahadła. Rezultaty otrzymane przez autorów [Lee93] [Lee94] wykazały, że algorytm uzyskuje lepsze wyniki niż algorytm elementarny.

Herrera i Lozano [Her01] zaproponowali mechanizm bazujący na sterowniku rozmytym do adaptacji operatorów genetycznych. Podstawowe założenia mechanizmu:

- połączenie adaptacyjnych operatorów genetycznych ze sterownikami rozmytymi działającymi na poziomie osobników. Wartości parametrów kontrolujących operatory genetyczne obliczane są dla każdego zestawu rodziców za pomocą sterownika rozmytego, uwzględniającego szczególne cechy podawanych na wejście rodziców,
- niejawne uczenie bazy reguł rozmytych w czasie pracy oddzielnego algorytmu genetycznego (nazywanego FBs-GA), koewoluującego z algorytmem zawierającym kontrolowane operatory genetyczne, nazywanym głównym algorytmem (main-GA). Każdy z algorytmów ma wpływ na pozostały. Z jednej strony algorytm FBs-GA indukuje wartości parametrów operatorów genetycznych używanych przez główny algorytm, z drugiej zaś strony oba algorytmy ewoluują w zależności od wyników uzyskiwanych przez operatory genetyczne algorytmu głównego. Podstawowym zadaniem algorytmu FBs-GA jest uzyskanie rozmytej bazy reguł, dającej użyteczne wartości parametrów, pozwalających operatorom genetycznym na rozwiązanie poszczególnych problemów. Algorytm FBs-GA nie operuje bezpośrednio na bazie reguł, lecz używa struktur nazywanych środowiskiem rozmytym (fuzzy behaviors) do reprezentowania reguł, dzięki

czemu mogą być one lepiej reprezentowane w postaci chromosomów. Środowisko rozmyte zawiera wektory wartości lingwistycznych reguł rozmytych.

Praktyczne badania przeprowadzono [Her01] na podstawie testów funkcji o różnym stopniu trudności z punktu widzenia osiągnięć algorytmu oraz z punktu widzenia możliwości uczenia środowiska. Wyniki były porównywane z wynikami algorytmów z miękkim modalnym krzyżowaniem oraz z algorytmem z adaptacją na poziomie osobników za pomocą sterownika rozmytego z różnymi bazami reguł rozmytych. Wyciągnięto wniosek, że zdolność uczenia związana z adaptacją miękkiego modalnego krzyżowania pozwala na tworzenie odpowiednich rozkładów FBs (uzyskano znaczącą poprawę osiągnięć dla wszystkich funkcji testowych, które osiągnęły najlepsze rezultaty w porównaniu z innymi metodami).

W kolejnej strategii zaproponowano algorytm genetyczny zawierający sterownik rozmyty wykonujący adaptację prawdopodobieństwa mutacji podczas pracy algorytmu. Parametr ten bezpośrednio wpływa na operator mutacji, który jest odpowiedzialny za zapewnienie różnorodności populacji, stanowiącej istotny element zapobiegania problemowi przedwczesnej zbieżności. Sterownik rozmyty uruchamiany jest po G pokoleniach i prawdopodobieństwo mutacji p_m jest zmieniane w takich samych odstępach czasowych. Brana jest pod uwagę wartość p_m w ostatnich G pokoleniach oraz przyrost funkcji przystosowania najlepszego dotychczasowego elementu. Następnie obliczana jest nowa wartość p_m , która obowiązywać będzie przez kolejne G pokoleń.

W artykule Herrera F., Lozano M., Verdegay J.L. [Her97a] autorzy prezentują propozycję konstrukcji operatorów krzyżowania dla algorytmów genetycznych z kodowaniem rzeczywistym (real coded genetic algorithms RCGA), które dzięki wykorzystaniu logiki rozmytej pozwolą na utrzymanie zróżnicowania populacji osobników w celu zapobieżenia przedwczesnej zbieżności algorytmu. Autorzy zaproponowali operatory krzyżowania odpowiadające funkcjom: T-normie, T-conormie, funkcji uśredniającej oraz uogólnionemu operatorowi kompresji.

W artykule Herrera F., Herrera-Viedma E., Lozano A., Verdegay L.J., [Her94b] autorzy prezentują niżej opisane dwa narzędzia do zwiększenia efektywności algorytmów genetycznych.

- Operatory krzyżowania bazujące na logice rozmytej – poprzez wybór odpowiedniej T-normy, T-conormy i funkcji uśredniającej użytkownik może zapobiegać przedwczesnej zbieżności algorytmu,
- System wykorzystujący logikę rozmytą monitorujący stopień zróżnicowania populacji, wykorzystujący dwa wskaźniki: odchylenie średnie chromosomów VAC

(variance average chromosomes) i średnie odchylenie alleli AVA (average variances alleles). Algorytm podzielony został na trzy fazy:

- inicjalizacji - wskaźniki zależne są od wartości początkowych populacji,
- poszukiwania - wskaźniki wpływają w sposób dynamiczny na wartości prawdopodobieństwa selekcji i mutacji,
- dostrajanie lokalne – parametry algorytmu sprzyjają przeszukiwaniu otoczenia lokalnego optimum.

Kruse R., Nauck D., [Kru95] przedstawiają dwie metody dostrajania sterowników rozmytych na podstawie danych pochodzących z pomiarów:

- metodę analizy klastrowej (cluster analysis),
- uczenie sterowników neuronowo-rozmytych (Neuro-Fuzzy Systems) – sterownik rozmyty przedstawiony jest w postaci wielowarstwowej sieci neuronowej i uczony metodami zaczerpniętymi z teorii sieci neuronowych.

Yen J., Lee B., Liao J., [Yen96] pokazali sposób wykorzystania logiki rozmytej i hybrydowego algorytmu genetycznego do modelowania procesów metabolicznych. W tym artykule autorzy proponują podejście do modelowania, które:

- wykorzystuje logikę rozmytą do rozszerzenia algebraicznego modelu wpływu enzymów,
- wykorzystuje hybrydowy algorytm genetyczny do identyfikowania niepewnych parametrów w modelu, gdzie w celu zwiększenia współczynnika zbieżności algorytmu genetycznego zastosowano metodę simpleksów.

Proponowana strategia modelowania pozwala na:

- włączenie elementów intuicyjnych do modelu matematycznego,
- adaptacyjne identyfikowanie i optymalizację kluczowych parametrów, aby dostosować środowisko systemu do warunków występujących w eksperymentach biochemicznych.

1.2. Wykorzystanie wiedzy nabytej we wcześniejszych pokoleniach

W elementarnym algorytmie genetycznym o przebiegu procesu ewolucji decyduje wartość funkcji przystosowania poszczególnych osobników. Rozważana jest tylko wartość funkcji przystosowania w bieżącym pokoleniu, pozostałe informacje są tracone. Z punktu widzenia ewolucji, przydatna natomiast może być również informacja o wartości funkcji przystosowania w poprzednim pokoleniu. W pracach dotyczących połączenia algorytmów genetycznych i logiki rozmytej, w większości przypadków adaptacja parametrów algorytmu

występuje na poziomie populacji albo dotyczy możliwości znalezienia i dobrania operatorów genetycznych zapewniających możliwie najlepsze warunki ewolucji.

Aby wykorzystać wiedzę zgromadzoną w poprzednim pokoleniu, autor proponuje wykorzystanie sterownika rozmytego. Zadaniem tego sterownika jest analiza wartości funkcji przystosowania poszczególnych osobników, wartości funkcji przystosowania najlepszego spośród dotychczasowych osobników oraz średniej wartości funkcji przystosowania w kolejnych pokoleniach i na podstawie tych danych zmodyfikowanie parametrów algorytmu genetycznego.

Autor niniejszej pracy proponuje adaptację prawdopodobieństwa selekcji i mutacji na poziomie osobników. W pracy oceniany będzie każdy osobnik we wszystkich pokoleniach, ponadto wykorzystane zostaną informacje pochodzące z poprzedniego pokolenia. O prawdopodobieństwie selekcji osobnika do puli rodzicielskiej oraz prawdopodobieństwie mutacji decydować będzie nie tylko wartość funkcji przystosowania osobnika, ale również średnia wartość funkcji przystosowania pokolenia bieżącego i poprzedniego oraz wartość funkcji przystosowania najlepszego spośród dotychczas znalezionych rozwiązań. Ponieważ wiedza dotycząca wyboru poszczególnych osobników do puli rodzicielskiej oraz oczekiwanej liczby osobników poddawanych mutacji ma charakter opisowy i trudno jest określić ścisłe reguły matematyczne, wykorzystany zostanie sterownik rozmyty. Baza reguł rozmytych sterownika zapewni przejrzystość reguł i pozwoli w przyszłości na ich modyfikacje w celu zapewnienia optymalnych warunków ewolucji osobników.

1.3. Cel pracy

Przegląd literatury zaprezentowany w rozdziale 1.1. pokazuje, że w wielu ośrodkach na świecie prowadzone są badania nad wykorzystaniem algorytmów genetycznych w różnych dziedzinach. W zależności od rozwiązywanego zadania stosowane są algorytmy o różnych właściwościach. Przedstawiona w niniejszej pracy koncepcja zakłada zbudowanie algorytmów o większej zdolności dopasowywania się do rozwiązywanego zadania, poprzez adaptację parametrów algorytmu w zależności od dotychczasowego przebiegu optymalizacji. Do adaptacji parametrów algorytmu wykorzystany zostanie sterownik rozmyty analizujący funkcję przystosowania osobników w pokoleniu bieżącym i poprzednim.

Zasadniczym celem tej pracy jest zbadanie, w jakim stopniu wykorzystanie wiedzy rozmytej do adaptacji parametrów może poprawić działanie algorytmu genetycznego. Z zasadniczego celu pracy wynikają cele szczegółowe:

Cel 1. Opracowanie własnej, nowej koncepcji sterowania parametrami algorytmu genetycznego przy pomocy sterownika rozmytego.

Skuteczność algorytmów genetycznych zależy jest od przyjętych parametrów algorytmu, takich jak prawdopodobieństwo selekcji i mutacji. Wprowadzenie do algorytmu adaptacji pozwalającej na dostosowanie tych parametrów w zależności od aktualnego przebiegu optymalizacji, pozwoli na zmianę relacji pomiędzy zdolnością algorytmu do przeszukiwania nowych obszarów przestrzeni rozwiązań (eksploracji) i eksploatacji odkrytych wcześniej obszarów.

Cel 2. Wykazanie, że adaptacja parametrów algorytmu genetycznego wpływa na relację pomiędzy poziomem eksploracji i eksploatacji przyspieszając uzyskanie satysfakcjonującego wyniku.

Algorytm genetyczny w czasie optymalizacji realizuje jednocześnie funkcję eksploracji i eksploatacji przestrzeni rozwiązań. Utrzymanie odpowiedniej relacji pomiędzy poziomem eksploracji i eksploatacji w zależności od rodzaju zadania, reprezentacji osobników i w różnych okresach działania algorytmu zwiększy efektywność tego algorytmu.

Cel 3. Wykazanie, że zmodyfikowane algorytmy genetyczne charakteryzują się większą niż algorytm elementarny zdolnością do opuszczania optimum lokalnego.

Algorytmy genetyczne wykorzystujące metody selekcji „twardej” są podatne na dominację superosobnika i przedwczesną zbieżność do lokalnego optimum. Zwiększenie odporności na przedwczesną zbieżność zapewni większą stabilność algorytmu i mniejsze różnice w wynikach uzyskanych w poszczególnych przebiegach algorytmu.

W oparciu o przedstawione powyżej rozważania stawiam następującą tezę:

Algorytm genetyczny wykorzystujący logikę rozmytą do wyznaczania kierunku ewolucji populacji osobników, w porównaniu z elementarnym algorytmem genetycznym, na ogół wykazuje większą szybkość w osiąganiu wartości optymalnej lub pozwala na osiągnięcie porównywalnych wyników przy mniejszej liczbie wywołań funkcji obliczającej przystosowanie osobnika.

1.4. Zakres pracy

Realizacja celów postawionych w pracy pozwoli na zbudowanie algorytmu, który może dostosowywać się do rozwiązywanego problemu i dzięki temu w bardziej efektywny sposób wykonywać optymalizację skomplikowanych problemów.

Aby opracować własną, nową koncepcję sterowania parametrami algorytmu genetycznego przy pomocy sterownika rozmytego zaplanowano następujące działania:

- studia literaturowe – zapoznanie się z aktualnym stanem wiedzy na temat algorytmów genetycznych, logiki rozmytej oraz sposobami ich połączenia wykorzystywanymi we wcześniejszych badaniach,
- wybór parametrów i opracowanie własnej, nowej koncepcji ich adaptacji,
- opracowanie koncepcji sterownika rozmytego – wybór modelu sterownika, jego wielkości wejściowych, wyjściowych i ich funkcji przynależności oraz na podstawie wiedzy o ewolucji ustalenie zbioru reguł dla sterownika i metody wnioskowania i defuzyfikacji.
- wykazanie teoretyczne prawdziwości twierdzenia o schematach w zmodyfikowanych algorytmach genetycznych oraz twierdzenia o punkcie stałym w zmodyfikowanym algorytmie genetycznym z odwzorowaniem zwięzającym, czyli wykazanie zbieżności tych algorytmów,
- teoretyczne oszacowanie maksymalnego nakładu obliczeniowego potrzebnego do wykonania adaptacji parametrów algorytmu.

Aby wykazać, że adaptacja parametrów algorytmu genetycznego wpływa na relację pomiędzy poziomem eksploracji i eksploatacji przyspieszając uzyskanie satysfakcjonującego wyniku zaplanowano następujące eksperymenty:

- doświadczalne sprawdzenie liczby osobników o wartości funkcji przystosowania lepszej, równej i gorszej od średniej w pokoleniu, wybieranych do puli rodzicielskiej oraz liczby mutacji zachodzących w tych osobnikach i porównanie ich z wynikami uzyskanymi przez algorytm elementarny. Jako funkcje testowe wykorzystane zostaną funkcje testowe F1 i F2 De Jonga [Jon75] (optymalizacja funkcji ciągłych) oraz problem komiwojażera (optymalizacja funkcji nieciągłych).
- doświadczalne wyznaczenie efektywności „on-line” i „off-line” algorytmów zmodyfikowanych i porównanie ich z wynikami uzyskanymi przez algorytm elementarny. Jako funkcje testowe wykorzystane zostaną funkcje testowe F1 i F2 De Jonga [Jon75] (optymalizacja funkcji ciągłych) oraz problem komiwojażera (optymalizacja funkcji nieciągłych).

- testowanie zmodyfikowanych algorytmów w rozwiązywaniu standardowych problemów optymalizacyjnych – przeprowadzenie badania wartości funkcji optymalizowanej i liczby wywołań funkcji obliczającej przystosowanie osobnika. Jako zadania testowe wykorzystane zostaną: problem optymalizacji funkcji testowych ciągłych, problem komiwojażera (optymalizacja funkcji nieciągłych) i zero-jedynkowy problem plecakowy (optymalizacja funkcji nieciągłych z ograniczeniami),
- porównanie wartości funkcji optymalizowanej uzyskanej przez zmodyfikowane algorytmy genetyczne, algorytm genetyczny elementarny oraz inne metody optymalizacji wzorowane na obserwacji natury – do porównania wybrano algorytmy mrówkowe, symulowanego wyżarzania i przeszukiwania tabu, jako zadanie testowe wybrano problem komiwojażera,
- porównanie wartości funkcji optymalizowanej, uzyskanej w algorytmach zmodyfikowanych i elementarnym po założonej liczbie pokoleń, przy różnej liczbie osobników w populacji,
- porównanie liczby pokoleń potrzebnych do uzyskania przez algorytmy zmodyfikowane i elementarny założonej wartości funkcji optymalizowanej przy różnej liczbie osobników w populacji.

Aby wykazać, że zmodyfikowane algorytmy genetyczne charakteryzują się większą niż algorytm elementarny zdolnością do opuszczania optimum lokalnego zaplanowano następujące eksperymenty:

- zbadanie zdolności algorytmów zmodyfikowanych do wyszukiwania rozwiązania optymalnego dla problemu komiwojażera (kodowanie binarne) i optymalizacji funkcji (kodowanie rzeczywiste),
- zbadanie zdolności algorytmów do wychodzenia z optimum lokalnego dla funkcji testowych o różnym stopniu trudności. Jako zadania testowe wybrano funkcje eksponencjalne zaproponowane w pracy [Kwa99g] o różnej liczbie zmiennych.
- zbadanie zdolności algorytmów do wychodzenia z optimum lokalnego dla funkcji testowych o dużym stopniu trudności. Jako zadanie testowe wybrano funkcję eksponencjalną o dużej liczbie optimumów lokalnych.
- opracowanie własnej metody optymalizacji kosztów funkcjonowania sieci komputerowej, wykorzystującej zmodyfikowane algorytmy genetyczne i porównanie ich wyników z dotychczas stosowaną metodą podziału i oszacowań.

1.5. Plan pracy

Aby udowodnić prawdziwość postawionych w pracy tez, autor przeprowadził szereg badań i eksperymentów opisanych w poszczególnych rozdziałach pracy.

W rozdziale 2 przedstawiono podstawowe informacje dotyczące algorytmów genetycznych i systemów wykorzystujących logikę rozmytą. Opisano w nim budowę i działanie elementarnego algorytmu genetycznego. W części dotyczącej logiki rozmytej zawarto informacje dotyczące budowy klasycznego sterownika rozmytego oraz jego podstawowych bloków (bazy reguł, bloku rozmywania, wnioskowania i defuzyfikacji). W oparciu o dotychczasowe prace, zaprezentowano propozycje autora wykorzystania logiki rozmytej do wyznaczania kierunku ewolucji osobników w algorytmach genetycznych.

W rozdziale 3 opierając się na twierdzeniu o schematach [Gol98] autor udowadnia, że dla algorytmów genetycznych wykorzystujących logikę rozmytą do wyznaczania kierunku ewolucji twierdzenie to jest również prawdziwe. W rozdziale tym autor udowadnia również, że algorytmy z odwzorowaniem zwężającym, wykorzystujące logiką rozmytą do wyznaczania kierunku ewolucji, są zbieżne. Dla zmodyfikowanego algorytmu dokonano oszacowania nakładu obliczeniowego koniecznego do adaptacji prawdopodobieństwa selekcji.

W rozdziale 4 przedstawiono propozycje budowy zmodyfikowanych algorytmów genetycznych, wykorzystujących bloki logiki rozmytej do sterowania zmianami prawdopodobieństwa selekcji i mutacji osobników w populacji. Przedstawiono koncepcje sterowników rozmytych, wykorzystywanych do wyznaczania kierunku ewolucji osobników wraz z funkcjami przynależności i bazami reguł rozmytych oraz metody wyznaczania zmodyfikowanego prawdopodobieństwa selekcji i mutacji dla poszczególnych osobników.

W rozdziale 5 przedstawiono przykładowe zadania testowe wykorzystywane w eksperymentach oraz zaprezentowano uzasadnienie wyboru tych zadań. Opisano przeprowadzone eksperymenty oraz przedstawiono wyniki uzyskane przez poszczególne algorytmy.

W rozdziale 6 przy pomocy algorytmów zmodyfikowanych rozwiązano zadanie minimalizacji średniego opóźnienia pakietu w sieci komputerowej. Zadanie to pozwala na obniżenie kosztów funkcjonowania sieci komputerowej, przy jednoczesnym zapewnieniu odpowiedniej wydajności transmisji pakietów danych. Czas rozwiązania zadania przy pomocy powszechnie stosowanej metody podziału i oszacowań jest bardzo długi. Wyniki uzyskane przez algorytmy zmodyfikowane porównano z wynikami uzyskanymi przez algorytm elementarny oraz metodę podziału i oszacowań.

W rozdziale 7 zebrano i sformułowano wnioski wynikające z przeprowadzonych eksperymentów oraz dokonano podsumowania pracy i wskazano na możliwe kierunki dalszych badań w tej dziedzinie.

2. Zasada działania algorytmu genetycznego i sterownika rozmytego

W rozdziale tym przedstawione zostaną podstawowe wiadomości dotyczące budowy i funkcjonowania algorytmów genetycznych i sterowników rozmytych.

2.1. Elementarny algorytm genetyczny

2.1.1. Wprowadzenie

Algorytmy genetyczne swoje istnienie zawdzięczają obserwacji i próbie naśladowania naturalnych procesów, zachodzących w świecie organizmów żywych - ewolucji i związanej z nią naturalnej selekcji, występującej w populacji żywych osobników. Opierają się na zasadzie przeżycia osobników najlepiej przystosowanych. Osobnik reprezentuje możliwe rozwiązanie danego problemu w formie zakodowanej. Każdy osobnik jest oceniany za pomocą funkcji przystosowania. Nowe pokolenia osobników tworzone są z puli rodzicielskiej wybieranej metodą probabilistyczną spośród wszystkich osobników pokolenia poprzedniego. Nowy osobnik powstaje na skutek krzyżowania (wymiany genów) rodziców. Ponadto każdy gen (cecha) może podlegać dodatkowo procesowi mutacji. Po określonej liczbie pokoleń w wyniku działania algorytmu powstają osobniki charakteryzujące się najwyższą wartością funkcji przystosowania – reprezentujące wynik optymalizacji.

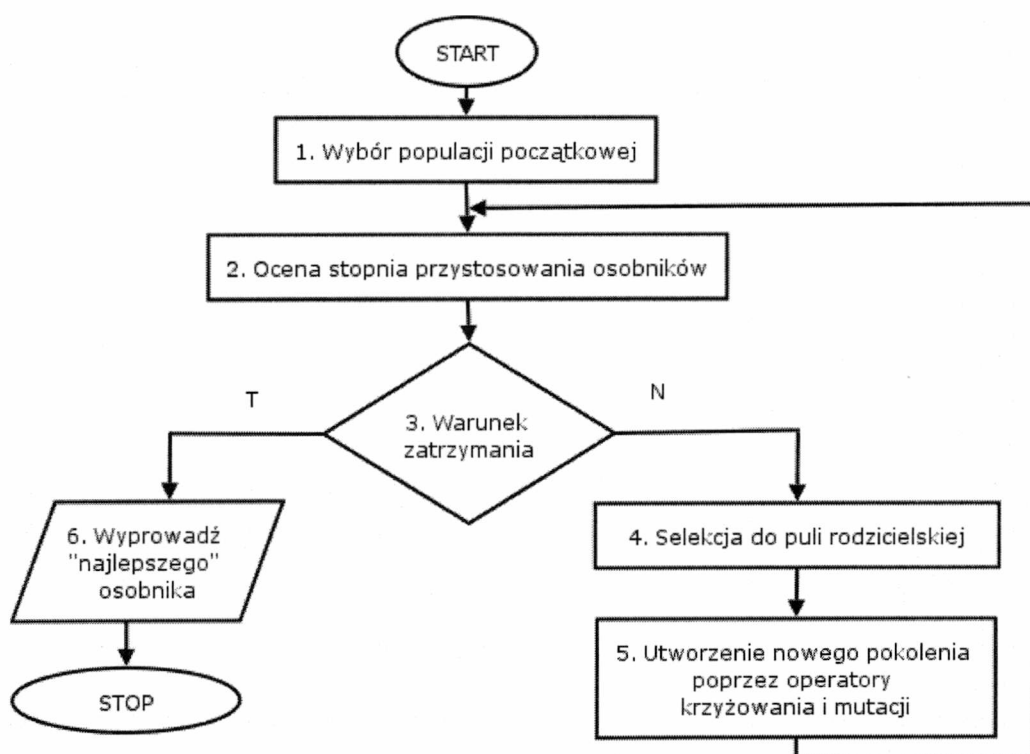
2.1.2. Budowa elementarnego algorytmu genetycznego

Elementarny algorytm genetyczny składa się z następujących kroków:

1. Wybór populacji początkowej - polega na losowym wyborze żądanej liczby chromosomów reprezentowanych przez ciągi znaków o określonej długości.
2. Ocena przystosowania poszczególnych osobników - polega na obliczeniu wartości funkcji przystosowania dla każdego osobnika z tym pokoleniu. Im większa jest wartość tej funkcji tym dany osobnik jest lepiej przystosowany.
3. Sprawdzenie warunku zatrzymania - w metodach optymalizacji, jeśli znana jest wartość maksymalna (lub minimalna) funkcji przystosowania, zatrzymanie algorytmu może nastąpić po uzyskaniu żądanej wartości optymalnej, ewentualnie z określoną dokładnością. Zatrzymanie algorytmu może również nastąpić, jeśli dalsze jego działanie nie poprawia już uzyskanej najlepszej wartości. Algorytm może też zostać zatrzymany po upływie określonego czasu działania lub po określonej liczbie iteracji. Jeśli warunek zatrzymania jest spełniony, następuje przejście do kroku 6, czyli

- wyprowadzenia chromosomu najlepiej przystosowanego. W przeciwnym wypadku następnym krokiem jest selekcja.
4. Selekcja osobników do puli rodzicielskiej - polega na wybraniu, na podstawie obliczonych wartości funkcji w kroku 2, tych chromosomów, które będą brały udział w tworzeniu potomków do następnego pokolenia. Wybór ten odbywa się zgodnie z zasadą naturalnej selekcji, zatem największą szansę na wybranie mają chromosomy o największej wartości funkcji przystosowania.
 5. Utworzenie nowego pokolenia osobników poprzez zastosowanie operatorów krzyżowania i mutacji (operator mutacji odgrywa drugoplanową rolę w stosunku do operatora krzyżowania, co wynika z analogii do świata organizmów żywych, gdzie mutacje zachodzą niezwykle rzadko). Przejście do kroku 2.
 6. Wyprowadzenie osobnika o najwyższej wartości funkcji przystosowania - jeżeli spełniony jest warunek zatrzymania algorytmu, należy wyprowadzić wynik działania algorytmu, czyli podać rozwiązanie problemu. Najlepszym rozwiązaniem jest chromosom o największej wartości funkcji przystosowania.

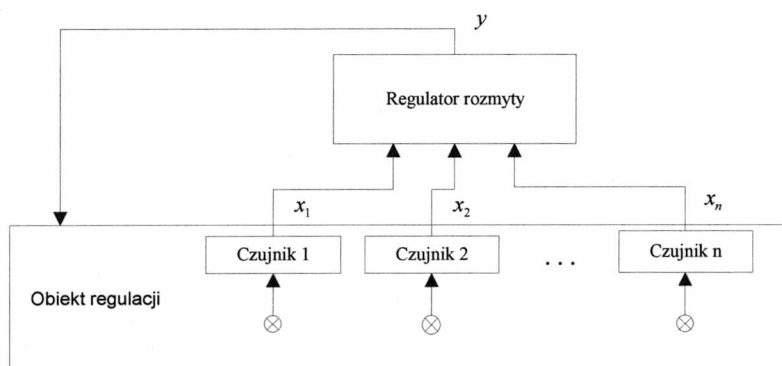
Schemat blokowy elementarnego algorytmu genetycznego przedstawiony jest na Rys. 2.1.



Rys.2.1. Schemat blokowy elementarnego algorytmu genetycznego

2.2. Budowa sterownika rozmytego

W wielu problemach, z którymi mamy do czynienia w technice, zachodzi konieczność zapewnienia regulacji obiektu przy braku matematycznego modelu zachodzących zjawisk. Dysponując jakościową wiedzą eksperta na temat zasad funkcjonowania urządzenia, możemy modelować działanie sterownika wykorzystującego bazę reguł rozmytych. Regulator rozmyty pełni takie same funkcje, jak każdy inny regulator. Schemat blokowy układu regulacji pokazano na Rys.2.2.

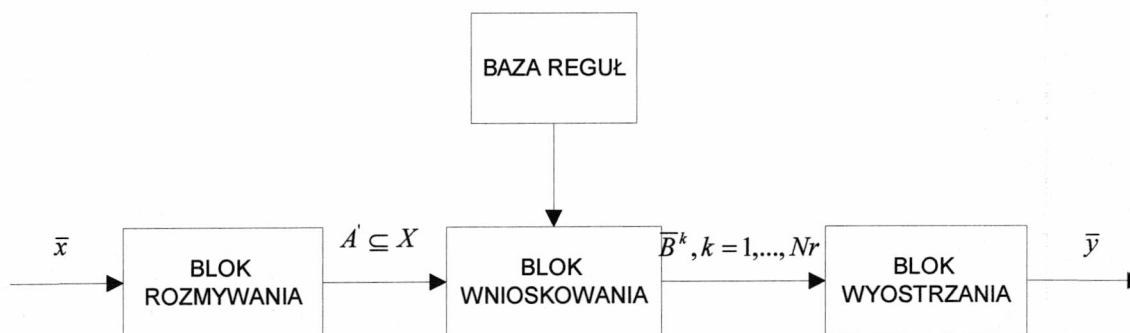


Rys.2.2. Schemat blokowy układu regulacji

Klasyczny sterownik rozmyty składa się z następujących bloków:

- baza reguł rozmytych,
- blok rozmywania (fuzyfikacji),
- blok wnioskowania,
- blok wyostrzania (defuzyfikacji).

Schemat blokowy klasycznego sterownika rozmytego pokazano na Rys.2.3.



Rys.2.3. Schemat blokowy klasycznego sterownika rozmytego

2.3. Propozycje zmian w elementarnym algorytmie genetycznym

W elementarnym algorytmie genetycznym parametrami mającymi wpływ na ewolucję i możliwości przeżycia osobnika są prawdopodobieństwo selekcji do puli rodzicielskiej i prawdopodobieństwo mutacji. Wartości te uzależnione są od funkcji przystosowania osobnika oraz całego bieżącego pokolenia. W oparciu o informacje pochodzące tylko z bieżącego pokolenia, metody selekcji osobników do puli rodzicielskiej mają za zadanie dokonanie wyboru osobników najbardziej obiecujących. Odpowiedni poziom mutacji powinien zwiększyć stopień zróżnicowania populacji, umożliwiając badanie całego obszaru poszukiwania oraz zabezpieczyć algorytm przed przedwczesną zbieżnością.

W niniejszej pracy rozważana będzie propozycja autora, polegająca na wykorzystaniu sterowników rozmytych, których zadaniem będzie dostosowanie prawdopodobieństwa selekcji i mutacji poszczególnych osobników w zależności od wartości funkcji przystosowania osobnika oraz od kierunku i szybkości zmian średniej wartości funkcji przystosowania populacji.

Dostosowanie prawdopodobieństwa selekcji polegać będzie na zwiększeniu prawdopodobieństwa selekcji osobników o wyższej niż przeciętna wartości funkcji przystosowania z pokolenia o szybko rosnącej średniej wartości funkcji przystosowania (zwiększenie liczby potomków w następnym pokoleniu). Osobniki o niższej niż przeciętna wartości funkcji przystosowania z pokoleń o malejącej średniej wartości funkcji przystosowania uzyskają mniejsze prawdopodobieństwo selekcji (zmniejszenie liczby potomków w następnym pokoleniu).

Dostosowanie prawdopodobieństwa mutacji polegać będzie na zwiększeniu prawdopodobieństwa mutacji osobników o niższej niż przeciętna wartości funkcji przystosowania z pokolenia o malejącej średniej wartości funkcji przystosowania (zwiększenie zróżnicowania osobników i zdolności algorytmu do przeszukiwania przestrzeni rozwiązań). Osobniki o wyższej niż przeciętna wartości funkcji przystosowania z pokolenia o rosnącej średniej wartości funkcji przystosowania uzyskają większe prawdopodobieństwo selekcji (zmniejszenie zróżnicowania osobników i zdolności algorytmu do przeszukiwania przestrzeni rozwiązań).

3. Teoretyczne podstawy działania zmodyfikowanego algorytmu genetycznego

3.1. Biologiczna inspiracja kierowania ewolucją osobników

Algorytmy genetyczne naśladują procesy ewolucji zachodzące w przyrodzie. Ewolucja dąży do uzyskania osobników jak najlepiej przystosowanych do środowiska, w którym żyją. Człowiek od dawna usiłował uzyskać wpływ na procesy ewolucji, aby osiągnąć własne cele, na przykład w hodowli zwierząt. Podstawową metodą działania była selekcja osobników i krzyżowanie ich w celu uzyskania potomstwa. W procesie selekcji wybierane były osobniki wykazujące pożądane przez hodowców cechy. Selekcji dokonywano różnymi metodami np.: poprzez organizowanie wystaw zwierząt hodowlanych lub wystaw psów i kotów, w których eksperci oceniali poszczególnych osobników i decydowali o ich wyborze do dalszej hodowli. Oprócz oceny ekspertów możliwe są i inne metody selekcji np.: wybór najszybszego lub najbardziej wytrzymałego konia w wyniku wyścigów. Hodowcy spodziewali się, że w wyniku krzyżowania osobników o określonych cechach ich potomstwo również posiadać będzie te cechy, które są pożądane. Dzięki tym zabiegom udało się wyhodować zwierzęta posiadające określone cechy np. wśród bydła spotyka się krowy mleczne i bydło rzeźne.

W algorytmach genetycznych zasada ta zostanie zrealizowana za pomocą sterownika rozmytego, który pełnić będzie rolę eksperta, oceniającego poszczególnych osobników pod kątem przydatności do procesów reprodukcji. Sterownik rozmyty oceniać będzie osobników w oparciu o ściśle określoną bazę reguł i w oparciu o ocenę modyfikowane będzie prawdopodobieństwo selekcji osobnika do puli rodzicielskiej.

Inną metodą wpływania na procesy ewolucji jest inżynieria genetyczna. Przykładem jej stosowania jest modyfikowanie określonych cech roślin uprawnych. Polega ona na zmianie genu określającego poszczególne cechy rośliny. Dzięki takiej modyfikacji można zmienić wygląd, kształt, kolor lub smak roślin albo zwiększyć ich odporność na mróz lub choroby. W inżynierii genetycznej wymagana jest dobra znajomość genotypu modyfikowanej rośliny – należy dokładnie wiedzieć, który gen należy zmienić i jaką nową wartość powinien on uzyskać. W algorytmach genetycznych na ogół nie jest znana wartość ani położenie genu, który należy zmodyfikować, dlatego geny osobników zmieniane będą w sposób losowy. Każdy z osobników będzie oceniony przez sterownik rozmyty i w zależności od oceny zmodyfikowane zostanie prawdopodobieństwo mutacji poszczególnych jego genów. Mutacje

osobników nieposiadających pożądaných cech będą przeprowadzane częściej niż osobników, posiadających określone cechy.

3.2. Definicje związane z kierowaniem ewolucją osobników

Teoretyczne podstawy funkcjonowania algorytmów genetycznych opierają się na pojęciu schematów i binarnej reprezentacji chromosomów. Teoria schematów opracowana została przez Hollanda [Hol68][Hol75].

Schemat powstaje poprzez wprowadzenie do alfabetu genów symbolu „*”, oznaczającego wartość nieistotną. Schemat reprezentuje wszystkie chromosomy, w których na wszystkich pozycjach nieoznaczonych symbolem „*” wartość chromosomu jest zgodna z wartością schematu. Schematy posiadają dwie istotne właściwości: rząd i długość definiującą. Przyjmijmy definicje pojęć związanych z algorytmami genetycznymi, używanych w dalszej części tej pracy.

Definicja

Schematem S nazywamy łańcuch zawierający k elementów, zdefiniowany nad alfabetem $\{0,1,*\}$, reprezentujący wszystkie chromosomy o liczbie genów równej k i identyczne z S na wszystkich pozycjach ustalonych, tzn. pozycjach zawierających „0” lub „1” w schemacie S .

Definicja

Chromosom ch_i jest *dopasowany* do schematu S wtedy, gdy:

- liczba elementów schematu S równa jest liczbie genów chromosomu ch_i ,
- chromosom ch_i i schemat S są identyczne na wszystkich pozycjach ustalonych, tzn. na pozycjach zawierających „0” lub „1” w schemacie S .

Definicja

Przystosowanie schematu S jest to średnia wartość przystosowania wszystkich chromosomów dopasowanych do schematu S .

Schematy, których przystosowanie jest większe niż średnia wartość przystosowania chromosomów w pokoleniu nazywane będą schematami dobrze przystosowanymi.

Definicja

Rząd schematu S (oznaczany symbolem $o(S)$) jest to liczba pozycji ustalonych w schemacie S , tzn. pozycji zawierających „0” lub „1”.

Schematy o małych, w stosunku do długości chromosomów, wartościach rzędu nazywane będą schematami niskiego rzędu.

Definicja

Długość definiująca $\delta(S)$ schematu jest to odległość między pierwszym a ostatnim ustalonym elementem schematu.

Schematy o małych, w stosunku do długości chromosomów, wartościach długości definiującej nazywane będą schematami wąskimi lub schematami o małej długości definiującej.

Na podstawie doświadczeń sformułowano podstawowe twierdzenie algorytmów genetycznych – twierdzenie o schematach [Gol98]:

Twierdzenie o schematach

„Wąskie, niskiego rzędu i dobrze przystosowane schematy rozprzestrzeniają się w kolejnych pokoleniach zgodnie z wykładniczym prawem wzrostu”.

W dalszej części tego rozdziału sformułowane i udowodnione zostanie twierdzenie o schematach w algorytmach genetycznych, w których zastosowano sterownik rozmyty zwięzający do kierowania ewolucją osobników.

Dla ułatwienia opisu zjawisk zachodzących w czasie działania zmodyfikowanego algorytmu genetycznego przyjmijmy, że algorytmy zmodyfikowane i algorytm elementarny operują na tej samej populacji osobników i przy jednakowej liczebności osobników wybieranych do puli rodzicielskiej. Zdefiniujmy następujące nowe pojęcia:

Definicja

Kierowanie procesem ewolucji osobników w algorytmie genetycznym – wpływanie na przebieg algorytmu poprzez zmianę zdolności poszczególnych osobników do reprodukcji i mutacji, w celu uzyskania założonej relacji pomiędzy możliwością przeszukiwania przestrzeni rozwiązań (eksploatacja) i możliwością odkrywania nowych obszarów przeszukiwania (eksploracja).

Definicja

Adaptacja prawdopodobieństwa selekcji – proces wyznaczania zmodyfikowanej wartości prawdopodobieństwa selekcji osobnika do puli rodziców, występujący w czasie działania

algorytmu, uwzględniający zarówno średnie wartości funkcji dopasowania populacji jak i indywidualne cechy każdego z osobników.

Definicja

Adaptacja prawdopodobieństwa mutacji – proces wyznaczania zmodyfikowanej wartości prawdopodobieństwa mutacji osobnika, występujący w czasie działania algorytmu, uwzględniający zarówno średnie wartości funkcji dopasowania populacji jak i indywidualne cechy każdego z osobników.

Definicja

Współczynnik adaptacji prawdopodobieństwa selekcji (wps) – współczynnik wyznaczany przez sterownik rozmyty zwężający, którego wartość wpływa na prawdopodobieństwo selekcji osobnika $ps(ch_i)$ do puli rodziców. Zmodyfikowana wartość prawdopodobieństwa selekcji osobnika $ps'(ch_i)$ do puli rodziców obliczana jest na podstawie wzoru:

$$ps'(ch_i) = wps * ps(ch_i) \quad (3.1)$$

Definicja

Współczynnik adaptacji prawdopodobieństwa mutacji (wpm) – współczynnik wyznaczany przez sterownik rozmyty zwężający, którego wartość wpływa na prawdopodobieństwo mutacji $pm(ch_i)$ każdej pozycji w chromosomie osobnika. Zmodyfikowana wartość prawdopodobieństwa mutacji osobnika $pm'(ch_i)$ obliczana jest na podstawie wzoru:

$$pm'(ch_i) = wpm * pm(ch_i) \quad (3.2)$$

Definicja

Względna skuteczność modyfikacji prawdopodobieństwa selekcji (η_{sel}) – iloraz oczekiwanej liczby osobników o wartości funkcji przystosowania wyższej niż wartość średnia w danym pokoleniu, wybranych do puli rodzicielskiej w algorytmie zmodyfikowanym i oczekiwanej liczby osobników o wartości funkcji przystosowania wyższej niż wartość średnia w danym pokoleniu, wybranych do puli rodzicielskiej w algorytmie elementarnym.

$$\eta_{sel} = \frac{L'_s}{L_s}, \quad (3.3)$$

gdzie:

L_s' - oczekiwana liczba osobników o wartości funkcji przystosowania wyższej niż wartość średnia w danym pokoleniu, wybranych do puli rodzicielskiej w algorytmie zmodyfikowanym,

L_s - oczekiwana liczba osobników o wartości funkcji przystosowania wyższej niż wartość średnia w danym pokoleniu, wybranych do puli rodzicielskiej w algorytmie elementarnym.

Definicja

Względna skuteczność modyfikacji prawdopodobieństwa mutacji osobników dobrze przystosowanych (η_{mut}^L) – iloraz oczekiwanej liczby mutacji genów w osobnikach, o wartości funkcji przystosowania wyższej niż wartość średnia w danym pokoleniu w algorytmie zmodyfikowanym i oczekiwanej liczby mutacji genów w osobnikach, o wartości funkcji przystosowania wyższej niż wartość średnia w danym pokoleniu w algorytmie elementarnym.

$$\eta_{mut}^L = \frac{L_m'}{L_m}, \quad (3.4)$$

gdzie:

L_m' - oczekiwana liczba mutacji genów osobników o wartości funkcji przystosowania wyższej niż wartość średnia w danym pokoleniu w algorytmie zmodyfikowanym,

L_m - oczekiwana liczba mutacji genów osobników o wartości funkcji przystosowania wyższej niż wartość średnia w danym pokoleniu w algorytmie elementarnym.

Definicja

Względna skuteczność modyfikacji prawdopodobieństwa mutacji osobników źle przystosowanych (η_{mut}^G) – iloraz oczekiwanej liczby mutacji genów w osobnikach, o wartości funkcji przystosowania niższej niż wartość średnia w danym pokoleniu w algorytmie zmodyfikowanym i oczekiwanej liczby mutacji genów w osobnikach, o wartości funkcji przystosowania niższej niż wartość średnia w danym pokoleniu w algorytmie elementarnym.

$$\eta_{mut}^G = \frac{G_m'}{G_m}, \quad (3.5)$$

gdzie:

G_m' - oczekiwana liczba mutacji genów osobników o wartości funkcji przystosowania niższej niż wartość średnia w danym pokoleniu w algorytmie zmodyfikowanym,

G_m - oczekiwana liczba mutacji genów osobników o wartości funkcji przystosowania niższej niż wartość średnia w danym pokoleniu w algorytmie elementarnym.

Definicja

Względna skuteczność modyfikacji prawdopodobieństwa mutacji (η_{mut}) – różnica między względną skutecznością modyfikacji prawdopodobieństwa mutacji osobników źle przystosowanych a względną skutecznością modyfikacji prawdopodobieństwa mutacji osobników dobrze przystosowanych.

$$\eta_{mut} = \frac{G'}{G} - \frac{L'}{L}. \quad (3.6)$$

Zdefiniujmy pewną klasę sterowników rozmytych, wykorzystywanych do modyfikacji parametrów algorytmu genetycznego:

Definicja

Sterownikiem rozmytym zwężającym nazywać będziemy sterownik rozmyty, wykorzystywany do modyfikacji prawdopodobieństwa selekcji lub mutacji osobników w algorytmie genetycznym posiadający następujące właściwości:

1. dla modyfikacji prawdopodobieństwa selekcji wartość wyjściowa wyznaczana przez sterownik musi przyjmować wartości:
 - a. większe od jedności dla osobników o wartości funkcji przystosowania powyżej średniej w pokoleniach, w których średnia wartość funkcji przystosowania rośnie w stosunku do pokolenia poprzedniego,
 - b. równe jedności dla osobników o wartości funkcji przystosowania równej średniej w pokoleniach, w których średnia wartość funkcji przystosowania nie zmienia się w stosunku do pokolenia poprzedniego,
 - c. z przedziału od zera do jedności dla osobników o wartości funkcji przystosowania poniżej średniej w pokoleniach, w których średnia wartość funkcji przystosowania zmniejsza się w stosunku do pokolenia poprzedniego,
2. dla modyfikacji prawdopodobieństwa mutacji wartość wyjściowa wyznaczana przez sterownik musi przyjmować wartości:
 - a. większe od jedności dla osobników o wartości funkcji przystosowania mniejszej od średniej w pokoleniach, w których średnia wartość funkcji przystosowania zmniejsza się w stosunku do pokolenia poprzedniego,

- b. równe jedności dla osobników o wartości funkcji przystosowania równej średniej w pokoleniach, w których średnia wartość funkcji przystosowania nie zmienia się stosunku do pokolenia poprzedniego,
- c. z przedziału od zera do jedności dla osobników o wartości funkcji przystosowania powyżej średniej w pokoleniach, w których średnia wartość funkcji przystosowania zwiększa się w stosunku do pokolenia poprzedniego.

W oparciu o zdefiniowane powyżej pojęcia sformułujemy i udowodnimy prawdziwość twierdzenia o schematach dla zmodyfikowanego algorytmu genetycznego.

Twierdzenie 3.1. o schematach w zmodyfikowanym algorytmie genetycznym

Jeżeli schemat w algorytmie genetycznym, w którym zastosowano kierowanie procesem ewolucji za pomocą sterownika rozmytego zwężającego, jest niskiego rzędu, o małej długości definiującej i dobrze przystosowany, to rozprzestrzenia się on w kolejnych pokoleniach nie wolniej niż wynika to z wykładniczego prawa wzrostu.

W celu udowodnienia prawdziwości powyższego twierdzenia sformułujemy lematy o schematach.

Lemat 1 o schematach

Jeżeli schemat w algorytmie genetycznym, w którym zastosowano kierowanie procesem ewolucji za pomocą sterownika rozmytego zwężającego, modyfikującego prawdopodobieństwo selekcji osobnika do puli rodzicielskiej, jest niskiego rzędu, o małej długości definiującej i dobrze przystosowany, to oczekiwana liczba m' jego reprezentantów, zwiększa się w kolejnych pokoleniach nie wolniej niż wynika to z wykładniczego prawa wzrostu.

Lemat 2 o schematach

Jeżeli schemat w algorytmie genetycznym, w którym zastosowano kierowanie procesem ewolucji za pomocą sterownika rozmytego zwężającego, modyfikującego prawdopodobieństwo selekcji osobnika do puli rodzicielskiej, jest wysokiego rzędu, o dużej długości definiującej i źle przystosowany, to oczekiwana liczba m' jego reprezentantów, zmniejsza się w kolejnych pokoleniach nie wolniej niż wynika to z wykładniczego prawa wzrostu.

Lemat 3 o schematach

Jeżeli schemat w algorytmie genetycznym, w którym zastosowano kierowanie procesem ewolucji za pomocą sterownika rozmytego zwężającego, modyfikującego prawdopodobieństwo mutacji osobnika, jest niskiego rzędu, o małej długości definiującej i dobrze przystosowany, to oczekiwana liczba m' jego reprezentantów, zwiększa się w kolejnych pokoleniach nie wolniej niż wynika to z wykładniczego prawa wzrostu.

Lemat 4 o schematach

Jeżeli schemat w algorytmie genetycznym, w którym zastosowano kierowanie procesem ewolucji za pomocą sterownika rozmytego zwężającego, modyfikującego prawdopodobieństwo mutacji osobnika, jest wysokiego rzędu, o dużej długości definiującej i źle przystosowany, to oczekiwana liczba m' jego reprezentantów, zmniejsza się w kolejnych pokoleniach nie wolniej niż wynika to z wykładniczego prawa wzrostu.

Aby udowodnić prawdziwość powyższych twierdzeń rozważmy wpływ reprodukcji, krzyżowania i mutacji na oczekiwaną liczbę reprezentantów schematu w kolejnych pokoleniach.

Dowody lematów 1 i 2 zostaną przedstawione w podrozdziale 3.3. Dowody lematów 3 i 4 przedstawione zostaną w podrozdziale 3.5. Dowód twierdzenia 3.1. zostanie przeprowadzony w podrozdziale 3.6.

3.3. Wpływ reprodukcji na oczekiwaną liczbę reprezentantów schematów

Początek dowodu lematu 1 i lematu 2 o schematach.

Założmy, że w elementarnym algorytmie genetycznym w pokoleniu t w populacji $P(t)$ złożonej z N osobników znajduje się

$$m = m(S, t) \quad (3.7)$$

reprezentantów schematu S . Podczas reprodukcji osobniki podlegają replikacji z prawdopodobieństwem

$$p_{si} = \frac{F(ch_i)}{\sum_{i=1}^N F(ch_i)}, \quad (3.8)$$

gdzie:

p_{si} – jest prawdopodobieństwem selekcji do puli rodzicielskiej osobnika i ,

N - jest liczbą osobników w populacji,

$F(ch_i)$ - jest wartością funkcji przystosowania osobnika i , dla $i=1...N$.

Uwzględniając współczynnik adaptacji prawdopodobieństwa selekcji, w zmodyfikowanym algorytmie genetycznym prawdopodobieństwo replikacji określone jest wzorem

$$p'_{si} = wps_i * \frac{F(ch_i)}{\sum_{i=1}^N F(ch_i)}, \quad (3.9)$$

gdzie: wps_i jest współczynnikiem modyfikacji prawdopodobieństwa selekcji osobnika i .

Tak więc, jeżeli w populacji $P(t)$ algorytmu elementarnego, w pokoleniu t znajduje się $m = m(S, t)$ reprezentantów schematu S , to w pokoleniu $(t+1)$ możemy oczekiwać $m(S, t+1)$ reprezentantów schematu S , przy czym

$$m(S, t+1) = m(S, t) * N * \frac{F(S)}{\sum_{i=1}^N F(ch_i)}, \quad (3.10)$$

gdzie:

$$F(S) = \frac{1}{m} \sum_{j=1}^m F(ch_j), \text{ dla } ch_j \in S, \quad (3.11)$$

jest średnią wartością funkcji przystosowania dla m osobników będących reprezentantami schematu S w pokoleniu t .

W algorytmie zmodyfikowanym oczekiwana liczba reprezentantów $m'(S, t+1)$ schematu S , w pokoleniu $(t+1)$ wyrażona jest wzorem:

$$m'(S, t+1) = m'(S, t) * N * wps(S) * \frac{F(S)}{\sum_{i=1}^N F(ch_i)}, \quad (3.12)$$

gdzie:

$$F(S) = \frac{1}{m'} \sum_{j=1}^{m'} F(ch_j), \text{ dla } ch_j \in S, \quad (3.13)$$

jest średnią wartością funkcji przystosowania dla m' osobników będących reprezentantami schematu S w pokoleniu t ,

$wps(S)$ - współczynnik adaptacji prawdopodobieństwa selekcji schematu S w pokoleniu t ,

$$wps(S) = \frac{\sum_{j=1}^{m'} wps_j * F(ch_j)}{\sum_{j=1}^{m'} F(ch_j)}, \text{ dla } ch_j \in S. \quad (3.14)$$

Jeżeli średnia funkcja przystosowania populacji

$$\bar{F} = \sum_{i=1}^N \frac{F(ch_i)}{N}, \quad (3.15)$$

to możemy zależność (3.10) wyrazić w postaci

$$m(S, t+1) = m(S, t) * \frac{F(S)}{\bar{F}}, \quad (3.16)$$

natomiast zależność (3.12) w postaci

$$m'(S, t+1) = m'(S, t) * wps(S) * \frac{F(S)}{\bar{F}}. \quad (3.17)$$

Jak widać z (3.17), po zastosowaniu operatora replikacji oczekiwana liczba m' reprezentantów dowolnego schematu S w algorytmach, w których zastosowano rozmyte kierowanie prawdopodobieństwem selekcji za pomocą sterownika rozmytego zwężającego, w kolejnych pokoleniach podlega zmianom ilościowym wynikającym z wartości współczynnika modyfikacji prawdopodobieństwa selekcji $wps(S)$ właściwego dla danego schematu. Wartości współczynnika $wps(S)$ dobierane są przez sterownik rozmyty zwężający spośród domkniętego zbioru wartości i nie zakłócają wykładniczego prawa wzrostu a jedynie go skalują. W celu określenia kierunku tego skalowania należy przeprowadzić badania zależności wartości współczynnika $wps(S)$ dla różnych klas schematów.

3.3.1. Wpływ współczynnika adaptacji prawdopodobieństwa selekcji

W eksperymentach przeprowadzonych przez autora współczynniki wps_i wyznaczane były przez sterownik rozmyty zwężający dla każdego osobnika ($i=1...N$). Jako wielkości wejściowe sterownika przyjęto:

- przyrost funkcji przystosowania populacji

$$\Delta F_{pop} = F_{pop}^{(T)} - F_{pop}^{(T-1)}, \quad (3.18)$$

gdzie: $F_{pop}^{(T)}, F_{pop}^{(T-1)}$ - wartości funkcji przystosowania populacji w pokoleniu T i $(T-1)$,

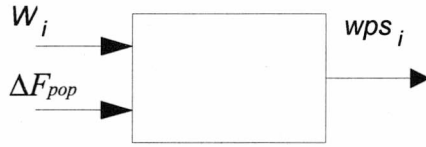
- wskaźnik stopnia przystosowania osobnika

$$W_i = F_i^{(T)} - F_{pop}^{(T)}, \quad (3.19)$$

gdzie: $F_i^{(T)}$ - wartości funkcji przystosowania osobnika.

Wartości współczynników wps_i wyznaczane były przez sterownik rozmyty zwężający z przyrostu funkcji przystosowania populacji ΔF_{pop} oraz z wartości wskaźnika stopnia przystosowania osobnika W_i w oparciu o bazę reguł. W konstrukcji sterownika rozmytego zakłada się, że wartości współczynników wps_i mogą przyjmować dodatnie wartości z

określonego przedziału zawierającego liczbę 1 (w niniejszej pracy przyjęto zakres wartości współczynników $wps_i \in <0,8, 1,1>$).



Rys. 3.1. Schemat ideowy sterownika rozmytego wykorzystywanego do adaptacji prawdopodobieństwa selekcji

Niech $\mu(\Delta F_{pop} \in \Delta F_{pop^k})$ oznacza stopień przynależności wartości ΔF_{pop} do zbioru wartości ΔF_{pop^k} .

Niech $\mu(W_i \in (W_i)_j)$ oznacza stopień przynależności wartości W_i do zbioru wartości W_{ij} .

Bazę reguł postaci

$$\begin{aligned}
 R^{(11)} &: IF(\Delta F_{pop} \in \Delta F_{pop^1}) AND (W_i \in (W_i)_1) THEN (y_{11} \in Y_{11}) \\
 R^{(jk)} &: IF(\Delta F_{pop} \in \Delta F_{pop^k}) AND (W_i \in (W_i)_j) THEN (y_{jk} \in Y_{jk}) \\
 R^{(JK)} &: IF(\Delta F_{pop} \in \Delta F_{pop^K}) AND (W_i \in (W_i)_J) THEN (y_{JK} \in Y_{JK})
 \end{aligned} \tag{3.20}$$

możemy przedstawić w postaci tabeli 3.1.

		$\Delta F_{pop} \in$					
$W_i \in$		ΔF_{pop^1}	ΔF_{pop^2}	...	ΔF_{pop^k}	...	ΔF_{pop^K}
	$(W_i)_1$	y_{11}	y_{12}	...	y_{1k}	...	y_{1K}
	$(W_i)_2$	y_{21}	y_{22}	...	y_{2k}	...	y_{2K}
	...	...	...	...	...	...	...
	$(W_i)_j$	y_{j1}	y_{j2}	...	y_{jk}	...	y_{jK}
	...	...	...	...	...	...	...
	$(W_i)_J$	y_{J1}	y_{J2}	...	y_{Jk}	...	y_{JK}

Tabela 3.1. Przynależności ΔF_{pop} , W_i i y_{jk} do poszczególnych przedziałów w sterowniku wykorzystywanym do adaptacji prawdopodobieństwa selekcji

Ponieważ przynależności ΔF_{pop} do ΔF_{pop^k} oraz W_i do $(W_i)_j$ są wzajemnie niezależne, stopień spełnienia przesłanki obliczyć możemy z zależności:

$$\mu_{jk} = \mu(W_i \in (W_i)_j) * \mu(\Delta F_{pop} \in \Delta F_{pop^k}). \tag{3.21}$$

Wartość współczynnika wps_i możemy obliczyć na podstawie zależności:

$$wps_i = \frac{\sum_{k=1}^K \sum_{j=1}^J y_{jk} * \mu_{jk}}{\sum_{k=1}^K \sum_{j=1}^J \mu_{jk}}. \quad (3.22)$$

Wartość współczynnika modyfikacji prawdopodobieństwa selekcji schematu $wps(S)$, obliczana jest jako wartość średnia współczynników modyfikacji prawdopodobieństwa selekcji wps_i osobników będących reprezentantami schematu S .

Przykładową bazę reguł, wykorzystywaną w eksperymentach autora przedstawiono w Tabeli 3.2. Tabela ta zawiera wartości rozmyte współczynników wps_i .

ΔF_{pop}

	UM	ZERO	DM	DD
W_i M	BM	M	S	S
S	M	S	S	D
D	M	S	D	D

Tabela 3.2. Wartości rozmyte współczynników wps_i dla sterownika do adaptacji prawdopodobieństwa selekcji

Symbole w tabeli oznaczają:

- zbiór wartości rozmytych przyrostu funkcji przystosowania populacji ΔF_{pop} ,
 - o UM – ujemny mały,
 - o ZERO – około zera,
 - o DM – dodatni mały,
 - o DD – dodatni duży,
- zbiór wartości rozmytych wskaźnika stopnia przystosowania osobnika W_i i współczynników wps_i
 - o BM – bardzo mały,
 - o M – mały,
 - o S – średni,
 - o D – duży.

Z konstrukcji sterownika rozmytego wynika, że wartość współczynnika $wps(S)$ musi przyjmować wartości:

- >1 dla schematów reprezentowanych przez osobniki o wartości funkcji przystosowania powyżej średniej w pokoleniach, w których średnia wartość funkcji przystosowania rośnie w stosunku do pokolenia poprzedniego (w bazie reguł pola na białym tle). Schematy te reprezentowane są przez osobniki, których obecność w puli rodzicielskiej jest pożądana oraz spełniają warunek dobrego przystosowania

$$\frac{F(S)}{\overline{F}} > 1. \quad (3.23)$$

Oczekiwana liczba osobników w kolejnym pokoleniu algorytmu elementarnego zmieni się zgodnie z zależnością

$$\frac{m(S, t+1)}{m(S, t)} = \frac{F(S)}{\overline{F}}, \quad (3.24)$$

natomiast w algorytmie z adaptacją prawdopodobieństwa selekcji

$$\frac{m'(S, t+1)}{m'(S, t)} = wps(S) * \frac{F(S)}{\overline{F}}. \quad (3.25)$$

Przyjmując, że w pokoleniu t w algorytmie elementarnym i w algorytmie z adaptacją prawdopodobieństwa selekcji była jednakowa liczba reprezentantów schematu S , możemy napisać:

$$\frac{m'(S, t+1)}{m(S, t+1)} = wps(S). \quad (3.26)$$

Wartość $wps(S) > 1$ powoduje, że w puli rodzicielskiej znajdzie się więcej osobników reprezentujących dany schemat, a więc zwiększenie prędkości rozprzestrzeniania się tego schematu. Oczekiwana liczba $m'(S, t+1)$ osobników reprezentujących dany schemat S w algorytmie z adaptacją prawdopodobieństwa selekcji będzie większa niż oczekiwana liczba $m(S, t+1)$ osobników reprezentujących dany schemat S w algorytmie elementarnym, tak więc

$$m'(S, t+1) > m(S, t+1). \quad (3.27)$$

- $=1$ dla schematów reprezentowanych przez osobniki o wartości funkcji przystosowania równej średniej w pokoleniach, w których średnia wartość funkcji przystosowania nie zmienia się stosunku do pokolenia poprzedniego (w bazie reguł pola na szarym tle). Wartość $wps(S) = 1$ powoduje, że w puli rodzicielskiej znajdzie się tyle samo osobników reprezentujących dany schemat S , ile byłoby ich w algorytmie elementarnym. Sterownik rozmyty nie wpływa na prędkość rozprzestrzeniania się schematu. Oczekiwana liczba $m'(S, t+1)$ osobników reprezentujących dany schemat S w algorytmie z adaptacją prawdopodobieństwa selekcji będzie równa oczekiwanej liczbie osobników $m(S, t+1)$ reprezentujących dany schemat S w algorytmie elementarnym, tak więc

$$m'(S, t+1) = m(S, t+1). \quad (3.28)$$

- <1 dla schematów reprezentowanych przez osobniki o wartości funkcji przystosowania poniżej średniej w pokoleniach, w których średnia wartość funkcji przystosowania zmniejsza się w stosunku do pokolenia poprzedniego (w bazie reguł pola na ciemnym tle). Schematy te reprezentowane są przez osobniki, których obecność w puli rodzicielskiej nie jest pożądana oraz spełniają warunek złego przystosowania

$$\frac{F(S)}{\overline{F}} < 1. \quad (3.29)$$

Oczekiwana liczba osobników w kolejnym pokoleniu algorytmu elementarnego zmieni się zgodnie z zależnością

$$\frac{m(S, t+1)}{m(S, t)} = \frac{F(S)}{\overline{F}}, \quad (3.30)$$

natomiast w algorytmie z adaptacją prawdopodobieństwa selekcji

$$\frac{m'(S, t+1)}{m'(S, t)} = wps(S) * \frac{F(S)}{\overline{F}}. \quad (3.31)$$

Przyjmując, że w pokoleniu t w algorytmie elementarnym i w algorytmie z adaptacją prawdopodobieństwa selekcji była jednakowa liczba reprezentantów schematu S , możemy napisać:

$$\frac{m'(S, t+1)}{m(S, t+1)} = wps(S). \quad (3.32)$$

Wartość $wps(S) < 1$ powoduje, że w puli rodzicielskiej znajdzie się mniej osobników reprezentujących dany schemat, a więc przyspiesza zanikanie tego schematu. Oczekiwana liczba $m'(S, t+1)$ osobników reprezentujących dany schemat S w algorytmie z adaptacją prawdopodobieństwa selekcji będzie mniejsza od oczekiwanej liczby $m(S, t+1)$ osobników reprezentujących dany schemat S w algorytmie elementarnym, tak więc

$$m'(S, t+1) < m(S, t+1). \quad (3.33)$$

W oparciu o powyższe rozważania możemy sformułować następujący wniosek.

Wniosek 3.1.

Oczekiwana liczba reprezentantów schematu S w algorytmach, w których zastosowano sterownik rozmyty zwięzający do adaptacji prawdopodobieństwa selekcji, w stosunku do oczekiwanej liczby reprezentantów tego schematu w algorytmie elementarnym jest:

- większa, jeżeli wartość funkcji przystosowania schematu S jest większa niż średnia wartość funkcji przystosowania dla wszystkich osobników,

- równa, jeżeli wartość funkcji przystosowania schematu S jest równa średniej wartości funkcji przystosowania dla wszystkich osobników,
- mniejsza, jeżeli wartość funkcji przystosowania schematu S jest mniejsza niż średnia wartość funkcji przystosowania dla wszystkich osobników.

Koniec dowodu lematu 1 i lematu 2 o schematach.

Zastosowanie sterownika rozmytego zwięzającego do kierowania procesem selekcji osobników do puli rodzicielskiej powinno spowodować przyspieszenie rozprzestrzeniania się schematów o wartości funkcji przystosowania wyższej niż wartość średnia w pokoleniu oraz przyspieszenie zanikania schematów o wartości funkcji przystosowania niższej niż wartość średnia w pokoleniu. Sterownik nie wpływa na rozprzestrzenianie się lub zanikanie schematów o wartości funkcji przystosowania równej średniej w pokoleniu.

3.4. Wpływ krzyżowania osobników na oczekiwaną liczbę reprezentantów schematów

Przyjmując długość ciągu kodowego osobnika l stwierdzamy, że ciąg ten może być przerywany w $(l-1)$ punktach, a więc prawdopodobieństwo zniszczenia schematu wyniesie:

$$p_d = \frac{\delta(S)}{l-1}, \quad (3.34)$$

gdzie: $\delta(S)$ - długość definiująca schematu.

Prawdopodobieństwo przeżycia schematu wyniesie:

$$p_{ds} = 1 - p_d. \quad (3.35)$$

Przyjmując, że krzyżowanie dla określonej pary osobników wykonywane jest z prawdopodobieństwem p_c , dolne oszacowanie prawdopodobieństwa przeżycia schematu

$$p_{ds} \geq 1 - p_c * \frac{\delta(S)}{l-1}. \quad (3.36)$$

Sterownik rozmyty nie wpływa na prawdopodobieństwo krzyżowania. Zakładając niezależność selekcji i krzyżowania możemy stwierdzić, że oczekiwana liczba osobników w kolejnym pokoleniu algorytmu, określona jest na podstawie wzoru:

$$m'(S, t+1) \geq m'(S, t) * wps(S) * \frac{F(S)}{F} (1 - p_c * \frac{\delta(S)}{l-1}). \quad (3.37)$$

Na podstawie powyższej nierówności stwierdzamy, że większe szanse przeżycia mają schematy wąskie (o małej długości definiującej).

3.5. Wpływ mutacji osobników na oczekiwaną liczbę reprezentantów schematów

Początek dowodu lematu 3 i lematu 4 o schematach.

Zakładając, że w elementarnym algorytmie genetycznym każdy z alleli może zostać poddany mutacji z prawdopodobieństwem p_m , prawdopodobieństwo, że allel nie zostanie zmutowany wynosi $(1 - p_m)$. Oznaczając liczbę pozycji ustalonych schematu $o(S)$ prawdopodobieństwo przeżycia mutacji przez schemat w algorytmie elementarnym wyniesie

$$p_{dm} = (1 - p_m)^{o(S)}. \quad (3.38)$$

Uwzględniając współczynnik adaptacji prawdopodobieństwa mutacji $wpm(S)$ dla algorytmu z modyfikowanym prawdopodobieństwem mutacji

$$p'_{dm} = (1 - wpm(S) * p_m)^{o(S)}. \quad (3.39)$$

Dla małych wartości p_m ($p_m \ll 1$) możemy dokonać aproksymacji

$$(1 - p_m)^{o(S)} \cong (1 - o(S) * p_m). \quad (3.40)$$

Wzór na prawdopodobieństwo przeżycia mutacji schematu w czasie mutacji przyjmie postać

$$p_{dm} = (1 - o(S) * p_m) \quad (3.41)$$

dla algorytmu elementarnego, oraz

$$p'_{dm} = (1 - wpm(S) * o(S) * p_m) \quad (3.42)$$

dla algorytmu z modyfikowanym prawdopodobieństwem mutacji.

Jak widać z (3.42), po zastosowaniu operatora mutacji oczekiwana liczba m' reprezentantów dowolnego schematu S w algorytmach, w których zastosowano rozmyte kierowanie prawdopodobieństwem mutacji za pomocą sterownika rozmytego zwężającego, w kolejnych pokoleniach podlega zmianom ilościowym wynikającym z wartości współczynnika modyfikacji prawdopodobieństwa mutacji $wpm(S)$. Wartości współczynnika $wpm(S)$ dobierane są przez sterownik rozmyty zwężający spośród domkniętego zbioru wartości i nie zakłócają wykładniczego prawa wzrostu a jedynie go skalują. W celu określenia kierunku tego skalowania należy przeprowadzić badania zależności wartości współczynnika $wpm(S)$ dla różnych klas schematów.

3.5.1. Wpływ współczynnika adaptacji prawdopodobieństwa mutacji

W eksperymentach przeprowadzonych przez autora współczynniki wpm_i wyznaczone były przez sterownik rozmyty zwężający dla każdego osobnika ($i=1...N$). Jako wielkości wejściowe sterownika przyjęto:

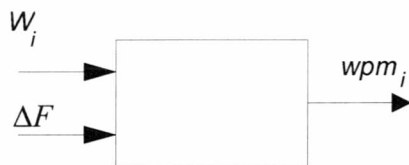
- jakość pokolenia (różnica pomiędzy funkcją przystosowania dla najlepszego z dotychczasowych rozwiązań i średnią funkcją przystosowania pokolenia)

$$\Delta F = F_{\max} - F_{pop}^{(T)}, \quad (3.43)$$

- wskaźnik stopnia przystosowania osobnika (określa na ile dany osobnik jest lepszy niż przeciętny osobnik w danym pokoleniu)

$$W_i = F_i^{(T)} - F_{pop}^{(T)}, \quad (3.44)$$

Wartości współczynników wpm_i wyznaczone są przez sterownik rozmyty z przyrostu funkcji przystosowania populacji ΔF oraz z wartości wskaźnika stopnia przystosowania osobnika W_i w oparciu o bazę reguł. W konstrukcji sterownika rozmytego zakłada się, że wartości współczynników wpm_i mogą przyjmować dodatnie wartości z określonego przedziału zawierającego liczbę 1 (w niniejszej pracy przyjęto zakres wartości współczynników $wpm_i \in <0,8, 1,1>$).



Rys. 3.2. Schemat ideowy sterownika rozmytego wykorzystywanego do adaptacji prawdopodobieństwa mutacji

Niech $\mu(\Delta F \in \Delta F_k)$ oznacza stopień spełnienia przesłanki, że wartość ΔF należy do zbioru wartości ΔF_k . Niech $\mu(W_i \in (W_i)_j)$ oznacza stopień spełnienia przesłanki, że wartość W_i należy do zbioru wartości W_{ij} .

Bazę reguł postaci

$$\begin{aligned} R^{(11)} &: IF(\Delta F \in \Delta F_1) AND (W_i \in (W_i)_1) THEN (y_{11} \in Y_{11}) \\ R^{(jk)} &: IF(\Delta F \in \Delta F_k) AND (W_i \in (W_i)_j) THEN (y_{jk} \in Y_{jk}) \\ R^{(JK)} &: IF(\Delta F \in \Delta F_K) AND (W_i \in (W_i)_J) THEN (y_{JK} \in Y_{JK}) \end{aligned} \quad (3.45)$$

możemy przedstawić w postaci tabeli 3.3.

		$\Delta F \in$					
$W_i \in$		ΔF_1	ΔF_2	...	ΔF_k	...	ΔF_K
	$(W_i)_1$	y_{11}	y_{12}	...	y_{1k}	...	y_{1K}
	$(W_i)_2$	y_{21}	y_{22}	...	y_{2k}	...	y_{2K}
	...	...	...	...	...	...	...
	$(W_i)_i$	y_{i1}	y_{i2}	...	y_{ik}	...	y_{iK}
	...	...	...	...	...	...	...
	$(W_i)_J$	y_{J1}	y_{J2}	...	y_{Jk}	...	y_{JK}

Tabela 3.3. Przynależności ΔF , W_i i y_{jk} do poszczególnych przedziałów w sterowniku wykorzystywanym do adaptacji prawdopodobieństwa mutacji

Ponieważ przynależności ΔF do ΔF_k oraz W_i do $(W_i)_j$ są wzajemnie niezależne, stopień spełnienia przesłanki obliczyć możemy z zależności:

$$\mu_{jk} = \mu(W_i \in (W_i)_j) * \mu(\Delta F \in \Delta F_k). \quad (3.46)$$

Wartość współczynnika wpm_i możemy obliczyć na podstawie zależności:

$$wpm_i = \frac{\sum_{k=1}^K \sum_{j=1}^J y_{jk} * \mu_{jk}}{\sum_{k=1}^K \sum_{j=1}^J \mu_{jk}}. \quad (3.47)$$

Wartość współczynnika modyfikacji prawdopodobieństwa mutacji schematu $wpm(S)$, obliczana jest jako wartość średnia współczynników modyfikacji prawdopodobieństwa mutacji wpm_i osobników będących reprezentantami schematu S .

Przykładową bazę reguł, wykorzystywaną w eksperymentach autora przedstawiono w Tabeli 3.4. Tabela ta zawiera wartości rozmyte współczynników wpm_i .

		ΔF			
		BM	M	D	BD
W_i	M	BD	D	S	S
	S	BD	S	S	M
	D	BD	S	M	M
	BD	BD	S	M	M

Tabela 3.4. Baza reguł dla sterownika realizującego adaptację prawdopodobieństwa mutacji w sterowniku wykorzystywanym do adaptacji prawdopodobieństwa mutacji

Symbole w tabeli oznaczają:

- zbiór wartości rozmytych jakości populacji ΔF ,
 - o BM – bardzo mały,
 - o M – mały,
 - o D – duży,
 - o BD – bardzo duży,
- zbiór wartości rozmytych wskaźników stopnia przystosowania osobnika W_i i współczynników wpm_i
 - o M – mały,
 - o S – średni,
 - o D – duży,
 - o BD – bardzo duży.

Z konstrukcji sterownika rozmytego wynika, że wartość współczynnika $wpm(S)$ musi przyjmować wartości:

- >1 dla schematów reprezentowanych przez osobniki o wartości funkcji przystosowania mniejszej od średniej w pokoleniach, w których średnia wartość funkcji przystosowania zmniejsza się w stosunku do pokolenia poprzedniego (w bazie reguł pola na białym tle). Schematy te reprezentowane są przez osobniki, których obecność w puli rodzicielskiej nie jest pożądana oraz spełniają warunek złego przystosowania

$$\frac{F(S)}{\overline{F}} < 1. \quad (3.48)$$

Prawdopodobieństwo przeżycia schematu S , w danym pokoleniu w algorytmie elementarnym w stosunku do prawdopodobieństwa przeżycia schematu w algorytmie, w którym zastosowano sterownik rozmyty do modyfikowania prawdopodobieństwa mutacji wyniesie:

$$\frac{p_{dm}}{p'_{dm}} = \frac{(1 - o(S) * p_m)}{(1 - o(S) * wpm(S) * p_m)}, \quad (3.49)$$

co oznacza, że jeżeli $wpm(S) > 1$ to $p_{dm} > p'_{dm}$.

Wartość $wpm(S) > 1$ powoduje, że prawdopodobieństwo przeżycia schematu S w algorytmie, w którym zastosowano sterownik rozmyty zwiężający do modyfikowania prawdopodobieństwa mutacji jest niższe niż w algorytmie elementarnym, przez co zwiększa się zdolność algorytmu do eksploracji przestrzeni rozwiązań. Prawdopodobieństwo przeżycia mutacji schematu w algorytmie z adaptacją prawdopodobieństwa mutacji p'_{dm} będzie mniejsze od prawdopodobieństwa przeżycia mutacji danego schematu p_{dm} w algorytmie elementarnym, tak więc

$$p'_{dm} < p_{dm}, \quad (3.50)$$

- $=1$ dla schematów reprezentowanych przez osobniki o wartości funkcji przystosowania równej średniej w pokoleniach, w których średnia wartość funkcji przystosowania nie zmienia się stosunku do pokolenia poprzedniego (w bazie reguł pola na szarym tle). Wartość $wpm(S) = 1$ nie powoduje zmian w prawdopodobieństwie mutacji osobników reprezentujących dany schemat. Prawdopodobieństwo przeżycia mutacji schematu w algorytmie z adaptacją prawdopodobieństwa mutacji p'_{dm} będzie równe prawdopodobieństwu przeżycia mutacji danego schematu p_{dm} w algorytmie elementarnym, tak więc

$$p'_{dm} = p_{dm}, \quad (3.51)$$

- <1 dla schematów reprezentowanych przez osobniki o wartości funkcji przystosowania powyżej średniej w pokoleniach, w których średnia wartość funkcji przystosowania zwiększa się w stosunku do pokolenia poprzedniego (w bazie reguł pola na ciemnym tle). Schematy te reprezentowane są przez osobniki, których obecność w puli rodzicielskiej jest pożądana oraz spełniają warunek dobrego przystosowania

$$\frac{F(S)}{\overline{F}} > 1. \quad (3.52)$$

Prawdopodobieństwo przeżycia schematu S , w danym pokoleniu w algorytmie elementarnym w stosunku do prawdopodobieństwa przeżycia schematu w algorytmie, w którym zastosowano sterownik rozmyty do modyfikowania prawdopodobieństwa mutacji wyniesie:

$$\frac{p_{dm}}{p'_{dm}} = \frac{(1 - o(S) * p_m)}{(1 - o(S) * wpm(S) * p_m)}, \quad (3.53)$$

co oznacza, że jeżeli $wpm(S) < 1$ to $p_{dm} < p'_{dm}$.

Wartość $wpm(S) < 1$ powoduje, że prawdopodobieństwo przeżycia schematu S w algorytmie, w którym zastosowano sterownik rozmyty zwiężający do modyfikowania prawdopodobieństwa mutacji, jest większe niż w algorytmie elementarnym, przez co zwiększa się zdolność algorytmu do eksploatacji przestrzeni rozwiązań. Prawdopodobieństwo przeżycia mutacji schematu w algorytmie z adaptacją prawdopodobieństwa mutacji p'_{dm} będzie większe od prawdopodobieństwa przeżycia mutacji danego schematu p_{dm} w algorytmie elementarnym, tak więc

$$p'_{dm} > p_{dm}. \quad (3.54)$$

W oparciu o powyższe rozważania możemy sformułować następujący wniosek.

Wniosek 3.2.

Prawdopodobieństwo przeżycia schematu S w algorytmach, w których zastosowano sterownik rozmyty zwiężający do adaptacji prawdopodobieństwa mutacji, w stosunku do prawdopodobieństwa przeżycia tego schematu w algorytmie elementarnym jest:

- większe, jeżeli wartość funkcji przystosowania schematu S jest większa niż średnia wartość funkcji przystosowania dla wszystkich osobników,
- równe, jeżeli wartość funkcji przystosowania schematu S jest równa średniej wartości funkcji przystosowania dla wszystkich osobników,

- mniejsze, jeżeli wartość funkcji przystosowania schematu S jest mniejsza niż średnia wartość funkcji przystosowania dla wszystkich osobników.

Koniec dowodu lematu 3 i lematu 4 o schematach.

Zastosowanie sterownika rozmytego zwięzającego powinno spowodować zwiększenie prawdopodobieństwa przeżycia schematów o wysokiej wartości funkcji przystosowania oraz zmniejszenie prawdopodobieństwa przeżycia schematów o niskiej wartości funkcji przystosowania. Sterownik ten powoduje również zmiany w relacjach pomiędzy zdolnością algorytmu do eksploracji i eksploatacji. Jeżeli różnica pomiędzy wartością funkcji przystosowania najlepszego osobnika a średnią wartością funkcji przystosowania jest duża, (np.: w początkowym okresie działania algorytmu) zwiększane jest prawdopodobieństwo mutacji (większy nacisk na eksplorację). Jeżeli różnica pomiędzy wartością funkcji przystosowania najlepszego osobnika a średnią wartością funkcji przystosowania jest mniejsza, (np.: w końcowym okresie działania algorytmu) zmniejszane jest prawdopodobieństwo mutacji (większy nacisk na eksploatację).

Ostatecznie możemy stwierdzić, że oczekiwana liczba reprezentantów schematu S w następnym pokoleniu, otrzymanym w wyniku selekcji, krzyżowania i mutacji, po uwzględnieniu współczynników adaptacji selekcji i mutacji spełnia następującą nierówność:

$$m'(S, t+1) \geq m'(S, t) * wps(S) * \frac{F(S)}{F} (1 - p_c * \frac{\delta(S)}{l-1} - wpm(S) * o(S) * p_m). \quad (3.55)$$

Z zależności (3.55) wynika, że schematy o wyższej niż średnia funkcji przystosowania, niskiego rzędu i o małej długości definiującej w kolejnych pokoleniach będą uzyskiwać rosnącą liczbę reprezentantów. Współczynniki adaptacji prawdopodobieństwa selekcji wps i prawdopodobieństwa mutacji wpm zaproponowane w pracy powinny zapewnić większą zbieżność algorytmów. Uwzględniając zależności (3.27), (3.28), (3.33), (3.50), (3.51), (3.54), możemy napisać, że oczekiwana liczba reprezentantów schematu S w następnym pokoleniu, w algorytmie z adaptacją prawdopodobieństwa selekcji i mutacji $m'(S, t+1)$ nie powinna być mniejsza niż oczekiwana liczba reprezentantów schematu S w następnym pokoleniu, w algorytmie elementarnym $m(S, t+1)$, a więc spełnia następującą nierówność

$$m'(S, t+1) \geq m(S, t+1). \quad (3.56)$$

Na podstawie zależności (3.55) i (3.56) oraz wniosków 3.1 i 3.2 możemy stwierdzić prawdziwość lematu 1, lematu 2, lematu 3 oraz lematu 4.

3.6. Dowód zmodyfikowanego twierdzenia o schematach

Jeżeli:

1. Liczba osobników w algorytmie jest stała w każdym pokoleniu,
2. Wprowadzone jest kierowanie procesem ewolucji osobników przy pomocy sterownika rozmytego zwięzającego, oceniającego każdego z osobników,
3. Sterownik rozmyty powoduje:
 - a. zwiększenie prawdopodobieństwa selekcji osobników reprezentujących schematy, których funkcja przystosowania jest wyższa niż średnia funkcja przystosowania wszystkich osobników w danym pokoleniu,
 - b. nie wpływa na prawdopodobieństwo selekcji osobników reprezentujących schematy, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,
 - c. zmniejszenie prawdopodobieństwa selekcji osobników reprezentujących schematy, których funkcja przystosowania jest niższa niż średnia funkcja przystosowania wszystkich osobników w danym pokoleniu,
 - d. zwiększenie prawdopodobieństwa mutacji osobników reprezentujących schematy, których funkcja przystosowania jest niższa niż średnia funkcja przystosowania wszystkich osobników w danym pokoleniu,
 - e. nie wpływa na prawdopodobieństwo mutacji osobników reprezentujących schematy, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,
 - f. zmniejszenie prawdopodobieństwa mutacji osobników reprezentujących schematy, których funkcja przystosowania jest wyższa niż średnia funkcja przystosowania wszystkich osobników w danym pokoleniu,

To:

Wąskie, niskiego rzędu i dobrze przystosowane schematy w algorytmach genetycznych, w których zastosowano kierowanie procesem ewolucji za pomocą sterownika rozmytego zwięzającego, rozprzestrzeniają się w kolejnych pokoleniach nie wolniej niż wynika to z wykładniczego prawa wzrostu.

Dowód:

Początek dowodu twierdzenia 3.1 dla adaptacji prawdopodobieństwa selekcji.

Wyberzmy dla kaźdego osobnika schemat o najwyźszej wartości funkcji przystosowania i przyporządkujmy osobnika do tego schematu. Kaźdy spośród osobników w dowolnym pokoleniu $t \in (1, t_{\max})$ algorytmu genetycznego moźe zostać zakwalifikowany do jednej spośród 3 grup:

1. osobników reprezentujących schemat, którego funkcja przystosowania jest wyźsza od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, oczekiwaną liczbę tych osobników oznaczmy L_s ,
2. osobników reprezentujących schemat, którego funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, oczekiwaną liczbę tych osobników oznaczmy J_s ,
3. osobników reprezentujących schemat, którego funkcja przystosowania jest niźsza od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, oczekiwaną liczbę tych osobników oznaczmy G_s .

Moźemy napisać, że dla elementarnego algorytmu genetycznego (przy spełnieniu założenia 3)

$$N = L_s + J_s + G_s, \quad (3.57)$$

gdzie: N – liczba wszystkich osobników w populacji,

L_s – oczekiwana liczba osobników, których funkcja przystosowania jest wyźsza od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,

J_s – oczekiwana liczba osobników, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,

G_s – oczekiwana liczba osobników, których funkcja przystosowania jest niźsza od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,

oraz dla algorytmu z modyfikowanym prawdopodobieństwem selekcji

$$N' = L_s' + J_s' + G_s', \quad (3.58)$$

gdzie: N' – liczba wszystkich osobników w populacji,

L_s' – oczekiwana liczba osobników, których funkcja przystosowania jest wyźsza od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,

J_s' – oczekiwana liczba osobników, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,

G_s' – oczekiwana liczba osobników, których funkcja przystosowania jest niźsza od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu.

Z lematu 1 o schematach wynika, że

$$\bigwedge_{2 \leq t \leq t_{\max}} (L_s')_t \geq (L_s)_t, \quad (3.59)$$

czyli

$$\bigwedge_{2 \leq t \leq t_{\max}} (\Delta L_s)_t = (L_s')_t - (L_s)_t \geq 0. \quad (3.60)$$

Z lematu 2 o schematach wynika, że

$$\bigwedge_{2 \leq t \leq t_{\max}} (G_s')_t \leq (G_s)_t, \quad (3.61)$$

czyli

$$\bigwedge_{2 \leq t \leq t_{\max}} (\Delta G_s)_t = (G_s')_t - (G_s)_t \leq 0. \quad (3.62)$$

Ponieważ sterownik rozmyty nie wpływa na prawdopodobieństwo selekcji osobników, których funkcja przystosowania jest równa średniej wszystkich osobników w danym pokoleniu (założenie 3.b.), to

$$\bigwedge_{2 \leq t \leq t_{\max}} (J_s')_t \cong (J_s)_t, \quad (3.63)$$

czyli

$$\bigwedge_{2 \leq t \leq t_{\max}} (\Delta J_s)_t = (J_s')_t - (J_s)_t \cong 0. \quad (3.64)$$

W całym przebiegu algorytmu różnica globalna wyniesie:

$$\Delta L_s = \sum_{t=2}^{t_{\max}} ((L_s')_t - (L_s)_t) \geq 0, \quad (3.65)$$

$$\Delta J_s = \sum_{t=2}^{t_{\max}} ((J_s')_t - (J_s)_t) \cong 0, \quad (3.66)$$

oraz

$$\Delta G_s = \sum_{t=2}^{t_{\max}} ((G_s')_t - (G_s)_t) \leq 0. \quad (3.67)$$

Ponieważ

$$\Delta J_s \cong 0, \quad (3.68)$$

to

$$\Delta L_s + \Delta G_s \cong 0, \quad (3.69)$$

czyli

$$\Delta L_s = -\Delta G_s. \quad (3.70)$$

Względna skuteczność modyfikacji prawdopodobieństwa selekcji wyniesie

$$\eta_{sel} = \frac{\Delta L_s}{L_s} + 1. \quad (3.71)$$

Z zależności (3.65) wynika, że $\Delta L_s \geq 0$, oznacza to, że w algorytmie z modyfikowanym prawdopodobieństwem selekcji oczekiwana liczba osobników o wartości funkcji przystosowania wyższej niż średnia wybieranych do puli rodzicielskiej, będzie nie mniejsza niż w algorytmie elementarnym

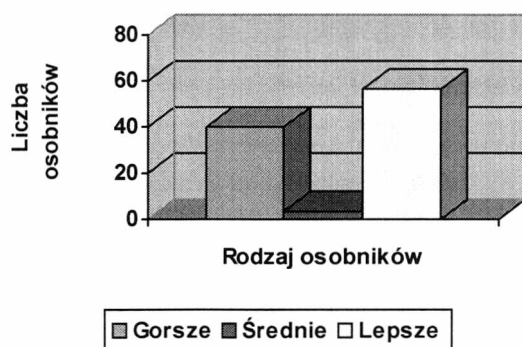
$$\eta_{sel} = \frac{\Delta L_s}{L_s} + 1 \geq 1. \quad (3.72)$$

Konkluzja 1.

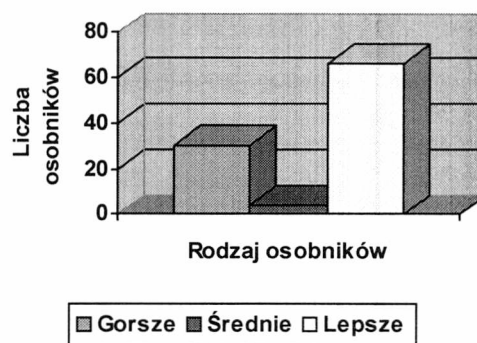
W oparciu o zależność (3.72) możemy stwierdzić, że wąskie, niskiego rzędu i dobrze przystosowane schematy w algorytmach genetycznych, w których zastosowano kierowanie procesem ewolucji za pomocą sterownika rozmytego zwięźającego, poprzez zmianę prawdopodobieństwa selekcji osobników do puli rodzicielskiej, rozprzestrzeniają się w kolejnych pokoleniach nie wolniej niż wynika to z wykładniczego prawa wzrostu. Tak więc, wykazano prawdziwość twierdzenia 3.1 w przypadku adaptacji prawdopodobieństwa selekcji. Koniec dowodu twierdzenia 3.1 dla adaptacji prawdopodobieństwa selekcji.

Dyskusja:

Ponieważ liczba osobników w każdym pokoleniu jest stała, sterownik rozmyty wpływa na ewolucję osobników tak, aby zapewnić większe prawdopodobieństwo przeżycia osobników dobrze przystosowanych, oraz zmniejszenie prawdopodobieństwa przeżycia osobników gorzej przystosowanych. Do puli rodzicielskiej wybieranych będzie więcej osobników dobrze przystosowanych, natomiast mniej osobników gorzej przystosowanych, niż miałyby to miejsce w algorytmie elementarnym. Najszybszy wzrost funkcji przystosowania powinien wystąpić, gdy $\Delta L_s \rightarrow \infty$, natomiast $\Delta G_s \rightarrow 0$, jednak sytuacja taka mogłaby doprowadzić do przedwczesnej zbieżności algorytmu. Na Rys. 3.3. przedstawiono szacunkowy wpływ sterownika rozmytego na oczekiwaną liczbę osobników wybieranych do puli rodzicielskiej (na rysunku Lepsze – oznacza osobniki w wartości funkcji przystosowania wyższej niż średnia w pokoleniu, Średnie - osobniki w wartości funkcji przystosowania równej średniej w pokoleniu, Gorsze - osobniki w wartości funkcji przystosowania niższej niż średnia w pokoleniu).



Algorytm elementarny



Algorytm zmodyfikowany

Rys. 3.3. Wpływ sterownika rozmytego na oczekiwaną liczbę osobników wybieranych do puli rodzicielskiej w algorytmach z modyfikacją prawdopodobieństwa selekcji

Sterownik rozmyty modyfikujący prawdopodobieństwo selekcji osobników do puli rodzicielskiej może być wykorzystany do zmiany zdolności algorytmu do eksploracji/eksploatacji przestrzeni rozwiązań. Poprzez zmianę parametrów sterownika możemy uzyskać zwiększenie lub zmniejszenie zdolności algorytmu do przeszukiwania przestrzeni rozwiązań albo przyspieszenie lub spowolnienie zbieżności algorytmu.

Początek dowodu twierdzenia 3.1 dla adaptacji prawdopodobieństwa mutacji.

Zdarzenia polegające na mutacji poszczególnych genów osobnika są wzajemnie niezależne. Każda spośród mutacji genu w osobniku w dowolnym pokoleniu $t \in (1, t_{\max})$ algorytmu genetycznego może zostać zakwalifikowana do jednej spośród 3 grup:

1. mutacji genów osobników reprezentujących schemat, którego funkcja przystosowania jest wyższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, liczbę tych mutacji oznaczmy L_m ,
2. mutacji genów osobników reprezentujących schemat, którego funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, liczbę tych mutacji oznaczmy J_m ,
3. mutacji genów osobników reprezentujących schemat, którego funkcja przystosowania jest niższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu, liczbę tych mutacji oznaczmy G_m .

Możemy napisać, że dla elementarnego algorytmu genetycznego (w którym dla każdego osobnika wybrano schemat o najwyższej wartości funkcji przystosowania i przyporządkowano osobnika do tego schematu)

$$M = L_m + J_m + G_m, \quad (3.73)$$

gdzie: M – oczekiwana liczba wszystkich mutacji osobników w populacji,

L_m – oczekiwana liczba mutacji osobników, których funkcja przystosowania jest wyższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,

J_m – oczekiwana liczba mutacji osobników, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,

G_m – oczekiwana liczba mutacji osobników, których funkcja przystosowania jest niższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,

oraz dla algorytmu z modyfikowanym prawdopodobieństwem selekcji

$$M' = L_m' + J_m' + G_m', \quad (3.74)$$

gdzie: M' – oczekiwana liczba wszystkich mutacji osobników w populacji,

L_s' – oczekiwana liczba mutacji osobników, których funkcja przystosowania jest wyższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,

J_s' – oczekiwana liczba mutacji osobników, których funkcja przystosowania jest równa średniej funkcji przystosowania wszystkich osobników w danym pokoleniu,

G_s' – oczekiwana liczba mutacji osobników, których funkcja przystosowania jest niższa od średniej funkcji przystosowania wszystkich osobników w danym pokoleniu.

Z konstrukcji sterownika rozmytego zwięźającego, wykorzystywanego do modyfikacji prawdopodobieństwa mutacji wynika, że w algorytmie zmodyfikowanym prawdopodobieństwo mutacji genów osobnika dobrze przystosowanego jest nie wyższe niż w algorytmie elementarnym

$$\bigwedge_{2 \leq t \leq t_{\max}} p_m'(L_t) \leq p_m(L_t). \quad (3.75)$$

Możemy napisać, że dla osobników dobrze przystosowanych oczekiwana liczba mutacji w algorytmie zmodyfikowanym będzie nie większa niż w algorytmie elementarnym

$$\bigwedge_{2 \leq t \leq t_{\max}} (L_m')_t \leq (L_m)_t, \quad (3.76)$$

czyli

$$\bigwedge_{2 \leq t \leq t_{\max}} (\Delta L_m)_t = (L_m')_t - (L_m)_t \leq 0. \quad (3.77)$$

W algorytmie zmodyfikowanym, prawdopodobieństwo mutacji genów osobnika gorzej przystosowanego jest nie mniejsze niż w algorytmie elementarnym

$$\bigwedge_{2 \leq t \leq t_{\max}} p'_m(G)_t \geq p_m(G)_t. \quad (3.78)$$

Możemy napisać, że dla osobników gorzej przystosowanych oczekiwana liczba mutacji w algorytmie zmodyfikowanym będzie nie mniejsza niż w algorytmie elementarnym

$$\bigwedge_{2 \leq t \leq t_{\max}} (G'_m)_t \geq (G_m)_t, \quad (3.79)$$

czyli

$$\bigwedge_{2 \leq t \leq t_{\max}} (\Delta G_m)_t = (G'_m)_t - (G_m)_t \geq 0. \quad (3.80)$$

Ponieważ sterownik rozmyty nie wpływa na prawdopodobieństwo mutacji osobników, których funkcja przystosowania jest równa średniej wszystkich osobników w danym pokoleniu (założenie 3.e.), to

$$\bigwedge_{2 \leq t \leq t_{\max}} p'_m(J)_t \cong p_m(J)_t. \quad (3.81)$$

Możemy napisać, że dla osobników o wartości funkcji przystosowania równej średniej w danym pokoleniu oczekiwana liczba mutacji w algorytmie zmodyfikowanym będzie taka sama jak w algorytmie elementarnym

$$\bigwedge_{2 \leq t \leq t_{\max}} (J'_m)_t \cong (J_m)_t, \quad (3.82)$$

czyli

$$\bigwedge_{2 \leq t \leq t_{\max}} (\Delta J_m)_t = (J'_m)_t - (J_m)_t \cong 0. \quad (3.83)$$

W całym przebiegu algorytmu różnica globalna wyniesie:

$$\Delta L_m = \sum_{t=2}^{t_{\max}} ((L'_m)_t - (L_m)_t) \leq 0, \quad (3.84)$$

$$\Delta J_m = \sum_{t=2}^{t_{\max}} ((J'_m)_t - (J_m)_t) \cong 0, \quad (3.85)$$

$$\Delta G_m = \sum_{t=2}^{t_{\max}} ((G'_m)_t - (G_m)_t) \geq 0. \quad (3.86)$$

Względna skuteczność modyfikacji prawdopodobieństwa mutacji wyrażona jest wzorem

$$\eta_{mut} = \frac{G'_m}{G_m} - \frac{L'_m}{L_m}, \quad (3.87)$$

który po uwzględnieniu (3.77) i (3.80) możemy przedstawić w postaci

$$\eta_{mut} = \frac{\Delta G_m}{G_m} - \frac{\Delta L_m}{L_m}. \quad (3.88)$$

Z zależności (3.84) wynika, że $\Delta L_m \leq 0$, oznacza to, że w algorytmie z modyfikowanym prawdopodobieństwem mutacji za pomocą sterownika rozmytego zwiężającego, oczekiwana

liczba mutacji genów osobników o wartości funkcji przystosowania wyższej niż średnia, będzie nie większa niż w algorytmie elementarnym. Z zależności (3.85) wynika, że $\Delta J_m \cong 0$, oznacza to, że w algorytmie z modyfikowanym prawdopodobieństwem mutacji, oczekiwana liczba mutacji genów osobników o wartości funkcji przystosowania równej średniej, będzie taka sama jak w algorytmie elementarnym. Z zależności (3.86), wynika że $\Delta G_m \geq 0$, oznacza to, że w algorytmie z modyfikowanym prawdopodobieństwem mutacji za pomocą sterownika rozmytego zwężającego, oczekiwana liczba mutacji genów osobników o wartości funkcji przystosowania niższej niż średnia, będzie nie mniejsza niż w algorytmie elementarnym.

$$\eta_{mut} = \frac{\Delta G_m}{G_m} - \frac{\Delta L_m}{L_m} \geq 0. \quad (3.89)$$

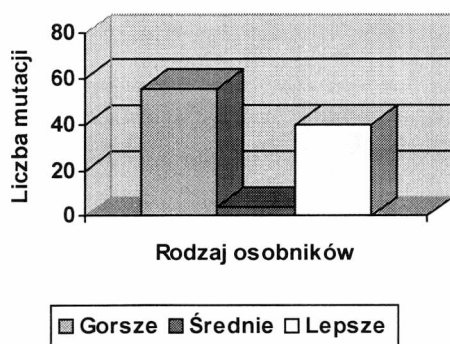
Konkluzja 2.

W oparciu o zależność (3.89) możemy stwierdzić, wąskie, niskiego rzędu i dobrze przystosowane schematy w algorytmach genetycznych, w których zastosowano kierowanie procesem ewolucji za pomocą sterownika rozmytego zwężającego, poprzez zmianę prawdopodobieństwa mutacji osobników, rozprzestrzeniają się w kolejnych pokoleniach nie wolniej niż wynika to z wykładniczego prawa wzrostu. Wykazano więc prawdziwość twierdzenia 3.1 w przypadku adaptacji prawdopodobieństwa mutacji.

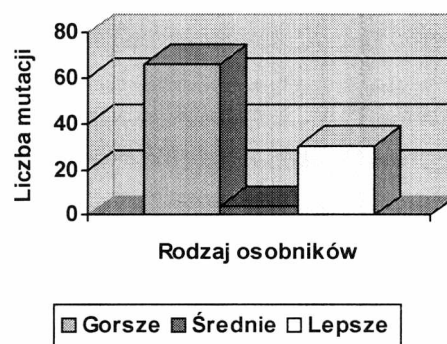
Koniec dowodu twierdzenia 3.1 dla adaptacji prawdopodobieństwa mutacji.

Dyskusja:

Ponieważ zdarzenia polegające na mutacji poszczególnych genów osobników są wzajemnie niezależne, sterownik rozmyty wpływa na ewolucję osobników tak, aby zapewnić większe prawdopodobieństwo mutacji osobników słabo przystosowanych, oraz mniejsze prawdopodobieństwo mutacji genów osobników dobrze przystosowanych. Częściej mutacji poddawane będą osobniki słabo przystosowane, rzadziej natomiast osobniki dobrze przystosowane, niż miałyby to miejsce w algorytmie elementarnym. Najszybszy wzrost funkcji przystosowania powinien wystąpić, gdy $\Delta L_m \rightarrow 0$, natomiast $\Delta G_m \rightarrow \infty$, jednak sytuacja taka mogłaby doprowadzić do ograniczenia zdolności algorytmu do eksploatacji przestrzeni rozwiązań. Na Rys. 3.4. przedstawiono szacunkowy wpływ sterownika rozmytego na oczekiwaną liczbę mutacji osobników (na rysunku Lepsze – oznacza osobniki w wartości funkcji przystosowania wyższej niż średnia w pokoleniu, Średnie - osobniki w wartości funkcji przystosowania równej średniej w pokoleniu, Gorsze - osobniki w wartości funkcji przystosowania niższej niż średnia w pokoleniu).



Algorytm elementarny



Algorytm zmodyfikowany

Rys. 3.4. Wpływ sterownika rozmytego na oczekiwaną liczbę mutacji osobników w algorytmach z modyfikacją prawdopodobieństwa mutacji

Sterownik rozmyty modyfikujący prawdopodobieństwo mutacji poszczególnych genów osobników może być wykorzystany do zmiany zdolności algorytmu do eksploracji/eksploatacji przestrzeni rozwiązań. Poprzez zmianę parametrów sterownika możemy uzyskać zwiększenie lub zmniejszenie zdolności algorytmu do przeszukiwania przestrzeni rozwiązań albo przyspieszenie lub spowolnienie zbieżności algorytmu.

Z konkluzji 1 i konkluzji 2 bezpośrednio wynika prawdziwość twierdzenia 3.1.

3.7. Dowód zbieżności zmodyfikowanego algorytmu genetycznego.

Twierdzenie o zbieżności algorytmu genetycznego z odwzorowaniem zwężającym zostało sformułowane i udowodnione przez Z. Michalewicza [Mic99]. Do udowodnienia zbieżności algorytmu wykorzystano twierdzenie Banacha o punkcie stałym [Żak84]. Aby udowodnić zbieżność zmodyfikowanego algorytmu genetycznego z odwzorowaniem zwężającym, wykorzystującego sterownik rozmyty zwężający do modyfikacji prawdopodobieństwa selekcji i mutacji wykażemy, że spełnia on założenia twierdzenia Banacha o punkcie stałym.

Przyjmijmy następujące założenia:

- kolejne pokolenia osobników otrzymanych w wyniku działania zmodyfikowanego algorytmu genetycznego są elementami pewnej przestrzeni Z ,
- funkcja $\delta : Z \times Z \rightarrow \mathbb{R}$, gdzie $\mathbb{R}$ jest zbiorem liczb rzeczywistych, pozwala mierzyć odległość między parą elementów zbioru Z ,
- $F : Z \rightarrow Z$ jest odwzorowaniem określonym na elementach zbioru Z , przekształcającym pokolenie osobników w nowe pokolenie osobników,
- rozpatrywane będzie zadanie maksymalizacji to znaczy takie, w którym rozwiązanie z_i jest lepsze od z_j wtedy i tylko wtedy, gdy wartość przystosowania osobnika:

$$eval(z_i) > eval(z_j), \quad (3.90)$$

- liczebność populacji jest ustalona i składa się z N osobników.

W oparciu o powyższe sformułować możemy twierdzenie:

Twierdzenie 3.2. o punkcie stałym w zmodyfikowanym algorytmie genetycznym

Jeżeli $\langle Z, \delta \rangle$ jest przestrzenią metryczną zupełną, taką, że jej elementy są kolejnymi pokoleniami zmodyfikowanego algorytmu genetycznego, wykorzystującego sterownik rozmyty zwiężający do modyfikacji prawdopodobieństwa selekcji i mutacji i odwzorowanie F jest odwzorowaniem zwiężającym, to odwzorowanie F ma jednoznaczny punkt stały $z \in Z$ taki, że dla każdego $z_0 \in Z$

$$z = \lim_{i \rightarrow \infty} F^i(z_0), \quad (3.91)$$

przy czym

$$F^0(z_0) = z_0 \text{ oraz } F^{i+1}(z_0) = F(F^i(z_0)). \quad (3.92)$$

Dowód twierdzenia 3.2.

Przyjmijmy funkcję $Eval$ oceniającą populację $P(t)$ w pokoleniu t , taką, że

$$Eval(P) = \frac{1}{N} \sum_{z \in P} eval(z). \quad (3.93)$$

gdzie $eval(z) \geq 0$.

Zbiór Z zawiera wszystkie możliwe populacje, czyli wszystkie wektory

$$\{z_1, \dots, z_N\} \in Z. \quad (3.94)$$

Zdefiniujmy odległość $\delta : Z \times Z \rightarrow \mathbb{R}$, taką że

$$\delta(P_1, P_2) = \begin{cases} 0 & \text{gdy } P_1 = P_2 \\ |1 + Max - Eval(P_1)| + |1 + Max - Eval(P_2)| & \text{gdy } P_1 \neq P_2 \end{cases}, \quad (3.95)$$

gdzie: Max – ograniczenie górne funkcji przystosowania osobnika $eval(z)$ w rozważanej dziedzinie.

Możemy stwierdzić, że

$$0 \leq eval(z) \leq Max, \text{ dla wszystkich osobników } z$$

oraz

$$0 \leq Eval(P) \leq Max, \text{ dla wszystkich pokoleń.}$$

Przyjmijmy trzy dowolne pokolenia P_1, P_2, P_3 . Odległość

$$\delta(P_1, P_2) + \delta(P_2, P_3) = |1 + Max - Eval(P_1)| + |1 + Max - Eval(P_2)| + |1 + Max - Eval(P_2)| + |1 + Max - Eval(P_3)| \quad (3.96)$$

jest nie mniejsza niż

$$\delta(P_1, P_3) = |1 + \text{Max} - \text{Eval}(P_1)| + |1 + \text{Max} - \text{Eval}(P_3)|. \quad (3.97)$$

Przestrzeń $\langle Z, \delta \rangle$ posiada następujące właściwości:

$$\delta(P_1, P_2) = 0 \text{ dla } P_1 = P_2, \quad (3.98)$$

$$\delta(P_1, P_2) \geq 0 \text{ dla każdego pokolenia } P_1 \text{ i } P_2, \quad (3.99)$$

$$\delta(P_1, P_2) = \delta(P_2, P_1), \quad (3.100)$$

$$\delta(P_1, P_2) + \delta(P_2, P_3) \geq \delta(P_1, P_3). \quad (3.101)$$

Możemy stwierdzić, że przestrzeń $\langle Z, \delta \rangle$ jest przestrzenią metryczną.

Ponieważ ciąg pokoleń $P_1, P_2, \dots$ jest ciągiem Cauchy'ego, to znaczy dla każdego ciągu pokoleń $P_1, P_2, \dots$ istnieje takie k , że dla wszystkich $n > k$ zachodzi $P_n = P_k$ (w każdym pokoleniu występuje skończona liczba elementów), to przestrzeń $\langle Z, \delta \rangle$ jest przestrzenią metryczną zupełną.

Przyjmijmy przekształcenie $F: Z \rightarrow Z$, będące pojedynczą iteracją zmodyfikowanego algorytmu genetycznego. Iteracja algorytmu będzie przyjęta przez operator przekształcenia, jeżeli nastąpi poprawa wartości funkcji $\text{Eval}(P)$ to znaczy

$$\text{Eval}(P+1) > \text{Eval}(P). \quad (3.102)$$

Jeżeli iteracja nie zostanie przyjęta przez operator przekształcenia następuje powtórzenie procesu selekcji, krzyżowania i mutacji, aż do momentu uzyskania iteracji akceptowanej przez operator przekształcenia.

Dla iteracji zaakceptowanej przez operator przekształcenia możemy napisać:

$$\text{Eval}(P_1(t)) < \text{Eval}(F(P_1(t))), \quad (3.103)$$

oraz

$$\text{Eval}(P_2(t)) < \text{Eval}(F(P_2(t))). \quad (3.104)$$

Ponieważ

$$\delta(F(P_1(t)), F(P_2(t))) = |1 + \text{Max} - \text{Eval}(F(P_1(t)))| + |1 + \text{Max} - \text{Eval}(F(P_2(t)))|, \quad (3.105)$$

oraz

$$\delta(P_1(t), P_2(t)) = |1 + \text{Max} - \text{Eval}(P_1(t))| + |1 + \text{Max} - \text{Eval}(P_2(t))|, \quad (3.106)$$

możemy napisać

$$\delta(F(P_1(t)), F(P_2(t))) < \delta(P_1(t), P_2(t)). \quad (3.107)$$

Oznacza to, że przekształcenie $F: Z \rightarrow Z$, będące pojedynczą iteracją zmodyfikowanego algorytmu genetycznego jest przekształceniem zwężającym.

Ponieważ przestrzeń $\langle Z, \delta \rangle$ jest przestrzenią metryczną zupełną, a przekształcenie $F: Z \rightarrow Z$, będące pojedynczą iteracją zmodyfikowanego algorytmu genetycznego jest przekształceniem zwężającym, to zmodyfikowany algorytm genetyczny z przekształceniem zwężającym, wykorzystujący sterownik rozmyty zwężający do modyfikacji prawdopodobieństwa selekcji i mutacji jest zbieżny do populacji P^* , która jest jednoznacznym punktem stałym dla wszystkich pokoleń czyli

$$P^* = \lim_{i \rightarrow \infty} F^i(P(0)). \quad (3.108)$$

Punkt P^* jest osiągany, gdy wszystkie osobniki w pokoleniu przyjmą tą samą wartość, a jego osiągnięcie nie zależy od populacji początkowej $P(0)$.

Koniec dowodu twierdzenia 3.2.

3.8. Analiza nakładu obliczeniowego adaptacji prawdopodobieństwa selekcji

W algorytmie genetycznym, w którym wykorzystywany jest sterownik rozmyty do modyfikowania prawdopodobieństwa selekcji, zwiększenie nakładu obliczeniowego potrzebnego do adaptacji prawdopodobieństwa selekcji występuje w procedurze realizującej selekcję osobników do puli rodzicielskiej. Zapis procedury przy użyciu pseudokodu, liczba wykonań poszczególnych instrukcji oraz czas wykonania tych instrukcji przedstawiony jest w tabeli 3.5. Pogrubioną czcionką zaznaczono te fragmenty kodu, które występują z algorytmie zmodyfikowanym i nie występują w algorytmie elementarnym.

	Czas wykonania	Liczba wykonań
select() { for (i=0 to N) do sum+=fitness[i] avg=sum/N for (i=0 to N) do wsp_sel[i]= $\ln(\Delta F_{pop}, W_i)$ for (i=0 to N) do sum_mod+=fitness[i]*wsp_sel[i] for (i=0 to N) do rfitness[i]=fitness[i]*wsp_sel[i]/sum_mod cfitness[0]=rfitness[0] for (i=1 to N) do cfitness[i]=cfitness[i-1]+rfitness[i] for (i=0 to N) do { losuj[i] za pomocą koła ruletki kopiuje[i] do puli rodzicielskiej } }	c1 c2 c3 c4 c5 c6	1 N 1 N 1 N

Tabela 3.5. Pseudokod procedury select()

Różnica pomiędzy czasem wykonania procedury select() w algorytmie zmodyfikowanym i elementarnym wyniesie:

$$T(N)=c1+c2*N+c3+c4*N+c5+c6*N=(c2+c4+c6)*N+(c1+c3+c5) \quad (3.109)$$

gdzie: N – liczba osobników w populacji.

Z analizy procedury select() wynika, że dodatkowy nakład na adaptację prawdopodobieństwa selekcji w procedurze select() jest liniowo zależny od liczby osobników w populacji.

W procedurze select() wywoływana jest procedura $lr(\Delta F_{pop}, W_i)$ wyznaczająca współczynniki modyfikacji prawdopodobieństwa selekcji. Procedura ta wywoływana jest jeden raz dla każdego osobnika.

3.8.1. Analiza nakładu obliczeniowego procedury $lr(\Delta F_{pop}, W_i)$

Procedura $lr(\Delta F_{pop}, W_i)$ nie występuje w elementarnym algorytmie genetycznym, więc czas jej wykonania w całości stanowić będzie dodatkowy nakład obliczeniowy związany z adaptacją prawdopodobieństwa selekcji. Zapis procedury przy użyciu pseudokodu, liczba wykonań poszczególnych instrukcji oraz koszt wykonania tych instrukcji przedstawiony jest w tabeli 3.6.

	Czas wykonania	Liczba wykonań
$lr(\Delta F_{pop}, W_i)$		
{ for (i=0 to N_{we})	c10	N_{we}
do rozmyj(wartosc)	c11	N_{we}
oblicz_przeslanki()	c12	1
wynik = wyostrz()	c13	1
}		

Tabela 3.6. Pseudokod procedury $lr(\Delta F_{pop}, W_i)$

Przed wykonaniem procedury $lr(\Delta F_{pop}, W_i)$ należy wczytać bazę danych relacji rozmytych oraz funkcje przynależności sygnałów wejściowych i wyjściowych sterownika. Czas ten jest stały i niezależny od liczby osobników w populacji (oznaczymy go przez c14).

Czas wykonania procedury $lr(\Delta F_{pop}, W_i)$ wyniesie

$$c2=c10*N_{we}+c11*N_{we}+c12+c13+c14=(c10+c11)*N_{we}+(c12+c13+c14) \quad (3.110)$$

gdzie: N_{we} – liczba sygnałów wejściowych sterownika,

c14 – czas wczytania danych

Z analizy wynika, że nakład obliczeniowy procedury $lr(\Delta F_{pop}, W_i)$ jest liniowo zależny od liczby sygnałów wejściowych sterownika.

W procedurze $lr(\Delta F_{pop}, W_i)$ wywoływane są procedury:

- rozmyj(wartosc) - wyznaczająca wartości rozmyte wielkości wejściowych sterownika. Procedura ta wywoływana jest tyle razy, ile wynosi liczba wartości wejściowych sterownika.
- oblicz_przeslanki() – wyznaczająca wartości stopnia spełnienia wszystkich przesłanek. Procedura ta wywoływana jest jeden raz.
- wyostrz() – wykonująca operację defuzyfikacji (wyznaczania wartości współczynnika modyfikacji prawdopodobieństwa selekcji). Procedura ta wywoływana jest jeden raz.

3.8.2. Analiza nakładu obliczeniowego procedury rozmyj(wartosc)

Procedura rozmyj(wartosc) nie występuje w elementarnym algorytmie genetycznym, więc czas jej wykonania w całości stanowić będzie dodatkowy nakład obliczeniowy związany z adaptacją prawdopodobieństwa selekcji. Zapis procedury przy użyciu pseudokodu, liczba wykonań poszczególnych instrukcji oraz czas wykonania tych instrukcji przedstawiony jest w tabeli 3.7.

	Czas wykonania	Liczba wykonań
rozmyj(wartosc)		
{for (i=0 to liczba_wartosci_ling)	c20	N_{wl}
do		
{if (wartosc <= a) wynik = 0	c21	N_{wl}
if ((wartosc > a) && (wartosc <= m)) wynik = ((wartosc - a)/(m - a))	c22	N_{wl}
if (wartosc == m) wynik = 1	c23	N_{wl}
if ((wartosc > m) && (wartosc < b)) wynik = ((b-wartosc)/(b-m))	c24	N_{wl}
if (wartosc >= b) wynik = 0	c25	N_{wl}
}		
}		

Tabela 3.7. Pseudokod procedury rozmyj(wartosc)

W obliczeniach przyjęto trójkątne funkcje przynależności o charakterystycznych punktach a, b, m, dla innych przebiegów funkcji przynależności czas wykonania może różnić się, co najwyżej o wartość współczynnika. Czas wykonania procedury rozmyj(wartosc) może być różny w zależności od przekazywanych do procedury danych. Najlepszy przypadek (optymistyczny) występuje, gdy wszystkie wartości są mniejsze od a, natomiast najgorszy przypadek (pesymistyczny), gdy wszystkie wartości są większe od b.

Maksymalny czas działania procedury wyniesie

$$c11 = (c20 + c21 + c22 + c23 + c24 + c25) * N_{wl} \quad (3.111)$$

gdzie: N_{wl} – liczba wartości lingwistycznych sygnałów wejściowych sterownika

Z analizy wynika, że górne ograniczenie nakładu obliczeniowego procedury rozmyj(wartosc) jest liniową funkcją liczby wartości lingwistycznych sygnałów wejściowych sterownika.

3.8.3. Analiza nakładu obliczeniowego procedury oblicz_przeslanki()

Procedura oblicz_przeslanki() nie występuje w elementarnym algorytmie genetycznym, więc czas jej wykonania w całości stanowić będzie dodatkowy nakład obliczeniowy związany z adaptacją prawdopodobieństwa selekcji. Zapis procedury przy użyciu pseudokodu, liczba wykonań poszczególnych instrukcji oraz czas wykonania tych instrukcji przedstawiony jest w tabeli 3.8. W tabeli przedstawiona jest procedura obliczająca przesłanki dla dwóch sygnałów wejściowych.

	Czas wykonania	Liczba wykonań
oblicz_przeslanki() {for (i=0 to liczba_wartosci_ling) do {for (j=0 to liczba_wartosc_ling) {for (k=0 to liczba_wartosc_ling) {iloczyn=(s[0].wynik_r[j]*s[1].wynik_r[k]) przeslanka[i][j][k]=iloczyn*relacja[i][j][k] } } } }	c30 c31 c32 c33 c34	N_{wl} $N_{wl} * N_{wl}$ $N_{wl} * N_{wl} * N_{wl}$ $N_{wl} * N_{wl} * N_{wl}$ $N_{wl} * N_{wl} * N_{wl}$

Tabela 3.8. Pseudokod procedury oblicz_przeslanki()

Czas wykonania procedury oblicz_przeslanki() może być różny w zależności od liczby sygnałów wejściowych. Jeżeli przyjmiemy, że wszystkie zmienne lingwistyczne posiadają jednakową liczbę wartości lingwistycznych, to dla dwóch sygnałów wejściowych czas działania procedury wyniesie:

$$c12 = c30 * N_{wl} + c31 * (N_{wl})^2 + c32 * (N_{wl})^3 + c33 * (N_{wl})^3 + c34 * (N_{wl})^3 = \quad (3.112)$$

$$= (c32 + c33 + c34) * (N_{wl})^3 + c31 * (N_{wl})^2 + c30 * N_{wl}$$

gdzie: N_{wl} – liczba wartości lingwistycznych sygnałów sterownika

Z analizy wynika, że nakład obliczeniowy procedury oblicz_przeslanki() jest funkcją potęgową trzeciego stopnia liczby wartości lingwistycznych sygnałów sterownika dla sterownika o dwóch wejściach i jednym wyjściu lub funkcją potęgową stopnia $(N_{we} + N_{wy})$ dla sterownika o N_{we} wejściach i N_{wy} wyjściach.

3.8.4. Analiza nakładu obliczeniowego procedury wyostrz()

Procedura wyostrz() nie występuje w elementarnym algorytmie genetycznym, więc czas jej wykonania w całości stanowić będzie dodatkowy nakład obliczeniowy związany z adaptacją prawdopodobieństwa selekcji. Zapis procedury przy użyciu pseudokodu, liczba wykonań poszczególnych instrukcji oraz czas wykonania tych instrukcji przedstawiony jest w tabeli. W tabeli 3.9. przedstawiona jest procedura obliczająca wartości współczynników modyfikacji prawdopodobieństwa selekcji dla dwóch sygnałów wejściowych.

	Czas wykonania	Liczba wykonań
wyostrz() { for (i=0 to liczba_wartosci_ling) { suma_y=0 waga_y=0 for (j = 0 to liczba_wartosc_ling) { for (k=0 to liczba_wartosc_ling) { suma_y+=przeslanka[i][j][k] } } } suma+=suma_y waga_y=suma_y * y.wart_lin[i].m waga+=waga_y } wynik=(waga/suma) }	c40 c41 c42 c43 c44 c45 c46 c47 c48 c49	N_{wl} N_{wl} N_{wl} $N_{wl} * N_{wl}$ $N_{wl} * N_{wl} * N_{wl}$ $N_{wl} * N_{wl} * N_{wl}$ N_{wl} N_{wl} N_{wl} 1

Tabela 3.9. Pseudokod procedury wyostrz()

Czas wykonania procedury wyostrz() może być różny w zależności od liczby sygnałów wejściowych. Jeżeli przyjmiemy, że wszystkie zmienne lingwistyczne posiadają jednakową liczbę wartości lingwistycznych, to dla dwóch sygnałów wejściowych czas działania procedury wyniesie

$$c13=c40*N_{wl}+c41*N_{wl}+c42*N_{wl}+c43*(N_{wl})^2+c44*(N_{wl})^3+c45*(N_{wl})^3+c46*N_{wl}+c47*N_{wl}+c48*N_{wl}+c49 \quad (3.113)$$

lub

$$c13=(c44+c45)*(N_{wl})^3+c43*(N_{wl})^2+(c40+c41+c42+c46+c47+c48)*N_{wl}+c49 \quad (3.114)$$

gdzie: N_{wl} – liczba wartości lingwistycznych sygnałów sterownika

Z analizy wynika, że nakład obliczeniowy procedury wyostrz() jest funkcją potęgową trzeciego stopnia liczby wartości lingwistycznych sygnałów sterownika.

Eliminując mniej znaczące składniki i podstawiając do wzoru możemy napisać, że nakład obliczeniowy jest postaci

$$O(N * N_{we} * N_{wl} + N_{wl}^{(N_{we} + N_{wy})}) \quad (3.115)$$

gdzie: N – liczba osobników w populacji,

N_{we} – liczba sygnałów wejściowych do sterownika,

N_{wy} – liczba sygnałów wyjściowych do sterownika,

N_{wl} – liczba wartości lingwistycznych sygnałów wejściowych i wyjściowych.

Liczba sygnałów wejściowych oraz liczba wartości lingwistycznych poszczególnych wielkości rozmytych w sterowniku jest stała i nie zmienia się w czasie działania algorytmu.

Na ogół liczba osobników w populacji jest większa niż liczba wejść sterownika oraz liczba wartości lingwistycznych, dlatego w większości przypadków decydujący o nakładzie obliczeniowym będzie pierwszy składnik równania.

Ostatecznie możemy oszacować, że nakład obliczeniowy rozmytego kierowania ewolucją osobników w algorytmach genetycznych jest rzędu $O(N * N_{we} * N_{wl})$.

4. Konstrukcja zmodyfikowanego algorytmu genetycznego

W rozdziale tym zaprezentowana zostanie propozycja budowy zmodyfikowanego algorytmu genetycznego, w którym zastosowano sterownik rozmyty do wyznaczania kierunku ewolucji populacji osobników.

Prowadzone są na świecie badania [Ala95], [Cor], [Gol97b], [Lan96], [Wei] mające na celu usprawnienie funkcjonowania algorytmów genetycznych lub dostosowania ich do sprawniejszego rozwiązania określonych problemów. Algorytmy genetyczne można usprawnić poprzez wykorzystanie wiedzy ekspertów do sterowania przebiegiem procesu ewolucji. Wiedza ekspertów na temat ewolucji ma charakter opisowy, więc do jej reprezentowania wykorzystane zostaną systemy z logiką rozmytą. Efektywność algorytmów genetycznych można zwiększyć poprzez modyfikację ich parametrów takich jak prawdopodobieństwo selekcji i prawdopodobieństwo mutacji.

W większości przypadków do testowania algorytmów genetycznych wykorzystywane są funkcje testowe o różnej liczbie zmiennych i różnym stopniu trudności, jednak są to funkcje ciągłe, w których stosowana jest zasada mówiąca, że prawdopodobieństwo małej zmiany wartości genu w czasie mutacji jest większe niż prawdopodobieństwo dużej zmiany (mutacja genu polega na dodaniu generowanej losowo według określonego rozkładu wartości). Istnieje jednak wiele problemów optymalizacyjnych, w których funkcja optymalizowana jest nieciągła (np.: problem komiwojażera, zero-jedynkowy problem plecakowy). Sposób kodowania osobników powoduje, że niemożliwe lub bardzo trudne jest określenie stopnia podobieństwa rozwiązań reprezentowanych przez poszczególne osobniki (nie można określić, które zmiany genów są małe, a które duże). W optymalizacji tego typu problemów stosowane mogą być operatory mutacji, w których prawdopodobieństwo dowolnej mutacji jest jednakowe. W dalszej części tej pracy chciałbym zwrócić szczególną uwagę na tego typu problemy optymalizacyjne i z tego powodu w operatorze mutacji stosowana będzie zasada jednakowego prawdopodobieństwa dla każdej z możliwych mutacji.

4.1. Modyfikacja prawdopodobieństwa selekcji

Parametrem mającym duży wpływ na efektywność algorytmu genetycznego jest prawdopodobieństwo selekcji. Jest to parametr decydujący o możliwości wyboru osobnika do puli rodzicielskiej. Wraz ze wzrostem prawdopodobieństwa selekcji zwiększa się szansa osobnika na przekazanie swojego materiału genetycznego do następnego pokolenia. Największe prawdopodobieństwo selekcji (szansę przeżycia) mają osobniki najlepiej

przystosowane, jednak do puli rodzicielskiej powinny trafiać również osobniki słabiej przystosowane, aby nie nastąpiła zbyt gwałtowna utrata ich materiału genetycznego i przedwczesna zbieżność algorytmu. Do realizacji tej strategii proponuję modyfikację algorytmu genetycznego, polegającą na wprowadzeniu dodatkowego bloku logiki rozmytej oceniającego poszczególnych osobników w populacji. Zadaniem tego bloku jest ustalenie wartości współczynnika adaptacji prawdopodobieństwa selekcji dla każdego z osobników. Współczynnik ten wyznaczany jest na podstawie różnicy między wartością funkcji przystosowania osobnika a średnią wartością funkcji przystosowania dla bieżącego pokolenia, oraz różnicy między średnią wartością funkcji przystosowania pokolenia bieżącego i poprzedniego. Zmodyfikowane prawdopodobieństwo selekcji osobnika i obliczane jest na podstawie wzoru (3.1), wynikającego z definicji przedstawionej w rozdziale 3:

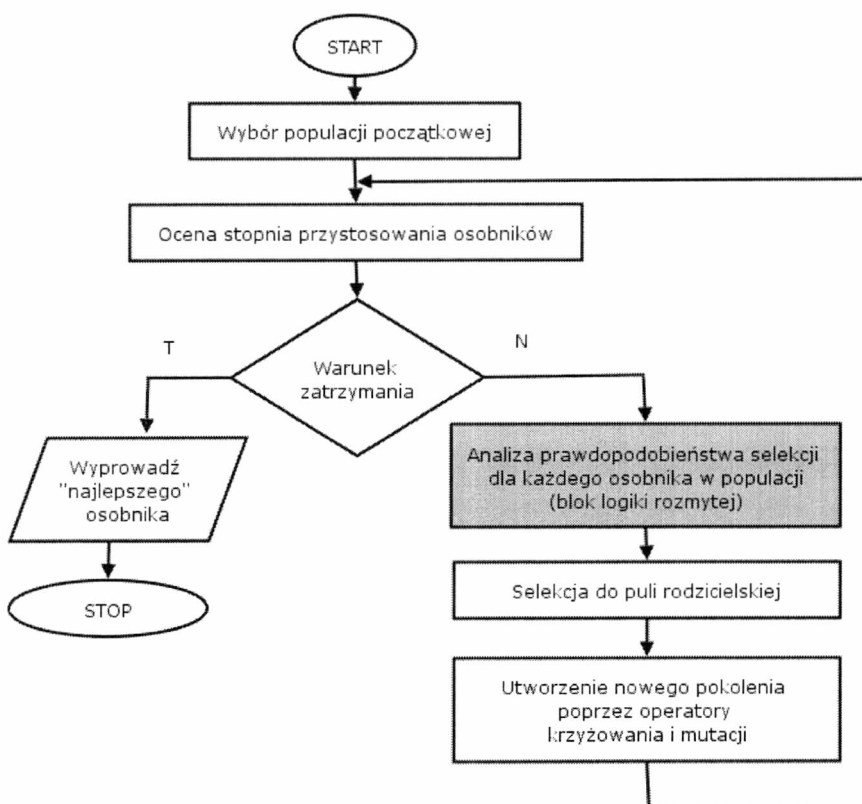
$$ps'(ch_i) = ps(ch_i) * wps_i \text{ dla } i=1...N, \quad (4.1)$$

gdzie:

$ps'(ch_i)$ - zmodyfikowane prawdopodobieństwo selekcji osobnika i ,

$ps(ch_i)$ - prawdopodobieństwo selekcji osobnika i ,

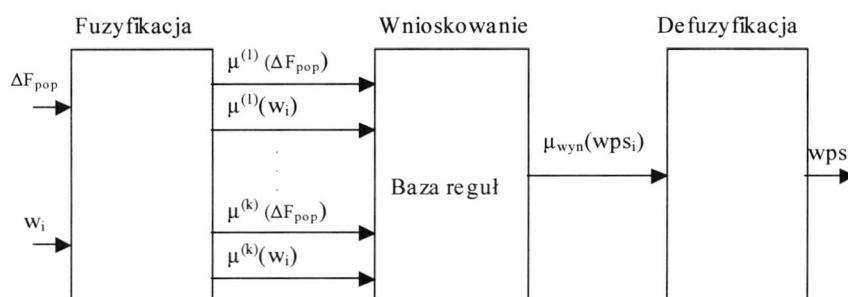
wps_i - współczynnik prawdopodobieństwa selekcji osobnika i .



Rys.4.1. Schemat blokowy zmodyfikowanego algorytmu genetycznego z adaptacją prawdopodobieństwa selekcji

4.1.1. Koncepcja sterownika rozmytego realizującego adaptację prawdopodobieństwa selekcji

W schemacie blokowym zmodyfikowanego algorytmu genetycznego (Rys.4.1.) blok sterownika rozmytego zaznaczono poprzez zacieniowanie. Zadaniem tego sterownika jest ocena poszczególnych osobników w danym pokoleniu i wyznaczenie wartości współczynników modyfikacji prawdopodobieństwa selekcji dla każdego z osobników. Schemat sterownika rozmytego wykorzystywanego do modyfikowania prawdopodobieństwa selekcji pokazany został na Rys.4.2.



Rys.4.2. Schemat sterownika rozmytego wykorzystywanego do modyfikowania prawdopodobieństwa selekcji

Sterownik rozmyty posiada:

* 2 wejścia:

- przyrost funkcji przystosowania populacji (określa szybkość zmian funkcji przystosowania)

$$\Delta F_{pop} = F_{pop}^{(T)} - F_{pop}^{(T-1)}, \quad (4.2)$$

- wskaźnik stopnia przystosowania osobnika (określa na ile dany osobnik jest lepszy niż przeciętny)

$$W_i = F_i^{(T)} - F_{pop}^{(T)}, \quad (4.3)$$

* 1 wyjście: wps_i - współczynnik modyfikacji prawdopodobieństwa selekcji osobnika,

* baza reguł (wartości wps_i) dla sterownika przedstawiona została w Tabeli 4.1

		ΔF_{pop}			
		UM	ZERO	DM	DD
W_i	M	BM	M	S	S
	S	M	S	S	D
	D	M	S	D	D

Tabela 4.1. Baza reguł dla sterownika realizującego adaptację prawdopodobieństwa selekcji

Symbole w tabeli oznaczają:

- zbiór wartości rozmytych przyrostu funkcji przystosowania populacji ΔF_{pop} ,
 - UM – ujemny mały,
 - ZERO – około zera,
 - DM – dodatni mały,
 - DD – dodatni duży,
- zbiór wartości rozmytych wskaźnika stopnia przystosowania osobnika W_i i współczynników wps_i ,
 - BM – bardzo mały,
 - M – mały,
 - S – średni,
 - D – duży.

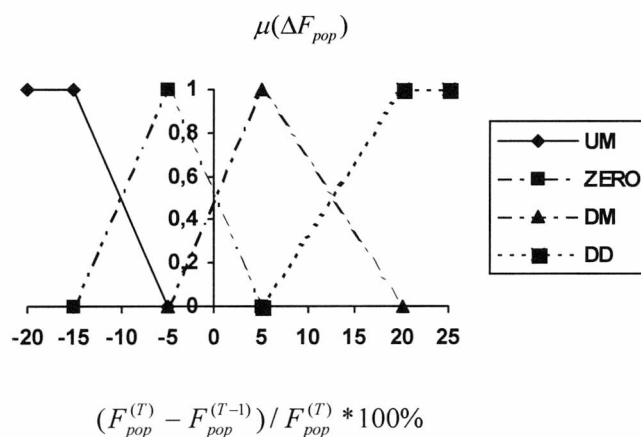
Baza reguł sterownika wykorzystywanego do adaptacji prawdopodobieństwa selekcji odzwierciedla dwie podstawowe zasady, wykorzystywane przez sterownik do oceny poszczególnych osobników:

- jeżeli różnica między wartością funkcji przystosowania osobnika a średnią wartością funkcji przystosowania pokolenia jest dodatnia (osobnik lepiej przystosowany niż średnia w pokoleniu), to prawdopodobieństwo selekcji tego osobnika jest zwiększane, natomiast jeżeli wartość jest ujemna – prawdopodobieństwo jest zmniejszane. Zasada ta powoduje „utwardzenie” selekcji i zwiększenie w puli rodzicielskiej liczby osobników dobrze przystosowanych, co może spowodować zwiększenie zbieżności algorytmu.
- jeżeli różnica między średnią wartością funkcji przystosowania pokolenia bieżącego a średnią wartością przystosowania pokolenia poprzedniego jest dodatnia (może to świadczyć o dobrym kierunku ewolucji), to zwiększamy prawdopodobieństwo selekcji osobników przystosowanych powyżej średniej i zmniejszamy prawdopodobieństwo selekcji osobników przystosowanych gorzej niż średnia w pokoleniu. Zasada ta powoduje, że jeżeli średnia wartość funkcji przystosowania w populacji rośnie, to do puli rodzicielskiej wybranych zostanie więcej osobników dobrze przystosowanych (strategia bardziej elitarna), co spowoduje zwiększenie zdolności eksploatacji algorytmu. Jeżeli średnia wartość funkcji maleje, to do puli rodzicielskiej wybranych zostanie więcej osobników gorzej przystosowanych, dzięki czemu zwiększy się zdolność algorytmu do eksploracji przestrzeni rozwiązań.

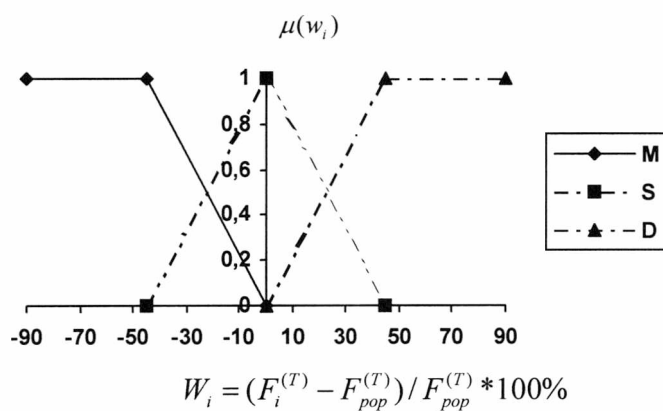
* blok defuzyfikacji – metoda wysokości,

* funkcje przynależności:

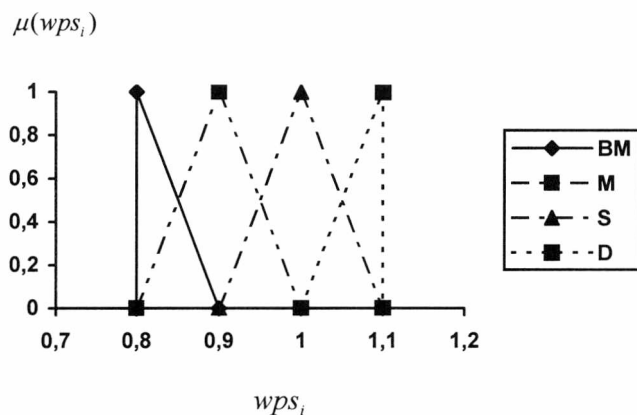
- przyrostu funkcji przystosowania populacji $\Delta F_{pop} = F_{pop}^{(T)} - F_{pop}^{(T-1)}$ - Rys.4.3,
- wskaźnika stopnia przystosowania osobnika $W_i = F_i^{(T)} - F_{pop}^{(T)}$ - Rys.4.4,
- współczynnika adaptacji prawdopodobieństwa selekcji osobnika wps_i - Rys.4.5,



Rys.4.3. Funkcja przynależności $\Delta F_{pop} = F_{pop}^{(T)} - F_{pop}^{(T-1)}$ w sterowniku do adaptacji prawdopodobieństwa selekcji



Rys.4.4. Funkcja przynależności $W_i = F_i^{(T)} - F_{pop}^{(T)}$ w sterowniku do adaptacji prawdopodobieństwa selekcji



Rys.4.5. Funkcja przynależności $\mu(wps_i)$ w sterowniku do adaptacji prawdopodobieństwa selekcji

4.2. Modyfikacja prawdopodobieństwa mutacji

Prawdopodobieństwo mutacji ma mniejszy niż prawdopodobieństwo selekcji, wpływ na efektywność algorytmu genetycznego. Wraz ze wzrostem prawdopodobieństwa mutacji zwiększa się zdolność algorytmu do przeszukiwania nowych obszarów rozwiązań. Największe prawdopodobieństwo mutacji należy przyjmować na początku działania algorytmu (okres przeszukiwania), aby zwiększyć zdolność algorytmu do penetracji nowych obszarów (tworzenia nowych genów). W końcowym okresie działania algorytmu prawdopodobieństwo mutacji może być zmniejszone, aby zwiększyć zdolność algorytmu do eksploatacji przestrzeni rozwiązań. Do realizacji tej strategii proponuję modyfikację algorytmu genetycznego, polegającą na wprowadzeniu dodatkowego bloku logiki rozmytej oceniającego poszczególnych osobników w populacji. Zadaniem tego bloku jest ustalenie wartości współczynnika adaptacji prawdopodobieństwa mutacji dla każdego z osobników. Wartość współczynnika ustalana jest na podstawie różnicy między wartością funkcji przystosowania osobnika a średnią wartością funkcji przystosowania bieżącego pokolenia oraz różnicy między średnią wartością funkcji przystosowania pokolenia bieżącego a średnią wartością funkcji przystosowania pokolenia poprzedniego. Zmodyfikowane prawdopodobieństwo mutacji osobnika i obliczane jest na podstawie wzoru (3.2), wynikającego z definicji przedstawionej w rozdziale 3:

$$pm'(ch_i) = pm(ch_i) * wpm_i \text{ dla } i=1...N, \quad (4.4)$$

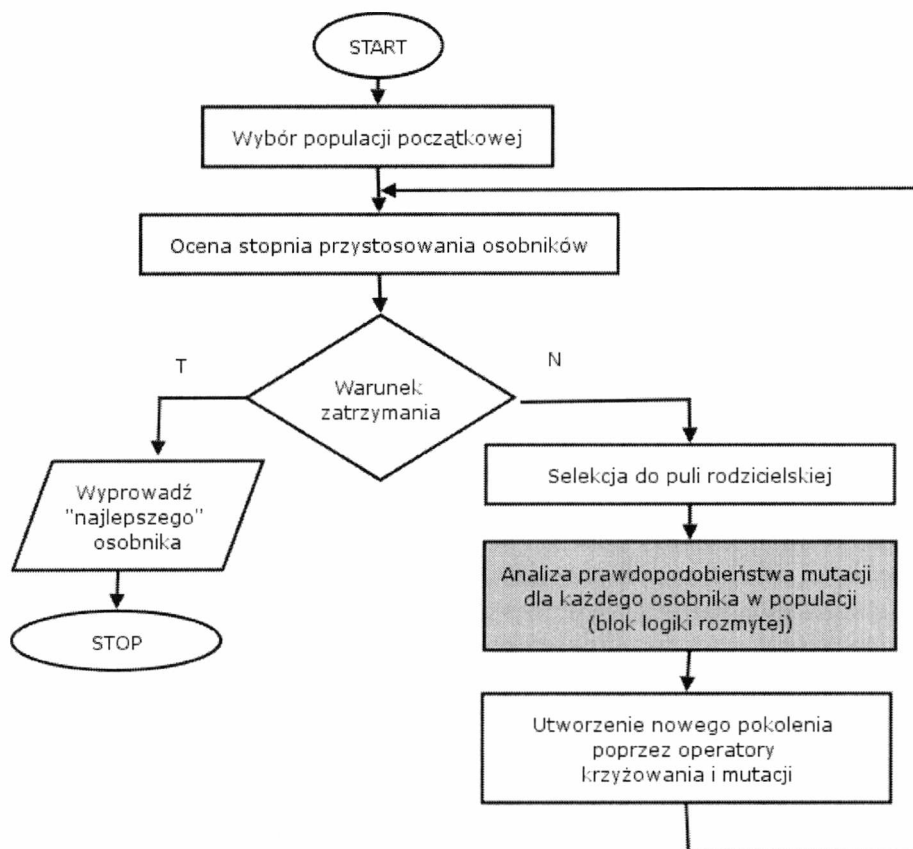
gdzie:

$pm'(ch_i)$ - zmodyfikowane prawdopodobieństwo mutacji osobnika i ,

$pm(ch_i)$ - prawdopodobieństwo mutacji osobnika i ,

wpm_i - współczynnik prawdopodobieństwa mutacji osobnika i .

Schemat blokowy zmodyfikowanego algorytmu genetycznego pokazany został na Rys.4.6.

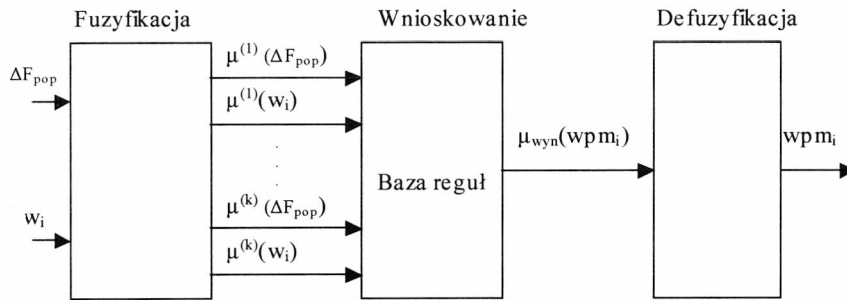


Rys.4.6. Zmodyfikowany algorytm genetyczny z adaptacją prawdopodobieństwa mutacji

4.2.1. Koncepcja sterownika rozmytego realizującego adaptację prawdopodobieństwa mutacji

W schemacie blokowym zmodyfikowanego algorytmu genetycznego (Rys.4.6.) blok sterownika rozmytego zaznaczono poprzez zacieniowanie. Zadaniem tego sterownika jest ocena poszczególnych osobników w danym pokoleniu i wyznaczenie wartości współczynników modyfikacji prawdopodobieństwa mutacji dla każdego z osobników.

Schemat sterownika rozmytego wykorzystywanego do modyfikowania prawdopodobieństwa selekcji pokazany został na Rys.4.7.



Rys.4.7. Schemat sterownika rozmytego wykorzystywanego do modyfikowania prawdopodobieństwa mutacji

Sterownik rozmyty posiada:

* 2 wejścia:

- jakość pokolenia (określa różnicę między funkcją przystosowania najlepszego z dotychczasowych rozwiązań a średnią funkcją przystosowania pokolenia)

$$\Delta F_{\max} = F_{\max} - F_{\text{pop}}^{(T)}, \quad (4.5)$$

- wskaźnik stopnia przystosowania osobnika (określa na ile dany osobnik jest lepszy niż przeciętny w danym pokoleniu)

$$W_i = F_i^{(T)} - F_{\text{pop}}^{(T)}, \quad (4.6)$$

* 1 wyjście: wpm_i - współczynnik zmiany prawdopodobieństwa mutacji,

* baza reguł dla sterownika przedstawiona została w Tabeli 4.2.

		$\Delta F_{\max}$			
		BM	M	D	BD
W_i	M	BD	D	S	S
	S	BD	S	S	M
	D	BD	S	M	M
	BD	BD	S	M	M

Tabela 4.2. Baza reguł dla sterownika realizującego adaptację prawdopodobieństwa mutacji
Symbole w tabeli oznaczają:

- zbiór wartości rozmytych jakości populacji ΔF ,
 - o BM – bardzo mały,
 - o M – mały,
 - o D – duży,
 - o BD – bardzo duży,
- zbiór wartości rozmytych wskaźników stopnia przystosowania osobnika W_i i współczynników wpm_i

- M – mały,
- S – średni,
- D – duży,
- BD – bardzo duży.

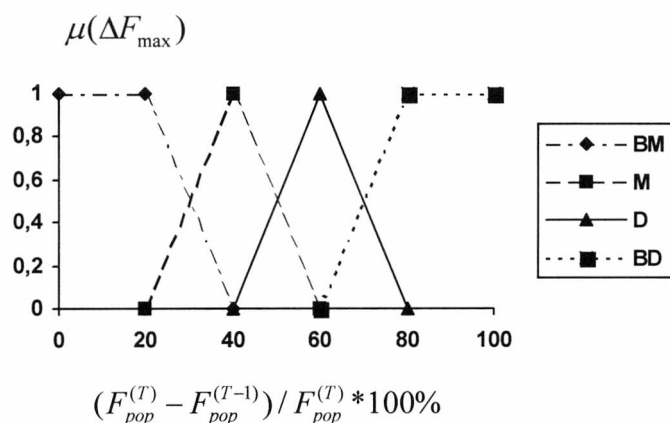
Baza reguł sterownika wykorzystywanego do adaptacji prawdopodobieństwa mutacji odzwierciedla dwie podstawowe zasady, wykorzystywane przez sterownik do oceny poszczególnych osobników:

- jeżeli różnica między wartością funkcji przystosowania osobnika a średnią wartością funkcji przystosowania pokolenia jest dodatnia (osobnik lepiej przystosowany niż średnia w pokoleniu), to prawdopodobieństwo mutacji tego osobnika jest zmniejszane, natomiast jeżeli wartość jest ujemna – prawdopodobieństwo jest zwiększane. Zasada ta powoduje, że osobniki lepiej przystosowane niż średnia w pokoleniu mutowane będą rzadziej, natomiast osobniki przystosowane gorzej niż średnia w pokoleniu mutowane będą częściej niż występowałyby to w algorytmie elementarnym.
- jeżeli różnica między średnią wartością funkcji przystosowania najlepszego spośród dotychczasowych rozwiązań a średnią wartością funkcji przystosowania pokolenia jest mała (może to świadczyć o zbliżaniu się do wartości optymalnej globalnej lub lokalnej), to zwiększamy prawdopodobieństwo mutacji osobników, aby zwiększyć zdolność eksploracji algorytmu i ułatwić opuszczenie obszaru optimum lokalnego. Jeżeli różnica ta jest duża (może to świadczyć o dużym zróżnicowaniu populacji), to zmniejszamy prawdopodobieństwo mutacji, aby zwiększyć zdolność do eksploatacji algorytmu i przyspieszyć zbieżność algorytmu do optimum.

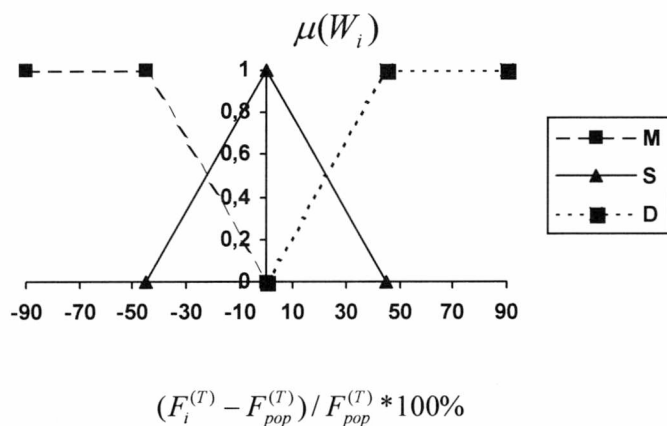
* blok defuzyfikacji – metoda wysokości.

* funkcje przynależności:

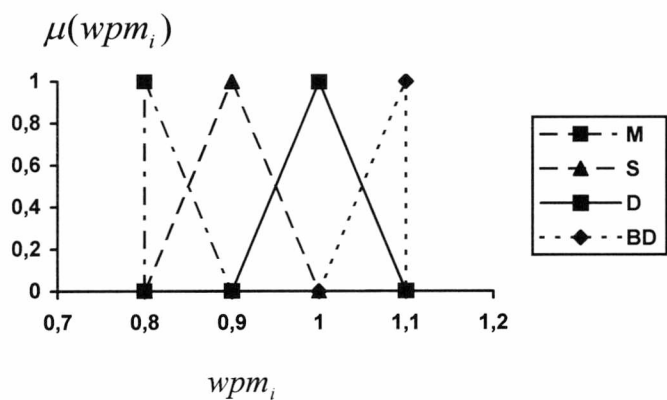
- przyrost funkcji przystosowania populacji $\Delta F_{\max} = F_{\max} - F_{\text{pop}}^{(T)}$ - Rys.4.8,
- wskaźnik stopnia przystosowania osobnika $W_i = F_i^{(T)} - F_{\text{pop}}^{(T)}$ - Rys.4.9,
- współczynnik zmiany prawdopodobieństwa selekcji osobnika wpm_i - Rys.4.10,



Rys.4.8. Funkcja przynależności $\Delta F_{\max} = F_{pop}^{(T)} - F_{pop}^{(T-1)}$ w sterowniku do adaptacji prawdopodobieństwa mutacji



Rys.4.9. Funkcja przynależności $W_i = F_i^{(T)} - F_{pop}^{(T)}$ w sterowniku do adaptacji prawdopodobieństwa mutacji



Rys.4.10. Funkcja przynależności wpm_i w sterowniku do adaptacji prawdopodobieństwa mutacji

4.3. Modyfikacja prawdopodobieństwa selekcji i mutacji

Dotychczas zakładaliśmy, że parametry algorytmu genetycznego (prawdopodobieństwo selekcji i mutacji) wpływają niezależnie na przebieg procesu ewolucji. Jednak w procesach ewolucji wszystkie czynniki wywierają wpływ na przebieg procesu jednocześnie oraz mogą wywierać wpływ również na siebie wzajemnie. Przeanalizujemy również przypadek, w którym jednocześnie modyfikujemy prawdopodobieństwo selekcji i mutacji. Do realizacji tej strategii proponuję modyfikację algorytmu genetycznego, polegającą na wprowadzeniu dwóch dodatkowych bloków logiki rozmytej oceniających poszczególnych osobników w populacji. Zadaniem tych bloków jest:

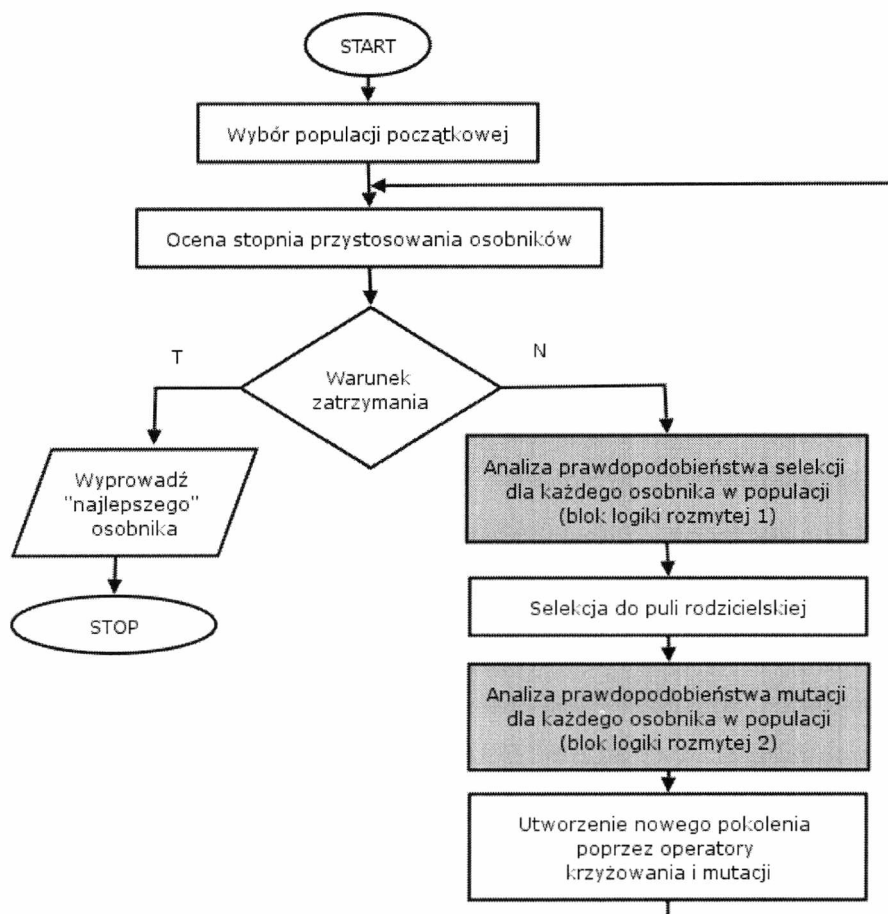
- ustalenie wartości współczynników adaptacji prawdopodobieństwa selekcji dla każdego osobnika na podstawie różnicy między wartością funkcji przystosowania osobnika a średnią wartością funkcji przystosowania bieżącego pokolenia oraz różnicy między średnią wartością funkcji przystosowania pokolenia bieżącego a średnią wartością funkcji przystosowania pokolenia poprzedniego,
- ustalenie wartości współczynników adaptacji prawdopodobieństwa mutacji dla każdego osobnika na podstawie różnicy między wartością funkcji przystosowania osobnika a średnią wartością funkcji przystosowania bieżącego pokolenia oraz różnicy między średnią wartością funkcji przystosowania pokolenia bieżącego i średnią wartością funkcji przystosowania pokolenia poprzedniego.

Zmodyfikowane prawdopodobieństwo selekcji osobnika i oraz prawdopodobieństwo mutacji genów osobnika, obliczane jest na podstawie wzorów (3.1) i (3.2), wynikających z definicji przedstawionych w rozdziale 3:

$$ps'(ch_i) = ps(ch_i) * wps_i \text{ dla } i=1...N \quad (4.7)$$

$$pm'(ch_i) = pm(ch_i) * wpm_i \text{ dla } i=1...N \quad (4.8)$$

Schemat blokowy zmodyfikowanego algorytmu genetycznego pokazany został na Rys.4.11. Bloki ilustrujące zaproponowane w pracy modyfikacje zaznaczono ciemniejszym tłem. Kierowanie ewolucją algorytmu genetycznego będzie realizowane za pomocą sterowników opisanych w rozdziałach 4.1. i 4.2.



Rys. 4.11. Zmodyfikowany algorytm genetyczny z adaptacją prawdopodobieństwa selekcji i mutacji

5. Symulacje działania zmodyfikowanych algorytmów genetycznych

W rozdziale tym omówiono i zaprezentowano wyniki przykładowych eksperymentów, mających na celu sprawdzenie prawdziwości tez postawionych na początku pracy. Eksperymenty przeprowadzono na komputerze z procesorem Athlon 1,33 GHz, wyposażonym w 256 MB RAM, pracującym pod kontrolą systemu RedHat Linux. W eksperymentach wykorzystano program realizujący algorytm genetyczny napisany przez Denisa Cormiera z North Carolina State University i zmodyfikowany przez Sitę S. Raghavana z University of North Carolina. Kod źródłowy tego programu opublikowany jest w pracy [Mic99]. W algorytmie tym stosowane jest jednopunktowe krzyżowanie osobników w losowo wybranym punkcie krzyżowania oraz mutacja polegająca na zamianie losowo wybranego genu na losowo wybraną wartość spośród wszystkich dopuszczalnych wartości genu. Selekcji osobników do puli rodzicielskiej dokonuje się metodą ruletki. Algorytm ten w dalszej części pracy nazywany będzie algorytmem elementarnym i będzie stanowił odniesienie dla porównania wyników uzyskiwanych przez zmodyfikowane algorytmy genetyczne. W elementarnym algorytmie wprowadzone zostały przez autora pracy modyfikacje mające na celu dostosowanie algorytmu do rozwiązywania zadań testowych.

Parametry algorytmów dobrano dla każdego zadania tak, aby zapewnić jego optymalne funkcjonowanie. W tym celu przeprowadzono serię eksperymentów wstępnych, w których każde z zadań testowych rozwiązano przy różnych wartościach parametrów algorytmów. W czasie eksperymentów wstępnych zmieniano wartości prawdopodobieństwa krzyżowania, prawdopodobieństwa mutacji i liczebność populacji, a następnie wybrano ten zestaw parametrów, przy którym uzyskano najlepsze wyniki.

Zasadniczym celem eksperymentów było sprawdzenie, czy zastosowanie sterownika rozmytego oceniającego poszczególnych osobników w populacji do kierowania ewolucją pozwala na poprawę efektywności algorytmów genetycznych.

Cele eksperymentów:

- porównanie wartości funkcji optymalizowanej uzyskiwanej przez algorytm elementarny oraz algorytmy, w których wykorzystano logikę rozmytą do kierowania ewolucją,
- porównanie liczby pokoleń lub liczby wywołań funkcji obliczającej przystosowanie osobnika, potrzebnych do uzyskania założonej wartości funkcji optymalizowanej przez

algorytm elementarny i algorytm, w którym wykorzystano logikę rozmytą do kierowania ewolucją,

- porównanie czasu działania algorytmu elementarnego oraz algorytmów wykorzystujących sterowniki rozmyte do kierowania ewolucją, potrzebnego do uzyskania założonej wartości funkcji optymalizowanej,
- porównanie skuteczności zmodyfikowanych algorytmów genetycznych z innymi metodami optymalizacyjnymi.

Jako podstawowe zadanie testowe wybrano problem komiwojażera. Zadanie to charakteryzuje się tym, że optymalizowana funkcja jest nieciągła. W zadaniu tym nie może być stosowana zasada, że mutacja powodująca niewielką zmianę genów jest bardziej prawdopodobna niż mutacja powodująca dużą zmianę, ponieważ niemożliwe lub trudne jest ustalenie stopnia podobieństwa rozwiązań reprezentowanych przez poszczególne osobniki. W pracy przeprowadzono również badania z wykorzystaniem zero-jedynkowego problemu plecakowego i problemu optymalizacji standardowych funkcji testowych o różnej liczbie zmiennych i poziomie trudności.

W eksperymentach wykorzystano następujące problemy i funkcje testowe:

- problem komiwojażera – problem komiwojażera zaliczany jest do grupy problemów NP-zupełnych. Problem ten polega na znalezieniu najkrótszej drogi, którą komiwojażer musi pokonać, aby odwiedzić jeden raz każde z miast i powrócić do punktu początkowego. Dane testowe dla problemu komiwojażera można pobrać ze strony internetowej projektu „Concorde”, dotyczącego różnych metod rozwiązywania problemu komiwojażera (<http://www.math.princeton.edu/tsp/concorde.html>). Funkcja optymalizowana jest nieciągła.
- zero-jedynkowy problem plecakowy – wybrany do eksperymentów problem polega na znalezieniu zbioru n przedmiotów o maksymalnej wartości P , które możemy „zabrać” do plecaka o ograniczonej pojemności C . Zakłada się, że przedmiot można albo zabrać w całości albo należy go pozostawić. Problemy tego typu zaliczamy do grupy zadań NP-zupełnych, ponadto w zadaniu występują dodatkowe ograniczenia dotyczące warunków nałożonych na rozwiązania. Problem ten możemy traktować jako zadanie polegające na maksymalizacji funkcji f , przy znanych wagach przedmiotów $W[i]$, wartościach przedmiotów $P[i]$, oraz pojemności plecaka C . Z każdym przedmiotem związana jest zmienna $x[i] \in \{0,1\}$, $x[i]=1$ oznacza, że przedmiot i zabieramy do plecaka, natomiast $x[i]=0$ oznacza, że przedmiot i pozostawiamy. Należy więc znaleźć

$$f = \max_{x[i]} \sum_{i=1}^n x[i] * P[i] \text{ dla } i = 1 \dots n, \quad (5.1)$$

przy jednoczesnym spełnieniu warunku

$$\sum_{i=1}^n x[i] * W[i] \leq C. \quad (5.2)$$

W eksperymentach wykorzystano zaproponowane w pracy [Mic99] zadania testowe. Dane testowe wygenerowane w sposób losowy, podzielone zostały na trzy zbiory o różnych korelacjach pomiędzy wartościami i wagami przedmiotów, zgodnie z zasadami zaproponowanymi w pracy [Mic99]:

- nieskorelowane

$W[i]$ – przypadkowe $\langle 1, v \rangle$ o rozkładzie jednostajnym,

$P[i]$ - przypadkowe $\langle 1, v \rangle$ o rozkładzie jednostajnym,

- słabo skorelowane

$W[i]$ – przypadkowe $\langle 1, v \rangle$ o rozkładzie jednostajnym,

$P[i] = W[i] +$ przypadkowe $\langle -r, r \rangle$ o rozkładzie jednostajnym (jeżeli dla pewnego i wartość $P[i] \leq 0$, to taka wartość zysku jest pomijana i obliczenia powtarza się tak długo, aż $P[i] > 0$),

- silnie skorelowane

$W[i]$ – przypadkowe $\langle 1, v \rangle$ o rozkładzie jednostajnym,

$$P[i] = W[i] + r. \quad (5.3)$$

Dane testowe używane w eksperymentach wygenerowano przy wartościach parametrów:

$$v = 100,$$

$$r = 50,$$

liczba przedmiotów $n \in \{50, 100\}$,

pojemność plecaka

$$C = 0,5 \sum_{i=1}^n W[i]. \quad (5.4)$$

- problem optymalizacji funkcji – polega na znalezieniu wartości ekstremum funkcji w określonym przedziale wartości argumentów funkcji:

- funkcja testowa zaproponowana w pracy Michalewicza [Mic99] M1 – jest to prosta funkcja trzech zmiennych posiadająca jedno maksimum globalne,

$$f(x_1, x_2, x_3) = x_1^2 - x_1 * x_2 + x_3 \quad (5.5)$$

gdzie $x_1, x_2, x_3 \in \langle 1, 02 \dots 1000 \rangle$.

Funkcja ta posiada maksimum:

$$f_{\max}(x_1, x_2, x_3) = 999980, \text{ dla } x_1 = 1000, x_2 = 1,02, x_3 = 1000.$$

- funkcja zaproponowana w pracy Michalewicz [Mic99] M2 – jest to funkcja posiadająca jedno maksimum lokalne i jedno maksimum globalne

$$f(v) = |11 * one(v) - 150|. \quad (5.6)$$

gdzie: $one(v)$ podaje liczbę jedynek w łańcuchu v .

Dla długości łańcucha $v=30$ funkcja ta posiada maksimum globalne o wartości 180 i maksimum lokalne o wartości 150.

- funkcja testowa De Jonga F1 – jest to prosta z punktu widzenia optymalizacji funkcja trzech zmiennych z jednym minimum globalnym

$$f(x_1, x_2, x_3) = \sum_{i=1}^3 x_i^2, \quad (5.7)$$

gdzie $-5,12 \leq x_i \leq 5,12$.

Funkcja ta przyjmuje globalną wartość minimalną zero w punkcie $(x_1, x_2, x_3) = (0, 0, 0)$.

- funkcja testowa De Jonga F2 – jest to trudna z punktu widzenia optymalizacji funkcja dwóch zmiennych z jednym minimum globalnym

$$f(x_1, x_2) = 100 * (x_1^2 - x_2)^2 + (1 - x_1)^2, \quad (5.8)$$

gdzie $-2,048 \leq x_i \leq 2,048$.

Funkcja ta przyjmuje globalną wartość minimalną zero w punkcie $(x_1, x_2) = (1, 1)$.

- funkcja Goldsteina-Price'a:

$$f(x_1, x_2) = [1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] * \\ * [30 + (2x_1 + 3x_2)^2 (18 + 32x_1 + 12x_1^2 + 48x_2 + 36x_1x_2 + 27x_2^2)], \quad (5.9)$$

gdzie $-2 \leq x_i \leq 2$.

Funkcja ta przyjmuje globalną wartość minimalną trzy w punkcie $(x_1, x_2) = (0, -1)$.

- funkcja Coldville'a:

$$f(x_1, x_2, x_3, x_4) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 + 90(x_4 + x_3^2)^2 + (1 - x_3)^2 + \\ + 10,1[(x_2 - 1)^2 + (x_4 - 1)^2] - 19,8(x_2 - 1)(x_4 - 1), \quad (5.10)$$

gdzie $-10 \leq x_i \leq 10$.

Funkcja ta przyjmuje globalną wartość minimalną zero w punkcie $(x_1, x_2, x_3, x_4) = (1, 1, 1, 1)$.

- funkcja Q_1 zaproponowana w pracy [Kwa99g]

$$Q_1(x_1, x_2) = h_1 * e^{(-n_1 * ((x_1 - x_1^1)^2 + (x_2 - x_2^1)^2))} + h_2 * e^{(-n_2 * ((x_1 - x_1^1)^2 + (x_2 - x_2^1)^2))}, \quad (5.11)$$

gdzie: $h_1 = 1$, $h_2 = 1,5$, $(x_1^1, x_2^1) = (5, 5)$, $(x_1^2, x_2^2) = (20, 20)$, $n_1 = n_2 = 0,02$.

Funkcja ta posiada maksimum lokalne w punkcie (5, 5) i maksimum globalne w punkcie (20, 20). Jako zakres zmiennych x_1 i x_2 przyjęto przedział [0, 30].

- funkcja Q_2 zaproponowana w pracy [Kwa99g]

$$Q_2(x_1, x_2, x_3, x_4, x_5) = \sum_{i=1}^2 h_i * e^{(-n_i * \sum_{j=1}^5 (x_i - x_j')^2)}, \quad (5.12)$$

gdzie: $h_1 = 1$, $h_2 = 1,5$, $(x_1^1, x_2^1, x_3^1, x_4^1, x_5^1) = (5, 5, 5, 5, 5)$,

$(x_1^2, x_2^2, x_3^2, x_4^2, x_5^2) = (20, 20, 20, 20, 20)$, $n_1 = 0,02$, $n_2 = 0,00225$.

Funkcja ta posiada maksimum lokalne w punkcie (5, 5, 5, 5, 5) i maksimum globalne w punkcie (20, 20, 20, 20, 20). Jako zakres zmiennych x_i przyjęto przedział [0, 30].

- funkcja Q_4 zaproponowana w pracy [Kwa99g]

$$Q_4(x_1, \dots, x_{30}) = \sum_{i=1}^2 h_i * e^{(-n_i * \sum_{j=1}^{30} (x_i - x_j')^2)}, \quad (5.13)$$

gdzie: $h_1 = 1$, $h_2 = 1,5$, $x_i^1 = 5$, $x_i^2 = 20$, $i = 1, \dots, 30$, $n_1 = 0,003$, $n_2 = 0,0004$.

Funkcja ta posiada maksimum lokalne w punkcie $(x_1, \dots, x_{30})$ gdzie $x_i = 5$, $i = 1, \dots, 30$ i maksimum globalne w punkcie $(x_1, \dots, x_{30})$ gdzie $x_i = 20$, $i = 1, \dots, 30$. Jako zakres zmiennych x_i przyjęto przedział [0, 30].

- funkcja Q_5 zaproponowana w pracy [Kwa99g]

$$Q_5(x_1) = \sum_{i=1}^2 h_i * e^{(-n_i * (x_i - x_i')^2)}, \quad (5.14)$$

gdzie: $h_1 = 1$, $h_2 = 1,5$, $x_1^1 = 5$, $x_2^2 = 60$, $n_1 = n_2 = 0,0018$.

Funkcja ta posiada maksimum lokalne w punkcie 5 i maksimum globalne w punkcie 60. Jako zakres zmiennych x_i przyjęto przedział [0, 70].

- funkcja Q_6 zaproponowana w pracy [Kwa99g]

$$Q_6(x_1, \dots, x_{10}) = \sum_{i=1}^2 h_i * e^{(-n_i * \sum_{j=1}^{10} (x_i - x_j')^2)}, \quad (5.15)$$

gdzie: $h_1 = 1$, $h_2 = 1,5$, $x_i^1 = 5$, $x_i^2 = 20$, $i = 1, \dots, 10$, $n_1 = 0,0025$, $n_2 = 0,0004$.

Funkcja ta posiada maksimum lokalne w punkcie $(x_1, \dots, x_{10})$ gdzie $x_i = 5, i=1, \dots, 10$ i maksimum globalne w punkcie $(x_1, \dots, x_{10})$ gdzie $x_i = 20, i=1, \dots, 10$. Jako zakres zmiennych x_1 i x_2 przyjęto przedział $[0, 30]$.

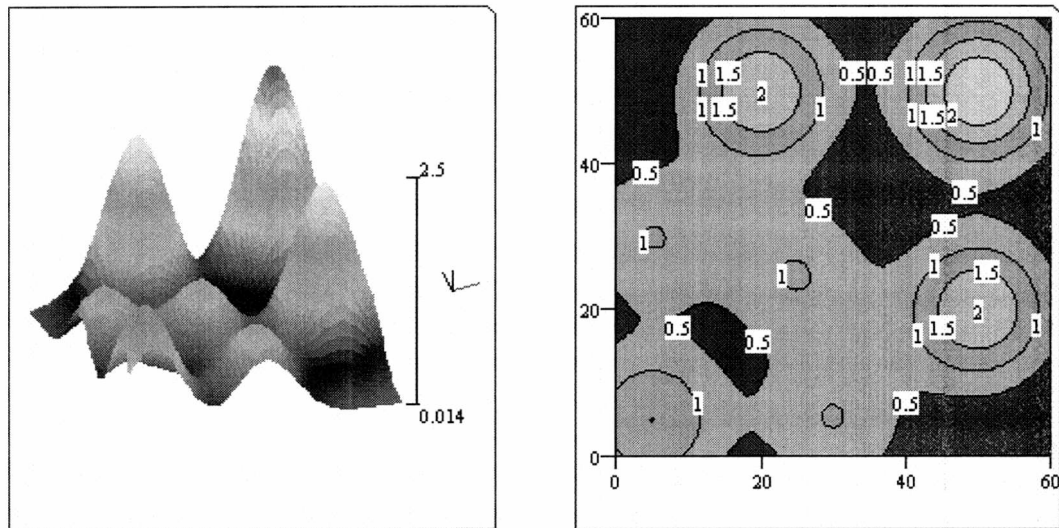
- funkcja eksponencjalna Q_7

$$Q_7(x_1, x_2) = \sum_{i=1}^7 h_i * e^{(-n_i * \sum_{j=1}^{10} (x_i - x_j^i)^2)}, \quad (5.16)$$

gdzie: $h_1 = 1,5, h_2 = h_3 = h_4 = 1, h_5 = h_6 = 2, h_7 = 2,5,$

$n_1 = n_2 = n_3 = n_4 = n_5 = n_6 = n_7 = 0,01, x_1^1 = 5, x_1^2 = 5, x_2^1 = 5, x_2^2 = 30, x_3^1 = 25,$
 $x_3^2 = 25, x_4^1 = 30, x_4^2 = 5, x_5^1 = 50, x_5^2 = 20, x_6^1 = 20, x_6^2 = 50, x_7^1 = 50, x_7^2 = 50$

Funkcja ta posiada maksimum globalne w punkcie $(50,50)$. Jako zakres zmiennych x_1 i x_2 przyjęto przedział $[0, 60]$. Przebieg funkcji przedstawiono na rysunku 5.1.



Rys. 5.1. Przebieg funkcji testowej Q_7

Weryfikację skuteczności działania zmodyfikowanego algorytmu genetycznego przeprowadzono w oparciu o program testów obejmujący sześć etapów:

- Etap 1 – porównanie liczby osobników, których wartość funkcji przystosowania jest większa, równa i mniejsza od wartości średniej w danym pokoleniu, wybieranych do puli rodzicielskiej, oraz liczby mutacji genów wykonywanych w osobnikach należących do tych grup w algorytmach zmodyfikowanych i elementarnych. Eksperymenty te przeprowadzono na przykładzie rozwiązywania problemu komiwojażera oraz optymalizacji funkcji.

- Etap 2 - sprawdzenie poprawności działania zmodyfikowanego algorytmu genetycznego na przykładzie rozwiązywania typowych, standardowych problemów optymalizacyjnych. W tym etapie badano zachowania algorytmów w czasie rozwiązywania problemu optymalizacji funkcji, rozwiązywania zero-jedynkowego problemu plecakowego oraz problemu komiwojażera. Jako dane testowe wykorzystane zostały zadania optymalizacyjne zaproponowane w innych pracach dotyczących optymalizacji.
- Etap 3 – porównanie czasu działania i wartości funkcji optymalizowanej uzyskanej przez algorytm genetyczny elementarny, algorytm genetyczny zmodyfikowany oraz inne metody optymalizacyjne oparte na obserwacji natury: metodzie przeszukiwania tabu (tabu search), symulowanego wyżarzania (simulated annealing) oraz systemów mrówkowych (ant system).
- Etap 4 - badanie wpływu parametrów algorytmu na rezultaty uzyskiwane przez algorytm elementarny i algorytmy zmodyfikowane. Do badań wybrano problem optymalizacji funkcji. Eksperyment składał się z:
 - badania wartości funkcji optymalizowanej po określonej liczbie wywołań funkcji obliczającej wartość przystosowania osobnika,
 - badania liczby wywołań funkcji obliczającej wartość przystosowania osobnika, potrzebnych aby algorytm osiągnął założoną wartość funkcji optymalizowanej – w tym etapie eksperymentu badano liczbę wywołań funkcji, po których wartość funkcji optymalizowanej osiągała około 99,5% wartości optymalnej.
- Etap 5 – badanie czasu i liczby wywołań funkcji obliczającej wartość przystosowania osobnika, potrzebnych do osiągnięcia przez algorytm wartości optymalnej. Do badań wybrano problem komiwojażera dla 12 miast rozmieszczonych na okręgu oraz problem optymalizacji funkcji.
- Etap 6 – badanie zdolności algorytmu do wychodzenia z optimum lokalnego. Do badań wybrano problem optymalizacji funkcji.

5.1. Weryfikacja skuteczności zmodyfikowanego algorytmu genetycznego

Sprawdzenie skuteczności zmodyfikowanego algorytmu genetycznego wykonano w oparciu o eksperymenty polegające na wielokrotnym wykonaniu algorytmu i porównaniu liczby osobników, których wartość funkcji przystosowania jest lepsza, równa i gorsza od wartości średniej w danym pokoleniu, wybieranych do puli rodzicielskiej oraz liczby mutacji

genów wykonywanych w osobnikach należących do tych grup w algorytmach zmodyfikowanych i elementarnym. Zadanie testowe polegało na rozwiązaniu problemu komiwojażera i optymalizacji funkcji. W eksperymencie tym badano również efektywność „on-line” i „off-line” algorytmów elementarnych i zmodyfikowanych.

5.1.1. Liczba osobników wybieranych do puli rodzicielskiej

Eksperyment 1

Celem eksperymentu jest zbadanie, w jaki sposób sterownik rozmyty wpływa na liczbę osobników, których wartość funkcji przystosowania jest lepsza, równa i gorsza od wartości średniej w danym pokoleniu, wybieranych do puli rodzicielskiej.

Jako zadania testowe wybrano problem komiwojażera dla 16 miast oraz dwa zadania o ciągłej funkcji optymalizowanej:

- problem optymalizacji funkcji De Jong’a F1 – łatwa w optymalizacji funkcja ciągła, z jednym optimum globalnym,
- problem optymalizacji funkcji De Jong’a F2 – trudna w optymalizacji funkcja ciągła, z jednym optimum globalnym,

Przyjęto następujące parametry algorytmu genetycznego:

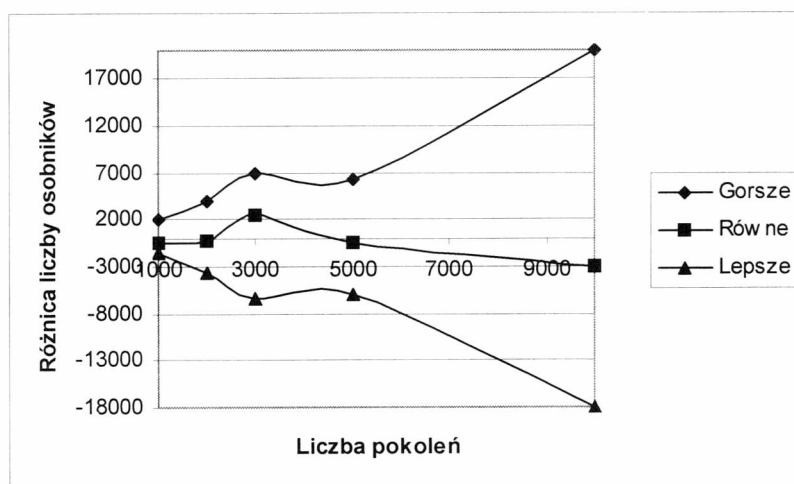
- geny reprezentowane przez liczby rzeczywiste dla optymalizacji funkcji, dla problemu komiwojażera przez ciągi 16 liczb z zakresu 0,...,15 – reprezentacja porządkowa [Mic99].
- prawdopodobieństwo krzyżowania: dla problemu komiwojażera 0,8, dla funkcji F1 0,9, dla funkcji F2 0,7,
- prawdopodobieństwo mutacji: dla problemu komiwojażera 0,15, dla funkcji F1 0,1, dla funkcji F2 0,2,
- liczebność populacji: dla problemu komiwojażera 25, dla optymalizacji funkcji 20,
- do grupy osobników o średniej wartości funkcji przystosowania zaliczane były te osobniki, których wartość funkcji przystosowania różniła się od wartości średniej o co najwyżej 1 % tej wartości,
- każdy z algorytmów uruchomiony został 10 razy.

W tabeli 5.1. przedstawiono liczbę osobników wybranych do puli rodzicielskiej w algorytmie elementarnym i zmodyfikowanym przy różnej liczbie pokoleń. Wykresy na Rys. 5.2.-5.4. pokazują różnice między liczbą osobników zakwalifikowanych do puli rodzicielskiej w algorytmie elementarnym a liczbą osobników zakwalifikowanych do puli rodzicielskiej w algorytmie zmodyfikowanym (wartość ujemna oznacza, że w algorytmie

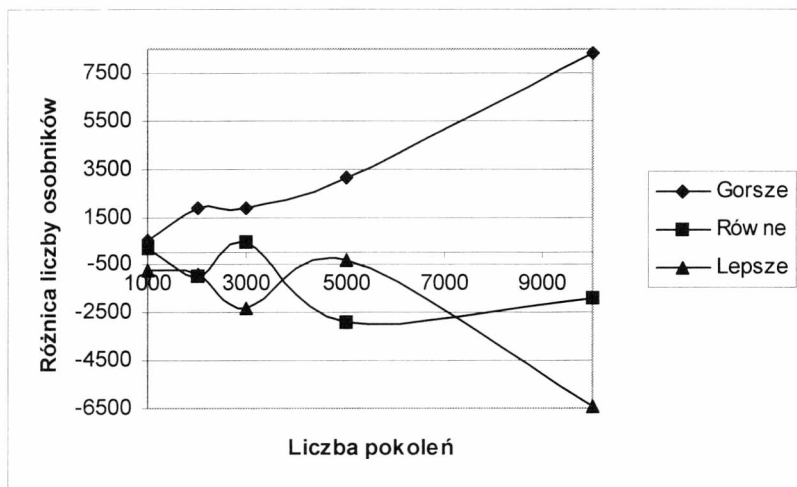
elementarnym zakwalifikowano do puli rodzicielskiej mniej osobników niż w algorytmie zmodyfikowanym).

			Rodzaj osobników					
			Gorsze		Równe		Lepsze	
			elementarny	modyfikowany	elementarny	modyfikowany	elementarny	modyfikowany
Liczba pokoleń	1000	F1	60452	58418	8826	9283	120722	122299
		F2	43060	42561	16831	16614	130109	130825
		komiwojażer	112440	111708	10614	5203	116946	123089
	2000	F1	119253	115285	18228	18542	242519	246073
		F2	87055	85165	32334	33318	260611	261517
		komiwojażer	224178	224791	21069	10588	234753	244621
	3000	F1	182105	175256	26903	24417	360992	367327
		F2	130086	128248	50008	49558	389906	392194
		komiwojażer	340782	338368	30578	15488	348640	366144
	5000	F1	298403	292025	45449	45860	606148	612115
		F2	215843	212676	80380	83259	653777	654065
		komiwojażer	562781	569255	51430	26509	585783	604236
	10000	F1	600452	580455	88993	92042	1209560	1227500
		F2	434009	425713	163982	165878	1302010	1308410
		komiwojażer	1135470	1133640	104317	51834	1160210	1214520

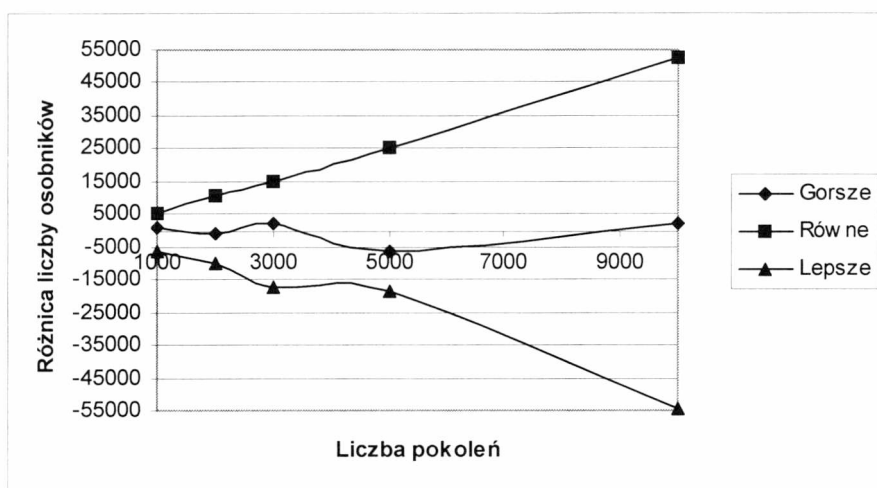
Tabela 5.1. Liczba osobników wybranych do puli rodzicielskiej przy danej liczbie pokoleń



Rys. 5.2. Różnica w liczbie osobników w algorytmie elementarnym i zmodyfikowanym zakwalifikowanych do puli rodzicielskiej w optymalizacji funkcji F1



Rys. 5.3. Różnica w liczbie osobników w algorytmie elementarnym i zmodyfikowanym zakwalifikowanych do puli rodzicielskiej w optymalizacji funkcji F2



Rys. 5.4. Różnica w liczbie osobników w algorytmie elementarnym i zmodyfikowanym zakwalifikowanych do puli rodzicielskiej w problemie komiwojażera

Spostrzeżenia z eksperymentu

1. W algorytmie z modyfikowanym prawdopodobieństwem selekcji do puli rodzicielskiej wybieranych jest:
 - a. od około 1% więcej (wartość gorsza) do około -3% mniej (wartość lepsza) osobników o wartości funkcji przystosowania niższej od średniej.
 - b. od około -50% mniej do około 9% więcej osobników o wartości funkcji przystosowania równej średniej w pokoleniu.
 - c. od około 0% (wartość jednakowa) do około 5% (wartość lepsza) więcej osobników o wartości funkcji przystosowania wyższej niż średnia w pokoleniu.

Wskaźniki procentowe odniesiono do wartości uzyskanych przez algorytm elementarny.

5.1.2. Liczba mutacji genów w algorytmach genetycznych

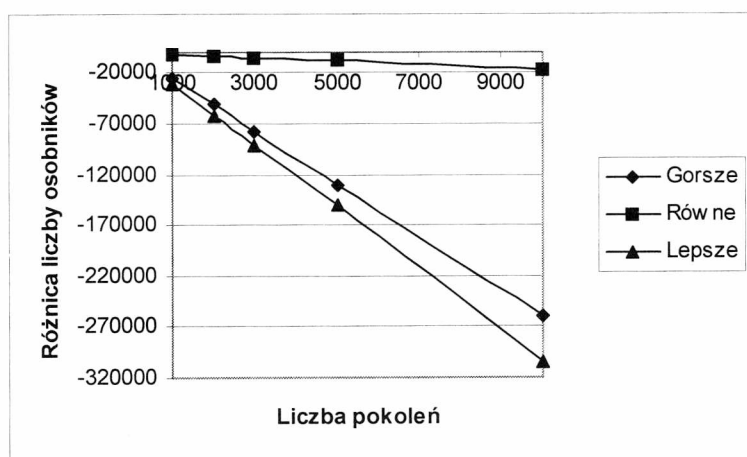
Eksperyment 2

Celem eksperymentu jest zbadanie, w jaki sposób sterownik rozmyty wpływa na liczbę mutacji genów osobników, których wartość funkcji przystosowania jest lepsza, równa i gorsza od wartości średniej w danym pokoleniu.

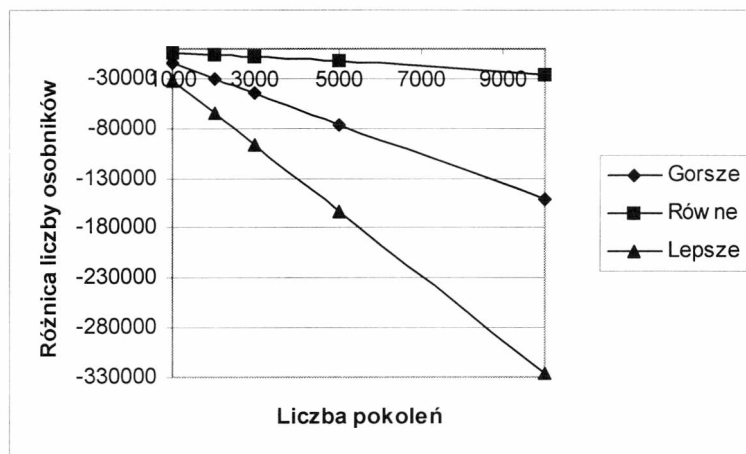
W eksperymencie tym przyjęto parametry algorytmu i rozwiązywano zadania testowe takie same jak w eksperymencie 1. W tabeli 5.2. przedstawiono liczbę mutacji genów osobników w algorytmie elementarnym i algorytmie z modyfikacją mutacji. Wykresy na Rys. 5.5.-5.7. pokazują różnice między liczbą mutacji genów osobników w algorytmie zmodyfikowanym a liczbą mutacji genów osobników w algorytmie elementarnym.

			Rodzaj osobników					
			Gorsze		Równe		Lepsze	
			elementarny	modyfikowany	elementarny	modyfikowany	elementarny	modyfikowany
Liczba pokoleń	1000	F1	35892	61319	6141	8056	58267	89175
		F2	21424	36310	9429	12520	48809	81096
		komiwojażer	324166	384490	26071	32088	249720	293407
	2000	F1	71324	122036	12550	15819	116113	177592
		F2	42781	72603	19880	25624	97497	162244
		komiwojażer	654663	758992	52989	58868	494387	562949
	3000	F1	106457	183962	18364	23588	175729	267305
		F2	63482	108677	30700	38455	146012	243374
		komiwojażer	993115	1137120	77172	88183	732287	840682
	5000	F1	176882	307372	30992	39667	293423	443979
		F2	105454	182281	51254	63901	243365	406692
		komiwojażer	1644410	1876640	129480	147493	1224120	1376300
	10000	F1	353987	613355	60843	79379	584678	888313
		F2	211281	362117	101789	127790	487409	813199
		komiwojażer	3318260	3732260	252521	290106	2428650	2728170

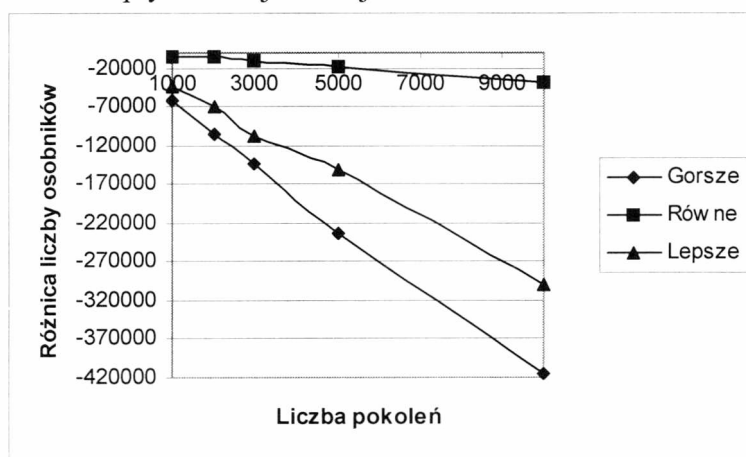
Tabela 5.2. Liczba mutacji osobników w algorytmie z modyfikacją mutacji i algorytmie elementarnym



Rys. 5.5. Różnica w liczbie mutacji genów w algorytmie zmodyfikowanym i elementarnym w zadaniu optymalizacji funkcji F1



Rys. 5.6. Różnica w liczbie mutacji genów w algorytmie zmodyfikowanym i elementarnym w zadaniu optymalizacji funkcji F2



Rys. 5.7. Różnica w liczbie mutacji genów w algorytmie zmodyfikowanym i elementarnym w problemie komiwojażera

Spostrzeżenia z eksperymentu

1. Liczba mutacji genów osobników w algorytmie z modyfikowanym prawdopodobieństwem mutacji była większa od liczby mutacji w algorytmie elementarnym.
2. W optymalizacji funkcji liczba mutacji osobników lepiej przystosowanych niż średnia w pokoleniu jest większa niż osobników gorzej przystosowanych, jednak w puli rodzicielskiej liczba osobników lepiej przystosowanych jest od około 110% do około 210% większa niż osobników gorzej przystosowanych.

5.1.3. Badanie efektywności zmodyfikowanych algorytmów genetycznych

Eksperyment 3

Celem eksperymentu jest zbadanie wpływu sterownika rozmytego na efektywność „on-line” i „off-line” algorytmu genetycznego z modyfikowanym prawdopodobieństwem selekcji, mutacji i jednocześnie selekcji i mutacji.

Jako zadania testowe wybrano problem komiwojażera dla 16 miast, oraz funkcje testowe De Jong’a F1 – łatwą z punktu widzenia optymalizacji i funkcję F2 – trudną do optymalizacji.

Przyjęto następujące parametry algorytmu genetycznego:

- geny reprezentowane są przez liczby rzeczywiste dla optymalizacji funkcji i przez ciągi liczb z zakresu 0, ..., 15 dla problemu komiwojażera – reprezentacja porządkowa [Mic99],
- prawdopodobieństwo krzyżowania: dla problemu komiwojażera 0,6, dla funkcji F1 0,9, dla funkcji F2 0,7,
- prawdopodobieństwo mutacji: dla problemu komiwojażera 0,2, dla funkcji F1 0,1, dla funkcji F2 0,2,
- liczebność populacji: 20,
- liczba pokoleń osobników 100,
- każdy z algorytmów uruchomiony został 10 razy.

Eksperyment polegał na wyznaczeniu miar efektywności „on-line” i „off-line” dla poszczególnych algorytmów. Efektywność „on-line” strategii s dla funkcji testowej j jest to średnia wartość funkcji celu ze wszystkich pokoleń włącznie z bieżącym.

$$x_j(s) = \frac{1}{T} \sum_{t=1}^T f_j(t), \quad (5.17)$$

gdzie: $f_j(t)$ jest wartością funkcji celu osobnika j w pokoleniu t .

Efektywność „off-line” dla funkcji testowej j jest to średnia wartości funkcji celu najlepszych osobników ze wszystkich pokoleń.

$$x_j^*(s) = \frac{1}{T} \sum_{t=1}^T f_j^*(t), \quad (5.18)$$

gdzie: $f_j^*(t)$ jest wartością funkcji celu najlepszego osobnika znalezionej do czasu t .

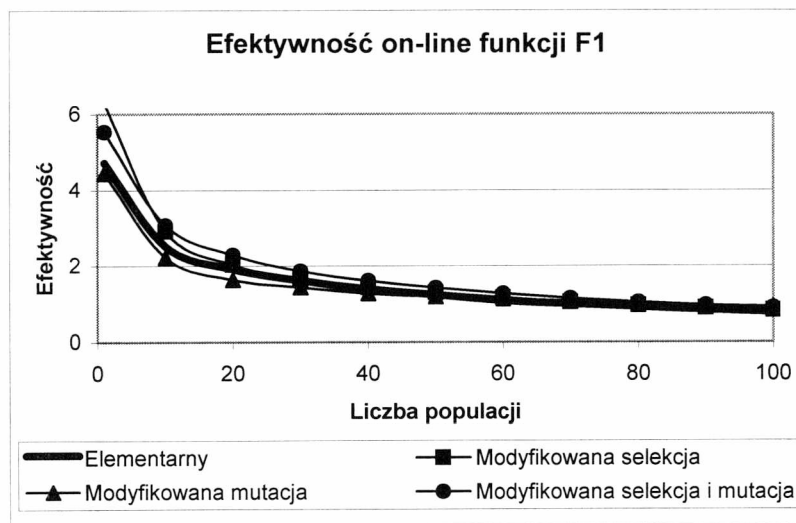
W tabelach 5.3. i 5.4. przedstawiono wartości efektywności „on-line” i „off-line” dla algorytmów elementarnego i zmodyfikowanych obliczone dla zadań testowych. Wykresy na Rys. 5.8. - 5.13. pokazują różnice między efektywnością „on-line” i „off-line” algorytmu elementarnego a efektywnością algorytmu zmodyfikowanego przy różnej liczbie pokoleń.

	Efektywność „on-line”											
Liczba pokoleń	Elementarny			Modyfikowana selekcja			Modyfikowana mutacja			Modyfikowana selekcja i mutacja		
	F1	F2	kom	F1	F2	kom	F1	F2	kom	F1	F2	Kom
1	4,726	7,166	92,32	6,285	5,556	92,32	4,442	14,35	92,32	5,532	7,413	92,32
10	2,521	2,937	93,99	2,911	2,040	92,82	2,223	3,313	92,05	3,080	2,387	93,45
20	1,919	1,860	95,59	2,038	1,793	91,40	1,641	2,967	90,18	2,290	1,908	92,33
30	1,619	1,570	96,24	1,640	1,676	90,40	1,434	2,329	89,56	1,865	1,595	91,59
40	1,396	1,518	96,52	1,439	1,474	89,73	1,272	1,983	88,75	1,613	1,366	91,12
50	1,238	1,380	96,92	1,304	1,346	88,68	1,189	1,782	88,30	1,424	1,246	90,38
60	1,091	1,252	97,21	1,163	1,258	87,84	1,135	1,666	87,88	1,278	1,177	89,63
70	1,007	1,268	97,01	1,047	1,280	87,51	1,069	1,676	87,48	1,147	1,093	88,97
80	0,929	1,294	96,66	0,964	1,246	87,04	1,020	1,715	87,50	1,051	1,032	88,72
90	0,851	1,287	96,68	0,899	1,162	86,56	0,964	1,643	87,24	0,975	1,025	88,31
100	0,796	1,246	96,91	0,836	1,110	86,27	0,927	1,610	87,22	0,900	0,979	88,02

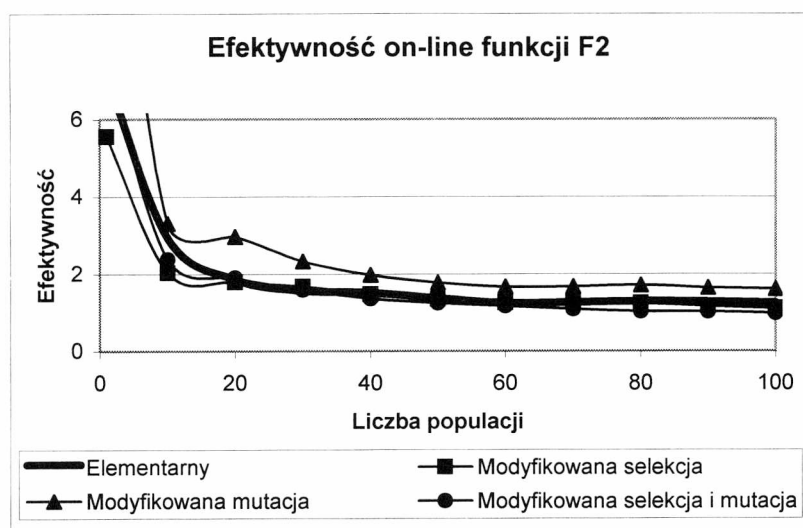
Tabela 5.3. Efektywność „on-line” algorytmów

	Efektywność „off-line”											
Liczba pokoleń	Elementarny			Modyfikowana selekcja			Modyfikowana mutacja			Modyfikowana selekcja i mutacja		
	F1	F2	kom	F1	F2	kom	F1	F2	kom	F1	F2	Kom
1	4,726	7,166	92,32	6,285	5,556	92,32	4,442	14,35	92,32	5,532	7,413	92,32
10	2,107	1,707	87,32	2,348	1,302	87,14	1,564	2,201	84,08	2,546	1,830	88,04
20	1,371	0,954	84,92	1,491	0,865	84,23	0,992	1,233	80,03	1,637	1,101	85,23
30	1,050	0,687	83,71	1,093	0,640	82,54	0,757	0,860	78,17	1,202	0,807	83,46
40	0,872	0,551	82,32	0,880	0,519	80,98	0,624	0,666	76,79	0,964	0,639	81,92
50	0,752	0,460	81,25	0,743	0,439	79,52	0,540	0,542	75,73	0,813	0,535	80,35
60	0,655	0,391	80,40	0,645	0,373	78,19	0,479	0,457	74,70	0,709	0,465	79,09
70	0,581	0,340	79,68	0,570	0,324	77,15	0,433	0,395	73,96	0,634	0,414	77,88
80	0,520	0,302	79,02	0,513	0,287	76,19	0,394	0,349	73,36	0,576	0,374	76,89
90	0,470	0,273	78,45	0,466	0,258	75,35	0,363	0,313	72,88	0,525	0,342	76,02
100	0,429	0,249	77,93	0,426	0,234	74,66	0,336	0,285	72,48	0,481	0,316	75,28

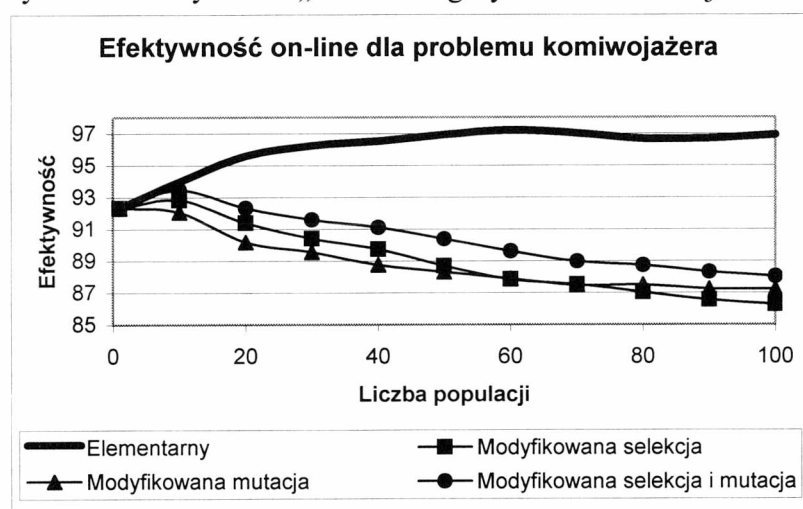
Tabela 5.4. Efektywność „off-line” algorytmów



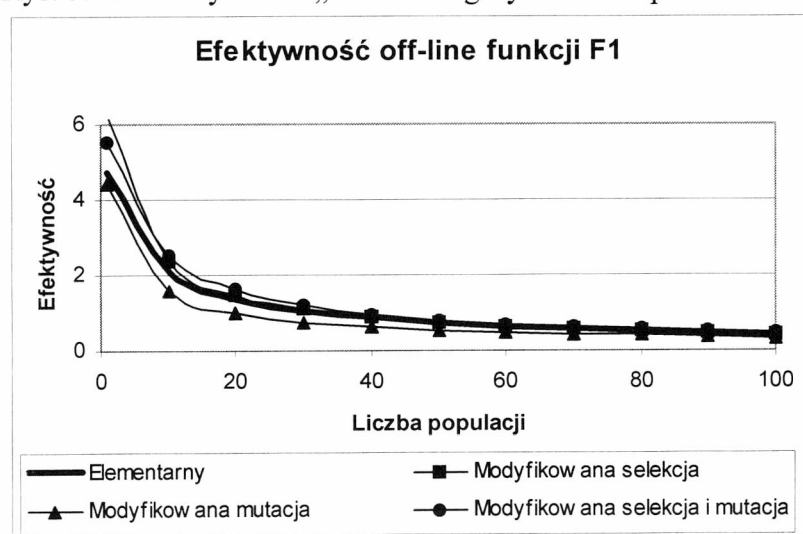
Rys. 5.8. Efektywność „on-line” algorytmów dla funkcji F1



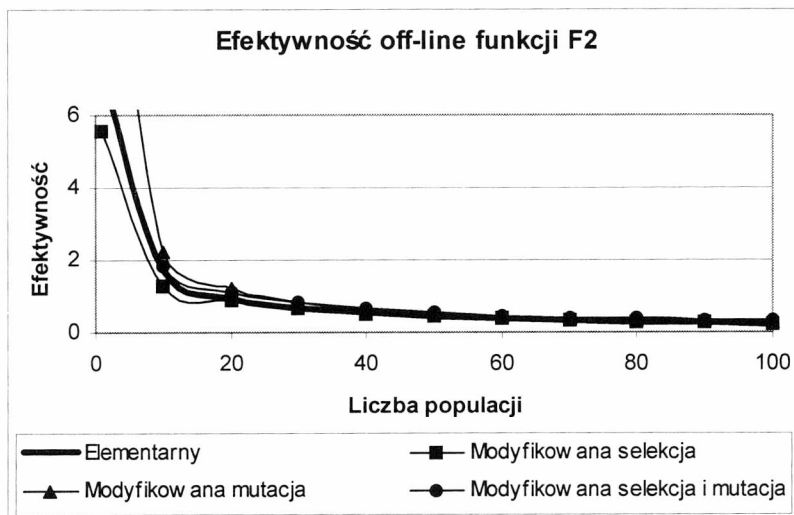
Rys. 5.9. Efektywność „on-line” algorytmów dla funkcji F2



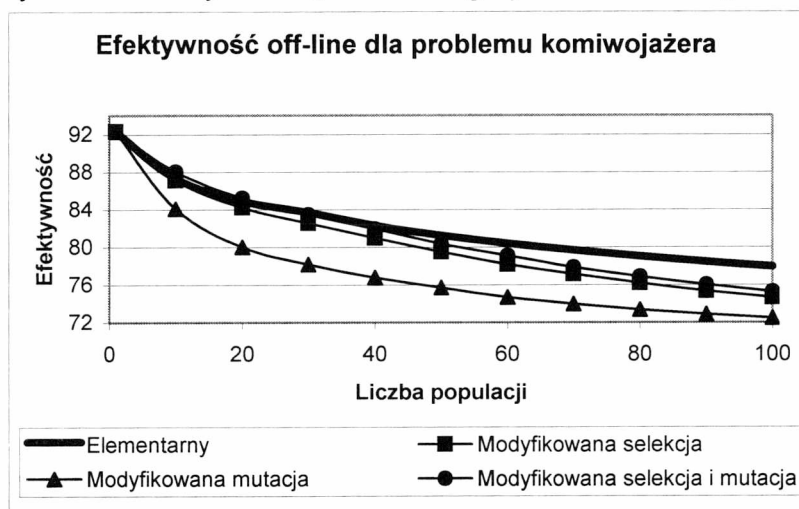
Rys. 5.10. Efektywność „on-line” algorytmów dla problemu komiwojażera



Rys. 5.11. Efektywność „off-line” algorytmów dla funkcji F1



Rys. 5.12. Efektywność „off-line” algorytmów dla funkcji F2



Rys. 5.13. Efektywność „off-line” algorytmów dla problemu komiwojażera

Spostrzeżenia z eksperymentu

1. Efektywność „on-line” algorytmów zmodyfikowanych była:
 - a. od około -4% mniejsza (wartość gorsza) do około 10% większa (wartość lepsza) dla algorytmu z modyfikacją selekcji niż algorytmu elementarnego,
 - b. od około -1% mniejsza (wartość gorsza) do około 9% większa (wartość lepsza) dla algorytmu z modyfikacją mutacji niż algorytmu elementarnego,
 - c. od około -11% mniejsza (wartość gorsza) do około 9% większa (wartość lepsza) dla algorytmu z modyfikacją selekcji i mutacji niż algorytmu elementarnego.
2. Efektywność „off-line” algorytmów zmodyfikowanych była:
 - a. od około -1% mniejsza (wartość gorsza) do około 4% większa (wartość lepsza) dla algorytmu z modyfikacją selekcji niż algorytmu elementarnego,

- b. od około -58% mniejsza (wartość gorsza) do około 10% większa (wartość lepsza) dla algorytmu z modyfikacją mutacji niż algorytmu elementarnego,
- c. od około -8% mniejsza (wartość gorsza) do około 15% większa (wartość gorsza) dla algorytmu z modyfikacją selekcji i mutacji niż algorytmu elementarnego.

Wskaźniki procentowe odniesiono do wartości uzyskanych przez algorytm elementarny.

5.1.4. Wnioski z eksperymentów

1. W algorytmach zmodyfikowanych w porównaniu do algorytmu elementarnego do puli rodzicielskiej wybieranych jest więcej osobników dobrze przystosowanych, mniej osobników gorzej i średnio przystosowanych.
2. W algorytmach zmodyfikowanych w porównaniu do algorytmu elementarnego częściej przeprowadzane są mutacje genów osobników gorzej i średnio przystosowanych, rzadziej osobników dobrze przystosowanych.
3. W algorytmach zmodyfikowanych efektywność on-line i off-line jest w większości przypadków większa w porównaniu do algorytmu elementarnego. Najlepsze wyniki uzyskał algorytm z modyfikacją prawdopodobieństwa selekcji.

5.2. Testowanie algorytmu w rozwiązywaniu standardowych problemów optymalizacyjnych

Zmodyfikowane algorytmy genetyczne wykorzystane zostały do rozwiązania standardowych problemów testowych: problemu komiwojażera, optymalizacji funkcji i zero-jedynkowego problemu plecakowego. Porównano czas rozwiązania zadania oraz liczbę wywołań funkcji obliczającej wartość przystosowania osobnika potrzebnych, aby algorytm osiągnął założoną wartość funkcji optymalizowanej. Wyniki porównane zostały z rezultatami uzyskanymi przez elementarny algorytm genetyczny.

5.2.1. Sprawdzenie działania algorytmu w rozwiązywaniu problemu komiwojażera

Eksperyment 4

Celem eksperymentu jest zbadanie wpływu sterownika rozmytego na liczbę wywołań funkcji obliczającej przystosowanie osobnika i czas działania, po którym zmodyfikowane algorytmy osiągnęły założoną wartość funkcji optymalizowanej oraz porównanie ich wyników z wynikami uzyskanymi przez algorytm elementarny na przykładzie problemu komiwojażera.

W eksperymencie tym badany jest wpływ sterownika rozmytego na działanie algorytmów podczas optymalizacji funkcji nieciągłych. Spośród opublikowanych danych testowych wybrano zadania o liczbie miast od 24 do 52. Zestawienie podstawowych informacji oraz parametry algorytmów stosowane w zadaniach testowych umieszczono w tabeli 5.5.

Nazwa zadania	Liczebność populacji	Prawdopodob. krzyżowania	Prawdopodob. mutacji	Liczba miast	Długość optymalnej drogi	Wartość najgorszego osobnika	Założona wartość drogi
Gr24	20	0,6	0,2	24	1272	4500	1900
Bays29	30	0,7	0,05	29	2020	7000	3300
Gr48	40	1,0	0,05	48	5046	23500	13000
Berlin52	25	0,5	0,05	52	7526	33500	18000

Tabela 5.5. Zadania testowe stosowane w eksperymentach z problemem komiwojażera

Przyjęto w algorytmach, że:

- geny reprezentowane są przez ciągi N_m liczb z przedziału $0...(N_m-1)$, gdzie N_m jest liczbą miast – reprezentacja porządkowa [Mic99],
- funkcja przystosowania dla osobnika obliczana jako suma odległości pomiędzy kolejnymi miastami na trasie komiwojażera.

W tabeli 5.6. przedstawiono średnie czasy działania poszczególnych algorytmów oraz wartości odchylenia standardowego czasu działania dla różnych zadań testowych. Wartości podane w tabeli stanowią średnią z dziesięciu przebiegów algorytmu.

Algorytm	Gr24		Bays29		Gr48		Berlin52	
	Czas [s]	δ	Czas [s]	δ	Czas [s]	δ	Czas [s]	δ
Elementarny	5,19	4,37	5,33	4,62	9,26	6,56	3,33	1,90
Zmodyfikowana selekcja	5,90	5,26	4,23	5,83	5,32	4,35	3,24	1,87
Zmodyfikowana mutacja	4,11	3,57	1,79	1,08	2,74	1,53	1,99	1,40
Zmodyfikowana selekcja i mutacja	5,56	3,99	4,77	5,40	4,91	4,47	1,70	1,47

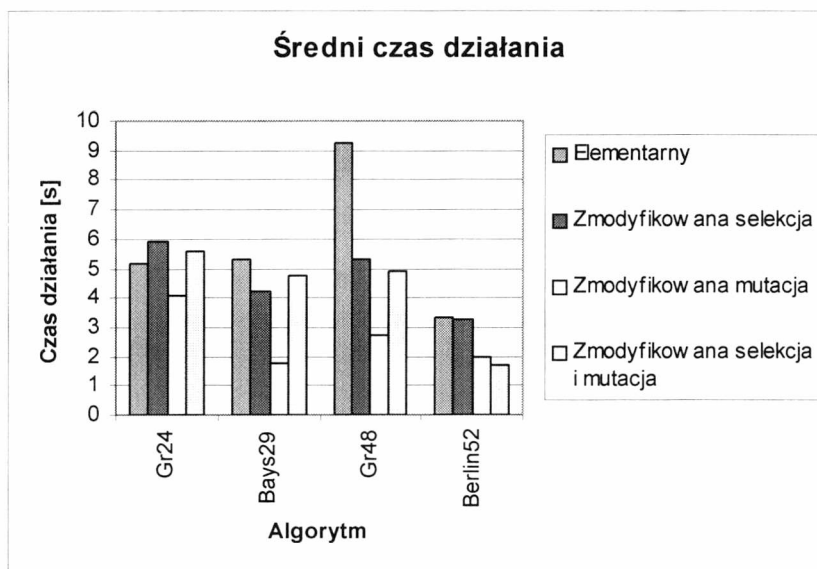
Tabela 5.6. Średnie czasy rozwiązania problemu komiwojażera przez poszczególne algorytmy

W tabeli 5.7. przedstawiono średnie liczby wywołań funkcji obliczającej przystosowanie osobnika, potrzebnych do uzyskania przez algorytmy założonej wartości funkcji optymalizowanej oraz wartości odchylenia standardowego tej wartości dla różnych zadań testowych. Wartości podane w tabeli stanowią średnią z dziesięciu przebiegów algorytmu.

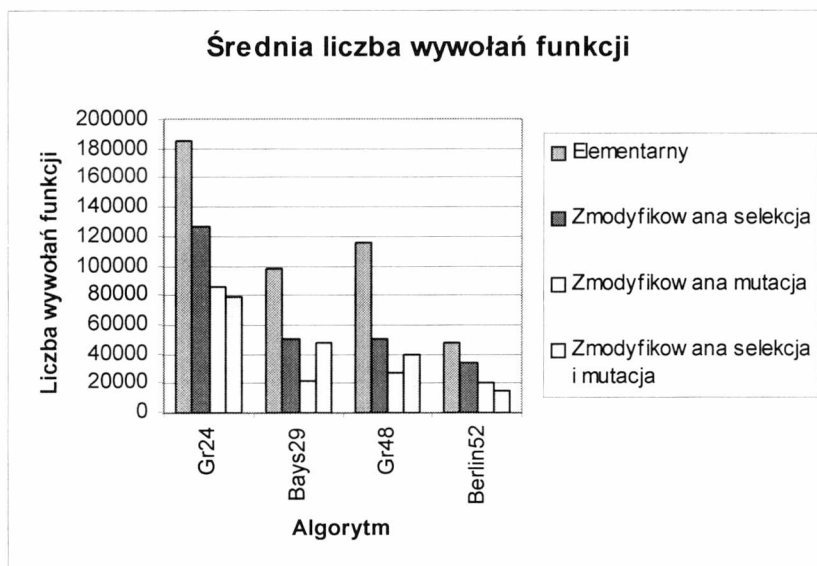
Algorytm	Gr24		Bays29		Gr48		Berlin52	
	Liczb. wyw.	δ	Liczb. wyw.	δ	Liczb. wyw.	δ	Liczb. wyw.	δ
Elementarny	185100	156020	98043	83970	115616	82520	47610	26925
Zmodyfikowana selekcja	126634	114960	50829	70410	50208	41440	34207	19925
Zmodyfikowana mutacja	85912	74420	22002	13470	26576	15080	20365	14625
Zmodyfikowana selekcja i mutacja	78456	56920	47778	54660	39748	36360	14460	12750

Tabela 5.7. Średnie liczby wywołań funkcji obliczającej przystosowanie potrzebnych do rozwiązania problemu komiwojażera

Wykresy na Rys. 5.14. – 5.15. pokazują różnice między czasem działania i liczbą wywołań funkcji obliczającej przystosowanie, potrzebnych do uzyskania założonej wartości funkcji optymalizowanej w poszczególnych algorytmach dla różnych zadań testowych.



Rys. 5.14. Średnie czasy rozwiązywania problemu komiwojażera przez algorytmy



Rys. 5.15. Średnia liczba wywołań funkcji obliczającej przystosowanie potrzebnych do uzyskania założonej wartości funkcji optymalizowanej

Spostrzeżenia z eksperymentu

1. Średni czas działania algorytmu, w którym wykorzystano logikę rozmytą do modyfikowania prawdopodobieństwa selekcji jest od około -13% dłuższy (wartość gorsza) do około 42% krótszy (wartość lepsza) niż algorytmu elementarnego.

2. Średnia liczba wywołań funkcji obliczającej przystosowanie, potrzebna do uzyskania założonej wartości funkcji optymalizowanej przez algorytm, w którym wykorzystano logikę rozmytą do modyfikowania prawdopodobieństwa selekcji jest od około 28% (wartość lepsza) do około 56% (wartość lepsza) mniejsza niż w algorytmie elementarnym.
3. Średni czas działania algorytmu, w którym wykorzystano logikę rozmytą do modyfikowania prawdopodobieństwa mutacji jest od około 20% (wartość lepsza) do około 66% (wartość lepsza) krótszy niż algorytmu elementarnego.
4. Średnia liczba wywołań funkcji obliczającej przystosowanie, potrzebna do uzyskania założonej wartości funkcji optymalizowanej przez algorytm, w którym wykorzystano logikę rozmytą do modyfikowania prawdopodobieństwa mutacji jest od około 53% (wartość lepsza) do około 77% (wartość lepsza) mniejsza niż w algorytmie elementarnym.
5. Średni czas działania algorytmu, w którym wykorzystano logikę rozmytą do modyfikowania prawdopodobieństwa selekcji i mutacji jest od około -7% dłuższy (wartość gorsza) do około 56% krótszy (wartość lepsza) niż algorytmu elementarnego.
6. Średnia liczba wywołań funkcji obliczającej przystosowanie, potrzebna do uzyskania założonej wartości funkcji optymalizowanej przez algorytm, w którym wykorzystano logikę rozmytą do modyfikowania prawdopodobieństwa selekcji i mutacji jest od około 51% (wartość lepsza) do około 69% (wartość lepsza) mniejsza niż w algorytmie elementarnym.
7. W algorytmach zmodyfikowanych zaobserwowano większą powtarzalność wyników. Odchylenie standardowe liczby wywołań funkcji obliczającej przystosowanie osobnika było we wszystkich przypadkach mniejsze, odchylenie standardowe czasu działania było w większości przypadków mniejsze niż w algorytmie elementarnym.

Wskaźniki procentowe odniesiono do wartości uzyskanych przez algorytm elementarny.

5.2.2. Sprawdzenie działania algorytmu na przykładzie optymalizacji funkcji

Eksperyment 5

Celem eksperymentu jest zbadanie wpływu sterownika rozmytego na liczbę wywołań funkcji obliczającej przystosowanie osobnika i czas działania, po którym zmodyfikowane algorytmy osiągnęły założoną wartość funkcji optymalizowanej oraz porównanie ich wyników z wynikami uzyskanymi przez algorytm elementarny na przykładzie optymalizacji funkcji.

W eksperymencie badano wyniki działania algorytmów, w których geny reprezentowane były przez liczby rzeczywiste. Optymalizacji poddano funkcje testowe ciągłe, posiadające jedno globalne optimum. Do eksperymentów wybrano cztery funkcje o liczbie zmiennych od 2 do 4 i różnym stopniu trudności:

- funkcja De Jong'a F1,
- funkcja De Jong'a F2,
- funkcja Coldville'a,
- funkcja Goldsteina-Price'a.

Zestawienie podstawowych informacji o wybranych zadaniach testowych umieszczono w tabeli 5.8.

Nazwa zadania	Liczebność populacji	Prawdop. krzyżowania	Prawdop. mutacji	Liczba zmiennych	Wartość stała C_{max}	Zakres zmiennych	Maks. błąd
De Jong F1	20	0,9	0,1	3	80	$-5,12 \leq x_i \leq 5,12$	0,001
De Jong F2	20	0,7	0,2	2	3905,9262	$-2,048 \leq x_i \leq 2,048$	0,01
Goldsteina-Price'a	20	0,8	0,15	2	1849350	$-2 \leq x_i \leq 2$	0,1
Coldville'a	30	0,7	0,05	4	2304082	$-10 \leq x_i \leq 10$	5,0

Tabela 5.8. Zadania testowe wykorzystywane w eksperymentach z optymalizacją funkcji

Przyjęto w algorytmach, że:

- geny reprezentowane są przez wartości rzeczywiste,
- funkcja przystosowania osobnika obliczana jako różnica pomiędzy wartością maksymalną C_{max} a funkcją optymalizowaną $f(x_1, x_2, \dots)$.

W tabeli 5.9. przedstawiono średnie czasy działania i wartość odchylenia standardowego czasu, potrzebne do uzyskania przez algorytm założonej wartości funkcji optymalizowanej dla poszczególnych algorytmów w różnych zadaniach testowych. Wartości podane w tabeli stanowią średnią z dziesięciu przebiegów algorytmu.

Algorytm	De Jong F1		De Jong F2		Goldsteina-Price'a		Coldville'a	
	czas [s]	δ	czas [s]	δ	czas [s]	δ	czas [s]	δ
Elementarny	0,19	0,08	0,24	0,14	0,11	0,08	0,57	0,73
Zmodyfikowana selekcja	1,03	0,30	0,92	0,77	0,48	0,32	1,27	0,81
Zmodyfikowana mutacja	0,69	0,39	1,04	0,63	0,17	0,15	2,23	1,44
Zmodyfikowana selekcja i mutacja	1,08	0,64	0,97	0,68	0,59	0,47	2,18	1,31

Tabela 5.9. Wyniki działania algorytmów rozwiązujących problemu optymalizacji funkcji

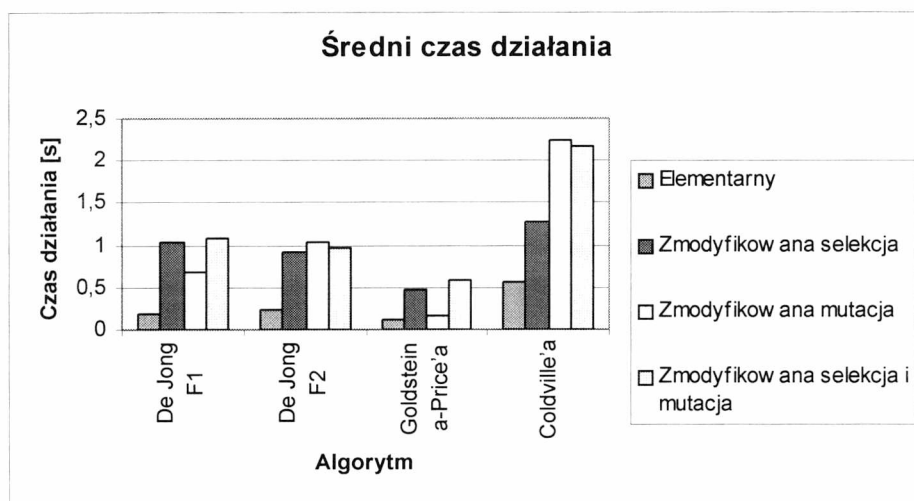
W tabeli 5.10. przedstawiono średnią liczbę wywołań funkcji obliczającej przystosowanie osobnika oraz odchylenie standardowe tej wartości, potrzebną do uzyskania przez algorytm założonej wartości funkcji optymalizowanej dla poszczególnych algorytmów

w różnych zadaniach testowych. Wartości podane w tabeli stanowią średnią z dziesięciu przebiegów algorytmu.

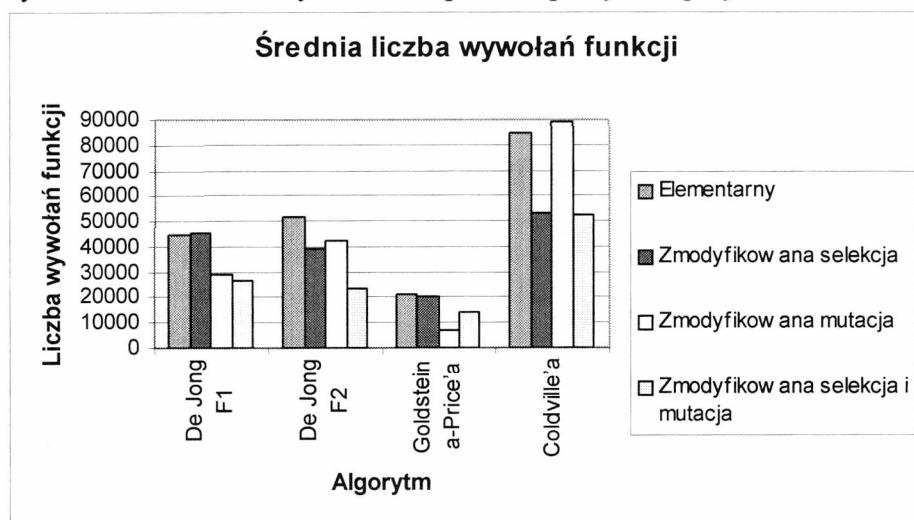
Algorytm	De Jong F1		De Jong F2		Goldsteina-Price'a		Coldville'a	
	liczba wywołań	δ	liczba wywołań	δ	liczba wywołań	δ	liczba wywołań	δ
Elementarny	44254	20166	51408	39964	21186	16741	84768	107811
Zmodyfikowana selekcja	45152	13384	39512	32702	20248	13868	53320	33070
Zmodyfikowana mutacja	29274	16631	42480	26162	6860	6416	89376	757418
Zmodyfikowana selekcja i mutacja	26852	15954	23712	16533	14284	11500	52329	31565

Tabela 5.10. Wyniki działania algorytmów rozwiązujących problemu optymalizacji funkcji

Wykresy na Rys. 5.16. – 5.17. pokazują różnice między czasem działania i liczbą wywołań funkcji obliczającej przystosowanie, potrzebnych do uzyskania założonej wartości funkcji optymalizowanej w poszczególnych algorytmach w różnych zadaniach testowych.



Rys. 5.16. Średnie czasy działania poszczególnych algorytmów



Rys. 5.17. Średnia liczba wywołań funkcji obliczającej przystosowanie potrzebnych do uzyskania założonej wartości funkcji optymalizowanej dla poszczególnych algorytmów

Spostrzeżenia z eksperymentu.

1. Średni czas działania algorytmu, w którym wykorzystano logikę rozmytą do modyfikacji prawdopodobieństwa selekcji jest od około -122% (wartość gorsza) do około -442% (wartość gorsza) dłuższy niż czas działania algorytmu elementarnego.
2. Średnia liczba wywołań funkcji obliczającej przystosowanie potrzebna do uzyskania założonej wartości funkcji optymalizowanej przez algorytm, w którym wykorzystano logikę rozmytą do modyfikacji prawdopodobieństwa selekcji jest od około -2% większa (wartość gorsza) do około 37% mniejsza (wartość lepsza) niż algorytmu elementarnego.
3. Średni czas działania algorytmu, w którym wykorzystano logikę rozmytą do modyfikacji prawdopodobieństwa mutacji jest od około -54% (wartość gorsza) do około -333% (wartość gorsza) dłuższy niż czas działania algorytmu elementarnego.
4. Średnia liczba wywołań funkcji obliczającej przystosowanie potrzebna do uzyskania założonej wartości funkcji optymalizowanej przez algorytm, w którym wykorzystano logikę rozmytą do modyfikacji prawdopodobieństwa mutacji jest od około -5% większa (wartość gorsza) do około 67% mniejsza (wartość lepsza) niż algorytmu elementarnego.
5. Średni czas działania algorytmu, w którym wykorzystano logikę rozmytą do modyfikacji prawdopodobieństwa selekcji i mutacji jest od około -282% (wartość gorsza) do około -468% (wartość gorsza) dłuższy niż czas działania algorytmu elementarnego.
6. Średnia liczba wywołań funkcji obliczającej przystosowanie potrzebna do uzyskania założonej wartości funkcji optymalizowanej przez algorytm, w którym wykorzystano logikę rozmytą do modyfikacji prawdopodobieństwa selekcji i mutacji jest od około 32% (wartość lepsza) do około 53% (wartość lepsza) mniejsza niż algorytmu elementarnego.
7. Odchylenie standardowe liczby wywołań funkcji obliczającej przystosowanie osobnika w algorytmach zmodyfikowanych było w większości przypadków mniejsze, odchylenie standardowe czasu działania było we wszystkich przypadkach większe niż w algorytmie elementarnym.

Wskaźniki procentowe odniesiono do wartości uzyskanych przez algorytm elementarny.

5.2.3. Sprawdzenie działania algorytmu na przykładzie rozwiązywania zero-jedynkowego problemu plecakowego

Eksperyment 6

Celem eksperymentu jest zbadanie wpływu sterownika rozmytego na liczbę wywołań funkcji obliczającej przystosowanie osobnika i czas działania, po którym zmodyfikowane algorytmy osiągnęły założoną wartość funkcji optymalizowanej oraz porównanie ich

wyników z wynikami uzyskanymi przez algorytm elementarny na przykładzie rozwiązywania zero-jedynkowego problemu plecakowego.

W eksperymencie jako zadanie testowe wybrano problem z ograniczeniami o różnym poziomie trudności.

Przyjęto następujące parametry algorytmu genetycznego:

- osobniki reprezentowane są przez ciągi n liczb 0, 1 (kodowanie binarne), gdzie 1 oznacza, że przedmiot i zabieramy, 0 – przedmiotu nie zabieramy,
- liczebność populacji 30,
- prawdopodobieństwo krzyżowania 0,7,
- prawdopodobieństwo mutacji 0,05,
- funkcja przystosowania dla osobnika i obliczana z zależności

$$F[i] = \sum_{i=1}^n x[i] * P[i] - kara(i), \quad (5.19)$$

- liniowa funkcja kary za przekroczenie pojemności plecaka obliczana na podstawie zależności

$$kara(i) = \rho * (\sum_{i=1}^n x[i] * W[i] - C), \quad (5.20)$$

$$\text{gdzie: } \rho = \max_{i=1..n} \{P[i]/W[i]\}, \quad (5.21)$$

C – pojemność plecaka.

Zestawienie podstawowych informacji o wybranych problemach testowych przedstawiono w tabeli 5.11. Wartość optymalna została wyznaczona przy pomocy procedury KNAPBACKTRACK opisanej w pracy [Sys99], stanowiącej implementację algorytmu podanego przez Martella i Totha [Mar79].

Nazwa problemu	Liczba przedm	Rodzaj korelacji	Wartość optymalna	$\sum_{i=1}^n W[i]$	$\sum_{i=1}^n P[i]$	Liczebność populacji	Prawdopodob krzyżowania	Prawdopodob mutacji	Pojemność plecaka	Wartość graniczna
50n	50	brak	2343	2108	2906	30	0,7	0,1	1054	2100
50s	50	słaba	2106	2894	3061	30	0,7	0,1	1447	1900
50m	50	silna	3019	2440	4940	30	0,7	0,1	1210	2850
100n	100	brak	4555	5103	5552	30	0,7	0,05	2551	3850
100s	100	słaba	4135	5613	5955	30	0,7	0,05	2806	3700
100m	100	silna	6102	5104	10104	30	0,7	0,05	2552	5600

Tabela 5.11. Podstawowe informacje o zadaniach testowych dla problemu plecakowego

W tabeli 5.12. przedstawiono średnie czasy działania oraz odchylenia standardowe czasu działania poszczególnych algorytmów w różnych zadaniach testowych. Wartości podane w tabeli stanowią średnią z dziesięciu przebiegów algorytmu.

Algorytm	50n		50s		50m		100n		100s		100m	
	czas	δ	czas	δ	czas	δ	czas	δ	czas	δ	czas	δ
Elementarny	0,24	0,12	0,21	0,15	1,03	0,48	0,51	0,25	1,38	0,27	1,17	0,6
Zmodyfikowana selekcja	0,38	0,15	0,39	0,29	1,38	1,11	0,79	0,43	1,49	0,72	1,25	0,76
Zmodyfikowana mutacja	0,59	0,25	0,68	0,34	1,93	1,39	0,98	0,33	2,2	0,86	2,13	1,25
Zmodyfikowana selekcja i mutacja	1,13	0,36	0,69	0,68	1,68	1,32	1,18	0,59	3,88	0,06	2,46	1,09

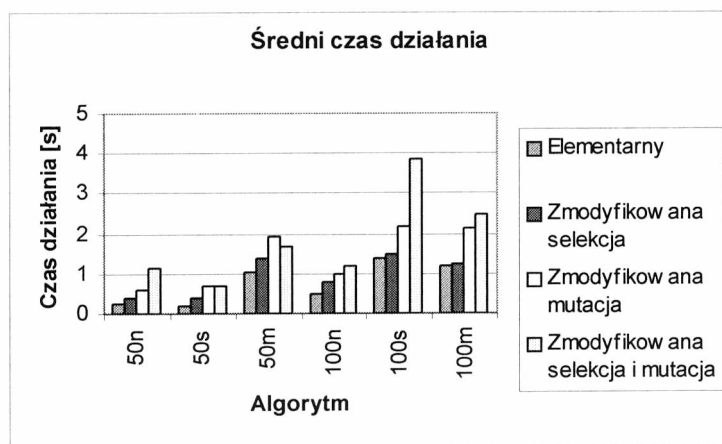
Tabela 5.12. Średnie czasy rozwiązywania problemów plecakowych przez algorytmy

W tabeli 5.13. przedstawiono średnie liczby wywołań funkcji obliczającej przystosowanie osobnika, potrzebne do uzyskania przez algorytm założonej wartości funkcji optymalizowanej w różnych zadaniach testowych. Wartości podane w tabeli stanowią średnią z dziesięciu przebiegów

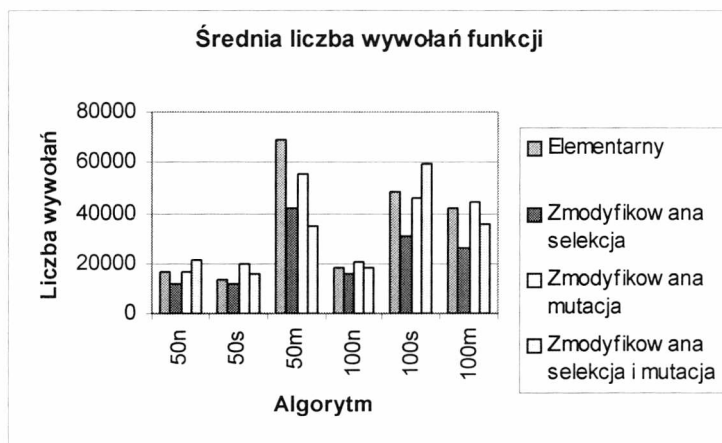
Algorytm	50n		50s		50m		100n		100s		100m	
	liczba wyw.	δ	liczba wyw.	δ	liczba wyw.	δ	liczba wyw.	δ	liczba wyw.	δ	liczba wyw.	δ
Elementarny	16419	8778	13827	10422	68655	32279	18075	8940	48462	10164	41997	21451
Zmodyf selekcja	11496	4917	11703	9043	41760	33861	16218	8942	30921	15305	25992	16009
Zmodyf mutacja	16836	7492	19485	9653	55407	39870	20520	7989	46299	18287	44628	26488
Zmodyf selekcja i mutacja	21597	7058	16170	12990	34890	26449	17964	9001	59100	16302	35847	16056

Tabela 5.13. Średnie liczby wywołań funkcji obliczającej przystosowanie potrzebnych do rozwiązania problemu plecakowego dla poszczególnych algorytmów

Wykresy na Rys. 5.18. – 5.19. pokazują różnice między czasem działania i liczbą wywołań funkcji obliczającej przystosowanie, potrzebnych do uzyskania założonej wartości funkcji optymalizowanej w poszczególnych algorytmach.



Rys. 5.18. Średnie czasy rozwiązywania problemów plecakowych przez algorytmy



Rys. 5.19. Średnia liczba wywołań funkcji obliczającej przystosowanie potrzebnych do uzyskania założonej wartości funkcji optymalizowanej

Spostrzeżenia z eksperymentu.

1. Różnica między średnim czasem działania algorytmu, w którym wykorzystano logikę rozmytą do modyfikowania prawdopodobieństwa selekcji a czasem działania algorytmu elementarnego wynosi od około -6% (wartość gorsza) do około -85% (wartość gorsza).
2. Różnica między średnią liczbą wywołań funkcji obliczającej przystosowanie potrzebną do uzyskania założonej wartości funkcji optymalizowanej przez algorytm, w którym wykorzystano logikę rozmytą do modyfikowania prawdopodobieństwa selekcji a liczbą wywołań w algorytmie elementarnym wynosi od około 10% (wartość lepsza) do około 39% (wartość lepsza).
3. Różnica między średnim czasem działania algorytmu, w którym wykorzystano logikę rozmytą do modyfikowania prawdopodobieństwa mutacji a czasem działania algorytmu elementarnego wynosi od około -59% (wartość gorsza) do około -223% (wartość gorsza).
4. Różnica między średnią liczbą wywołań funkcji obliczającej przystosowanie potrzebną do uzyskania założonej wartości funkcji optymalizowanej przez algorytm, w którym wykorzystano logikę rozmytą do modyfikowania prawdopodobieństwa mutacji a liczbą wywołań w algorytmie elementarnym wynosi od około -40% (wartość gorsza) do około 19% (wartość lepsza).
5. Różnica między średnim czasem działania algorytmu, w którym wykorzystano logikę rozmytą do modyfikowania prawdopodobieństwa selekcji i mutacji a czasem działania algorytmu elementarnego wynosi od około -63% (wartość gorsza) do około -370% (wartość gorsza).
6. Różnica między średnią liczbą wywołań funkcji obliczającej przystosowanie potrzebną do uzyskania założonej wartości funkcji optymalizowanej przez algorytm, w którym

wykorzystano logikę rozmytą do modyfikowania prawdopodobieństwa selekcji i mutacji a liczbą wywołań algorytmu elementarnego wynosi od około -31% (wartość gorsza) do około 49% (wartość lepsza).

7. Odchylenie standardowe liczby wywołań funkcji obliczającej przystosowanie osobnika w algorytmach zmodyfikowanych było w większości przypadków mniejsze, odchylenie standardowe czasu działania było we wszystkich przypadkach większe niż w algorytmie elementarnym.

Wskaźniki procentowe odniesiono do wartości uzyskanych przez algorytm elementarny.

5.2.4. Wnioski z eksperymentów

1. W uzyskanych wynikach zaobserwować można zależność czasu działania zmodyfikowanego algorytmu od rozwiązywanego zadania. W zadaniach, w których wyznaczenie funkcji przystosowania osobników nie wymaga dużego nakładu obliczeniowego (badane problemy optymalizacji funkcji), algorytm elementarny kończył działanie wcześniej. W zadaniach, w których wyznaczenie funkcji przystosowania osobników wymaga większego nakładu obliczeniowego, algorytm zmodyfikowany kończył działanie szybciej (koszt wykonania obliczeń adaptacji parametrów algorytmu był mniejszy niż koszt wynikający ze zmniejszenia liczby pokoleń).
2. Liczba wywołań funkcji obliczającej przystosowanie osobnika w algorytmach zmodyfikowanych w większości przypadków jest mniejsza niż w algorytmie elementarnym, mniejszy jest również rozrzut tych wartości.

5.3. *Porównanie skuteczności zmodyfikowanego algorytmu genetycznego i innych metod optymalizacyjnych*

Eksperyment 7

Celem eksperymentu jest zbadanie wartości funkcji optymalizowanej uzyskanej przez algorytmy genetyczne elementarne, zmodyfikowane i inne metody optymalizacyjne wzorowane na obserwacji natury.

Porównanie skuteczności działania algorytmu genetycznego elementarnego, algorytmu genetycznego zmodyfikowanego oraz metod symulowanego wyżarzania (simulated annealing), przeszukiwania tabu (tabu search) i systemów mrówkowych (ant system) przeprowadzono na przykładzie rozwiązywania problemu komiwojażera. Porównywano średnią wartość funkcji optymalizowanej, uzyskaną po upływie określonego

czasu działania algorytmu. Każdy z algorytmów uruchomiony został dziesięciokrotnie. Jako dane testowe wykorzystano wybrane zadania o liczbie miast od 24 do 29.

Do porównania wykorzystano następujące algorytmy:

- ❖ Algorytm genetyczny elementarny oraz algorytmy, w których wykorzystano logikę rozmytą do modyfikowania prawdopodobieństwa selekcji, mutacji i jednocześnie selekcji i mutacji. W algorytmie geny reprezentowane są przez liczby rzeczywiste.

W czasie obliczeń przyjęto następujące parametry pracy algorytmu:

Zadanie	Liczebność populacji	Prawdopodobieństwo krzyżowania	Prawdopodobieństwo mutacji
Gr24	20	0,6	0,20
Fri26	20	0,6	0,05
Bays29	30	0,7	0,05

Tabela 5.14. Parametry algorytmów w eksperymencie z problemem komiwojażera

- ❖ Algorytm symulowanego wyżarzania.

W czasie obliczeń przyjęto następujące parametry pracy algorytmu:

- temperatura początkowa procesu: 801,
- temperatura końcowa procesu: 1,
- liczba zmian temperatury między temperaturą początkową i końcową: 800,
- temperatura zmieniana jest liniowo.

- ❖ Algorytm przeszukiwania tabu. W algorytmie przyjęto długość listy tabu 50.

- ❖ Algorytm rozwiązywania problemu komiwojażera przy pomocy systemu mrówkowego.

- liczba mrówek: 25,
- współczynnik parowania feromonu (γ): 0,5,
- współczynnik dawki feromonu (q_3): 10000,
- współczynnik odpowiedzialny za wagę wartości heurystycznych (β): 5.

W tabelach 5.15 – 5.17. zebrano wartości funkcji optymalizowanej uzyskane przez poszczególne algorytmy, po upływie określonego czasu działania. Przedstawione wartości stanowią średnią z dziesięciu przebiegów algorytmu. Puste miejsca w tabeli oznaczają, że algorytm w określonym czasie nie uzyskał żadnego rozwiązania.

Algorytm	Czas działania algorytmu [s]									
	0,01	0,02	0,05	0,1	0,2	0,5	1	2	5	10
Algorytm elementarny	2915	2701	2491	2385	2283	2119	1995	1947	1884	1821
Algorytm z modyfikacją selekcji	2963	2757	2527	2413	2252	2095	1976	1878	1814	1778
Algorytm z modyfikacją mutacji	2926	2759	2416	2205	2058	1939	1870	1810	1736	1702
Algorytm z modyfikacją selekcji i mutacji	3007	2833	2619	2552	2495	2428	2365	2251	2148	2003
Algorytm symulowanego wyżarzania	3036	2615	2510	2510	2510	2510	2510	2510	2510	2510
Algorytm przeszukiwania tabu	3164	2898	2897	2887	2867	2839	2830	2785	2751	2753
Algorytm mrówkowy				1415	1352	1324	1294	1287	1278	1277

Tabela 5.15. Wartości funkcji optymalizowanej uzyskane przez algorytmy genetyczne i inne metody optymalizacyjne dla zadania gr24

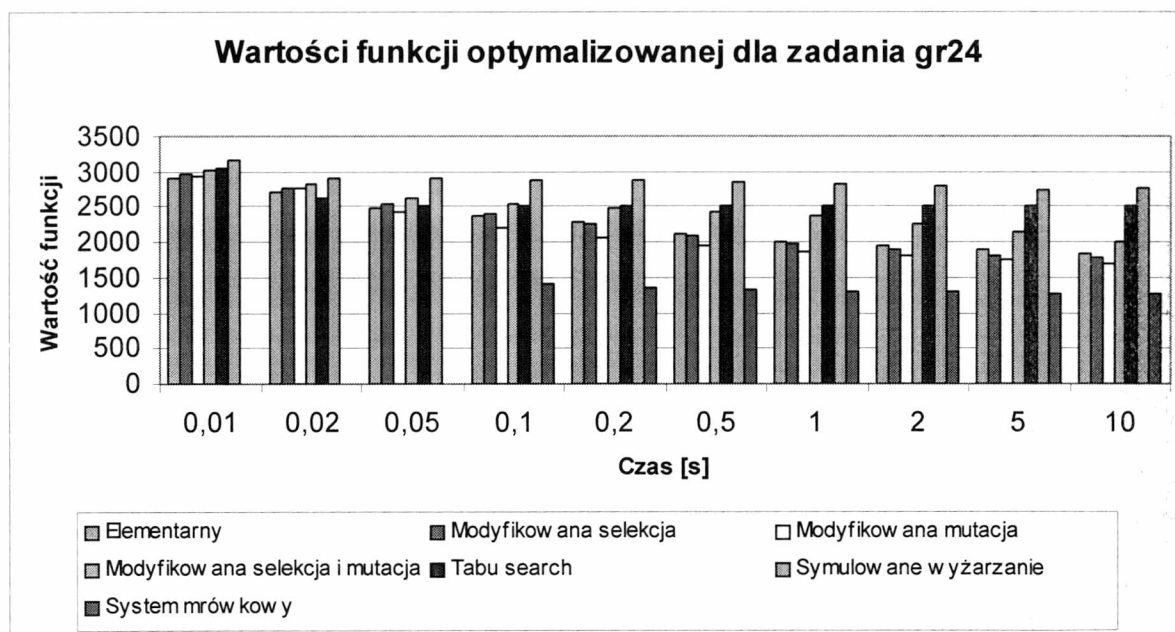
Algorytm	Czas działania algorytmu [s]									
	0,01	0,02	0,05	0,1	0,2	0,5	1	2	5	10
Algorytm elementarny		2063	1905	1787	1666	1514	1440	1380	1271	1238
Algorytm z modyfikacją selekcji		2133	1830	1758	1654	1513	1452	1338	1247	1201
Algorytm z modyfikacją mutacji		2127	1826	1755	1590	1434	1369	1280	1209	1174
Algorytm z modyfikacją selekcji i mutacji		2204	1966	1896	1845	1748	1677	1608	1513	1452
Algorytm symulowanego wyżarzania	2336	2139	2107	2107	2107	2107	2107	2107	2107	2107
Algorytm przeszukiwania tabu	2438	2411	2371	2367	2348	2336	2329	2327	2315	2294
Algorytm mrówkowy				1038	966	947	937	937	937	937

Tabela 5.16. Wartości funkcji optymalizowanej uzyskane przez algorytmy genetyczne i inne metody optymalizacyjne dla zadania fri26

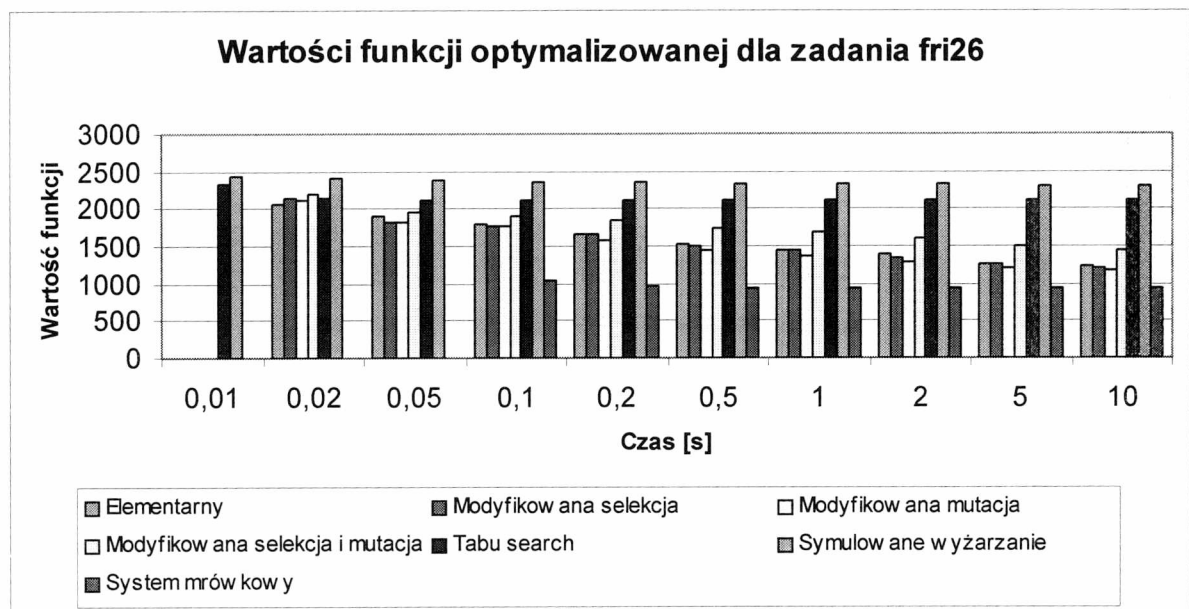
Algorytm	Czas działania algorytmu [s]									
	0,01	0,02	0,05	0,1	0,2	0,5	1	2	5	10
Algorytm elementarny		4849	4531	4242	4053	3811	3659	3602	3429	3383
Algorytm z modyfikacją selekcji		4964	4538	4411	4180	3956	3737	3541	3350	3281
Algorytm z modyfikacją mutacji		4910	4549	4418	4202	3994	3808	3623	3384	3282
Algorytm z modyfikacją selekcji i mutacji		4874	4566	4346	4122	3914	3740	3665	3527	3379
Algorytm symulowanego wyżarzania	5313	4482	4304	4304	4304	4304	4304	4304	4304	4304
Algorytm przeszukiwania tabu	5504	5375	5335	5267	5245	5215	5195	5195	5184	5158
Algorytm mrówkowy				2469	2268	2140	2091	2080	2071	2065

Tabela 5.17. Wartości funkcji optymalizowanej uzyskane przez algorytmy genetyczne i inne metody optymalizacyjne dla zadania bays29

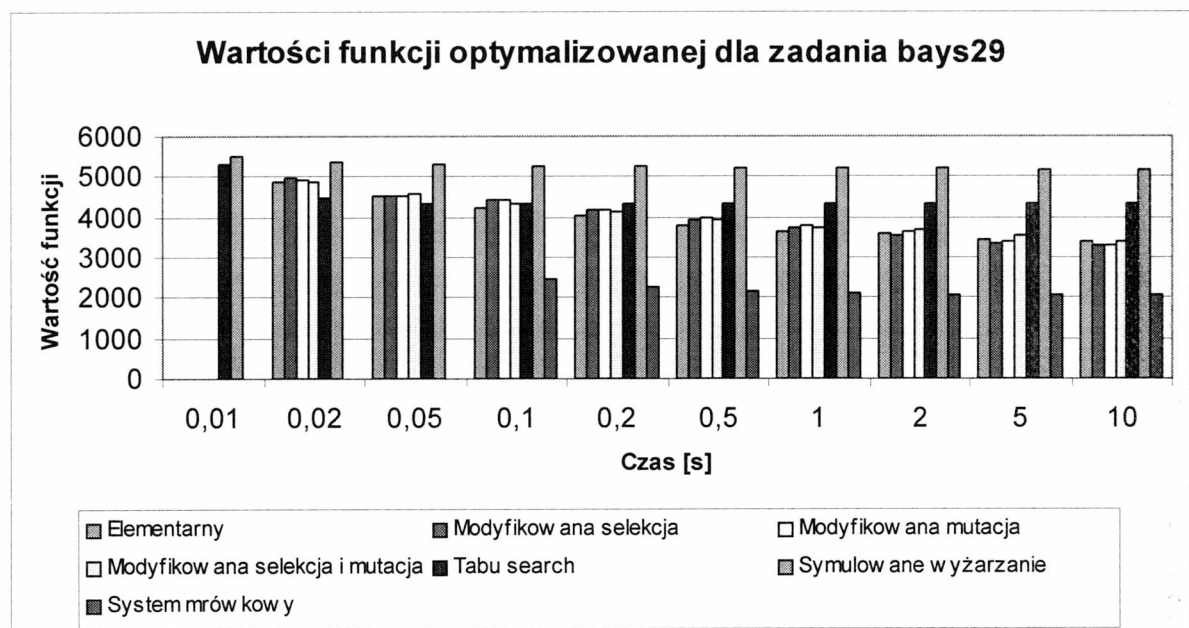
Wykresy na Rys. 5.20. – 5.22. pokazują różnice między wartościami funkcji optymalizowanej uzyskanymi przez poszczególne algorytmy po upływie określonego czasu.



Rys. 5.20. Średnie wartości funkcji optymalizowanej uzyskane przez różne metody optymalizacji dla zadania gr24



Rys. 5.21. Średnie wartość funkcji optymalizowanej uzyskane przez różne metody optymalizacji dla zadania fri26



Rys. 5.22. Średnie wartość funkcji optymalizowanej uzyskane przez różne metody optymalizacji dla zadania bays29

Spostrzeżenia z eksperymentu.

1. Wartości funkcji optymalizowanej uzyskane przez algorytmy wspomagane logiką rozmytą w porównaniu z algorytmem elementarnym są:
 - od około -3% gorsze do około 4% lepsze w algorytmie, w którym wykorzystano logikę rozmytą do modyfikacji prawdopodobieństwa selekcji,

- od około -4% gorsze do około 9% lepsze w algorytmie, w którym wykorzystano logikę rozmytą do modyfikacji prawdopodobieństwa mutacji,
 - od równej do około -18% gorsze w algorytmie, w którym wykorzystano logikę rozmytą do modyfikacji prawdopodobieństwa selekcji i mutacji.
2. Wartości funkcji optymalizowanej uzyskane przez algorytm genetyczny były lepsze od wartości uzyskanych przez algorytmy symulowanego wyżarzania i przeszukiwania tabu. Algorytmy te stosunkowo szybko uzyskiwały wartość funkcji, która w późniejszym okresie nie ulegała większym zmianom.
 3. Wartości funkcji optymalizowanej uzyskane przez algorytm mrówkowy były lepsze od wartości uzyskanych przez algorytmy genetyczne. Przy większej liczbie miast czas działania algorytmu mrówkowego gwałtownie rośnie.

Wskaźniki procentowe odniesiono do wartości uzyskanych przez algorytm elementarny.

5.3.1. Wnioski z eksperymentu

1. Największe korzyści (poprawę otrzymanej wartości funkcji optymalizowanej) z zastosowania kierowania ewolucją osobników w algorytmach genetycznych uzyskano na skutek modyfikacji prawdopodobieństwa mutacji.
2. Algorytm genetyczny, w którym wykorzystano logikę rozmytą uzyskał wyniki lepsze niż algorytm genetyczny elementarny, jednak w porównaniu z metodami symulowanego wyżarzania i przeszukiwania tabu stwierdzić należy, że metody te uzyskują rezultaty lepsze pod względem szybkości działania. Algorytm mrówkowy uzyskał rezultaty najlepsze pod względem wartości funkcji optymalizowanej, jednak czas działania tego algorytmu przy dużej liczbie miast był znacznie dłuższy niż w innych metodach.

5.4. Działanie sterownika rozmytego przy różnych parametrach algorytmu genetycznego

W tym etapie zmodyfikowane algorytmy genetyczne wykorzystane zostały do rozwiązania zadania optymalizacyjnego przy różnej liczebności populacji i liczbie pokoleń. Jako zadania testowe wykorzystano problem optymalizacji funkcji zaproponowany w pracy [Mic99]. Dla zadania testowego przeprowadzono serię eksperymentów obejmującą:

- badanie wartości funkcji optymalizowanej po określonej liczbie pokoleń $MAXGENS \in \{50, 100, 150, 200, 250\}$,

- badanie liczby pokoleń potrzebnych, aby algorytm osiągnął założoną wartość funkcji optymalizowanej.

W eksperymentach przyjęto jednakowe parametry algorytmów:

- prawdopodobieństwo krzyżowania 0,7,
- prawdopodobieństwo mutacji 0,1,
- liczebność populacji POPSIZE $\in \{20, 25, 30, 40, 50\}$.

W eksperymencie tym badano zachowanie zmodyfikowanych algorytmów genetycznych w zadaniach optymalizacji, w których wykorzystano kodowanie rzeczywiste.

Do rozważań wybrano prostą, ciągłą funkcję testową o trzech zmiennych i jednym optimum globalnym (funkcja M1 zaproponowana w pracy [Mic99]).

5.4.1. Badanie wartości funkcji optymalizowanej

Eksperyment 8

Celem eksperymentu jest zbadanie wpływu sterownika rozmytego na wartości funkcji optymalizowanej osiągniętej przez algorytmy zmodyfikowane oraz porównanie ich wyników z wynikami uzyskanymi przez algorytm elementarny przy różnych wielkościach populacji i liczbie pokoleń.

W tabelach 5.18 – 5.21. przedstawione zostały uśrednione dla 10 przebiegów algorytmu, oraz różnych wartości liczebności populacji (POPSIZE) i liczby pokoleń (MAXGENS), wartości funkcji optymalizowanej otrzymane w wyniku działania poszczególnych algorytmów.

Algorytm elementarny						
		POPSIZE				
		20	25	30	40	50
MAXGENS	50	964581,2	971366,7	971918,4	974687,0	974866,1
	100	982765,7	980934,4	981517,4	983732,1	982180,3
	150	984843,2	984643,1	986047,3	985524,8	985063,7
	200	984980,4	987545,4	987115,0	990218,1	988009,9
	250	987830,9	988310,0	987616,2	992093,2	989486,3

Tabela 5.18. Wartości funkcji optymalizowanej uzyskane przez algorytm elementarny w zadaniu optymalizacji funkcji M1

Algorytm z modyfikacją prawdopodobieństwa selekcji						
		POPSIZE				
		20	25	30	40	50
MAXGENS	50	970266,1	962682,7	958077,7	969133,8	971478,5
	100	978223,7	981227,1	973037,7	979312,1	982591,9
	150	982787,6	987855,3	980048,6	982871,2	989387,9
	200	986918,9	989489,3	983608,8	988771,0	991428,6
	250	987497,9	989599,9	988752,9	990394,0	992644,7

Tabela 5.19. Wartości funkcji optymalizowanej uzyskane przez algorytm z modyfikacją prawdopodobieństwa selekcji w zadaniu optymalizacji funkcji M1

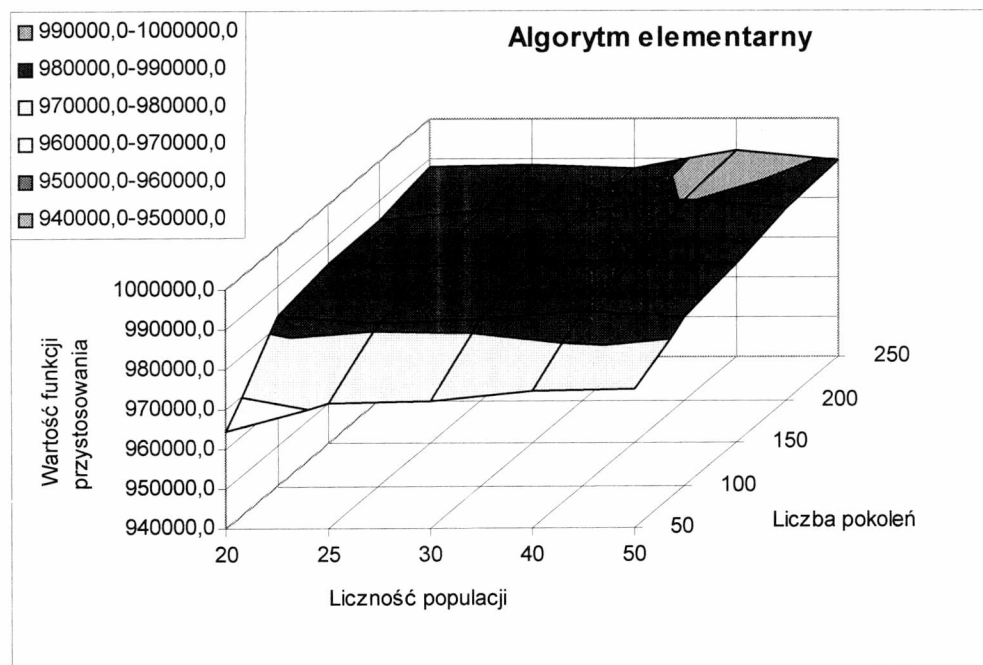
Algorytm z modyfikacją prawdopodobieństwa mutacji						
		POPSIZE				
		20	25	30	40	50
MAXGENS	50	972236,8	956917,1	958702,8	968400,8	971765,0
	100	985663,0	978053,4	980216,4	981809,8	976239,2
	150	988205,1	983011,9	986019,0	988185,6	984953,4
	200	990077,8	987394,3	987492,5	991722,4	989191,3
	250	993460,4	991581,5	990549,9	991879,6	993885,7

Tabela 5.20. Wartości funkcji optymalizowanej uzyskane przez algorytm z modyfikacją prawdopodobieństwa mutacji w zadaniu optymalizacji funkcji M1

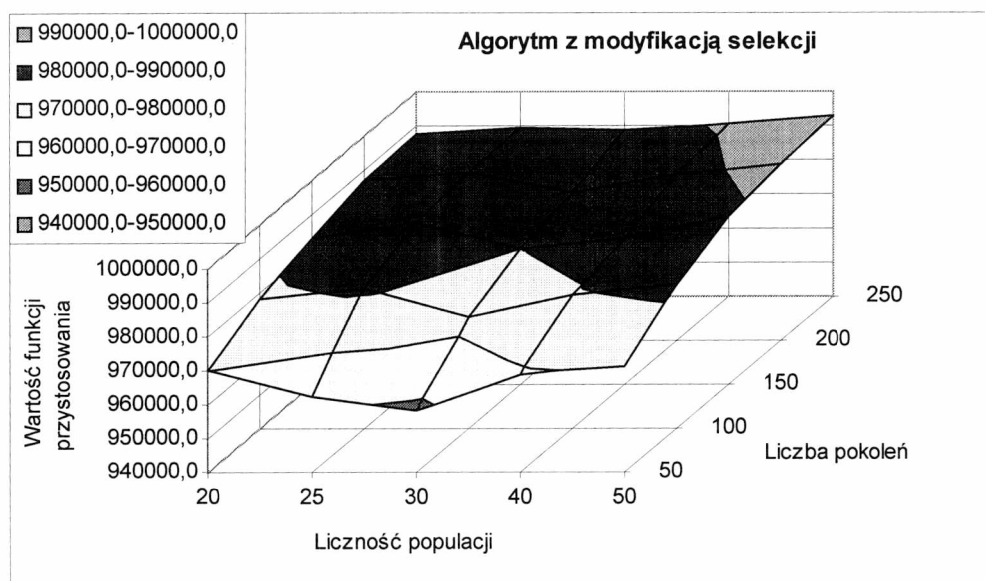
Algorytm z modyfikacją prawdopodobieństwa selekcji i mutacji						
		POPSIZE				
		20	25	30	40	50
MAXGENS	50	962709,3	967458,3	971828,1	974071,7	966115,2
	100	977434,1	974890,4	983777,6	985761,7	979883,9
	150	984806,2	984699,7	986595,5	988011,6	984381,3
	200	987530,5	988297,6	988484,0	989911,1	991262,8
	250	990116,3	990274,8	991855,3	992169,4	994310,9

Tabela 5.21. Wartości funkcji optymalizowanej uzyskane przez algorytm z modyfikacją prawdopodobieństwa selekcji i mutacji w zadaniu optymalizacji funkcji M1

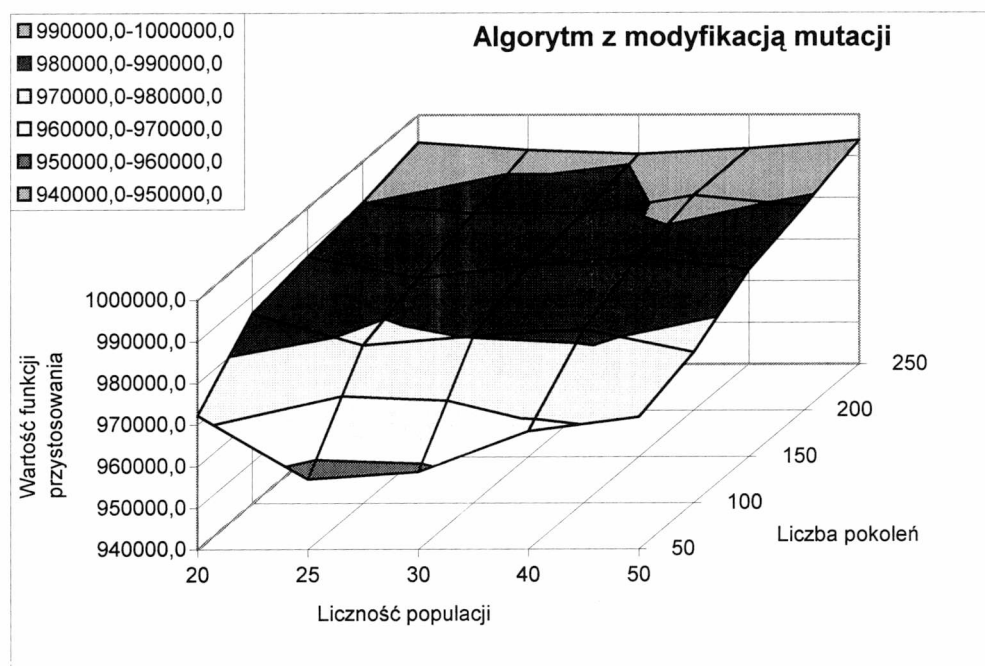
Wykresy na Rys. 5.23. – 5.26. pokazują wartości funkcji optymalizowanej parametrów zależności od przyjętych parametrów algorytmu genetycznego.



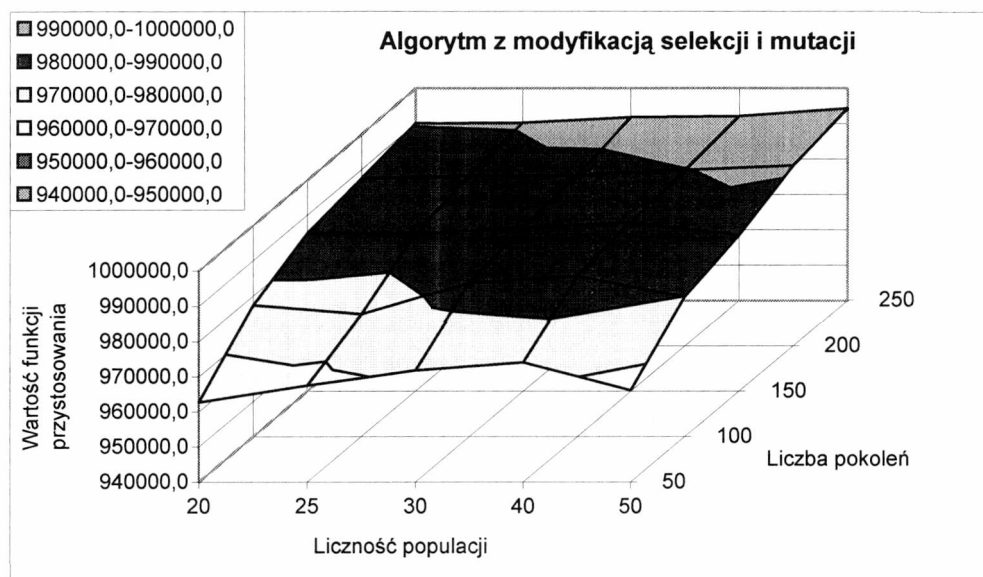
Rys. 5.23. Funkcja przystosowania najlepszego osobnika w elementarnym algorytmie genetycznym



Rys. 5.24. Funkcja przystosowania najlepszego osobnika w algorytmie z modyfikacją selekcji



Rys. 5.25. Funkcja przystosowania najlepszego osobnika w algorytmie z modyfikacją mutacji



Rys. 5.26. Funkcja przystosowania najlepszego osobnika w algorytmie genetycznym z modyfikacją selekcji i mutacji

Spostrzeżenia z eksperymentu.

1. Wartości funkcji optymalizowanej uzyskane przez algorytm, w którym wykorzystano logikę rozmytą do adaptacji prawdopodobieństwa selekcji, były w 44% przypadków lepsze od wartości funkcji uzyskanych przez algorytm elementarny. Różnica między średnimi wartościami funkcji optymalizowanej uzyskanymi przez algorytm, w którym wykorzystano logikę rozmytą a wartościami uzyskanymi przez algorytm elementarny wynosi od -1,42% (wartość gorsza) do 0,59% (wartość lepsza), w zależności od liczebności populacji i liczby pokoleń.
2. Wartości funkcji optymalizowanej, uzyskane przez algorytm, w którym wykorzystano logikę rozmytą do adaptacji prawdopodobieństwa mutacji, były w 48% przypadków lepsze od wartości funkcji uzyskanych przez algorytm elementarny. Różnica między wartościami funkcji optymalizowanej uzyskanymi przez algorytm, w którym wykorzystano logikę rozmytą a wartościami uzyskanymi przez algorytm elementarny wynosi od -1,48% (wartości gorsze) do 0,79% (wartości lepsze) w zależności od liczebności populacji i liczby pokoleń.
3. Wartości funkcji optymalizowanej, uzyskane przez algorytm, w którym wykorzystano logikę rozmytą do adaptacji prawdopodobieństwa selekcji i mutacji, były w 56% przypadków lepsze od wartości funkcji uzyskanych przez algorytm elementarny. Różnica między wartościami funkcji optymalizowanej uzyskanymi przez algorytm, w którym wykorzystano logikę rozmytą a wartościami uzyskanymi przez algorytm elementarny

wynosi od -0,89% (wartości gorsze) do 0,48% (wartości lepsze), w zależności od liczebności populacji i liczby pokoleń.

Wskaźniki procentowe odniesiono do wartości uzyskanych przez algorytm elementarny.

5.4.2. Badanie liczby pokoleń potrzebnych, aby algorytm osiągnął założoną wartość funkcji optymalizowanej

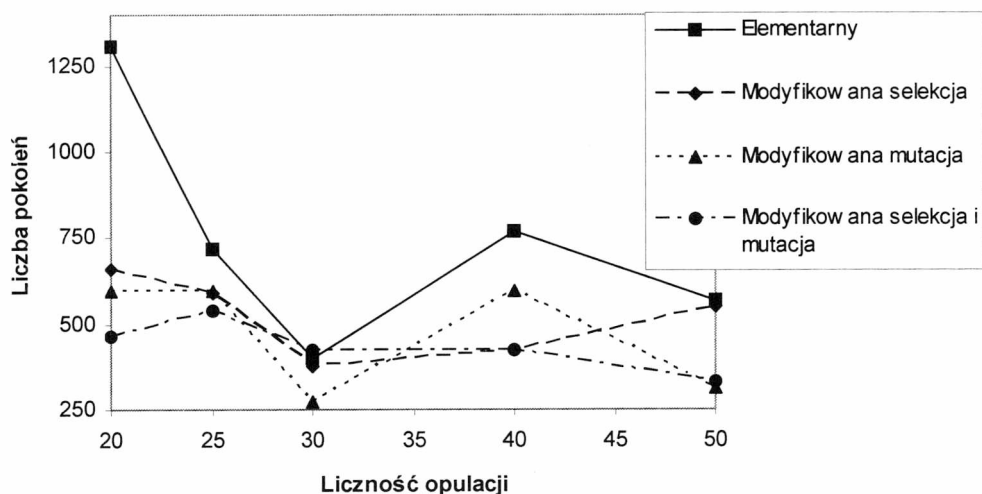
Eksperyment 9

Celem eksperymentu jest zbadanie wpływu sterownika rozmytego na liczbę pokoleń, po których wartość funkcji optymalizowanej osiągnięta przez algorytmy zmodyfikowane przekroczy założoną wartość oraz porównanie ich wyników z wynikami uzyskanymi przez algorytm elementarny przy różnych wielkościach populacji.

W eksperymencie przyjęto, że zakończenie działania algorytmu nastąpi po uzyskaniu wartości funkcji przystosowania $\text{fitness}=995000$. W tabeli 5.22. przedstawione zostały uśrednione dla 10 różnych przebiegów algorytmu, liczby pokoleń potrzebne do osiągnięcia założonej wartości funkcji optymalizowanej, otrzymane w wyniku działania poszczególnych algorytmów przy różnej liczebności populacji. Wykres na Rys. 5.27. ilustruje różnicę między liczbą pokoleń potrzebną do osiągnięcia założonej wartości w poszczególnych algorytmach.

Algorytm	POPISZE				
	20	25	30	40	50
Algorytm elementarny	1308,7	717,9	401,4	771,0	569,5
Algorytm z modyfikacją selekcji	661,6	589,6	378,5	423,0	551,4
Algorytm z modyfikacją mutacji	597,9	595,1	275,8	597,9	314,2
Algorytm z modyfikacją selekcji i mutacji	463,5	537,5	422,8	425,6	330,3

Tabela 5.22. Liczba pokoleń potrzebna do osiągnięcia założonej wartości funkcji przystosowania w zadaniu optymalizacji funkcji



Rys. 5.27. Liczba pokoleń potrzebna, aby algorytm osiągnął założoną wartość w zadaniu optymalizacji funkcji

Spostrzeżenia z eksperymentu.

1. Różnica między liczbą pokoleń potrzebną do osiągnięcia założonej wartości funkcji optymalizowanej w algorytmie elementarnym a liczbą pokoleń w algorytmie, w którym wykorzystano logikę rozmytą, odniesiona do wartości uzyskanej przez algorytm elementarny wynosi od -5,33% (wartość gorsza) do 64,58% (wartość lepsza) w zależności od liczebności populacji i sposobu wykorzystania bloków logiki rozmytej.
2. Tylko w 1 spośród 15 przypadków (6,66% ogółu przebiegów algorytmu) nie uzyskano zmniejszenia liczby pokoleń, potrzebnych do osiągnięcia założonej wartości funkcji, w pozostałych przypadkach algorytm, w którym wykorzystano logikę rozmytą był lepszy od algorytmu elementarnego.

5.4.3. Wnioski z eksperymentów

1. Większy spadek liczby pokoleń, potrzebnych do uzyskania założonej wartości funkcji optymalizowanej, spowodowany zastosowaniem logiki rozmytej do kierowania ewolucją, uzyskano przy większej liczbie osobników w populacji, przy małej liczbie osobników w populacji uzyskane wyniki były mniej stabilne a różnice pomiędzy poszczególnymi przebiegami algorytmu większe.
2. W przeprowadzonym eksperymencie najwyższe wartości funkcji optymalizowanej uzyskano w algorytmach, w których zastosowano bloki logiki rozmytej do modyfikowania prawdopodobieństwa selekcji i mutacji poszczególnych osobników.
3. W przeprowadzonym eksperymencie najmniej pokoleń do uzyskania założonej wartości funkcji optymalizowanej potrzebował algorytm, w którym wykorzystano bloki logiki rozmytej do modyfikowania prawdopodobieństwa selekcji i mutacji poszczególnych osobników.
4. Elementarny algorytm jak i algorytm, w którym wykorzystano logikę rozmytą są zbieżne, zauważyć można, że wraz ze wzrostem liczby pokoleń, po której dokonywano porównania, różnica pomiędzy wartościami funkcji optymalizowanej uzyskanymi przez poszczególne algorytmy zmniejsza się.

5.5. Zdolność algorytmu do wyszukiwania rozwiązania optymalnego

Eksperyment 10

Celem eksperymentu jest zbadanie zdolności zmodyfikowanych algorytmów do wyszukiwania rozwiązania optymalnego dla funkcji ciągłych i nieciągłych.

Algorytmy genetyczne na ogół wyznaczają rozwiązanie bliskie optymalnemu a nie rozwiązanie optymalne. W tym eksperymencie algorytm kończył działanie po osiągnięciu wartości optymalnej. Ze względu na te właściwości algorytmów genetycznych jako zadania testowe wybrano problemy łatwe, w których algorytmy są w stanie znaleźć rozwiązanie optymalne. W eksperymencie badano czas i liczbę wywołań funkcji obliczającej wartość przystosowania osobnika, potrzebne do osiągnięcia przez algorytm wartości optymalnej.

W tym eksperymencie rozważano dwa zadania testowe:

- problem komiwojażera dla 12 miast rozmieszczonych na okręgu (zadanie podobne do rozpatrywanego w pracy [Dol]). Miasta rozmieszczono w równych odstępach na okręgu o promieniu 100.
- zaproponowany w pracy [Mic99] problem optymalizacji funkcji M2

Przyjęto parametry algorytmu genetycznego:

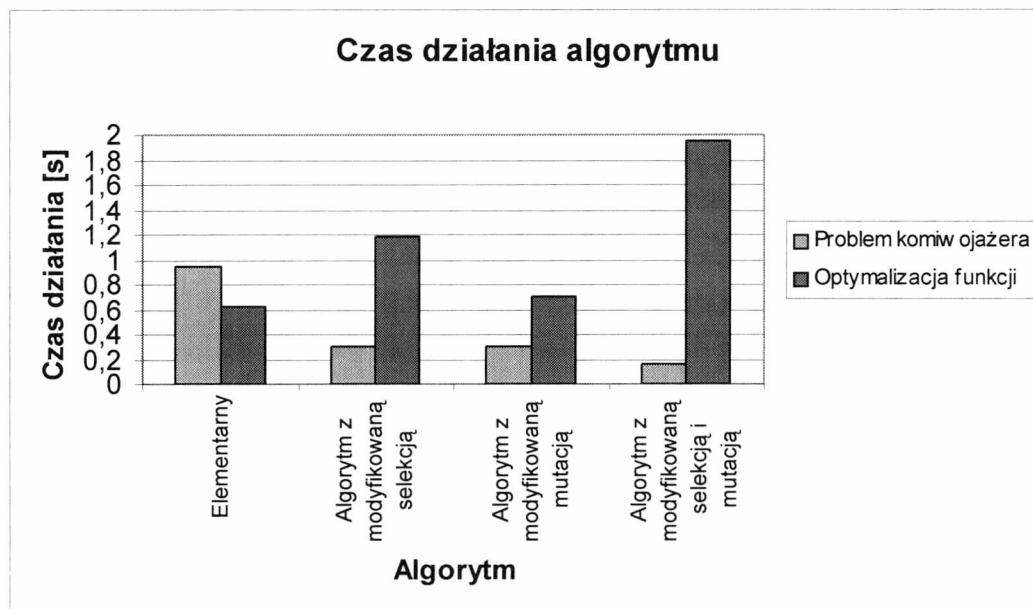
- dla problemu komiwojażera:
 - o geny reprezentowane są przez ciągi 12 liczb z przedziału (0, 11) – reprezentacja porządkowa [Mic99],
 - o funkcja przystosowania dla osobnika obliczana jako suma odległości między kolejnymi miastami na trasie komiwojażera,
- dla optymalizacji funkcji: kodowanie rzeczywiste,
- liczebność populacji: problem komiwojażera 25, optymalizacja funkcji 20,
- prawdopodobieństwo krzyżowania: problem komiwojażera 0,8, optymalizacja funkcji 0,5,
- prawdopodobieństwo mutacji: problem komiwojażera 0,15, optymalizacja funkcji 0,25,
- działanie algorytmu kończone po uzyskaniu wartości optymalnej.

W tabeli 5.23. przedstawione zostały, uśrednione dla 10 różnych przebiegów, wartości czasu działania poszczególnych algorytmów i liczby wywołań funkcji obliczającej przystosowanie osobnika (oraz odchylenia standardowe tych wartości), po których algorytmy osiągnęły wartość optymalną.

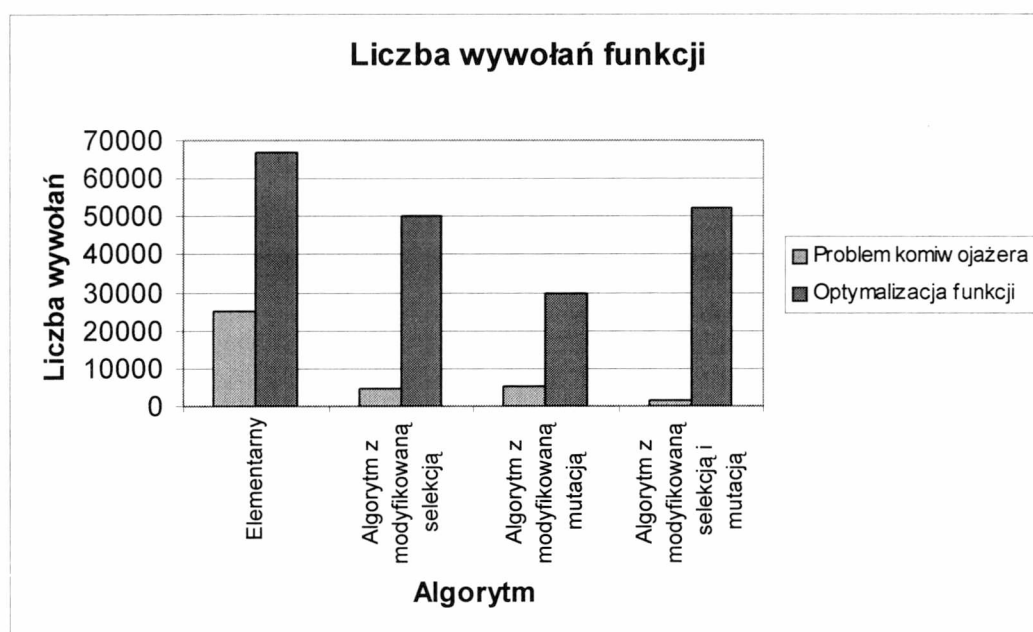
Algorytm	Problem komiwojażera				Optymalizacja funkcji			
	Czas [s]	δ	Liczba wyw.	δ	Czas [s]	δ	Liczba wyw.	δ
Elementarny	0,94	0,79	25220	21085	0,62	0,29	66952	30442
Algorytm z modyfikowaną selekcją	0,3	0,10	4840	1677	1,18	0,67	49905	28456
Algorytm z modyfikowaną mutacją	0,31	0,23	5095	3988	0,71	0,27	29700	11968
Algorytm z modyfikowaną selekcją i mutacją	0,16	0,11	1825	1439	1,96	0,83	52437	22213

Tabela 5.23. Czas działania i liczba pokoleń potrzebna do uzyskania rozwiązania optymalnego

Wykresy na Rys. 5.28-5.29. pokazują wartości czasu działania i liczby wywołań funkcji obliczającej przystosowanie osobnika potrzebne do uzyskania wartości optymalnej w zależności od rodzaju algorytmu.



Rys. 5.28. Czas działania potrzebny do osiągnięcia wartości optymalnej



Rys. 5.29. Liczba wywołań funkcji potrzebna do osiągnięcia wartości optymalnej

Spostrzeżenia z eksperymentu.

1. Czas działania algorytmu zmodyfikowanego był:
 - a. około 68% krótszy (wartość lepsza) w problemie komiwojażera i około -90% dłuższy (wartość gorsza) w optymalizacji funkcji w algorytmie z adaptacją selekcji,

- b. około 67% krótszy (wartość lepsza) w problemie komiwojażera i około -14% dłuższy (wartość gorsza) w optymalizacji funkcji w algorytmie z adaptacją mutacji,
 - c. około 82% krótszy (wartość lepsza) w problemie komiwojażera i około -216% dłuższy (wartość gorsza) w optymalizacji funkcji w algorytmie z adaptacją selekcji i mutacji.
2. Liczba wywołań funkcji obliczającej przystosowanie osobnika potrzebna do osiągnięcia wartości optymalnej uzyskana przez algorytmy zmodyfikowane była w porównaniu z algorytmem elementarnym:
- a. około 25% mniejsza (wartość lepsza) w optymalizacji funkcji i około 80% mniejsza (wartość lepsza) w problemie komiwojażera, w algorytmie z adaptacją selekcji,
 - b. około 55% mniejsza (wartość lepsza) w optymalizacji funkcji i około 79% mniejsza (wartość lepsza) w problemie komiwojażera, w algorytmie z adaptacją mutacji,
 - c. około 21% mniejsza (wartość lepsza) w optymalizacji funkcji i około 92% mniejsza (wartość lepsza) w problemie komiwojażera, w algorytmie z adaptacją selekcji i mutacji.

Wskaźniki procentowe odniesiono do wyników algorytmu elementarnego.

5.5.1. Wnioski z eksperymentu

W obu zadaniach testowych liczba wywołań funkcji obliczającej przystosowanie osobnika w algorytmach z adaptacją parametrów jest mniejsza niż w algorytmie elementarnym. W optymalizacji funkcji wybrano łatwą w optymalizacji funkcję, co powoduje, że algorytm elementarny jest wystarczająco skuteczny, zastosowanie adaptacji parametrów nie powoduje skrócenia czasu działania. W problemie komiwojażera funkcja optymalizowana jest nieciągła i trudna w optymalizacji, co może powodować, że algorytmy z adaptacją parametrów działają krócej niż algorytm elementarny.

5.6. Zdolność algorytmu do wychodzenia z optimum lokalnego

W tym etapie badano zdolność algorytmu do opuszczenia obszaru, w którym występuje optimum lokalne na przykładzie problemu optymalizacji funkcji. W zadaniach tych poszczególne geny osobników reprezentowane były za pomocą liczb rzeczywistych.

Działanie algorytmu kończono po znalezieniu założonej wartości funkcji optymalizowanej. Zadanie rozwiązywano przy pomocy algorytmu genetycznego elementarnego i algorytmów, w którym wykorzystano logikę rozmytą. Wyniki porównano z rezultatami działania algorytmów pracujących w oparciu o K-model, opublikowanymi w pracy [Kwa99g].

Przyjęto parametry algorytmu genetycznego:

- geny reprezentowane są przez liczby rzeczywiste,
- populacja początkowa umieszczona została w maksimum lokalnym,
- liczebność populacji: 100,

Parametr	Funkcja testowa					
	Q1	Q2	Q4	Q5	Q6	Q7
Prawdopodobieństwo krzyżowania	0,5	0,6	0,9	0,5	0,7	0,6
Prawdopodobieństwo mutacji	0,2	0,05	0,05	0,05	0,05	0,15

Tabela 5.24. Parametry algorytmów stosowane do optymalizacji funkcji eksponencjalnych

Eksperyment 11

Celem eksperymentu jest zbadanie wpływu sterownika rozmytego na średnią liczbę pokoleń potrzebnych, aby średnia wartość funkcji przystosowania populacji osiągnęła założoną wartość.

Jako zadania testowe wybrano problem optymalizacji funkcji od 1 do 5 zmiennych, posiadających jedno maksimum lokalne i jedno maksimum globalne. Działanie algorytmu kończono po osiągnięciu przez średnią wartość funkcji przystosowania populacji założonej wartości funkcji, co może świadczyć, że algorytm opuścił optimum lokalne.

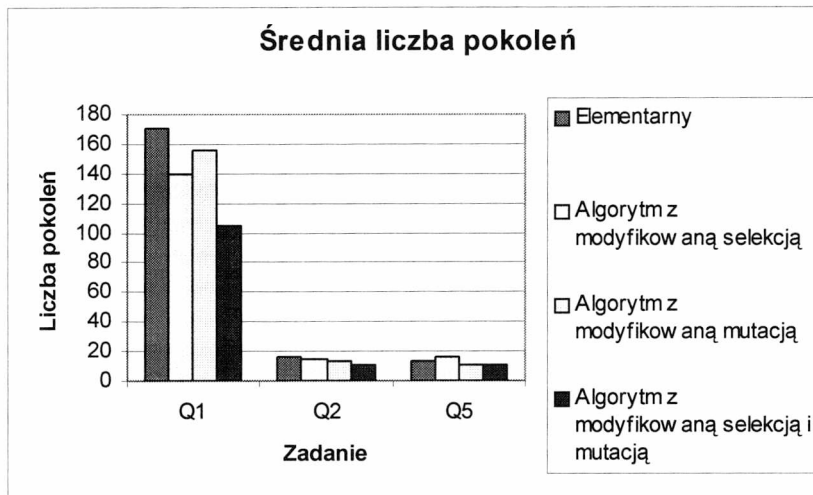
W tabeli 5.25. przedstawione zostały, uśrednione dla 10 różnych przebiegów, wartości liczby pokoleń oraz odchylenia standardowe tej wartości, potrzebnych aby średnia wartość funkcji przystosowania osobników uzyskała założoną wartość. W algorytmach z modyfikacją prawdopodobieństwa mutacji i jednocześnie selekcji i mutacji lepsze wyniki uzyskano przy zmniejszonych wartościach prawdopodobieństwa mutacji. Dla funkcji Q1 przyjęto prawdopodobieństwo mutacji 0,1 w algorytmie z modyfikacją prawdopodobieństwa mutacji i 0,15 w algorytmie z modyfikacją prawdopodobieństwa selekcji i mutacji. Dla funkcji Q2 i Q5 przyjęto prawdopodobieństwo mutacji 0,02.

Algorytm	Q1		Q2		Q5	
Założona wartość funkcji	1,15		1,15		4,7	
	Liczba pokoleń	δ	Liczba pokoleń	δ	Liczba pokoleń	δ
Elementarny	171	101	16,7	4,64	13,5	3,82
Algorytm z modyfikowaną selekcją	140	88	14,4	2,28	15,5	3,26
Algorytm z modyfikowaną mutacją	156	80	12,8	3,78	11,3	2,57
Algorytm z modyfikowaną selekcją i mutacją	105	59	10,1	1,81	10,9	2,16
K-model ¹	3601	-	371	-	195	-

Tabela 5.25. Średnia liczba pokoleń potrzebnych, aby algorytm opuścił lokalne optimum

¹ Wyniki badań na podstawie pracy: Kwaśnicka H., *Obliczenia ewolucyjne w sztucznej inteligencji*. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław, 1999.

Wykres na Rys. 5.30. pokazuje różnice w liczbie pokoleń potrzebnych, aby średnia wartość funkcji przystosowania osobników uzyskała założoną wartość.



Rys. 5.30. Liczba pokoleń potrzebna, aby średnia wartość funkcji przystosowania osobników uzyskała założoną wartość

Spostrzeżenia z eksperymentu.

1. Liczba pokoleń potrzebnych do uzyskania założonej wartości funkcji była:
 - a. od około -14% większa (wartość gorsza) do około 18% mniejsza (wartość lepsza) dla algorytmu z adaptacją selekcji,
 - b. od około 8% mniejsza (wartość lepsza) do około 23% mniejsza (wartość lepsza) dla algorytmu z adaptacją mutacji,
 - c. od około 19% mniejsza (wartość lepsza) do około 39% mniejsza (wartość gorsza) dla algorytmu z adaptacją selekcji i mutacji.

Wskaźniki procentowe odniesiono do wyników algorytmu elementarnego.

W algorytmie elementarnym i algorytmach zmodyfikowanych występowały problemy dla funkcji testowych o większej liczbie zmiennych. Na skutek dużego zróżnicowania populacji algorytmy te miały kłopoty z uzyskaniem założonej wartości średniej funkcji przystosowania. Problem ten rozwiązano poprzez zmniejszenie prawdopodobieństwa mutacji, co pozwoliło na zmniejszenie zróżnicowania populacji (przy jednoczesnym zmniejszeniu zdolności algorytmu do eksploracji). Dla ciągłych funkcji testowych lepsze wyniki uzyskuje się stosując mutację pełzającą.

W porównaniu z wynikami działania algorytmu opartego o K-model możemy stwierdzić, że algorytmy osiągnęły założoną wartość funkcji optymalizowanej przy mniejszej liczbie pokoleń. Zdolność do opuszczenia optimum lokalnego algorytmu opartego o K-model

może być większa dzięki makromutacjom oraz wykorzystaniu efektu pleiotropowego i poligenicznego.

Eksperyment 12

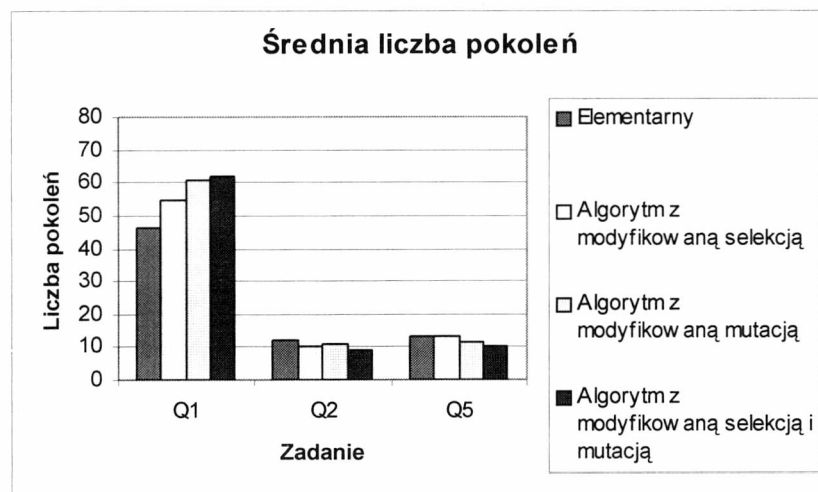
Celem eksperymentu jest zbadanie wpływu sterownika rozmytego na średnią liczbę pokoleń potrzebnych, aby populacja osobników przemieściła się do obszaru maksimum globalnego.

Jako zadania testowe wybrano problem optymalizacji funkcji od 1 do 5 zmiennych, posiadających jedno maksimum lokalne i jedno maksimum globalne. Działanie algorytmu kończono, gdy wartość funkcji przystosowania przynajmniej połowy osobników przekroczyła założoną wartość, co może świadczyć, że populacja zasiedliła obszar, w którym występuje maksimum globalne. W tabeli 5.26. przedstawione zostały, uśrednione dla 10 różnych przebiegów, wartości liczby pokoleń potrzebnych, aby funkcja przystosowania przynajmniej połowy osobników przekroczyła założoną wartość oraz odchylenia standardowe tej wartości.

Algorytm	Q1		Q2		Q5	
Założona wartość funkcji	1,15		1,15		4,9	
	Liczba pokoleń	δ	Liczba pokoleń	δ	Liczba pokoleń	δ
Elementarny	46,5	20,9	12,1	2,62	13,2	12,4
Algorytm z modyfikowaną selekcją	55	24,0	10,3	1,67	13,5	5,0
Algorytm z modyfikowaną mutacją	61	21,4	10,8	4,01	11,4	3,8
Algorytm z modyfikowaną selekcją i mutacją	62	31,5	8,9	1,44	10	3,1

Tabela 5.26. Średnia liczba pokoleń potrzebnych, aby algorytm opuścił lokalne optimum

Wykres na Rys. 5.31. pokazuje różnice w liczbie pokoleń potrzebnych, aby wartość funkcji przystosowania przynajmniej połowy osobników przekroczyła założoną wartość.



Rys. 5.31. Liczba pokoleń potrzebna, aby wartość funkcji przystosowania przynajmniej połowy osobników przekroczyła założoną wartość

Spostrzeżenia z eksperymentu.

1. Liczba pokoleń potrzebna do uzyskania założonej wartości funkcji w porównaniu z algorytmem elementarnym była:
 - a. od około -18% większa (wartość gorsza) do około 14% mniejsza (wartość lepsza) dla algorytmu z adaptacją selekcji,
 - b. od około -31% większa (wartość gorsza) do około 13% mniejsza (wartość lepsza) dla algorytmu z adaptacją mutacji,
 - c. od około -33% większa (wartość gorsza) do około 26% mniejsza (wartość lepsza) dla algorytmu z adaptacją selekcji i mutacji.

Wskaźniki procentowe odniesiono do wyników algorytmu elementarnego.

Algorytmy szybko znajdują rozwiązanie bliskie optymalnemu jednak w populacji występuje duże zróżnicowanie osobników.

Eksperyment 13

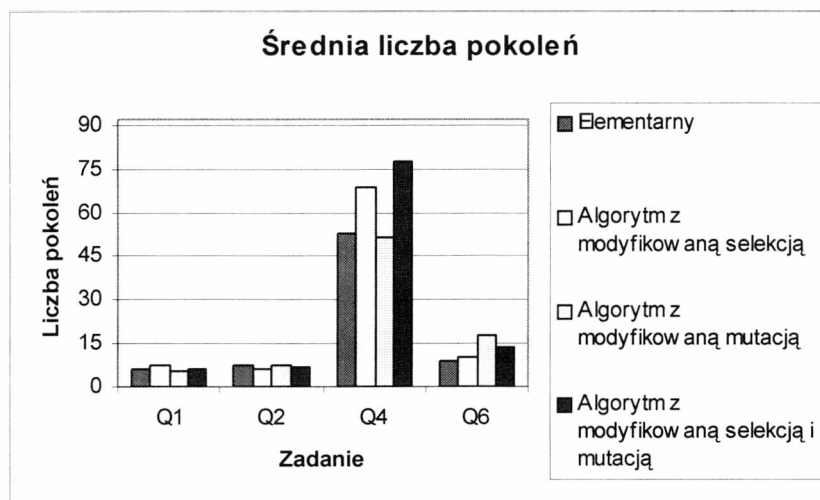
Celem eksperymentu jest zbadanie wpływu sterownika rozmytego na liczbę pokoleń, po których następowała zmiana wartości funkcji przystosowania najlepszego osobnika.

Jako zadania testowe wybrano problem optymalizacji funkcji od 2 do 30 zmiennych, posiadających jedno maksimum lokalne i jedno maksimum globalne. Działanie algorytmu kończono, gdy wartość funkcji przystosowania najlepszego osobnika przekroczyła założoną wartość. Badano średnią liczbę pokoleń, po których następowała zmiana wartości najlepszego osobnika, co może świadczyć o opuszczeniu lokalnego optimum. W tabeli 5.27. przedstawione zostały, uśrednione dla 10 różnych przebiegów, wartości liczby pokoleń (oraz odchylenia standardowe tej liczby), po których następowała zmiana wartości najlepszego osobnika.

Algorytm	Q1		Q2		Q4		Q6	
Założona wartość funkcji	1,499		1,47		1,35		1,45	
	Liczba pokoleń	δ	Liczba pokoleń	δ	Liczba pokoleń	δ	Liczba pokoleń	δ
Elementarny	6,1	2,34	7,3	3,40	53,2	11,8	8,7	2,68
Algorytm z modyfikowaną selekcją	7,7	2,68	6,2	2,67	69	20,9	10,1	3,23
Algorytm z modyfikowaną mutacją	5,4	3,03	7,6	2,69	51,5	20,7	17,8	6,47
Algorytm z modyfikowaną selekcją i mutacją	5,9	2,80	7	2,79	77,9	12,9	13,4	4,52

Tabela 5.27. Średnia liczba pokoleń, po których następowała zmiana wartości najlepszego osobnika

Wykres na Rys. 5.32. pokazuje różnice w liczbie pokoleń, po których następowała zmiana wartości najlepszego osobnika w zależności od rodzaju algorytmu.



Rys. 5.32. Liczba pokoleń, po których następowała zmiana wartości najlepszego osobnika

Spostrzeżenia z eksperymentu.

1. Liczba pokoleń, po których następowała zmiana wartości funkcji optymalizowanej w porównaniu z algorytmem elementarnym była:
 - a. od około -29% większa (wartość gorsza) do około 15% mniejsza (wartość lepsza) dla algorytmu z adaptacją selekcji,
 - b. od około -104% większa (wartość gorsza) do około 11% mniejsza (wartość lepsza) dla algorytmu z adaptacją mutacji,
 - c. od około -54% większa (wartość gorsza) do około 4% mniejsza (wartość lepsza) dla algorytmu z adaptacją selekcji i mutacji.

Wskaźniki procentowe odniesiono do wyników algorytmu elementarnego.

5.6.4. Wnioski z eksperymentów

1. Adaptacja prawdopodobieństwa selekcji powoduje, że do puli rodzicielskiej wybieranych jest więcej osobników o funkcji przystosowania powyżej średniej, dzięki czemu średnia wartość funkcji przystosowania może rosnąć szybciej niż w algorytmie elementarnym. Przejście do kolejnego optimum lokalnego odbywa się po większej liczbie pokoleń, co oznacza, że różnice pomiędzy wartościami kolejno osiągniętych optimów są większe (zwiększenie zdolności eksploatacji przestrzeni rozwiązań).
2. Adaptacja prawdopodobieństwa mutacji powoduje, że osobniki gorzej przystosowane mutowane są częściej niż w algorytmie elementarnym, dzięki czemu mogą one uzyskać wysoką wartość przystosowania, co może powodować szybszy wzrost średniej wartości przystosowania populacji. Przejście do kolejnego optimum lokalnego następuje po

mniej liczbie pokoleń, co może świadczyć o większej zdolności do eksploracji w porównaniu z algorytmem elementarnym.

5.7. Sprawdzenie skuteczności zmodyfikowanego algorytmu genetycznego w rozwiązywaniu skomplikowanych zadań

Eksperyment 14

Celem eksperymentu jest zbadanie, w jaki sposób sterownik rozmyty wpływa na liczbę wywołań funkcji obliczającej przystosowanie osobnika, potrzebnych do opuszczenia optimum lokalnego dla funkcji trudnych z punktu widzenia optymalizacji.

W eksperymencie wybrano jako zadanie testowe problem optymalizacji funkcji dwóch zmiennych Q7. Funkcja ta posiada 6 optimum lokalnych i jedno optimum globalne. Przebieg tej funkcji pokazany jest na rysunku 5.1. Zadanie testowe rozwiązano również za pomocą algorytmów genetycznych wykorzystujących metody selekcji turniejowej i rankingowej (metody selekcji „twardej”).

Populacja początkowa umieszczona została w optimum lokalnym w punkcie (5, 5). W eksperymencie badano:

- liczbę wywołań funkcji obliczającej przystosowanie osobnika, potrzebną aby algorytm osiągnął wartość najlepszego osobnika 2,499,
- liczbę wywołań funkcji obliczającej przystosowanie osobnika, potrzebnych aby średnia wartość funkcji przystosowania w populacji osiągnęła wartość 2,25.

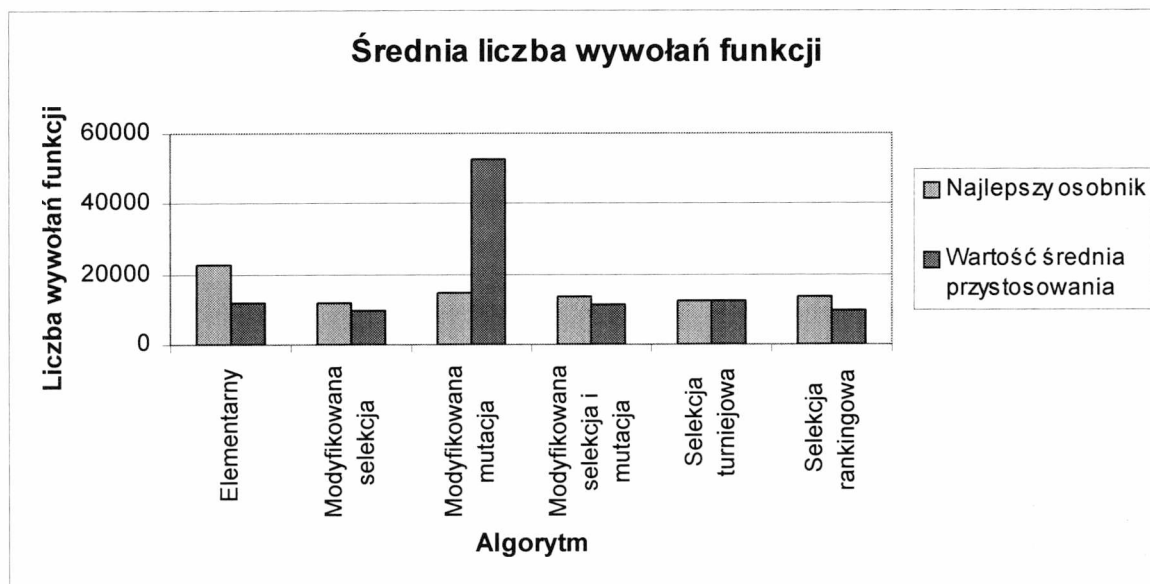
Przyjęto następujące parametry algorytmu genetycznego:

- geny reprezentowane przez liczby rzeczywiste.
- prawdopodobieństwo krzyżowania 0,8,
- prawdopodobieństwo mutacji 0,15,
- liczebność populacji 100,
- każdy z algorytmów uruchomiony został 10 razy.

W tabeli 5.28. przedstawiono średnią liczbą wywołań funkcji obliczającej przystosowanie osobnika, potrzebną do uzyskania przez poszczególne algorytmy założonego warunku zatrzymania oraz odchylenie standardowe tej wartości. Wykres na Rys. 5.33. pokazuje liczbę wywołań funkcji obliczającej przystosowanie osobnika dla poszczególnych algorytmów.

Algorytm	Rodzaj kryterium zatrzymania			
	Najlepszy osobnik	δ	Wartość średnia przystosowania	δ
Elementarny	22580	97,33	11830	30,09
Modyfikowana selekcja	11910	87,91	9600	23,43
Modyfikowana mutacja	14780	110,5	52710	287,31
Modyfikowana selekcja i mutacja	13480	44,94	11230	28,67
Selekcja turniejowa	12490	28,75	12500	35,53
Selekcja rankingowa	13620	51,43	9470	47,96

Tabela 5.28. Liczba wywołań funkcji obliczającej przystosowanie osobnika w zadaniu Q7



Rys. 5.33. Liczba wywołań funkcji obliczającej przystosowanie osobnika w zadaniu Q7

Spostrzeżenia z eksperymentu

- Liczba wywołań funkcji obliczającej przystosowanie osobnika, potrzebna do osiągnięcia przez najlepszego osobnika założonej wartości funkcji optymalizowanej w stosunku do algorytmu elementarnego jest:
 - około 47% mniejsza (wartość lepsza) w algorytmie z modyfikacją prawdopodobieństwa selekcji,
 - około 34% mniejsza (wartość lepsza) w algorytmie z modyfikacją prawdopodobieństwa mutacji,
 - około 44% mniejsza (wartość lepsza) w algorytmie z modyfikacją prawdopodobieństwa selekcji i mutacji.
- Liczba wywołań funkcji obliczającej przystosowanie osobnika, potrzebna do osiągnięcia przez średnią wartość funkcji przystosowania założonej wartości funkcji optymalizowanej w stosunku do algorytmu elementarnego jest:

- a. około 18% mniejsza (wartość lepsza) w algorytmie z modyfikacją prawdopodobieństwa selekcji,
 - b. około -345% większa (wartość gorsza) w algorytmie z modyfikacją prawdopodobieństwa mutacji,
 - c. około 5% mniejsza (wartość lepsza) w algorytmie z modyfikacją prawdopodobieństwa selekcji i mutacji.
3. Liczba wywołań funkcji obliczającej przystosowanie osobnika, potrzebna do osiągnięcia przez średnią wartość funkcji przystosowania założonej wartości funkcji optymalizowanej w stosunku do algorytmu z selekcją turniejową jest:
- a. około 4% mniejsza (wartość lepsza) w algorytmie z modyfikacją prawdopodobieństwa selekcji,
 - b. około -18% większa (wartość gorsza) w algorytmie z modyfikacją prawdopodobieństwa mutacji,
 - c. około -7% większa (wartość gorsza) w algorytmie z modyfikacją prawdopodobieństwa selekcji i mutacji.
4. Liczba wywołań funkcji obliczającej przystosowanie osobnika, potrzebna do osiągnięcia przez średnią wartość funkcji przystosowania założonej wartości funkcji optymalizowanej w stosunku do algorytmu z selekcją turniejową jest:
- a. około 23% mniejsza (wartość lepsza) w algorytmie z modyfikacją prawdopodobieństwa selekcji,
 - b. około -321% większa (wartość gorsza) w algorytmie z modyfikacją prawdopodobieństwa mutacji,
 - c. około 10% mniejsza (wartość lepsza) w algorytmie z modyfikacją prawdopodobieństwa selekcji i mutacji.
5. Liczba wywołań funkcji obliczającej przystosowanie osobnika, potrzebna do osiągnięcia przez średnią wartość funkcji przystosowania założonej wartości funkcji optymalizowanej w stosunku do algorytmu z selekcją rankingową jest:
- a. około 12% mniejsza (wartość lepsza) w algorytmie z modyfikacją prawdopodobieństwa selekcji,
 - b. około -8% większa (wartość gorsza) w algorytmie z modyfikacją prawdopodobieństwa mutacji,
 - c. około 1% mniejsza (wartość lepsza) w algorytmie z modyfikacją prawdopodobieństwa selekcji i mutacji.

6. Liczba wywołań funkcji obliczającej przystosowanie osobnika, potrzebna do osiągnięcia przez średnią wartość funkcji przystosowania założonej wartości funkcji optymalizowanej w stosunku do algorytmu z selekcją rankingową jest:
- a. około -1% większa (wartość gorsza) w algorytmie z modyfikacją prawdopodobieństwa selekcji,
 - b. około -456% większa (wartość gorsza) w algorytmie z modyfikacją prawdopodobieństwa mutacji,
 - c. około -18% większa (wartość gorsza) w algorytmie z modyfikacją prawdopodobieństwa selekcji i mutacji.

Wskaźniki procentowe odniesiono do wartości uzyskanych przez algorytm elementarny.

7. W algorytmie z adaptacją prawdopodobieństwa mutacji, kryterium najlepszego osobnika spełnione jest wcześniej niż w algorytmie elementarnym, natomiast kryterium średniej wartości później. Może to być spowodowane większym zróżnicowaniem populacji, co zwiększa zdolność algorytmu do eksploracji, zapobiegając jednocześnie przedwczesnej zbieżności.

6. Zastosowanie algorytmów genetycznych do optymalizacji sieci komputerowych

Optymalizacja sieci komputerowych posiada duże znaczenie praktyczne. Podstawowym celem optymalizacji sieci jest zapewnienie jej sprawnego i efektywnego funkcjonowania. W tym rozdziale przeanalizujemy możliwość zastosowania algorytmów genetycznych, w których zastosowano sterowniki rozmyte do kierowania ewolucją osobników, do rozwiązywania zadań optymalizacji sieci komputerowych. Rezultaty działania tych algorytmów porównamy z rezultatami uzyskanymi przez elementarne algorytmy genetyczne oraz algorytm wykorzystujący metodę podziału i oszacowań.

6.1. Zadanie minimalizacji średniego opóźnienia pakietu

Niech sieć jest reprezentowana przez graf właściwy i zorientowany $G(N_w, L, C_l)$, gdzie: N_w jest zbiorem węzłów reprezentujących przełączniki, L jest zbiorem łuków reprezentujących łącza, natomiast C_l jest wektorem wag. Wagi łuków mogą reprezentować odległości, opóźnienia, przepustowości lub inną wybraną miarę.

Oznaczmy:

$(x, y) \in L$ - łuk o początku w węźle x i końcu w węźle y , gdzie $x, y \in N$,

$c(x, y) \in C_l$ - przepustowość łuku (x, y) ,

$N_l(x, y)$ - liczba przepustowości dostępnych dla łuku,

$d(x, y) \in ZD$ - koszt budowy lub dzierżawy przepustowości łuku (x, y) ,

$f(x, y)$ - przepływ w łuku (x, y) ,

DB - całkowity koszt budowy sieci.

Oznaczmy przez $R = [r_{ij}]_{m \times n}$ macierz z zewnątrz wprowadzanych natężeń oraz r_{ij} średnie natężenie strumienia danych kierowanych z węzła i do węzła j . Poszczególne składniki r_{ij} możemy ponumerować od 1 do q . Z k -tym składnikiem związana będzie para węzłów s_k i u_k jako źródło i ujście. Przez r_k oznaczmy średnie natężenie k -tego składnika tzn. $r_k = r_{ij}$ gdzie $i = s_k$, $j = u_k$. Niech $f^k(x, y)$ oznacza przepływ składnika k w łuku (x, y) od węzła x do węzła y , natomiast f^k przepływ składnika k w sieci. Przepływ wieloskładnikowy w sieci możemy przedstawić jako:

$$f = \sum_{i=1}^q f^i, \quad (6.1)$$

natomiast przepływ w łuku (x,y)

$$f(x,y) = \sum_{i=1}^q f^i(x,y). \quad (6.2)$$

Przepływem wieloskładnikowym w sieci S realizującym macierz z zewnątrz wprowadzanych natężeń R nazywamy zespół funkcji

$$f^k : L \rightarrow R^+ \cup \{0\} \quad k=1, \dots, q, \quad (6.3)$$

których wartości $f^k(x,y) \quad k=1, \dots, q$ przypisane poszczególnym łukom $(x,y) \in L$ spełniają następujący układ warunków:

$$\sum_{y \in A(x)} f^k(x,y) - \sum_{y \in B(x)} f^k(y,x) = \begin{cases} r_k & \text{dla } x = s_k \\ -r_k & \text{dla } x = u_k \\ 0 & \text{w pozostałych przypadkach} \end{cases} \quad (6.4)$$

dla każdego $x \in N, k=1, \dots, q$.

Zbiór $A(x) = \{y : y \in N \text{ i } (x,y) \in L\}$ jest zbiorem węzłów sieci, do których prowadzą łuki wychodzące z węzła x , zbiór $B(x) = \{y : y \in N \text{ i } (y,x) \in L\}$ jest zbiorem węzłów sieci, do których prowadzą łuki skierowane do węzła x .

Problem minimalizacji średniego opóźnienia pakietu polega na minimalizacji funkcji

$$T = \frac{1}{\gamma} \sum_{(x,y) \in L} \frac{f(x,y)}{c(x,y) - f(x,y)}, \quad (6.5)$$

gdzie: $\gamma = \sum_{i=1}^q r_i$ jest natężeniem sumarycznego strumienia pakietów wprowadzanych do sieci,

przy spełnieniu warunków:

$$\sum_{y \in A(x)} f^k(x,y) - \sum_{y \in B(x)} f^k(y,x) = \begin{cases} r_k & \text{dla } x = s_k \\ -r_k & \text{dla } x = u_k \\ 0 & \text{w pozostałych przypadkach} \end{cases} \quad (6.6)$$

dla każdego $x \in N, k=1, \dots, q$,

$$f^k(x,y) \geq 0 \text{ dla każdego } (x,y) \in L, k=1, \dots, q, \quad (6.7)$$

$$f(x,y) \leq c(x,y) \text{ dla każdego } (x,y) \in L, \quad (6.8)$$

$$\sum_{(x,y) \in L} d(x,y) \leq DB, \quad (6.9)$$

W eksperymencie tym rozwiązywano zadanie testowe przy pomocy algorytmu genetycznego elementarnego i wykorzystującego logikę rozmytą do modyfikowania prawdopodobieństwa selekcji, mutacji i jednocześnie selekcji i mutacji. Zadanie rozwiązano

przy różnych wartościach parametrów algorytmu, a następnie wybrano ten zestaw parametrów, przy którym uzyskano najlepsze wyniki.

W czasie eksperymentu przyjęto następujące parametry algorytmu genetycznego:

- chromosom osobnika podzielony został na:
 - o część reprezentującą trasę pakietów - ciąg q liczb z przedziału $\langle 0, k \rangle$,
 - o część reprezentującą przepustowości łuków – ciąg L liczb z przedziału $\langle 0, N_l \rangle$,
- dla każdego składnika ustalono m tras po których mogą być przesyłane dane,
- funkcja przystosowania dla osobnika obliczana na podstawie zależności (6.5),
- prawdopodobieństwo krzyżowania 0,8,
- prawdopodobieństwo mutacji dla genów reprezentujących trasy 0,15,
- prawdopodobieństwo mutacji dla genów reprezentujących przepustowości 0,15,
- liczebności populacji 25,
- każdy z algorytmów uruchomiony został 10 razy, jako wynik końcowy przyjęto średnią z dziesięciu przebiegów.

Eksperyment polegał na wyznaczeniu liczby wywołań funkcji obliczającej przystosowanie osobnika i czasu działania algorytmu, po którym osiągnięto założoną wartość funkcji optymalizowanej. Podstawowe informacje dotyczące optymalizowanych w eksperymencie sieci zebrano w tabeli 6.1.

	Sieć		
	S1	S2	S3
Liczba węzłów	12	14	28
Liczba łuków	17	28	67
Liczba strumieni danych	9	7	19
Założona wartość dla kryterium stopu	0.00042	0.00035	0,0005
Wartość optymalna ²	0.00039	0.00030	-

Tabela 6.1. Charakterystyka optymalizowanych sieci

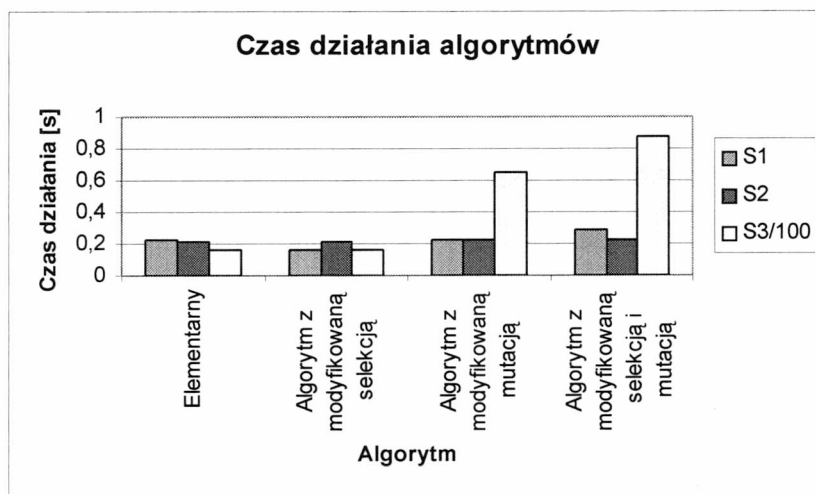
W tabeli 6.2. przedstawione zostały, uśrednione dla 10 różnych przebiegów, wartości czasu działania i liczby wywołań funkcji obliczającej przystosowanie osobnika, potrzebnych do uzyskania założonej wartości funkcji optymalizowanej w poszczególnych algorytmach. Algorytm wykorzystujący metodę podziału i oszacowań nie rozwiązał zadania S3.

² Obliczona za pomocą algorytmu wykorzystującego metodę podziału i oszacowań

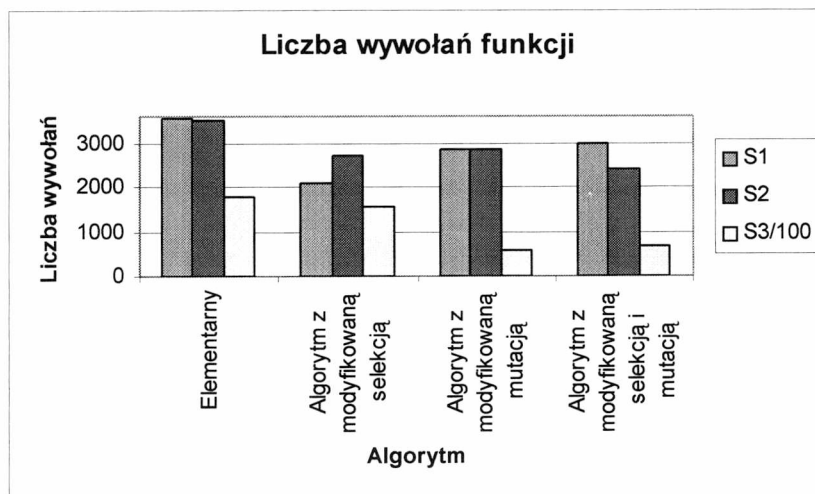
Algorytm	Czas działania [s]			Liczba pokoleń		
	S1	S2	S3	S1	S2	S3
Elementarny	0,22	0,21	16,46	3550	3500	178575
Algorytm z modyfikowaną selekcją	0,16	0,21	16,73	2075	2725	154350
Algorytm z modyfikowaną mutacją	0,22	0,23	6,55	2825	2825	57475
Algorytm z modyfikowaną selekcją i mutacją	0,29	0,23	8,69	3000	2400	66550
Podziału i oszacowań	20,7	278	-	-	-	-

Tabela 6.2. Czas i liczba wywołań funkcji potrzebne do uzyskania założonej wartości funkcji optymalizowanej

Wykres na Rys. 6.1. pokazuje czasy działania algorytmów a wykres na rys 6.2. pokazuje średnią liczbę wywołań funkcji obliczającej przystosowanie osobnika, potrzebną do uzyskania założonej wartości funkcji optymalizowanej (dla zwiększenia czytelności wykresu czas działania algorytmu i liczbę wywołań funkcji dla zadania S3 podzielono przez 100).



Rys. 6.1. Czasy działania algorytmów potrzebne do osiągnięcia założonej wartości funkcji optymalizowanej



Rys. 6.2. Średnia liczba wywołań funkcji obliczającej przystosowanie, potrzebna do uzyskania założonej wartości funkcji optymalizowanej

Wnioski i spostrzeżenia

1. Czas działania algorytmów zmodyfikowanych różnił się od czasu działania algorytmu elementarnego (wskaźniki procentowe odniesiono do wyników algorytmu elementarnego)
 - a. od około -1% (wartość gorsza) do około 27% (wartość lepsza) dla algorytmu z adaptacją selekcji,
 - b. od około -9% (wartość gorsza) do około 60% (wartość lepsza) dla algorytmu z adaptacją mutacji,
 - c. około -31% (wartość gorsza) do około 47% (wartość lepsza) dla algorytmu z adaptacją selekcji i mutacji.
2. Liczba wywołań funkcji obliczającej przystosowanie osobnika, potrzebna do uzyskania założonej wartości funkcji optymalizowanej różniła się od liczby wywołań funkcji obliczającej przystosowanie osobnika dla algorytmu elementarnego (wskaźniki procentowe odniesiono do wyników algorytmu elementarnego)
 - a. od około 13% (wartość lepsza) do około 41% (wartość lepsza) dla algorytmu z adaptacją selekcji,
 - b. od około 19% (wartość lepsza) do około 68% (wartość lepsza) dla algorytmu z adaptacją mutacji,
 - c. około 31% (wartość lepsza) do około 63% (wartość lepsza) dla algorytmu z adaptacją selekcji i mutacji.
3. Algorytmy genetyczne zakończyły działanie w krótszym czasie niż algorytm wykorzystujący metodę podziału i oszacowań. Jednak w przyjętym modelu, algorytm genetyczny nie sprawdza wszystkich możliwych tras, po których pakiety mogą być przesyłane, a tylko te, które zostały zdefiniowane przez administratora sieci, co powoduje zawężenie obszaru przeszukiwania.
4. Zaproponowana metoda może być wykorzystana w czasie projektowania nowych sieci komputerowych lub do optymalizacji ruchu pakietów danych w już istniejących sieciach.

7. Wnioski i podsumowanie pracy

W pracy sformułowano podstawowy jej cel: zbadanie, w jakim stopniu wykorzystanie wiedzy rozmytej do adaptacji parametrów może poprawić działanie algorytmu genetycznego. Z celu podstawowego wynikają cele szczegółowe:

- opracowanie własnej, nowej koncepcji sterowania parametrami algorytmu genetycznego przy pomocy sterownika rozmytego,
- wykazanie, że adaptacja parametrów algorytmu genetycznego wpływa na relację pomiędzy poziomem eksploracji i eksploatacji przyspieszając uzyskanie satysfakcjonującego wyniku,
- wykazanie, że zmodyfikowane algorytmy genetyczne charakteryzują się większą niż algorytm elementarny zdolnością do opuszczania optimum lokalnego.

Aby osiągnąć założone w pracy cele wykonano:

- studia literaturowe – pozwoliły mi na zapoznanie się z aktualnym stanem wiedzy na temat algorytmów genetycznych, logiki rozmytej oraz sposobami ich połączenia, wykorzystywanymi we wcześniejszych badaniach,
- wybór parametrów i opracowanie własnej, nowej koncepcji ich adaptacji – spośród wielu parametrów algorytmu genetycznego do optymalizacji wybrałem prawdopodobieństwo selekcji i prawdopodobieństwo mutacji oraz zaproponowałem własną metodę ich adaptacji poprzez współczynnik modyfikacji prawdopodobieństwa, obliczany za pomocą sterownika rozmytego,
- opracowanie koncepcji sterownika rozmytego – wybrałem model sterownika Mamdaniego ze względu na jego prostotę. Dla sterownika wybrałem wielkości wejściowe i wyjściowe, oraz ich funkcje przynależności. Na podstawie wiedzy o ewolucji ustaliłem zbiór reguł dla sterownika oraz metodę wnioskowania i defuzyfikacji.
- wykazanie teoretyczne prawdziwości twierdzenia o schematach w zmodyfikowanych algorytmach genetycznych oraz wykazanie zbieżności tych algorytmów – przeprowadziłem dowody prawdziwości twierdzenia o schematach w zmodyfikowanym algorytmie genetycznym oraz twierdzenia o punkcie stałym w zmodyfikowanym algorytmie genetycznym z odwzorowaniem zwężającym.
- teoretyczne oszacowanie nakładu obliczeniowego potrzebnego do wykonania adaptacji parametrów algorytmu – w sposób teoretyczny wyprowadziłem wzór pozwalający

wyznaczyć górne oszacowanie nakładu obliczeniowego adaptacji prawdopodobieństwa selekcji.

- dokonałem implementacji algorytmów w języku C++. Zaprezentowałem szczegółowy opis oraz wykonałem serię eksperymentów zakładając różne klasy problemów testowych: problem optymalizacji funkcji, zero-jedynkowy problem plecakowy oraz problem komiwojażera.
- doświadczalne sprawdzenie liczby osobników o wartości funkcji przystosowania lepszej, równej i gorszej od średniej w pokoleniu, wybieranych do puli rodzicielskiej oraz liczby mutacji zachodzących w tych osobnikach i porównanie ich z wynikami uzyskanymi przez algorytm elementarny. Eksperymenty te przeprowadziłem na przykładzie optymalizacji funkcji i problemu komiwojażera.
- doświadczalne wyznaczenie efektywności „on-line” i „off-line” algorytmów zmodyfikowanych i porównanie ich z wynikami uzyskanymi przez algorytm elementarny. Eksperymenty te przeprowadziłem na przykładzie optymalizacji funkcji i problemu komiwojażera.
- testowanie zmodyfikowanych algorytmów w rozwiązywaniu standardowych problemów optymalizacyjnych – przeprowadziłem badania wartości funkcji optymalizowanej i liczby wywołań funkcji obliczającej przystosowanie osobnika. Jako zadania testowe wykorzystałem problem optymalizacji funkcji testowych ciągłych, problem komiwojażera (optymalizacja funkcji nieciągłych) i zero-jedynkowy problem plecakowy (optymalizacja funkcji nieciągłych z ograniczeniami).
- porównanie wartości funkcji optymalizowanej uzyskanej przez zmodyfikowane algorytmy genetyczne, algorytm genetyczny elementarny oraz inne metody optymalizacji wzorowane na obserwacji natury – do porównania wybrałem algorytmy mrówkowe, symulowanego wyżarzania i przeszukiwania tabu.
- porównanie wartości funkcji optymalizowanej, uzyskanej w poszczególnych algorytmach, po założonej liczbie pokoleń, przy różnej liczbie osobników w populacji.
- porównanie liczby pokoleń potrzebnych do uzyskania przez algorytmy założonej wartości funkcji optymalizowanej, przy różnej liczbie osobników w populacji.
- zbadanie zdolności algorytmów zmodyfikowanych do wyszukiwania rozwiązania optymalnego dla binarnego i rzeczywistego kodowania osobników.
- zbadanie zdolności algorytmów do wychodzenia z optimum lokalnego dla funkcji testowych o różnym stopniu trudności.

- opracowanie własnej metody optymalizacji średniego opóźnienia pakietów w sieci komputerowej o ograniczonych kosztach funkcjonowania, wykorzystujące zmodyfikowane algorytmy genetyczne i porównanie ich wyników z dotychczas stosowaną metodą podziału i oszacowań.

Badania przeprowadzone w pracy pozwoliły na wyciągnięcie wniosku potwierdzającego prawdziwość postawionej w rozdziale 1 tezy, dla wybranych typów zadań.

Algorytm genetyczny wykorzystujący logikę rozmytą do kierowania ewolucją populacji osobników, w porównaniu z elementarnym algorytmem genetycznym, na ogół wykazuje większą szybkość w osiąganiu wartości optymalnej lub pozwala na osiągnięcie porównywalnych wyników, przy mniejszej liczbie wywołań funkcji obliczającej przystosowanie osobnika.

Wyniki przeprowadzonych eksperymentów wskazują na przewagę algorytmów, w których wykorzystano logikę rozmytą do kierowania ewolucją nad algorytmami elementarnymi dla wybranych typów zadań. Możemy w związku z tym przypuszczać, że dla innych zadań również wystąpi poprawa efektywności algorytmów, co dowodzi słuszności postawionych w pracy tez. W czasie eksperymentów nie stwierdzono wzrostu tendencji algorytmów, w których wykorzystano logikę rozmytą do przedwczesnej zbieżności spowodowanej dominacją superosobnika. Na uwagę zasługuje również fakt, że sterownik rozmyty wykorzystywany do wszystkich eksperymentów posiadał identyczną budowę, co dowodzi niezależności metody od wiedzy specyficznej dla danego zadania. Poprawa efektywności algorytmów występowała w różnym stopniu w zależności od wielu czynników:

* dotyczących algorytmu genetycznego:

- rodzaju rozważanego problemu (typu zadania optymalizacyjnego),
- parametrów algorytmu genetycznego (prawdopodobieństwa selekcji i mutacji),
- wielkości populacji i liczby pokoleń,

* dotyczących bloku logiki rozmytej:

- bazy reguł rozmytych,
- kształtu i położenia funkcji przynależności.

Na podstawie przeprowadzonych badań sformułować możemy następujące wnioski:

1. W pracy sformułowano i udowodniono twierdzenie o schematach w zmodyfikowanych algorytmach genetycznych, w których wykorzystano sterownik rozmyty do kierowania

ewolucją osobników oraz wykazano, że zmodyfikowane algorytmy z odwzorowaniem zwięzającym są zbieżne.

2. Wykorzystanie bloków logiki rozmytej, oceniających poszczególnych osobników populacji, do sterowania parametrami algorytmu genetycznego jest możliwe. Przeprowadzone eksperymenty z zastosowaniem algorytmów genetycznych, w których wykorzystano sterowniki rozmyte do adaptacji prawdopodobieństwa selekcji i mutacji na podstawie oceny poszczególnych osobników populacji, wskazują na zmniejszenie liczby wywołań funkcji obliczającej przystosowanie osobnika, potrzebnych do uzyskania założonej wartości funkcji przystosowania najlepszego osobnika oraz w większości przypadków nieznacznie wyższe wartości funkcji przystosowania najlepszego osobnika. Pozytywny wpływ adaptacji parametrów występował niezależnie od przyjętej reprezentacji chromosomów (kodowanie rzeczywiste i binarne) i typu zdania optymalizacyjnego.
3. Wartości funkcji przystosowania najlepszego osobnika uzyskiwane przez algorytmy, w których wykorzystano logikę rozmytą są nieznacznie lepsze niż uzyskiwane przez algorytmy elementarne (w niektórych przypadkach wartości funkcji przystosowania uzyskiwane przez algorytm elementarny mogą być lepsze). Poprawa efektywności algorytmów genetycznych na skutek zastosowania bloków logiki rozmytej do sterowania parametrami algorytmu daje różne efekty w zależności od rodzaju rozważanego problemu, przyjętych parametrów algorytmu oraz rodzaju parametru algorytmu poddawanego adaptacji. Spośród badanych problemów największą poprawę wartości funkcji przystosowania najlepszego osobnika uzyskano dla problemu plecakowego (wartości funkcji przystosowania uzyskane przez algorytm, w którym wykorzystano logikę rozmytą do ok. 4% lepsze niż algorytmu elementarnego).
4. Zastosowanie logiki rozmytej do kierowania ewolucją osobników nie gwarantuje uzyskania przez algorytm lepszych wartości funkcji przystosowania, lecz powoduje uzyskanie wartości porównywalnych z algorytmem elementarnym, przy znacznie mniejszej liczbie wywołań funkcji obliczającej przystosowanie osobnika. Różnica między wartościami funkcji przystosowania najlepszego osobnika uzyskanymi przez algorytm elementarny i algorytm, w którym wykorzystano logikę rozmytą, przy dużej liczbie pokoleń jest niewielka. Jednak, liczba wywołań funkcji obliczającej przystosowanie osobnika, potrzebna do osiągnięcia założonej wartości funkcji przystosowania, w algorytmie, w którym wykorzystano logikę rozmytą była do ok. 64% mniejsza niż w algorytmie elementarnym (dla przypadku optymalizacji funkcji). Zmniejszenie liczby

pokoleń, po których uzyskujemy satysfakcjonujący wynik, związane jest jednak z koniecznością przeprowadzenia dodatkowych obliczeń, mających na celu adaptację parametrów algorytmu oraz ocenę poszczególnych osobników i populacji.

5. Adaptacja parametrów algorytmu wpływa na relację między poziomem eksploracji i eksploatacji. Osobniki o wysokiej wartości funkcji przystosowania, w pokoleniach o szybko rosnącej średniej funkcji przystosowania częściej są wybierane do reprodukcji (zwiększenie poziomu eksploatacji), natomiast osobniki o niskiej wartości funkcji przystosowania, w populacjach o malejącej średniej funkcji przystosowania charakteryzują się zwiększonym prawdopodobieństwem mutacji (zwiększenie poziomu eksploracji). Zmiany relacji między poziomem eksploracji i eksploatacji zmniejszają ryzyko przedwczesnej zbieżności algorytmu, spowodowanego dominacją supersobnika i w konsekwencji powodują zbieżność algorytmu do optimum globalnego.
6. Adaptacja parametrów algorytmu przynosi lepsze efekty (skrócenie czasu działania, zmniejszenie liczby pokoleń lub liczby wywołań funkcji obliczającej przystosowanie, potrzebnych do uzyskania zadowalającego wyniku) w problemach, w których do wyznaczenia funkcji oceny osobników wymagane są większe nakłady obliczeniowe.
7. Spośród zaproponowanych metod adaptacji parametrów algorytmu najlepsze efekty (skrócenie czasu działania, zmniejszenie liczby pokoleń lub liczby wywołań funkcji obliczającej przystosowanie, potrzebnych do uzyskania zadowalającego wyniku) uzyskano w algorytmach z adaptacją prawdopodobieństwa selekcji.

Przeprowadzone badania wskazują na możliwości wykorzystania sterowników rozmytych do kierowania ewolucją osobników w algorytmach genetycznych, jednak czas potrzebny do oceny każdego z osobników we wszystkich populacjach jest znaczący. Dla zmniejszenia nakładu obliczeniowego można poddawać ocenie osobniki tylko z wybranych, ustalanych według określonego algorytmu pokoleń. Przedmiotem dalszych badań może być zachowanie algorytmów dla innych typów zadań jak np.: optymalizacja wielokryterialna, optymalizacja funkcji wielomodalnych itp.

Za najważniejsze własne osiągnięcia autor pracy uważa:

1. sformułowanie i udowodnienie twierdzenia o schematach dla zmodyfikowanych algorytmów genetycznych,
2. udowodnienie zbieżności zmodyfikowanego algorytmu genetycznego z odwzorowaniem zwężającym,
3. oszacowanie nakładu obliczeniowego potrzebnego do adaptacji prawdopodobieństwa selekcji,

4. opracowanie własnej koncepcji adaptacji parametrów algorytmu genetycznego za pomocą współczynnika adaptacji prawdopodobieństwa selekcji i współczynnika adaptacji prawdopodobieństwa mutacji, wyznaczanych przez bloki logiki rozmytej dla każdego osobnika w populacji,
5. opracowanie koncepcji sterownika rozmytego (dobór wielkości wejściowych i wyjściowych oraz ich funkcji przynależności, bazy reguł rozmytych i metod defuzyfikacji) wykorzystywanego do adaptacji parametrów algorytmu genetycznego,
6. wykonanie implementacji algorytmów w języku C++ oraz przeprowadzenie badań efektów działania algorytmów dla wybranych typów zadań,
7. porównanie wartości funkcji przystosowania najlepszego osobnika i liczby pokoleń lub liczby wywołań funkcji obliczającej przystosowanie, potrzebnych do osiągnięcia założonej wartości funkcji przystosowania dla algorytmów elementarnych i algorytmów, w których wykorzystano logikę rozmytą w różnych typach zadań optymalizacyjnych,
8. analizę wyników uzyskanych w czasie eksperymentów oraz sformułowanie wniosków.

8. Literatura

- [Abe91] Abel D., *Fuzzy Control – eine Einführung ins Unscharfe*. Automatisierungstechnik 1991, vol. 39, No. 12, pp. 433-438, 1991.
- [Ala95] Alander J.T., *An indexed bibliography of genetic programming*. Report Series no 94-1-GP, Department of Information Technology and Industrial Management, University of Vaasa, Finland, 1995. <ftp://ftp.uwasa.fi/cs/report94-1/README>
- [Alt92] von Altrock C., Krause B., Zimmermann H. J., *Advanced fuzzy logic control of model car in extreme situations*. Fuzzy Sets and Systems 1992, vol.48, No.1, pp.41-52, 1991.
- [Alt95] Altenberg L., *The Schema Theorem and Price's Theorem*. In Foundations of Genetic Algorithms 3, D. Whitley (Ed.). San Mateo, CA: Morgan Kaufmann 1995.
- [And96] Andre D., Teller A., *A study in program response and the negative effects of introns in genetic programming*. In John R. Koza, David E. Goldberg, David B. Fogel, and Rick L. Riolo, editors, Genetic Programming 1996: Proceedings of the First Annual Conference, pp. 12-20, Stanford University, CA, USA, 28-31 July 1996.
- [Ang92] Angeline P.J. Pollack J.B., *Evolutionary Induction of subroutines*. 14th Annual Cognitive Science Conference, pp.236-241, 1992.
- [Ang93] Angeline P.J., Pollack J.B., *Competitive environments evolve better solutions to complex problems*. Fifth International Conference on Genetic Algorithms. pp.264-270, 1993.
- [Ang95] Angeline P.J., *Adaptive and Self-Adaptive Evolutionary Computations*, In Computational Intelligence: A Dynamic Systems Perspective, M. Palaniswami, Y Attikiouzel, R. Marks, D. Fogel and T. Fukuda (eds.), Piscataway, NJ: IEEE Press, pp.152-163, 1995.
- [Ang96] Angeline P.J., *Two Self-Adaptive Crossover Operations for Genetic Programming*, In Advances in Genetic Programming: Volume 2, P. Angeline and K. Kinnear (eds.), MIT Press. In Press. 1996
- [Ara01] Arabas J., *Wykłady z algorytmów ewolucyjnych*, Wydawnictwa Naukowo-Techniczne, Warszawa, 2001.
- [Are99] Arendt D., Szczepaniak P. S., *Algorytm genetyczny w wytyczaniu tras datagramów IP*, Zeszyty Naukowe Politechniki Śląskiej, Seria Informatyka z. 36, Gliwice 1999.
- [Ari90] Arita S., Tsutsui T., *Fuzzy logic control of blood pressure through inhalational anesthesia*. Proc. Of the Int'l Conf. On Fuzzy Logic and Neural Networks, II Zuka, Japan, pp.545-547, 1990.
- [Ari91] Arita S., *Development of an ultrasonic cancer diagnosis system using fuzzy theory*. Japanese Journal of Fuzzy Theory and Systems 1991, vol.3, No.3, pp.215-230, 1991.
- [Arn94] Arnone S., Dell'Orto M., Tettamanzi A., *Toward a fuzzy government of genetic populations*. In Proc. of the 6th IEEE Conference on Tools with Artificial Intelligence, pp.585-591, Los Alamitos, CA: IEEE Computer Society Press, 1994.
- [Ass74] Assilian S., *Artificial Intelligence in the control of real dynamical systems*, (Praca doktorska), London University, 1974.
- [Ata95] Atanassov K.T., *Remark on Intuitionistic Fuzzy Logic and Intuitionistic Logic*. Mathware and Soft Computing 2(1995), pp.151-156, 1995.

- [Bab99] Babuska R., *Data-driven fuzzy modeling: transparency and complexity issues*. In Proceedings European Symposium on Intelligent Techniques ESIT'99, AB-01, Crete, Greece, June 1999.
- [Bäc91a] Bäck T., Hoffmeister F., *Global Optimization by Means of Evolutionary Algorithms*. In Antamoshkin Random Search as a Method for Adaptation and Optimization of Complex Systems, Divnogorsk, USSR, Krasnoyarsk, Space Technology University, pp.17-21, March 1991.
- [Bäc91b] Bäck T., Hoffmeister F., Schwefel H-P., *A survey of evolution strategies*. In Lashon B. Belew, Richard K.; Booker, editor, Proceedings of the 4th International Conference on Genetic Algorithms, pp. 2-9, San Diego, CA, July 1991. Morgan Kaufmann.
- [Bäc92a] Bäck T., *The Interaction of Mutation Rate, Selection, and Self-Adaptation Within a Genetic Algorithm*, R. Männner and B. Manderick, editors: Parallel Problem Solving from Nature, 2, pp. 85-94, Elsevier, Amsterdam, 1992.
- [Bäc92b] Bäck T., *Self-Adaptation in Genetic Algorithms*, F.J. Varela and P. Bourguine, editors: Proceedings of the 1st European Conference on Artificial Life, pp.263-271, The MIT Press, Cambridge MA, 1992.
- [Bäc92c] Bäck T., Schwefel H-P., *Evolutionary Algorithms: Some Very Old Strategies for Optimization and Adaptation*. In Perret-Gallix New Computing Techniques in Physics Research II. Proceedings 2nd Internatinal Workshop on Software Engineering, Artificial Intelligence and Expert Systems for High Energy and Nuclear Physics, La Londe-Les-Maures, France, January 13-18, pp.247-254, World Scientific, Singapore, 1992.
- [Bäc94] Bäck T., *Selective pressure in evolutionary algorithms: A characterization of selection mechanisms*. In Proceedings of the First IEEE Conference on Evolutionary Computation, Volume 1 (pp. 57--62). Piscataway, NJ: IEEE Service Center, 1994.
- [Bäc95] Bäck T., Kursawe F., *Evolutionary Algorithms for Fuzzy Logic: A Brief Overview*. In: Bouchon-Meunier, B., Yager, R.R., Zadeh, L.A. (eds.): Fuzzy Logic and Soft Computing, Vol. IV. World Scientific, pp.3-10, Singapore 1995.
- [Bäc96] Bäck T., Schutz M., *Intelligent Mutation Rate Control in Canonical Genetic Algorithms*, In W. Ras and M. Michalewicz, editors: Foundation of Intelligent Systems 9th International Symposium, ISMIS '96, pp.158-167, Springer, Berlin, 1996.
- [Bag67] Bagley J. D., *The behavior of adaptive systems with employ genetic and correlation algorithms*. (Doctoral dissertation, University of Michigan). *Dissertation Abstracts International*, 28(12), 5106B. (University Microfilms No.68-7556), 1967.
- [Bat83] Batycki T., Kasprzak T., *Wybrane Algorytmy optymalizacji przepływów w sieciach teleinformatycznych*. Wydawnictwo Politechniki Wrocławskiej, Wrocław 1983.
- [Bea93a] Beasley D., Bull D.R., Martin R.R., *An Overview of Genetic Algorithms: Part 1: Fundamentals*, University Computing vol. 15(4), pp.58-69, 1993.
- [Bea93b] Beasley D., Bull D.R., Martin R.R., *An Overview of Genetic Algorithms: Part 2: Research Topics*, University Computing, vol. 15(4), pp.170-181, 1993.
- [Ber98] van den Berg J., Ettes D., *Representation and Learning Capabilities of Additive Fuzzy Systems*, Proceedings of the 1998 IEEE International Conference on Intelligent Engineering Systems (INES'98), eds. P. Kopacek and I.J. Rudas, pp.121-126, September 1998, Vienna, Austria.

- [Bie92] Bien Z., Hwang D.H., Lee J.H., Ryu H.K., *An automatic start-up and shut-down control of a drum type boiler using fuzzy logic*. Proc. 2nd Int'l Conf. On Fuzzy Logic and Neural Networks, II Zuka, Japan, pp.465-468, 1992.
- [Bli95] Blickle T., Thiele L., *A comparison of selection schemes used in genetic algorithms*. TIK-Report 11, TIK Institut für Technische Informatik und Kommunikationsnetze, Computer Engineering and Networks Laboratory, ETH, Swiss Federal Institute of Technology, Gloriastrasse 35, 8092 Zurich, Switzerland, December 1995.
- [Bos72] Bosworth J., Foo N., Zeigler B. P., *Comparison of genetic algorithms with conjugate gradient methods* (CR-2093). Washington, DC: National Aeronautics and Space Administration, 1972.
- [Bri81] Brindle A., *Genetic algorithms for function optimization*. Unpublished doctoral dissertation, University of Alberta, Edmonton, 1981.
- [Can97] Cantu-Paz E., *A Survey of Parallel Genetic Algorithms*, IlliGAL Report 97003, 1997.
- [Cav70] Cavicchio D. J., *Adaptive search using simulated evolution*. Unpublished doctoral dissertation, University of Michigan, Ann Arbor, 1970.
- [Che98] Chellapilla K., *Combining Mutation Operators in Evolutionary Programming*. IEEE Trans. on Evolutionary Computation, vol.2, no.3, pp. 91-97, Sept, 1998.
- [Chen99] Chen Y., Kawaji S., *Evolutionary Control of Discrete-Time Nonlinear System using PIPE Algorithm*. Proc. of IEEE Int. Conference on SMC'99, pp.1078-1083, Tokyo.
- [Chm97] Chmiel W., Wala K., *Algorytmy ewolucyjne z dynamiczną zmianą parametrów*, Materiały II Krajowej Konferencji Algorytmy ewolucyjne i optymalizacja globalna, Ryto 1997, str. 33 - 40.
- [Ciu00] Ciura M., *Rozwiązywanie problemu dostawy za pomocą algorytmu genetycznego*, Studia Informatica, Volume 21, Number 3(41), Gliwice 2000.
- [Chr95] Chromiec J., Strzemieczna E., *Sztuczna inteligencja. Metody konstrukcji i analizy systemów eksperckich*, Akademicka Oficyna Wydawnicza EXIT, Warszawa 1995.
- [Chu98] Chudzik K., Janiak A., *Algorytm tabu search dla wybranych problemów szeregowania zadań na pojedynczej maszynie z przebrojeniami* w XI Krajowa Konferencja Automatyzacji Dyskretnych Procesów Przemysłowych, Zakopane, Referaty sekcji 1. Podstawy teoretyczne. Gliwice: Wydaw. PŚI. s.97-109, Zeszyty Naukowe-Politechnika Śląska nr 1389, Automatyka, z.123, 1998.
- [Cob91] Cobb H.G., Grefenstette J.J., *Learning the persistence of actions in reactive control rules*, Proc. Eighth Intl. Machine Learning Workshop, San Mateo: Morgan Kaufmann, pp.293-297, 1991.
- [Con] Concorde. *A Code for Solving Traveling Salesman Problem*. <http://www.math.princeton.edu/tsp/concorde.html>.
- [Cor] Cordon O., Herrera F., Lozano M., *Combination Fuzzy Logic-Genetic Algorithms Bibliography*. ftp://decsai.ugr.es/pub/arai/tech_rep/ga-fl/rev-bib.ps.Z
- [Cor96] Cordon O., Herrera F., Herrera-Viedma E., Lozano M., *Genetic Algorithms and Fuzzy Logic in Control Processes*. Archives of Control Sciences. Vol. 5, pp.135-168, 1996.
- [Cor97a] Cordon O., Herrera F., Peregrin A., *Applicability of the Fuzzy Operators in the Design of Fuzzy Logic Controllers*. Fuzzy Sets and Systems. Vol.86: 1, pp.15-41, 1997.

- [Cor97b] Cordon O., Herrera F., Lozano M., *On the Combination of Fuzzy Logic and Evolutionary Computation: A Short Review and Bibliography*. Fuzzy Evolutionary Computation, W. Pedrycz (Ed.), Kluwer Academic, pp.57-77, 1997.
- [Cor01] Cordon O., Herrera F., *Hybridizing Genetic Algorithms with Sharing Scheme and Evolution Strategies for Designing Approximate Fuzzy Rule-Based Systems*. Fuzzy Sets and Systems 118:2, pp. 235-255, 2001.
- [Cyt96] Cytowski J., *Algorytmy genetyczne - Podstawy i zastosowania*, Akademicka Oficyna Wydawnicza PLJ, Warszawa, 1996.
- [Cyt99] Cytowski J., *Metody i algorytmy sztucznej inteligencji w cyfrowym przetwarzaniu sygnałów*, Akademicka Oficyna Wydawnicza EXIT, Warszawa 1999.
- [Czo85] Czogała E., Pedrycz W., *Elementy i Metody Teorii Zbiorów Rozmytych*, Warszawa: PWN, 1985.
- [Dar95] Darwen P.J., Yao X., *A Dilemma for Fitness Sharing with a Scaling Function*. Second IEEE International Conference on Evolutionary Computation. 1995.
- [Deb91] Deb K., Goldberg, D.E., *mGA in C: A Messy Genetic Algorithm in C*. Illinois Genetic Algorithms Laboratory (IlligAL), report no. 91008, 1991.
- [Dol] Dolan A., *Ariel Dolan's Web Site: Artificial Life and Other Experiments*, www.aridolan.com.
- [Dvo97] Dvorak A., *Computational Properties of Fuzzy Logic Deduction*. In: Reusch, B.(Ed.): Computational Intelligence. Theory and Applications. Proceedings of 5th Fuzzy Days Dortmund, Springer-Verlag 1997, pp.189-196.
- [Dri96] Driankov D., Hellendoorn H., Reinfrank M., *Wprowadzenie do sterowania rozmytego*, Warszawa WNT 1996.
- [Dub96a] Dubois D., Prade H., *A fuzzy set approach to case-based decision*. Institut de Recherche en Informatique de Toulouse (I.R.I.T)-C.N.R.S. Toulouse Cedex-France, 1996.
- [Dub96b] Dubois D., Prade H., *What Are Fuzzy Rules and How to Use Them*, Fuzzy Sets and Systems, Vol. 84, pp. 169-185. 1996.
- [Eib98] Eiben A.E., *Multi-parent recombination*. In T. Bäck, D. Fogel, and M. Michalewicz, editors, Handbook of Evolutionary Computation, pp.25-33, IOP Publishing Ltd. and Oxford University Press, 1998.
- [Eib99] Eiben A.E., Hinterding R., Michalewicz, Z., *Parameter Control in Evolutionary Algorithms*, IEEE Transactions on Evolutionary Computation, Vol.3, No.2, 1999, pp.124-141.
- [Ekl92] Eklund P., Klawonn F., Nauck D., *Distributing Errors in Neural Fuzzy Control*. Proc. 2nd International Conference on Fuzzy Logic and Neural Networks (Iizuka'92). Fuzzy Logic Systems Institute, Iizuka 1992, pp.1139-1142.
- [Fed92] Fedrizzi M., Fullér R., *Stability in possibilistic linear programming problems with continuous fuzzy number parameters*, Fuzzy Sets and Systems 47, pp.187-191, 1992.
- [Fic00a] Ficici S.G., Melnik O., Pollack J.B., *A Game-Theoretic Investigation of Selection Methods Used in Evolutionary Algorithms*. Proceedings of the 2000 Congress on Evolutionary Computation, A. Zalzala, et al, (eds.), IEEE Press, 2000.

- [Fic00b] Ficici S.G., Pollack J.B., *Effects of Finite Populations on Evolutionary Stable Strategies*. Proceedings of the 2000 Genetic and Evolutionary Computation Conference, L. Darrell Whitley (ed.), Morgan-Kaufmann, 2000.
- [For85] Forrest S., *Documentation for PRISONERS DILEMMA and NORMS programs that use the genetic algorithm*. Unpublished manuscript, University of Michigan, Ann Arbor, 1985.
- [For93] Forrest S., Mitchell M., *Relative building block fitness and the building block hypothesis*. In L.Darrell Whitley (Ed.), *Foundation of Genetic Algorithms-2*, pp.109-126, San Mateo: Morgan Kaufman, 1993.
- [Fra72] Franz D. R., *Non-linearities in genetic adaptive search*. (Doctoral dissertation, University of Michigan). *Dissertation Abstract International*, 33(11), 5240B-5241B. (University Microfilms No. 73-11,116), 1972.
- [Ful89] Fullér R., *On stability in fuzzy linear programming problems*, *Fuzzy Sets and Systems* 30, pp.339-344. 1989.
- [Ful92] Fullér R., *A law of large numbers for fuzzy numbers*, *Fuzzy Sets and Systems*, 45. pp.299-303. 1992.
- [Ful93] Fullér R., Triesch E., *A note on law of large numbers for fuzzy variables*, *Fuzzy Sets and Systems*, 55 pp.235-236. 1993.
- [Ful] Fullér R. *Fuzzy Logic Resources* <http://www.abo.fi/~rfuller/fuzs.html>.
- [Fun00] Funes P., Pollack J.B., (2000). *Measuring Progress in Coevolutionary Competition*. From Animals to Animats 6: Proceedings of the Sixth International Conference on Simulation of Adaptive Behavior. Meyer, J. et. al (eds.) MIT Press. pp.450-459, 2000.
- [Gol91] Goldberg D.E., *Real-coded Genetic Algorithm, Virtual Alphabets, and Blocking*, *Complex Systems* 5(2), pp. 139-169, (IlliGAL Report 90001), 1991.
- [Gol97a] Goldberg D.E., Wang L., *Adaptive niching via coevolutionary sharing*, In Quagliarella et al. (Eds.), *Genetic Algorithms in Engineering and Computer Science*, pp.21-38, John Wiley and Sons Ltd. 1997.
- [Gol97b] Goldberg D.E., Zakrzewski K., Sutton B., Gradient R., Chang C., Gallego P., Miller B., Cantu-Paz E., *Genetic algorithms: a bibliography*. IlliGAL Report No. 97011.
- [Gol98] Goldberg D.E., *Algorytmy genetyczne i ich zastosowania*, Wydawnictwa Naukowo Techniczne, Warszawa 1998.
- [Gol01] Goldberg D.E., Sastry K., *A Practical Schema Theorem for Genetic Algorithms Design and Tuning*, IlliGAL Report 2001017, January 2001.
- [Goo94] Goodman E., Tetelbaum A., Kureichik V., *A Genetic Algorithm Approach to Compaction, Bin Packing, and Nesting Problems*, Tech. report #940702. Case center for computer-aided engineering and manufacturing. Michigan State University, 1994.
- [Gre86] Greffenstette J.J., *Optimization of control parameters for genetic algorithms*, *IEEE Transactions on Systems, Man and Cybernetics* 16, pp.122-128, 1986.
- [Gre92] Grefenstette J.J., *Genetic algorithms for changing environments*, *Proc. Parallel Problem Solving from Nature-2*, R. Maenner and B. Manderick (Eds.), North-Holland, 137-144, 1992.

- [Gre93] Grefenstette J.J., *Genetic algorithms and machine learning*. Invited talk, Proc. Sixth Annual ACM Conf. Computational Learning Theory (COLT 93), ACM, 1993.
- [Gwi98] Gwiazda T.D., *Algorytmy genetyczne. Zastosowania w finansach*. Wydawnictwo Wyższej Szkoły Przedsiębiorczości i Zarządzania im. L. Koźmińskiego, Warszawa, 1998.
- [Háj95] Hájek P., Godo L., Esteva F., *Fuzzy logic and Probability*. (P. Besnard & S. Hanks, Eds.) Uncertainty in Artificial Intelligence Conference. Morgan Kaufmann. San Francisco, pp. 237-244, 1995.
- [Har97] Harik G., Cantu-Paz E., Goldberg D.E., Miller B., *The gambler's ruin problem, genetic algorithms, and the sizing of populations*. In Bäck, T., editor, Proceedings of the Fourth International Conference on Evolutionary Computation, pp.7-12, IEEE Press, Piscataway, New Jersey, 1997.
- [Her94a] Herrera F., Lozano M., Verdegay J.L., *The Use of Fuzzy Connectives to Design Real-Coded Genetic Algorithms*. Mathware and Soft Computing 3(1994), pp.239-251, 1994.
- [Her94b] Herrera F., Herrera-Viedma E., Lozano A., Verdegay J.L., *Fuzzy tools to improve genetic algorithms*. Second European Congress on Fuzzy and Intelligent Technologies. EUFIT'94. Aachen (Germany), 1994, Vol. III, pp. 1532-1539.
- [Her95a] Herrera F., Lozano M., Verdegay J.L., *Generating Fuzzy Rules from Examples using Genetic Algorithms*. B. Bouchon-Meunier, R.R. Yager, L.A. Zadeh (Eds.), Fuzzy Logic and Soft Computing, pp.11-20. World Scientific, 1995.
- [Her95b] Herrera F., Verdegay J.L., *On Group Decision Making under Linguistic Preferences and Fuzzy Linguistic Quantifiers*. Fuzzy Logic and Soft Computing, B. Bouchon, R. Yager, L. Zadeh (Eds.), pp. 173-180, Word Scientific, 1995.
- [Her96a] Herrera F., Lozano M., *Adaptation of Genetic Algorithm Parameters based on Fuzzy Logic Controllers*. Genetic Algorithms and Soft Computing, F. Herrera, J. L. Verdegay (Eds.), pp.95-125, Physica-Verlag, 1996.
- [Her96b] Herrera F., Lozano M., Verdegay J.L., *Dynamic and Heuristic Fuzzy Connectives-Based Crossover Operators for Controlling the Diversity and Convergence of Real Coded Genetic Algorithms*. Int. Journal of Intelligent Systems Vol. 11, pp.1013-1041, 1996.
- [Her97a] Herrera F., Lozano M., Verdegay J.L., *Fuzzy connectives based crossover operators to model genetic algorithms population diversity*. Fuzzy Sets and Systems, 92(1), 1997, pp.21-30.
- [Her97b] Herrera F., Magdalena L., *Genetic Fuzzy Systems*. Tatra Mountains Mathematical Publications Vol. 13, 1997, pp.93-121. R. Mesiar, B. Riecan (Eds.) Fuzzy Structures. Current Trends. Lecture Notes of the Tutorial: Genetic Fuzzy Systems. Seventh IFSA World COngress (IFSA97), Prague, June 1997.
- [Her98a] Herrera F., Lozano M., *Fuzzy Genetic Algorithms: Issues and Models*. Technical Report DECSAI-98116, Dept. of Computer Science and A.I., University of Granada, June 1998.
- [Her98b] Herrera F., Lozano M., Verdegay J.L., *Tackling Real-Coded Genetic Algorithms: Operators and tools for the Behaviour Analysis*. Artificial Intelligence Review 12, pp.265-319, 1998.
- [Her99] Herrera F., Lozano M., Moraga C., *Hierarchical Distributed Genetic Algorithms*. International Journal of Intelligent Systems 14:9, pp.1099-1121, 1999.

- [Her00a] Herrera F., Lozano M., *Gradual Distributed Real-Coded Genetic Algorithms*. IEEE Transactions on Evolutionary Computation 4:1, pp.43-63, 2000.
- [Her00b] Herrera F., Lozano M., *Two-Loop Real-Coded Genetic Algorithms with Adaptive Control of Mutation Step Sizes*. Applied Intelligence 13:3 pp. 187-204, 2000.
- [Her01] Herrera F., Lozano M., *Adaptive Genetic Operators Based on Coevolution with Fuzzy Behaviours*. IEEE Transactions on Evolutionary Computation 2001.
- [Hin96] Hinterding R., Michalewicz Z., Eiben A.E., *Adaptation in Evolutionary Computation: A Survey*. In Proceedings of the Forth IEEE International Conference on Evolutionary Computation, Indianapolis, pp.13-16 April, 1996.
- [Hol68] Holland J.H., *Hierarchical descriptions of universal spaces and adaptive systems* (Technical Report ORA Projects 01252 and 08226), Ann Arbor: University of Michigan, Department of Computer and Communication Sciences, 1968.
- [Hol75] Holland J.H., *Adaptation in Natural and Artificial Systems*, Ann Arbor: University of Michigan Press, 1975.
- [Hol71] Hollstien R. B., *Artificial genetic adaptation in computer control systems*. (Doctoral dissertation, University of Michigan). *Dissertation Abstract International*, 32(3), 1510B. (University Microfilms No. 71-23,773), 1971.
- [Jan94] Janiak A., Kobylański P., *Algorytm genetyczny rozwiązujący przepływowy problem obsługi z zasobami* w IX Krajowa Konferencja Automatyzacji Dyskretnych Procesów Przemysłowych, PŚl. Kozubnik k. Porąbki, 14-17.09.1994. Referaty sekcji I. Gliwice: Wydaw. PŚl. s. 99-109, Zeszyty Naukowe - Politechnika Śląska nr 1250, Automatyka z. 114 I06/94/I-076, 1994
- [Jan96a] Janiak A., Chudzik K., *Algorytm genetyczny dla jednomaszynowego problemu szeregowania zadań z zasobami*, w X Krajowa Konferencja Automatyzacji Dyskretnych Procesów Przemysłowych. Zakopane, 18-21.09.1996. Referaty sekcji 1. Podstawy teoretyczne. Gliwice: Wydaw. PŚl. S.111-122, Zeszyty Naukowe-Politechnika Śląska nr1337, Automatyka z.117, 1996,
- [Jan96b] Janiak A., Chudzik K., Portmann M.C., *Porównanie czterech algorytmów genetycznych na bazie sekwencyjnego problemu przepływowego*, w X Krajowa Konferencja Automatyzacji Dyskretnych Procesów Przemysłowych, Zakopane, 18-21.09.1996. Referaty sekcji 1. Podstawy teoretyczne. Gliwice: Wydaw. PŚl. 1996 s. 123-135, Zeszyty Naukowe - Politechnika Śląska nr 1337, Automatyka z. 117, 1996.
- [Jan99] Janiak A., *Wybrane problemy i algorytmy szeregowania zadań i rozdziału zasobów*, Akademicka Oficyna Wydawnicza PLJ, Warszawa, 1999.
- [Jan00] Janiak A., Marek M., *Porównanie technik aproksymacyjnych: tabu search, ewolucyjnej i sieci neuronowej na przykładzie problemu szeregowania zadań z zasobami na pojedynczej maszynie* w XII Krajowa Konferencja Automatyzacji Procesów Dyskretnych. Zakopane, 13-16.09.2000. Gliwice: Wydaw. PŚl. 2000 s.157-166, Zeszyty Naukowe - Politechnika Śląska nr 1474, Automatyka, ISSN 0434-0760; z. 129, 2000.
- [Joh] John B., *Fuzzy Logic Sources of Information* <http://www.cms.dmu.ac.uk/~rij/fuzzy.html>.
- [Jon75] De Jong K.A., *An analysis of the behavior of a class of genetic adaptive systems*. Doctoral dissertation, University of Michigan, 1975.

- [Jon89] De Jong K.A., Spears W.M., *Using Genetic Algorithms to Solve NP-Complete Problems*, Proceedings of the Third International Conference on Genetic Algorithms, pp.124-132, 1989.
- [Jon91] De Jong K.A., Spears W.M., *Learning Concept Classification Rules Using Genetic Algorithms*, In Proceedings of the Int'l Joint Conference on Artificial Intelligence, pp.651-656, 1991.
- [Jon92] De Jong K.A., Spears W.M., *A Formal Analysis of the Role of Multi-Point Crossover in Genetic Algorithms*. In Annals of Mathematics and Artificial Intelligence, Volume 5, #1, pp.1-26, 1992.
- [Jon93] De Jong K.A., Spears W.M., *On the state of evolutionary computation*, Proceedings of the Fifth International Conference on Genetic Algorithms, pp.618-623, 1993.
- [Jon97] De Jong K.A., Potter M.A., Spears W.M., *Using Problem Generators to Explore the Effects of Epistasis*. In Proceedings of the Seventh International Conference on Genetic Algorithms, pp.338-345. Morgan Kaufmann, 1997.
- [Kac86] Kacprzyk J., *Zbiory rozmyte w analizie systemowej*, Warszawa: PWN, 1986.
- [Kar96] Kargupta H., *The gene expression messy genetic algorithm*. Proceedings of the 1996 IEEE International Conference on Evolutionary Computation. Nagoya University, Japan., 1996, 631-636.
- [Kark93] Karkowski I., *Circuit Delay Optimization as a Multiple Choice Linear Knapsack Program*. Proceedings of the EDAC 1993, pp.419-423.
- [Kas89] Kasprzak A., *Optymalizacja przepływów i przepustowości kanałów w sieciach teleinformatycznych*, Archiwum Automatyki i Telemechaniki, Tom XXXIV, pp. 233-252,
- [Kas99] Kasprzak A., *Rozległe sieci komputerowe z komutacją pakietów*. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 1999.
- [Kel] Keller P.E., *Activity in Fuzzy Logic World Wide* <http://www.emsl.pnl.gov:2080/proj/neuron/fuzzy>.
- [Kin94] Kinzel J., Klawonn F., Kruse R., *Modifications of Genetic Algorithms for Designing and Optimizing Fuzzy Controllers*. Proc. IEEE Conference on Evolutionary Computation, IEEE, Orlando 1994, pp.28-33.
- [Kla95a] Klawonn F., Kruse R., *Clustering Methods in Fuzzy Control*. In: W. Gaul, D. Pfeifer (eds.): From Data to Knowledge: Theoretical and Practical Aspects of Classification, Data Analysis and Knowledge Organization. Springer-Verlag, Berlin 1995, pp.195-202.
- [Kla95b] Klawonn F., Nauck D., Kruse R., *Generating Rules from Data By Fuzzy and Neuro-Fuzzy Methods*. In Proc. Fuzzy-Neuro-Systeme'95, pp.223-230, Darmstadt 1995.
- [Kla95c] Klawonn F., Kruse R., *Derivation of Fuzzy Classification Rules from Multidimensional Data*. In: G.E. Lasker, X. Liu (eds.): Advances in Intelligent Data Analysis. The International Institute for Advanced Studies in Systems Research and Cybernetics, Windsor, Ontario 1995, pp.90-94.
- [Kla97a] Klawonn F., Kruse R., *Constructing a Fuzzy Controller from Data*. Fuzzy Sets and Systems 85 1997, pp.177-193.
- [Kla97b] Klawonn F., Klement E.P., *Mathematical Analysis of Fuzzy Classifiers*. In: X. Liu, P. Cohen, M. Berthold (eds.): Advances in Intelligent Data Analysis. Springer, Berlin 1997, pp.359-370.
- [Kla97c] Klawonn F., Keller A., *Fuzzy Clustering and Fuzzy Rules*. Proc. 7th International Fuzzy Systems Association World Congress (IFSA'97) Vol. I, Academia, Prague 1997, pp.193-198.
- [Kla97d] Klawonn F., *Fuzzy Clustering with Evolutionary Algorithms*. Proc. 7th International Fuzzy Systems Association World Congress (IFSA'97) Vol. II, Academia, Prague 1997, pp.312-317.

- [Klu92] Kluska J., *Sterowanie z logiką rozmytą*, Rzeszów: Zeszyty Naukowe Politechniki Rzeszowskiej Nr.104, Elektrotechnika z.12, 1992.
- [Kos96] Kosko B., Mitaim S., *What is the Best Shape for a Fuzzy Set in Function Approximation?*. Proceedings of the 5th IEEE International Conference on Fuzzy Systems (FUZZ-96), pp.1237-1243, September 1996.
- [Koz89] Koza J.R., *Hierarchical genetic algorithms operating on populations of computer programs*. In Proceedings of the 11th International Joint Conference on Artificial Intelligence. San Mateo, CA: Morgan Kaufmann. Volume I, 1989, pp.768-774.
- [Koz90] Koza, J.R., *Genetically breeding populations of computer programs to solve problems in artificial intelligence*. In Proceedings of the Second International Conference on Tools for AI. Herndon, Virginia, November 6-9, Los Alamitos, CA:IEEE Computer Society Press, pp.819-827, 1990.
- [Koz95a] Koza J.R., *Two ways of discovering the size and shape of a computer program to solve a problem*. In Eshelman L. (editor). Genetic Algorithms: Proceedings of the Fifth International Conference. San Francisco, CA: Morgan Kaufmann, pp.287-294, 1995.
- [Koz95b] Koza J.R., *Survey of genetic algorithms and genetic programming*. Proceedings of 1995 WESCON Conference, Piscataway, NJ: IEEE. pp.589 – 594, 1995.
- [Koz97] Koza, J.R., *Future work and practical applications of genetic programming*. In Baeck, T., Fogel, D. B., and Michalewicz, Z. (editors) Handbook of Evolutionary Computation. Bristol, UK: Institute of Physics Publishing and New York: Oxford University Press, pp. H1.1:1-6 1997.
- [Kru95] Kruse R., Nauck D., *Learning Methods for Fuzzy Systems*, In: Proc. Fuzzy-Neuro-Systeme'95, pp.7-22, Darmstadt, 1995.
- [Kwa98] Kwaśnicka H., *Algorytmy genetyczne w uczeniu się maszyn*, VI konferencja "Pozyskiwanie wiedzy z baz danych", Rzeszów 1998, Materiały konferencyjne, str. 148-169, Akademia Ekonomiczna, Wrocław.
- [Kwa99a] Kwaśnicka H., *K-MODEL - an Evolutionary Algorithm with New Schema of Representation*, II Symposium on Artificial Intelligence, Adaptive Systems, March 22-26, 1999, International Conference Center of Havana, Cuba.
- [Kwa99b] Kwaśnicka H., *Algorytmy ewolucyjne w grach menedżerskich*, VII Konferencja Pozyskiwanie wiedzy z baz danych, 14-16 maja 1999, Szklarska Poręba. Materiały Konferencyjne, Akademia Ekonomiczna, Wrocław.
- [Kwa99c] Kwaśnicka H., *K-Model: rozmiar populacji a charakter ewolucji*, III Krajowa Konferencja Algorytmy Ewolucyjne i Optymalizacja Globalna KEAiOG'99, 25-28 maja 1999, Potok Złoty.
- [Kwa99d] Kwaśnicka H., Dąbrowski T., Piasecki M., *Zastosowanie algorytmu genetycznego do konstrukcji funkcji oceniającej w grze OTHELLO*, III Krajowa Konferencja Algorytmy Ewolucyjne i Optymalizacja Globalna KEAiOG'99, 25-28 maja 1999, Potok Złoty.
- [Kwa99e] Kwaśnicka H., Molecki B., *Timetabling of the city tram service using a genetic algorithm*, Intelligent Information Systems VIII Proceedings of the Workshop, Ustroń, Poland, June 14-18, 1999.
- [Kwa99f] Kwaśnicka H., *Knowledge Acquisition For Intelligent Control Of Cooperative Evolutionary Algorithms*, MENDEL'99, 4th International Mendel Conference on Genetic Algorithms, Technical

- University of Brno, Faculty of Mechanical Engineering Institute of Automation and Computer Science, Brno, Czech Republic, June 9-12, 1999.
- [Kwa99g] Kwaśnicka H., *Obliczenia ewolucyjne w sztucznej inteligencji*. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław, 1999.
- [Kwa04] Kwaśnicka H., Spirydowicz A., *Uczący się komputer. Programowanie gier logicznych*. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław, 2004.
- [Lan96] Langdon W.B., *A bibliography for genetic programming*. In Peter J. Angeline and K. E. Kinneer, Jr., editors, *Advances in Genetic Programming 2*, chapter B, pages 507-532. MIT Press, Cambridge, MA, USA, 1996. <http://www.cs.bham.ac.uk/~wbl/biblio/>.
- [Lau00] Laumanns M., Zitzler E., Thiele L., *A Unified Model for Multi-Objective Evolutionary Algorithms with Elitism*. Congress on Evolutionary Computation (CEC-2000), July 2000, pp. 46-53.
- [Lee93] Lee M.A., Takagi H., *Dynamic control of genetic algorithms using fuzzy logic techniques*. In S. Forrest (Ed.), *Proc. of the Fifth Int. Conf. On Genetic Algorithms* pp.76-83. San Mateo, Morgan Kaufmann. 1993.
- [Lee94] Lee M.A., Takagi H., *A framework for studying the effects of dynamic crossover, mutation, and population sizing in genetic algorithms*, In Furuhashi (Ed.), *Advances in Fuzzy Logic, Neural Networks and Genetic Algorithms*, pp.111-126, Lecture Notes in Computer Science 1011, Berlin: Springer-Verlag, 1994.
- [Luk] Luke B.T., *Application of Simulated Annealing to the Traveling Salesman Problem*, <http://btluke.netfirms.com/apsatsp.html>.
- [Luk97] Luke S., Spector L., *A comparison of crossover and mutation in genetic programming*. In John R. Koza, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max Garzon, Hitoshi Iba, and Rick L. Riolo, editors, *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pages 240-248, Stanford University, CA, USA, 13-16 July 1997. Morgan Kaufmann.
- [Łac01] Łachwa A., *Rozmyty świat zbiorów, liczb, relacji, faktów, reguł i decyzji*, Akademicka Oficyna Wydawnicza EXIT, Warszawa, 2001.
- [Mah92] Mahfound S.W., *Crowding and preselection revised*. In R. Manner & B. Manderick (Ed.), *Parallel Problem Solving from Nature 2*, pp.27-36, Amsterdam, Elsevier Science Publishers, 1992.
- [Mam77] Mamdani E. H., *Application of fuzzy algorithms for control of simple dynamic plant*. *Proceedings IEEE* 1974, No. 121 (12), pp.1585-1588, 1977.
- [Mar79] Martello S., Toth P., *Algorithm 37: Algorithm for the solution of the 0-1 single knapsack problem*, *Computing* 21, pp.81-86, 1979.
- [Mic96] Michalewicz Z., Schoenauer M., *Evolutionary Algorithms for Constrained Parameter Optimization Problems*, *Evolutionary Computation*, Vol.4, No.1, 1996, pp.1-32.
- [Mic99] Michalewicz Z., *Algorytmy genetyczne + struktury danych = programy ewolucyjne*, Wydawnictwa Naukowo Techniczne, Warszawa 1999.
- [Mit92] Mitchell M., Forrest S., Holland J.H., *The royal road for genetic algorithms: Fitness landscapes and GA performance*. In *Proceedings of the First European Conference on Artificial Life*. Cambridge, MA: MIT Press/Bradford Books, 1992.

- [Mos94] Moscato P., Tinetti F., *Blending Heuristics with a Population-Based Approach: A "Memetic" Algorithm for the Traveling Salesman Problem*, Universidad Nacional de La Plata, La Plata Argentina, 1994.
- [Müh93a] Mühlenbein H., Schlierkamp-Voosen D., *Predictive Models for the Breeder Genetic Algorithm*, Appeared in: *Evolutionary Computation*, 1(1): 25-49, 1993.
- [Müh93b] Mühlenbein H., Schlierkamp-Voosen D., *Optimal Interaction of Mutation and Crossover in the Breeder Genetic Algorithm*, Proc. of Fifth Int. Conf. on Genetic Algorithms (ICGA-93), Morgan Kaufmann, pp.648-658, 1993.
- [Mul97] Mulawka J.J., *Systemy ekspertowe*, Wydawnictwa Naukowo Techniczne, Warszawa 1997.
- [Nau92] Nauck D., Klawonn F., Kruse R., *Fuzzy Sets, Fuzzy Controllers, and Neural Networks*. *Wiss. Zeitschrift der Humboldt-Universität zu Berlin, R. Medizin* 41-4 1992, pp.99-120.
- [Nau94] Nauck D., Kruse R., *Choosing Appropriate Neuro-Fuzzy Models*. In Proc. Second European Congress on Fuzzy and Intelligent Technologies (EUFIT94), pp.552-557, Aachen.
- [Nau96] Nauck D., Klawonn F., *Neuro-Fuzzy Classification Initialized by Fuzzy Clustering*. Proc. Of Fourth European Congress on Intelligent Techniques and Soft Computing (EUFIT'96), Aachen 1996, pp.1551-1555.
- [Obu03] Obuchowicz A., *Evolutionary Algorithms for Global Optimization and Dynamic System Diagnosis*, Lubusky Scientific Society in Zielona Gora, Zielona Gora, 2003.
- [Ope] *OpenTS – Java Tabu Search*, biblioteka klas,
<http://oss.software.ibm.com/developerworks/opensource/coin/OpenTS/index.html>
- [Pic97] Pickert L., Klawonn F., Wingender E., *Fuzzy Cluster Analysis for Identification of Gene Regulating Regions*. Proc. 7th International Fuzzy Systems Association World Congress (IFSA'97) Vol. IV, Academia, Prague 1997, pp.56-61.
- [Pie99] Piegat A., *Modelowanie i sterowanie rozmyte*, Akademicka Oficyna Wydawnicza EXIT, Warszawa 1999.
- [Pil97] Piliński M., *Metodologia automatycznego uczenia sterowników rozmytych z wykorzystaniem sieci neuronowych*, Praca doktorska, Politechnika Częstochowska w Częstochowie, Częstochowa 1997.
- [Pol96] Poli R., Logan B., *The evolutionary computation cookbook: Recipes for designing new algorithms*. In Proceedings of the Second Online Workshop on Evolutionary Computation, Nagoya, pp.33-36, March 1996.
- [Pol97] Poli R., Langdon W.B., *A New Schema Theory for Genetic Programming with One-Point Crossover and Point Mutation*, Proceedings of the Second International Conference on Genetic Programming, GP'97, pp. 278-285, Stanford, July 1997. Morgan Kaufmann.
- [Pol98] Poli R., Langdon W.B., *Schema Theory for Genetic Programming with One-point Crossover and Point Mutation*, *Evolutionary Computation Journal*, 6(3), pp. 231-252, 1998.
- [Pol00a] Poli R., *Hyperschema theory for GP with one-point crossover, building blocks, and some new results in GA theory*. In Riccardo Poli, Wolfgang Banzhaf, and et al., editors, *Genetic Programming*, Proceedings of EuroGP 2000. Springer-Verlag, 15-16 April 2000.

- [Pol00b] Poli R., *Exact Schema Theorem and Effective Fitness for GP with One-Point Crossover*. In D. Whitley et al., editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, Las Vegas, July 2000, Morgan Kaufmann.
- [Pol00c] Poli R., *Recursive Conditional Schema Theorem, Convergence and Population Sizing in Genetic Algorithms*. In *Proceedings of the Foundation of Genetic Algorithm (FOGA) Workshop*, Charlottesville, Virginia, USA, July 2000.
- [Pot94] Potter M.A., De Jong K.A., *A Cooperative Coevolutionary Approach to Function Optimization*. In *Proceedings of the Third Conference on Parallel Problem Solving from Nature*, pp.249-257, Springer-Verlag, 1994.
- [Pot95] Potter M.A., De Jong K.A., Grefenstette J.J., *A Coevolutionary Approach to Learning Sequential Decision Rules*. In *Proceedings of the Sixth International Conference on Genetic Algorithms*, pp.366-372. Morgan Kaufmann, 1995.
- [Pyt02] Pytel K., Kluka G., *Wykorzystanie logiki rozmytej do wspomaganie ewolucji osobników w algorytmach genetycznych*, w *Studia z Automatyki i Informatyki*, Wydawnictwo Poznańskiego Towarzystwa Przyjaciół Nauk, Poznań, 2002.
- [Pyt04a] Pytel K., Kluka G., Szymonik A., *Fuzzy Methods of Driving Genetic Algorithms*, in *Proceedings of the Fourth International Workshop on Robot Motion and Control*, Poznan, 2004.
- [Pyt04b] Pytel K., Kluka G., Kisilewicz J., *Zastosowanie algorytmów genetycznych do optymalizacji sieci komputerowych*. w *Materiały konferencyjne Multimedialne i sieciowe systemy informacyjne*. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław, 2004.
- [Ram93] Ramsey C.L., Grefenstette J.J., *Case-based initialization of genetic algorithms*. *Genetic Algorithms: Proc. Fifth Intl. Conf. (ICGA93)*, San Mateo: Morgan Kaufmann, pp.84-91, 1993.
- [Reb97] Rebajn B., *Zastosowanie algorytmów genetycznych do problemu układania rozkładu zajęć w szkołach wyższych*, *Zeszyty Naukowe Politechniki Śląskiej, Seria Informatyka z.33*, Gliwice 1997.
- [Rey94] Reynolds R.G., *An Introduction to Cultural Algorithms*, *Proceedings of Evolutionary Programming(EP-94)*, San Diego, CA, Feb. 1994.
- [Rey97] Reynolds R.G., Chung C-J., *Regulating the amount of information used for self-adaptation in cultural algorithms*, In T. Back (Ed.) *Proc. of the Seventh Int. Conf. No Genetic Algorithms*, pp.401-408, San Francisco: Morgan Kaufman Publishers, 1997.
- [Roa95] Roache E.A., Hickok K.A., Loje K.F., Hunt M.W., Grefenstette J.J., (1995). *Genetic algorithms for expert system validation*. *Proc. 1995 Western MultiConf.. Society for Computer Simulation*, Las Vegas, NV, 1995.
- [Ros67] Rosenberg R. S., *Simulation of genetic populations with biochemical properties*, (Doctoral dissertation, University of Michigan). *Dissertation Abstracts International*, 28(7), 2731B. (University Microfilms No. 67-17,836), 1967.
- [Rut97a] Rutkowska D., *Inteligentne systemy obliczeniowe. Algorytmy genetyczne i sieci neuronowe w systemach rozmytych*, Akademicka Oficyna Wydawnicza PLJ, Warszawa 1997.
- [Rut97b] Rutkowska D., Piliński M., Rutkowski L., *Sieci neuronowe, algorytmy genetyczne i systemy rozmyte*, PWN, Warszawa, Lipiec 1997.

- [Scha02] Schaefer R., *Podstawy genetycznej optymalizacji globalnej*, Wydawnictwo Uniwersytetu Jagiellońskiego, Kraków, 2002.
- [Scho97a] Schoenauer M., Michalewicz Z., *Evolutionary Computation: An Introduction*. In Control and Cybernetics Special Issue on Evolutionary Computation 26(3) pp. 307-338, 1997.
- [Scho97b] Schoenauer M., Michalewicz Z., *Boundary Operators for Constrained Parameter Optimization Problems*, Proceedings of the Seventh International Conference on Genetic Algorithms, Morgan Kaufmann, pp.322-329, 1997.
- [Schr92] Schraudolph N.N., Belew R.K., *Dynamic parameter encoding for genetic algorithms*. Machine Learning, 9(1), 1992, pp.9-21.
- [Schu92] Schultz A.C., Grefenstette J.J., De Jong, K. A., *Adaptive testing of controllers for autonomous vehicles*, Proc. Symposium on Autonomous Underwater Vehicle Technology, Washington: IEEE, pp.158-164, 1992.
- [Sco95] Scott S.D., Seth S., Samal A., *A Hardware-Based Genetic Algorithm*, In Proc. of the ACM/SIGDA Third Int. Symposium on Field-Programmable Gate Arrays, pp.53-59, 1995.
- [Seb97a] Sebag M., Schoenauer M., *A Society of Hill-Climbers*, Proceedings of 1997 IEEE International Conference on Evolutionary Computation, pp.319-324, Indianapolis, April 1997.
- [Seb97b] Sebag M., Schoenauer M., Ravisé C., *Inductive Learning of Mutation Step-size in Evolutionary Parameter Optimization*, Proceedings of the Sixth Annual Conference on Evolutionary Programming, pp.247-261, Indianapolis, April 1997.
- [Shi99] Shi Y., Eberhart R., Chen Y., *Implementation of evolutionary fuzzy systems*, IEEE Transaction on Fuzzy Systems 7 (2), pp.109-119, 1999.
- [Sla] Slany W., Nakatsuyama M., *Fuzzy-Mail Archives*
<http://www.dbai.tuwien.ac.at/marchives/fuzzy-mail/index.html>.
- [Smi97] Smith J.E., Fogarty T.C., *Operator and parameter adaptation in genetic algorithms*. Soft Computing 1(2), pp.81-87, 1987.
- [Spe91] Spears W.M., Jong K.A., *On the Virtues of Parameterized Uniform Crossover*. In Proceedings of the Int'l Conference on Genetic Algorithms, pp.230-236, 1991.
- [Spe92] Spears W.M. *Crossover or Mutation?* In Proceedings of Foundations of Genetic Algorithms Workshop, pp.221-237, 1992.
- [Spe93] Spears W.M., De Jong K.A., Back T., Fogel D.B., de Garis H., *An Overview of Evolutionary Computation*, Proceedings of the European Conference on Machine Learning, v667, pp.442-459, 1993.
- [Spe95] Spears W.M., *Adapting Crossover in Evolutionary Algorithms*. To appear in Proceedings of the Fourth Annual Conference on Evolutionary Programming, San Diego, CA, 1995.
- [Spe98] Spears W.M., *The Role of Mutation and Recombination in Evolutionary Algorithms*. Ph.D. Dissertation, George Mason University, Fairfax, Virginia, 1998.
- [Spe99] Spears W.M. (1999). *An Overview of Multidimensional Visualization Techniques*. Proceedings of the Genetic and Evolutionary Computation Conference, 1999.
- [Sta] Stamer H., *Ant System (AS) for the Traveling Salesman Problem (TSP)*,
http://stinfwww.informatik.uni-leipzig.de/~mai97ixb/index_en.html.

- [Ste97] Stęclik P., *Zastosowanie algorytmów genetycznych przy uczeniu sieci neuronowych i sterowników rozmytych*, Praca dyplomowa, Politechnika Częstochowska w Częstochowie, Częstochowa 1997.
- [Str99] Streifel R.J., Marks II R.J., Reed R., Choi J.J., Healy M., *Dynamic fuzzy control of genetic algorithm parameter coding*, IEEE Trans. On Systems, Man and Cybernetics – Part B: Cybernetics 29(3), pp.426-433, 1999.
- [Sys99] Sysło M.M, Deo N., Kowalik J.S., *Algorytmy optymalizacji dyskretnej*, Wydawnictwo Naukowe PWN, Warszawa, 1999.
- [Tel94] Teller A., *Genetic Programming, Indexed Memory, The Halting Problem and Other Curiosities*, The 7th Annual Florida AI Research Symposium, pp. 270-274, Published by: IEEE Press. 1994.
- [Tel96] Teller A., *Evolving Programmers: The Co-evolution of Intelligent Recombination Operators*, Advances in Genetic Programming II, Edited by P. Angeline and K. Kinnear, MIT Press, 1996.
- [Tet95] Tettamanzi A.G., *Evolutionary algorithms and fuzzy logic: a two-way integration*. In 2nd Joint Conference on Information Sciences, Wrigthsville Beach, pp.464-467, 1995.
- [Tom95] Tomassini M., *Genetic Algorithms: a Survey*, In Annual Reviews of Computational Physics Vol. III, World Scientific, 87-117, 1995.
- [Tri94] Trillas E., Cubillo S., De Soto A.R., *On the Identity of Fuzzy Material Conditionals*. Mathware and Soft Computing 3(1994), pp.309-314, 1994.
- [Tus98] Tuson A.L., Ross P., *Adapting operator settings in genetic algorithms*. Evolutionary Computation, 6 (2), pp.161-184, 1998.
- [Vog96] Voget S., *Multiobjective optimization with genetic algorithms and fuzzy control*, In Proc. of the Fourth European Congress on Intelligent Techniques and Soft Computing, pp.391-394, 1996.
- [Vog98] Voget S., Kolonko M., *Multidimensional optimization with a fuzzy genetic algorithm*, Journal of Heuristic 4(3), pp.221-224, 1998.
- [Voi92] Voigt H.M., *Fuzzy evolutionary algorithms*. Technical Report tr-92-038, International Computer Science Institute (ICSI), Berkeley, 1992.
- [Voi93] Voigt H.M., Born J., Santibanez-Koref I., *A Multivalued Evolutionary Algorithm*. Technical Report tr93-022, International Computer Science Institute (ICSI), Berkeley, 1993.
- [Voi95] Voigt H.M., Muhlenbein H., Cvetkovic D., *Fuzzy recombination for the breeder genetic algorithm*. To appear in Sixth Int. Conf. on Genetic Algorithms Morgan Kaufmann, San Mateo, 1995.
- [Wat98] Watson R.A., Hornby G.S., Pollack J.B., *Modeling Building-Block Interdependency*. Parallel Problem Solving from Nature, proceedings of Fifth International Conference /PPSN V, Springer 1998, pp.97-106, 1998.
- [Wat99] Watson R.A., Pollack J.B., *Incremental Commitment in Genetic Algorithms*. publication, GECCO, Proceedings of 1999 Genetic and Evolutionary Computation Conference (GECCO 99). Banzhaf, Daida, Eiben, Garzon, Honavar, Jakiela, Smith, eds, Morgan Kauffmann, pp.710-717, 1999.
- [Wat00] Watson R.A., Reil T. Pollack J.B., *Mutualism, Parasitism, and Evolutionary Adaptation*. Proceedings of Artificial Life VII, Bedau, M, McCaskill J, Packard N, Rasmussen, S (eds.), 2000.
- [Wei] Weicker K., Weicker N., *Keyword Indexed Bibliography of Evolutionary Computation* <http://www.informatik.uni-stuttgart.de/ifi/fk/evolalg/eabib.html>

- [Wei70] Weinberg R., *Computer simulations of a living cell* (Doctoral dissertation, University of Michigan). *Dissertation Abstract International*, 31(9), 5312B. (University Microfilms No. 71-4766), 1970.
- [Whi93] Whitley D., *A genetic algorithm tutorial*, Tech. Rep. CS-93-103, Department of Computer Science, Colorado State University, Fort Collins, CO 8052, March 1993.
- [Wie01] Wierzchoń S.T., *Sztuczne systemy immunologiczne. Teoria i zastosowania*. Akademicka Oficyna Wydawnicza WXIT, Warszawa, 2001.
- [Xu93] Xu H.Y., Vukovich G., *A fuzzy genetic algorithm with effective search and optimization*, In Proc. of 1993 Int. Joint Conference on Neural Networks, pp.2967-2970, 1993.
- [Xu94] Xu H.Y., Vukovich G., Ichikawa Y., Ishii Y., *Fuzzy evolutionary algorithms and automatic robot trajectory generation*, In Z. Michalewicz, J.D. Schaffer, H.-P. Schwefel, D.B. Fogel & H. Kitano (Eds.), Proc. of the First IEEE Int. Conference on Evolutionary Computation, pp.595-600, Piscataway, Nj: IEEE Press, 1994.
- [Yag95] Yager R.R., Filev D.P., *Podstawy modelowania i sterowania rozmytego*, WNT, 1995.
- [Yen96] Yen J., Lee B., Liao J., *Using a Hybrid Genetic Algorithm and Fuzzy Logic for Metabolic Modeling*, in Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-96), Portland, Oregon, pp 743-749, August 4-8, 1996.
- [Yen99] Yen J., *Fuzzy Logic: A Mordern Perspective*, IEEE Transactions on Knowledge and Data Engineering, Vol 11, No. 1, January 1999.
- [Zad65] Zadeh L.A., *Fuzzy Sets, Information and Control*, vol.8, pp.338-353, 1965.
- [Zad83] Zadeh L.A., *A Fuzzy-Set-Theoretic Approach to the Compositionality of Meaning: Propositions, Dispositions and Canonical Forms*, Univerisity of California at Berkeley, Institute of Cognitive Sciences - Research report, 1983.
- [Zen97] Zeng X., Rabenasolo B., *A fuzzy login based design for adaptive genetic algorithms*, In Proc. of the European Congress on Intelligent Techniques and Soft Computing, pp.660-664, 1997.
- [Zit99] Zitzler E., Deb K., Thiele L., *Comparison of Multiobjective Evolutionary Algorithms on Test Functions of Different Difficulty*. Genetic and Evolutionary Computation Conference (GECCO-99): Bird-of-a-feather Workshop on Multi-criterion Optimization Using Evolutionary Methods, July 1999, pp.121-122.
- [Żak84] Żakowski W., Kołodziej M., *Matematyka*, Wydawnictwa Naukowo-Techniczne, Warszawa 1984.
- [Żur96] Żurada J., Barski M., Jędruch W., *Sztuczne sieci neuronowe*, PWN, Warszawa, 1996.