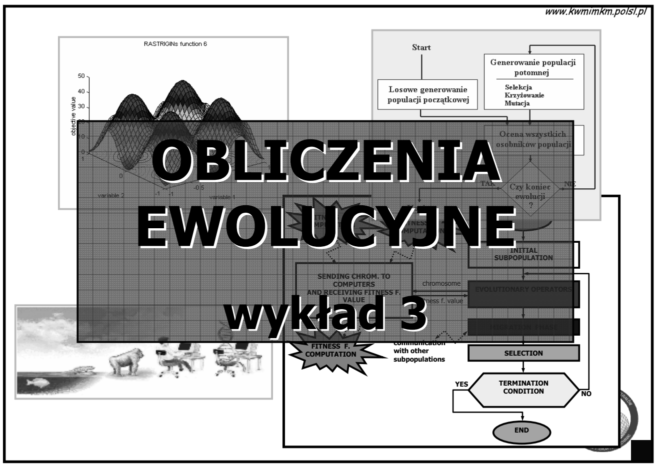


OBLICZENIA EWOLUCYJNE

wykład 3

SCHEMATY - problemy

S_1 = **(111********)**

S_2 = **(*********11)**

Założenie 1:

Dopasowanie ich liniowej kombinacji**:** S_3 = **(111******11)**

jest gorsze, niż: S_4 = **(000******00)**

Założenie 2:

Optymalny osobnik: v_0 = **(11111111111)**

Algorytm genetyczny może mieć trudności ze zbiegnięciem się do v_0 i może się zbiegać do punktów w rodzaju:

v_1 = **(00011111100)**

Zjawisko zawodu – zadanie zwodnicze

EPISTAZA:

- **W biologii** (genetyce) – efekt maskowania
  (obecność genu epistatycznego tłumi efekt innego genu);
- **W AG**: silne uzależnienie między genami w chromosomie
  (określa, na ile wpływ jednego genu na dopasowanie chromo- somu zależy od wartości innych genów).

Sposoby postępowania w przypadku zadań zwodniczych:

- Specyficzny dla danego zadania sposób kodowania zadania *(zakłada się odpowiednią wiedzę o przebiegu funkcji celu);*
- Użycie operatora inwersji;
- Nieporządny algorytm genetyczny**.**

Inwersja

- operator jednoargumentowy;
- wybór 2 punktów w łańcuchu i odwracana kolejność bitów między nimi;
- konieczność pamiętania początkowej pozycji bitów.

np.:

v = **(00011010001)**

v=((**1,0**)(**2,0**)(**3,0**)|(**4,1**)(**5,1**)(**6,0**)(**7,1**)|(**8,0**)(**9,0**)(**10,0**)(**11,1**))

v'=((**1,0**)(**2,0**)(**3,0**)|(**7,1**)(**6,0**)(**5,1**)(**4,1**)|(**8,0**)(**9,0**)(**10,0**)(**11,1**))

Algorytm korzystający z inwersji poszukuje najlepszego położenia bitów do formowania cegiełek.

NIEPORZĄDNY ALGORYTM GENETYCZNY

- Każdy bit chromosomu jest oznaczony;
- Łańcuchy mają zmienną długość i nie jest wymagana pełna liczba genów;
- Możliwa jest:
  nadmiarowość, powtórzenia i niedookreślenie:
  v_1 = ((7,1)(1,0))
  v_2 = ((3,1)(9,0)(3,1)(3,1)(3,1))
  v_3 = ((2,1)(2,0)(4,1)(5,0)(6,0)(7,1)(8,1));
- Stosuje się zmodyfikowane operatory (łączenie i cięcie);
- Operator mutacji jak w AG;
- Selekcja turniejowa;

Łączenie:

Skleja dwa łańcuchy z pewnym prawdopodobieństwem:

v_1 = ((7,1)(1,0)) v_2 = ((3,1)(9,0)(3,1)(3,1)(3,1))

v_4 = ((7,1)(1,0)(3,1)(9,0)(3,1)(3,1)(3,1))

Cięcie:

Rozcina (z pewnym prawdopodobieństwem) łańcuch w losowo wybranym miejscu na dwa osobniki potomne:

v_3 = ((2,1)(2,0)(4,1)(5,0)(6,0)(7,1)(8,1))

v_5 = ((2,1)(2,0)(4,1)) v_6 = ((5,0)(6,0)(7,1)(8,1))

„NAG dają zaskakująco dobre rezultaty dla niektórych funkcji zawodnych”.

ZBIEŻNOŚĆ AG

Przedwczesna zbieżność:

- Przez przedwczesną zbieżność rozumie się utratę przez algorytm optymalizacyjny zdolności przeszukiwania przestrzeni poszukiwań przed osiągnięciem ekstremum globalnego.
- Zjawisku temu nie można zapobiec całkowicie, lecz można je ograniczać (np. stosując AE).

Przyczyny braku zbieżności AG:

- Kodowanie powoduje, iż algorytmy pracują w innej przestrzeni, niż przestrzeń zadania;
- Teoretycznie nieograniczona liczba iteracji w praktyce musi być skończona;
- Teoretycznie nieograniczona liczebność populacji w praktyce musi być skończona.

Strategie zwalczania przedwczesnej zbieżności:

- Zapobieganie kazirodztwu;
- Jednorodne krzyżowanie;
- Wykrywanie jednakowych łańcuchów w populacji.

Większość badań koncentruje się na:

1. Określeniu zakresu i rodzaju błędów związanych z wyborem punktów próbnych (mechanizm próbkowania);
2. Charakterystyce funkcji celu (skalowanie funkcji celu).

Ad. 1. Mechanizm próbkowania

(modyfikacje procedury selekcji).

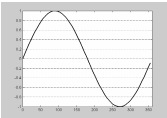
Istotne czynniki wpływające na znalezienie optimum globalnego:

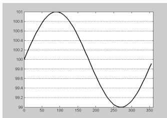
- różnorodność populacji;
- napór selekcyjny.

Celem jest osiągnięcie równowagi.

Selekcja proporcjonalna – nieodporna na dodanie stałej do funkcji celu:

$y=\sin(x)$; | $y^*=\sin(x)+100$





$x_1=80^0$, $y_1=0.9848$ | $x_1=80^0$, $y^*_1=100.9848$

$x_2=50^0$, $y_2=0.766$ | $x_2=50^0$, $y^*_2=100.766$

$y_1/y_2=1.29$ | $y_1/y_2=1.002$

W ekstremalnym przypadku – błądzenie przypadkowe...

Modele selekcji:

- **Model elitarny** – nacisk kładziony na uzyskanie najlepszych osobników (forsowanie najlepszych do następnych pokoleń);
- **Model wartości oczekiwanej** – licznik ustawiany na wartość początkową $[\mathrm{eval}(v)/\mathrm{eval}_{śr}(v)]$ i odpowiednio zmniejszany, gdy osobnik podlega reprodukcji. Gdy licznik osiąga 0 – śmierć osobnika;
- **Elitarny model wartości oczekiwanej** – połączenie powyższych metod;
- **Model ze współczynnikiem zatłoczenia**: nowy chromosom wymienia stary (stary jest wybierany spośród takich, które są podobne do nowego);

Modele selekcji:

- **Metoda stochastycznego próbkowania na podstawie reszty z zamianą** - osobniki potomne otrzymuje się uwzględniając część całkowitą wartości oczekiwanej pojawienia się osobnika w następnym pokoleniu. O pozostałe miejsca w populacji osobniki rywalizują na podstawie części ułamkowej.
- **Stochastyczne próbkowanie uniwersalne** – koło ruletki o równych polach;
- **Selekcja turniejowa** – wybór k osobników (k – rozmiar turnieju, zwykle k=2) i selekcja najlepszego z grupy. Powtarzane pop_size razy.
- **Selekcja rankingowa** – szeregowanie osobników wg wartości przystosowania i selekcja zgodnie z kolejnością (wg tzw. linii rangi) – związana z tzw. superosobnikami.

Superosobniki:

- Niepożądane w początkowej fazie działania **(przedwczesna zbieżność)**;
- Pozytywne pod koniec pracy algorytmu **(zawężenie przestrzeni poszukiwań)**.

Ad. 2. Skalowanie funkcji celu

(regulacja liczby kopii)

Skalowanie liniowe:

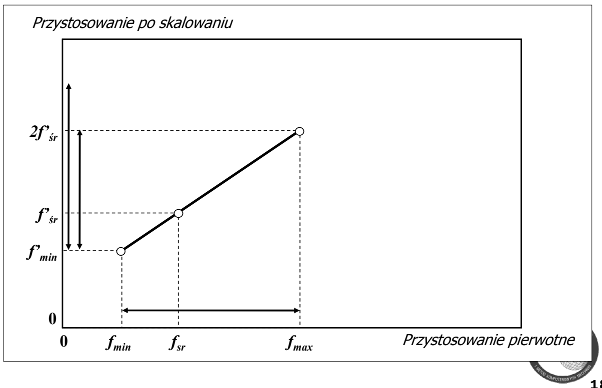
$$\boxed{f' = a f + b}$$

f – przystosowanie pierwotne;
f' – przystosowanie po skalowaniu;
a, b – współczynniki.

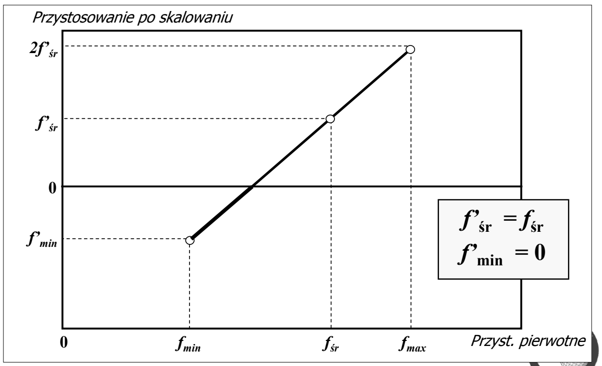
$$f'_{śr} = f_{śr}$$
$$f'_{max} = c \cdot f_{śr}$$

c – współczynnik zwielokrotnienia (typowo $c = 1.2 \div 2$)

Populacja młoda:



Populacja dojrzała:



Inne skalowania:

- Obcinanie na poziomie odchylenia standardowego:

$$f' = f + (f_{śr} - d \cdot \sigma)$$

d – liczba całkowita (zwykle $d = 1 \div 5$);
σ – odchylenie standardowe populacji.

- Skalowanie zgodne z prawem potęgowym:

$$f' = f^k$$

k – parametr bliski 1 (np. $k = 1.005$);

LICZEBNOŚĆ POPULACJI

Istotny parametr AG...

Możliwości regulacji liczebności populacji:

- Metaalgorytmy genetyczne sterujące parametrami innego AG;
- Skorzystanie z „różnych algorytmów ustalania liczebności populacji";
- AG ze zmienną liczebnością populacji.

procedure AGzZLP
begin
  t:=0
  wybierz populację początkową $P(t)$
  oceń $P(t)$
  while (**not** warunek zakończenia) **do**
  begin
    t:=t+1
    podnieś wiek każdego osobnika o 1
    zmień $P(t)$ *(utwórz $P'(t)$)*
    oceń $P'(t)$
    usuń osobniki o *wieku* większym od ich *czasu życia*
  end
end

Populacja pomocnicza $P'(t)$:

$$pop\_size'(t) = \rho \cdot pop\_size(t)$$

ρ – *współczynnik reprodukcji*

procedure AGzZLP
begin
  t:=0
  wybierz populację początkową $P(t)$
  oceń $P(t)$
  while (**not** warunek zakończenia) **do**
  begin
    t:=t+1
    podnieś wiek każdego osobnika o 1
    zmień $P(t)$
    oceń $P'(t)$
    usuń osobniki o *wieku* większym od ich *czasu życia*
  end
end

- Prawdopodobieństwo wyboru do $P'(t)$ niezależne od dopasowania osobnika.
- Krzyżowanie i mutacja – tylko w populacji $P'(t)$;
- Czas życia przypisywany jest jednorazowo każdemu osobnikowi;
- Wiek osobnika rośnie od 0;
- Śmierć osobnika – gdy wiek przekracza czas jego życia.

Liczebność populacji po jednej iteracji:

$$pop\_size(t+1) = pop\_size(t) + pop\_size'(t) - D(t)$$

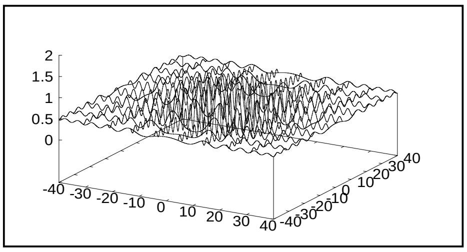
$D(t)$ – liczba osobników, które wyginęły w pokoleniu t.

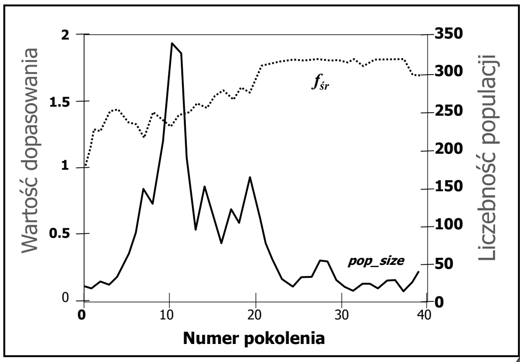
Ustalanie czasu życia osobników:

- Wzmacnianie osobników z wartością dopasowania powyżej średniej i osłabianie gorszych od średniej;
- Dostrajanie liczebności populacji do bieżącego etapu poszukiwań (szczególnie ochrona przed wykładniczym wzrostem populacji).

Przykład:

$$f(x) = 0.5 + \left[\frac{\left(\sin^2 \sqrt{x^2 + y^2} - 0.5 \right)}{1 + 0.001(x^2 + y^2)} \right]^2 \quad , -100 \le x, y \le 100$$





AG i dylemat więźnia

Dylemat więźnia (zdrada czy współpraca):

Gracz 1	Gracz 2	P1	P2	Uwagi
Z	Z	1	1	Kara za wspólną zdradę
Z	W	5	0	Zapłata przez naiwnego i zachęta do zdrady
W	Z	0	5	
W	W	3	3	Nagroda za współpracę

Użycie AG do nauki strategii postępowania (Axelrod '87)

A. Reprezentacja zadania:

- Korzystamy z informacji o 3 poprzednich ruchach
- 64 bity (4x4x4 możliwości);
- 3 ruchy wstępne (przesłanki): +6 bitów
- **Łańcuch 70-bitowy**

Łańcuch:

- całkowicie reprezentuje strategię;
- reprezentuje gracza.

B. Algorytm

1. Wybór początkowej populacji.
2. Sprawdzenie każdego gracza.

   Każdy gracz rozgrywa gry z pozostałymi graczami. Wynik – średnia ze wszystkich rozgrywanych gier.

3. Wybór graczy do rozmnożenia.

   Gracz ze średnim wynikiem dostaje jednego partnera, gracz z wynikiem o 1 odchylenie standardowe lepszym dwóch partnerów, gracz z wynikiem o 1 odch. st. gorszym nie dostaje żadnego partnera.

4. Kojarzenie wygrywających graczy.

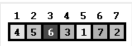
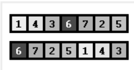
   Krzyżowanie i mutacja wśród wybranych.

C. Efekty

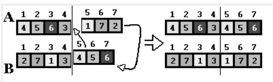
1. Przedłużaj współpracę po 3 kolejnych współpracach
   W po **(WW)(WW)(WW)**
2. Zdradzaj, jeśli zdradzono cię po dłuższej współpracy
   Z po **(WW)(WW)(WZ)**
3. Akceptuj przeprosiny
   W po **(WW)(WZ)(ZW)**
4. Zapominaj
   W po **(ZW)(WW)(WW)**
5. Zdradzaj po 3 kolejnych zdradach
   Z po **(ZZ)(ZZ)(ZZ)**

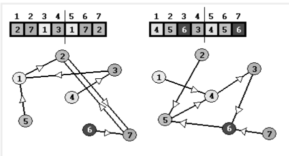
METODY UWZGLĘDNIANIA OGRANICZEŃ

Np. Problem komiwojażera:

- Kodowanie klasyczne: 
  (W genie danego miasta zapisany jest numer miasta docelowego).
- Kodowanie permutacyjne: 
  (W kolejnych genach zapisane są kolejne miasta)

Dla kodowania klasycznego:





1. Metoda kary

- Kara stała (w tym kara śmierci);
- Kara zależna od stopnia naruszenia ograniczenia
  (zależność liniowa, logarytmiczna, wykładnicza itp).

2. Algorytmy naprawy

- korygowanie rozwiązań niedopuszczalnych.

Wady:

- znaczne wydłużenie czasu obliczeń;
- algorytm naprawy musi być dopasowany do konkretnego zadania;
- proces korygowania może być równie złożony, jak zadanie początkowe.

3. Użycie dekoderów

Dekodery – przekształcenia reprezentacji gwarantujące (lub zwiększających prawdopodobieństwo) generowania osobników spełniających ograniczenia.

Wady:

- wysokie wymagania obliczeniowe;
- nie wszystkie ograniczenia mogą być w ten sposób uwzględnione;
- konieczność stosowania dedykowanych dekoderów.

Zadanie załadunku (plecakowe)

- Istnieje cały szereg zadań załadunku (zadań plecakowych).
- Dany jest zbiór rzeczy o nadanych rozmiarach i wartościach;
- Należy wybrać jeden lub wiele rozłącznych podzbiorów tak, by suma rozmiarów w każdym podzbiorze nie przekroczyła zadanego ograniczenia (pojemność plecaka) i by suma wartości była maksymalna.
- *Wiele zadań z tej klasy jest NP-trudnych.*

PROBLEMY NP

Problem NP (*nondeterministic polynomial*):

- problem decyzyjny, dla którego rozwiązanie można zweryfikować w czasie wielomianowym.

Problem P_0 jest **NP-zupełny**, gdy:

1. P_0 należy do klasy NP,
2. Każdy problem z klasy NP da się sprowadzić w czasie wielomianowym do problemu P_0.

Problem **NP-trudny** spełnia tylko punkt 2.

- Problemy NP-zupełne maja postać pytania *„czy istnieje"*.
- Problemy NP-trudne to zwykle ich optymalizacyjne wersje („znajdź najmniejszy").

Zero-jedynkowe zadanie załadunku

Wybierz wektor binarny $\mathbf{x} = \langle x[1], \ldots, x[n]\rangle$

spełniający warunki:

$$\sum_{i=1}^{n} x[i]\cdot W[i] \le C$$

$$\mathbf{P}(\mathbf{x}) = \sum_{i=1}^{n} x[i]\cdot P[i] = \max$$

gdzie: $W[i]$ – zbiór wag (rozmiarów);
$P[i]$ – zbiór zysków;
C – pojemność.

Wylosowano 3 zbiory danych:

$W[i]$ – zbiór wag (rozmiarów);
$P[i]$ – zbiór zysków;

$W[i]$ – przypadkowe $([1, v])$ o rozkładzie jednostajnym;

1. Nieskorelowane:

$P[i]$ – przypadkowe $([1, v])$ o rozkładzie jednostajnym.

2. Słabo skorelowane:

$P[i] = W[i]$ + przypadkowe $([-r, r])$ o rozkładzie jednostajnym.

2. Silnie skorelowane:

$P[i] = W[i] + r$.

Przyjęto parametry:
$v = 10$, $r = 5$

Większa korelacja to mniejsza wartość różnicy:

$$\max(P[i]/W[i]) - \min(P[i]/W[i]), \quad i = 1 \ldots n$$

co może powodować większe problemy przy optymalizacji.

Rozpatrzono 2 rodzaje zadania załadunku:

(Michalewicz)

- Z ograniczoną pojemnością (C_1)
  (Rozwiązanie optymalne zawiera bardzo mało artykułów).
- Ze średnią pojemnością (C_2)
  (Rozwiązanie optymalne zawiera ok. połowy artykułów).

Rodzaje użytych algorytmów:

- Algorytmy z funkcją kary ($A_p[\mathrm{k}]$);
- Algorytmy z metodami naprawy ($A_r[\mathrm{k}]$);
- Algorytmy z dekoderami ($A_d[\mathrm{k}]$).

k – *numer algorytmu*

Ad. 1 ALGORYTMY Z FUNKCJĄ KARY ($A_p[\mathrm{k}]$)

- Wektor $\mathbf{x}$ jest reprezentowany przez łańcuch binarny o długości n.
- Artykuł i-ty jest ładowany do plecaka tylko wtedy, gdy:

$$x[i]=1$$

Dopasowanie:

$$eval(\mathbf{x}) = \sum_{i=1}^{n} x[i]\cdot P[i] - Pen(\mathbf{x})$$

gdzie: $Pen(\mathbf{x})$ – funkcja kary:

a) $Pen(\mathbf{x}) = \mathbf{0}$ dla rozw. dopuszczalnych;
b) $Pen(\mathbf{x}) > \mathbf{0}$ dla pozostałych rozwiązań.

Wzrost kary (ze stopniem przekroczenia ograniczenia):

a. logarytmiczny:

$$A_p[1]:\quad Pen(\mathbf{x}) = \log_2\left[1 + \rho\cdot\left(\sum_{i=1}^{n} x[i]\cdot W[i] - C\right)\right]$$

b. liniowy:

$$A_p[2]:\quad Pen(\mathbf{x}) = \rho\cdot\left(\sum_{i=1}^{n} x[i]\cdot W[i] - C\right)$$

c. kwadratowy:

$$A_p[3]:\quad Pen(\mathbf{x}) = \left[\rho\cdot\left(\sum_{i=1}^{n} x[i]\cdot W[i] - C\right)\right]^2$$

$$\rho = \max\{P[i]/W[i]\},\quad i = 1 \ldots n$$

Ad. 2 ALGORYTMY Z MET. NAPRAWY ($A_r[k]$)

- Wektor **x** jest reprezentowany przez łańcuch binarny o dł. n.
- Artykuł i-ty jest ładowany do plecaka tylko wtedy, gdy $x[i]=1$.

Dopasowanie:

gdzie: $\mathbf{x}'$ – naprawiona wersja wektora **x**.

Można stosować różne metody naprawy.

Zastosowane algorytmy różnią jedynie procedurą wybierającą artykuł do wyjęcia z plecaka:

- $A_r[1]$ – naprawa losowa;
- $A_r[2]$ – naprawa zachłanna *(Wszystkie artykuły w plecaku są ustawione w porządku malejącym względem stosunków zysków do wagi. Procedura wyboru wybiera ostatni artykuł na liście).*

Ad. 3 ALGORYTMY Z DEKODERAMI ($A_d[k]$)

- Wektor **x** jest reprezentowany przez łańcuch całkowito-liczbowy o długości n, gdzie i-ty składnik wektora to liczba z zakresu $[1, n-i+1]$;
- Artykuł i-ty jest ładowany do plecaka tylko wtedy, gdy znaj-duje się na bieżącej liście.

Reprezentacja porządkowa korzysta z listy L artykułów, deko-dowanie za pomocą wektora następuje poprzez wybór artyku-łów z bieżącej listy, np:

dla listy: **L = (1, 2, 3, 4, 5, 6)**

dekodowanej za pomocą wektora: **< 4, 3, 4, 1, 1, 1 >**

otrzymujemy ciąg artykułów : **4,3,6,1,2,5**

- $A_d[1]$ – **dekodowanie losowe**

Procedura dekodowania tworzy listę L artykułów takich, że ich kolejność na liście odpowiada kolejności artykułów w zbiorze wejściowym (który jest przypadkowy).

- $A_d[2]$ – **dekodowanie zachłanne**

Procedura dekodowania tworzy listę L artykułów w porządku malejącym względem stosunków zysków do wagi. Dekodowanie wektora **x** następuje na podstawie uporządkowanego zbioru.

Parametry programu:

- liczebność populacji $pop\_size = 100$ (stała);
- $p_m = 0.05$;
- $p_c = 0.65$;
- liczba pokoleń $gen\_number = 500$.

Wyniki – średnia z 25 obliczeń

Korelacja	Liczba artykułów	Typ pojemności	Metoda						
			$A_p[1]$	$A_p[2]$	$A_p[3]$	$A_r[1]$	$A_r[2]$	$A_d[1]$	$A_d[1]$
BRAK	100	C_1 (ograniczona)	*	*	*	63	94	64	60
		C_2 (średnia)	398	341	243	345	371	355	353
	250	C_1	*	*	*	123	136	58	60
		C_2	920	837	826	842	894	867	858
	500	C_1	*	*	*	64	156	61	61
		C_2	1712	1571	1565	1577	1663	1603	1597
SŁABA	100	C_1	*	*	*	38	51	38	38
		C_2	409	327	328	330	358	334	332
	250	C_1	*	*	*	44	74	43	45
		C_2	921	791	789	798	852	804	799
	500	C_1	*	*	*	45	94	43	45
		C_2	1729	1532	1532	1539	1624	1548	1574
MOCNA	100	C_1	*	*	*	62	90	60	60
		C_2	742	565	565	567	577	576	576
	250	C_1	*	*	*	66	117	65	64
		C_2	1632	1340	1343	1346	1364	1366	1359
	500	C_1	*	*	*	68	120	67	64
		C_2	3052	2704	2701	2710	2748	2738	2744

Wnioski:

- Dla zadań ze średnią pojemnością algorytm $A_p[1]$ (z logarytmiczną funkcją kary) jest najskuteczniejszy;
- Dla zadań z ograniczoną pojemnością zachłanna metoda naprawy ($A_r[2]$) jest najskuteczniejsza.

BINARNIE CZY INACZEJ?

Binarnie	Niebinarnie
00000	A
00001	B
...	...
11001	Z
11010	1
...	...
11111	6

Ciąg binarny	Ciąg niebinarny	Wartość	Dopasowanie
11000	**Y**	24	**576**
01011	**L**	11	**121**
01000	**I**	8	**64**
10011	**T**	19	**361**

Porównanie liczby schematów:
- jednakowa liczba osobników;
- ciągi kodowe o różnych długościach.

By liczba punktów w obu przestrzeniach była jednakowa:

$$\boxed{2^l = k^{l'}}$$

l – długość osobnika zakodowanego binarnie,

l' – dł. osobnika zakodowanego w alfabecie k-elementowym

tu:

$$2^5 = k^1 \Rightarrow k = 32$$

Liczba schematów:
- 3^l dla alfabetu dwójkowego
- $(k+1)^{l'}$ dla alfabetu k-elementowego.

tu:

$3^5 = \mathbf{243}$ dla alfabetu dwójkowego

$(32+1)^1 = 33$ dla alfabetu k-elementowego.

Kod dwójkowy charakteryzuje się największą ze wszyst-kich liczbą schematów przypadającą na bit informacji.

Jednakże jeżeli:
- 100 zmiennych;
- dziedzina z zakresu [-500 500];
- żądana dokładność 6 miejsc po przecinku;

To:
- długość łańcucha binarnego wynosi **3000**;
- przestrzeń poszukiwań rzędu $\mathbf{10^{1000}}$.

Dla tak wielkich przestrzeni AG działają słabo...

Zasada znaczących cegiełek:

Kod należy dobierać w taki sposób, by schematy niskiego rzędu i o małej rozpiętości wyrażały własności zadania oraz pozostawały względnie niezależne od schematów na pozycjach ustalonych.

Zasada minimalnego alfabetu:

Należy wybrać najmniejszy alfabet, w którym zadanie wyraża się w sposób naturalny.

Jeden z celów zmodyfikowanego kodowania:
przybliżenie algorytmu do przestrzeni zadania.

Dogodne jest, by dwa punkty leżące blisko siebie w przestrzeni reprezentacji (genotyp) leżały również blisko siebie w przestrzeni zadania (fenotyp).

(*Nie zawsze prawdziwe przy kodowaniu binarnym*)

np.:

Binarnie	Całkowitoliczbowo
0111	7
1000	8

KOD GRAYA

```
procedure GrayToBin
begin
    value := g_1
    b_1 := value
    for k := 2 to m do
    begin
        if g_k = 1 then value := NOT value
        b_k := value
    end
end
```

```
procedure BinToGray
begin
    g_1 := b_1
    for k := 2 to m do
        g_k := b_{k-1} XOR b_k
end
```

a	**b**	**a** XOR **b**
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**

$\mathbf{b} = \langle b_1, b_2, \ldots, b_m \rangle$ – liczba binarna
$\mathbf{g} = \langle g_1, g_2, \ldots, g_m \rangle$ – liczba w kodzie Graya;
m – długość ciągu kodowego.

Binarnie	Kod Graya
0000	0000
0001	0001
0010	0011
0011	0010
0100	0110
0101	0111
0110	0101
0111	0100
1000	1100
1001	1101
1010	1111
1011	1110
1100	1010
1101	1011
1110	1001
1111	1000

Zmiana 1 bitu w kodzie powoduje, że otrzymana liczba ma szansę być liczbą bezpośrednio bliską liczbie przed zmianą.

KODOWANIE LOGARYTMICZNE

Stosowane w celu zmniejszenia długości łańcucha binarnego.

$$[b_1 b_2 \text{bin}] = (-1)^{b_2} e^{(-1)^{b_1}[\text{bin}]_{10}}$$

b_1 – bit znaku wykładnika funkcji wykładniczej;
b_2 – bit znaku funkcji wykładniczej;
bin – reprezentacja wykładnika funkcji wykładniczej
$[\text{bin}]_{10}$ – wartość dziesiętna liczby zakodowanej binarnie.

$$[10110] = (-1)^0 e^{(-1)^1[110]_{10}} = e^{-6} = 0.002478752$$

$$[01011] = (-1)^1 e^{(-1)^0[011]_{10}} = -e^3 = -20.08553692$$

- Za pomocą 5 bitów możliwe jest zakodowanie liczb z zakresu $[-e^7, e^7]$
  (w kodowaniu binarnym $[0, 31]$).

- Dalszą modyfikacją jest zastosowanie **KODOWANIA ZMIENNOPOZYCYJNEGO.**

KODOWANIE CHROMOSOMU:

Podział chromosomów uwzględniający wartości:
- binarne (np. zadanie plecakowe);
- całkowitoliczbowe (np. TSP);
- zmiennopozycyjne (typowe inżynierskie zadania optymalizacji);
- tekstowe.

KODOWANIE CHROMOSOMU:

Kodowanie (reprezentacja danych) to zbiór stanów z przestrzeni zadania przedstawiony w postaci skończonego alfabetu znaków.

Podział chromosomów uwzględniający strukturę:
- standardowe (jak w klasycznym AG);
- permutacyjne (np. problem komiwojażera - TSP);
- drzewiaste;
- macierzowe.