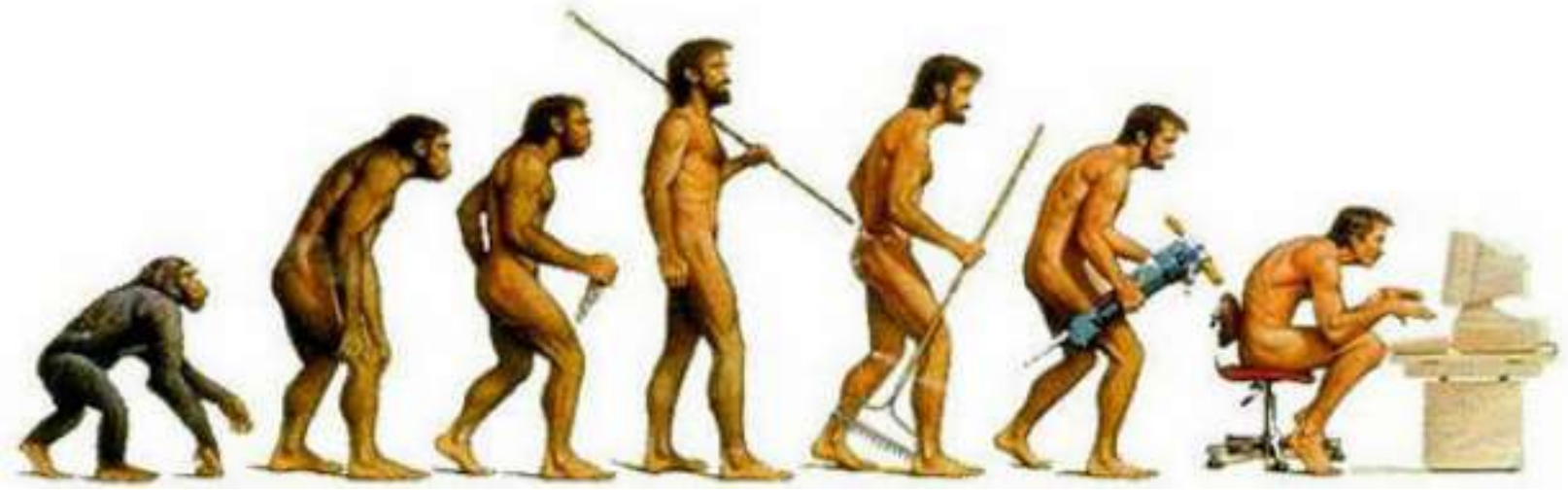
Wydział Zarządzania AGH
Katedra Informatyki Stosowanej

Algorytmy ewolucyjne

Metaheurystyki

Treść wykładu

- Wprowadzenie
- Zasada działania
- Podział EA
- Algorytm genetyczny
- Przykład



EA - wprowadzenie

- Algorytmy ewolucyjne (ang. Evolutionary Algorithms - EA) są szeroko stosowaną heurystyką przeszukiwania i opty-malizacji opartą na zasadach przejętych z teorii ewolucji. Naturalność oraz prostota działania sprawiły, że algorytmy te są chętnie wykorzystywane w naukach zarządzania do rozwiązywania problemów optymalizacji kombinatorycznej, a w szczególności - do szeroko rozumianych problemów alokacji zasobów.
- Algorytmy ewolucyjne nie gwarantują znalezienia optimum globalnego, jednak generalnie zapewniają znalezienie rozwiązania wystarczająco dobrego w akceptowalnym przedziale czasu. Stąd głównym zastosowaniem tych algorytmów powinny być problemy, dla których nie istnieją techniki specjalizowane. Nawet jeśli techniki takie istnieją, można osiągnąć poprawę ich działania poprzez ich połączenie z algorytmami ewolucyjnymi.

Zasada działania

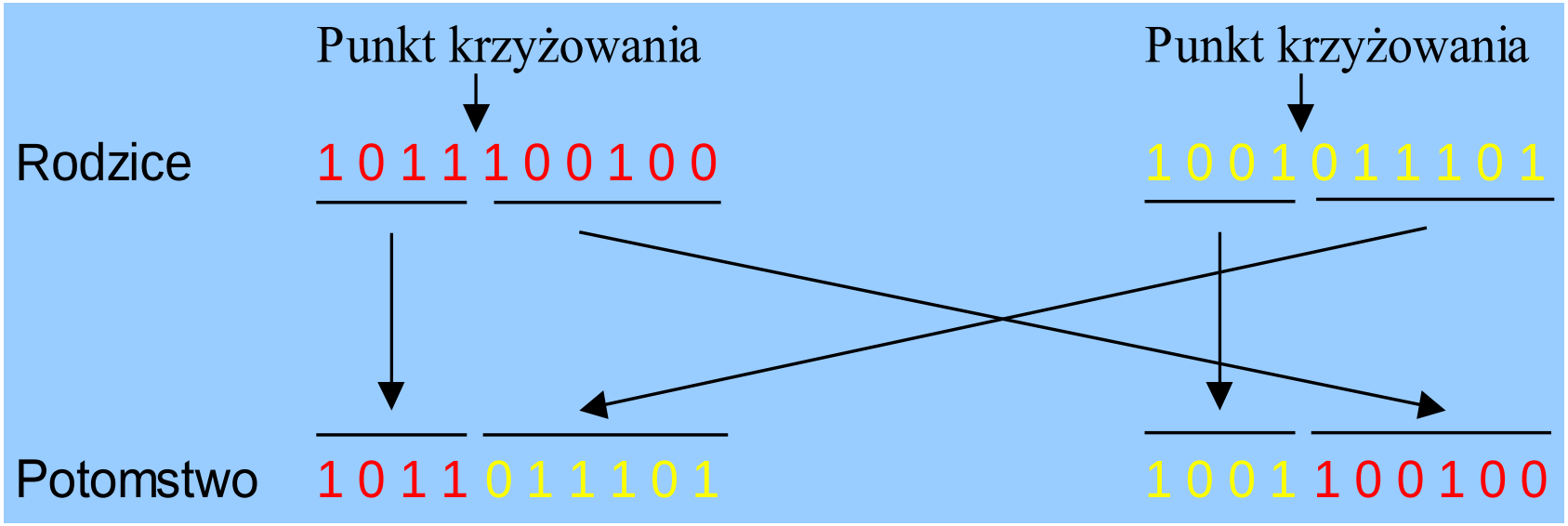
- Drogą rozwoju przyrody ożywionej jest ewolucja czyli metoda prób i błędów oparta na doborze naturalnym. Najlepiej udokumentowanym mechanizmem doboru naturalnego jest proces genetyczny: można go postrzegać jako proces optymalizacyjny, w którym osobniki najlepiej dostosowane do środowiska mają największe szanse przeżycia i utworzenia potomków. Nośnikiem informacji o cechach indywidualnych osobnika są geny - bloki DNA - tworzące chromosomy: kod ten determinuje budowę osobnika i jego rozwój, a w szczególności - jego dopasowanie do środowiska naturalnego. Istnieje silna zależność między chromosomem osobnika a jego żywotnością i zdolnością do przekazywania genotypu kolejnym pokoleniom.

Zasada działania

- Ewolucyjny rozwój populacji chromosomów odbywa się poprzez mechanizm reprodukcji, na który składają się procesy krzyżowania (ang. crossover), mutacji (ang. mutation) i inwersji (ang. inversion). W procesie krzyżowania, z dwóch chromosomów rodzicielskich wybierane są geny, które po zespoleniu tworzą jeden lub więcej chromosomów potomnych. W procesie mutacji dochodzi do przekłamania kodu poprzez zmianę jednego genu lub ich ciągu, natomiast inwersja odwraca fragment chromosomu. Przy pomocy tych mechanizmów tworzą się kolejne pokolenia (populacje chromosomów), zawierające coraz "doskonalsze" osobniki.

Zasada działania

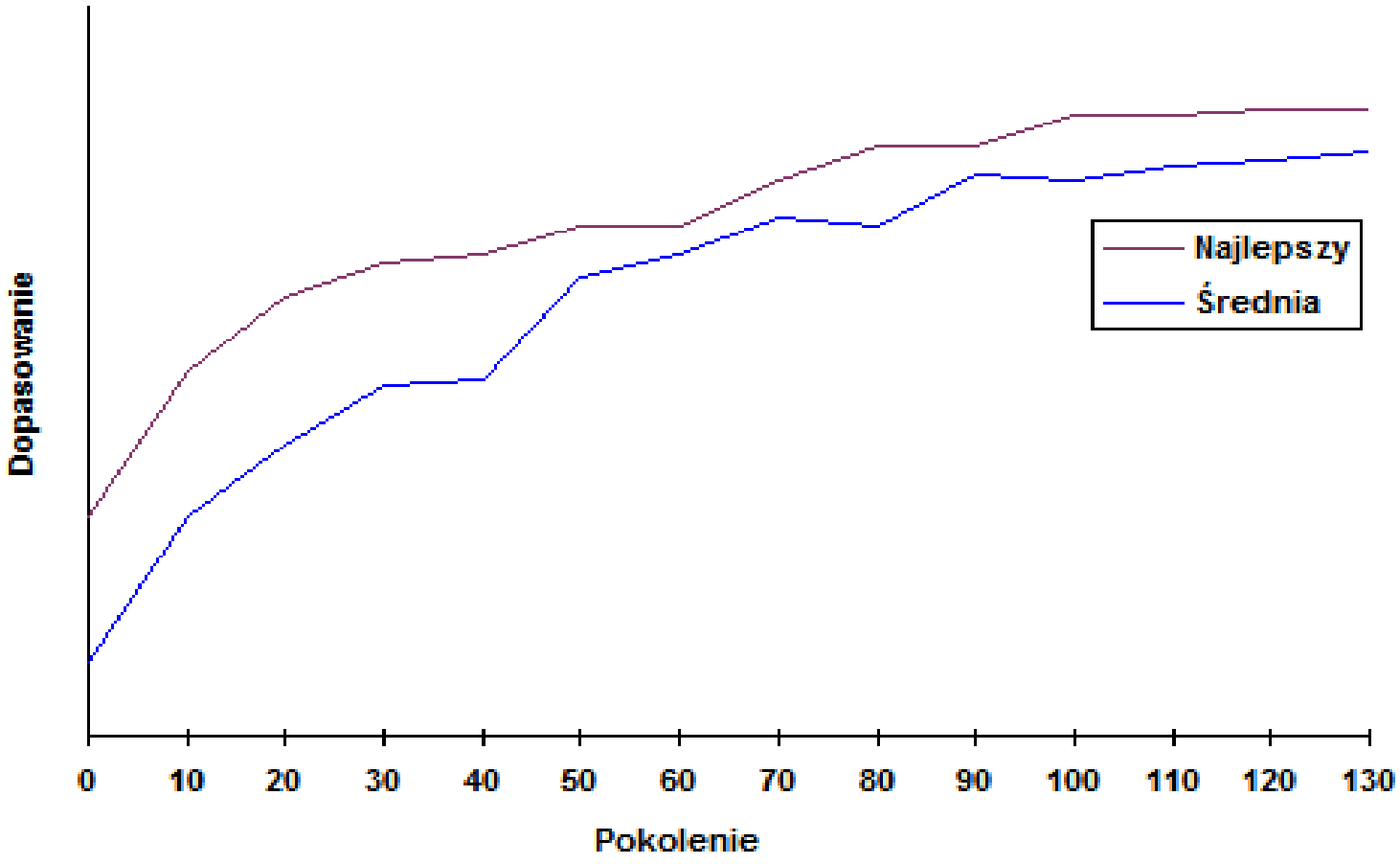
Krzyżowanie osobników



Zasada działania

Mutacja i inwersja

Zasada działania



Podział AE

Ta ogólnie przedstawiona zasada działania jest wykorzystywana w różnych wariantach algorytmów ewolucyjnych; wyróżnia się 4 typy EA:

- algorytmy genetyczne (ang. Genetic Algorithms - GA),
- strategie ewolucyjne (ang. Evolution Strategies - ES),
- programowanie ewolucyjne (ang. Evolutionary Programming - EP),
- programowanie genetyczne (ang. Genetic Programming - GP).

Cechy algorytmów ewolucyjnych

Algorytmy powyższe, oprócz terminologii oraz operatorów przeniesionych z biologii, różnią się istotnie od innych technik przeszukiwania:

- wykorzystują operatory pseudoewolucyjne, które działają na sprecyzowanych reprezentacjach rozwiązań (osobnikach),
- przetwarzają całą populację rozwiązań, badając przy tym przestrzeń przeszukiwania równocześnie z wielu punktów,
- do prawidłowego działania nie potrzebują żadnej szczegółowej wiedzy o charakterze problemu, a jedynie informację o jakości rozwiązań,
- w sposób celowy stosują procesy stochastyczne; oznacza to nie losowe, lecz inteligentne badanie przestrzeni przeszukiwania: wysiłek obliczeniowy koncentruje się na obiecujących regionach przestrzeni przeszukiwań.

Zalety algorytmów ewolucyjnych

- efektywna technika, o szerokich możliwościach zastosowania,
- wiarygodność,
- dobrze przystosowane do przeszukiwania wielowymiarowej, złożonej przestrzeni rozwiązań,
- względnie łatwe do opracowania i implementacji,
- nie ma ograniczeń co do postaci funkcji celu,
- wstępna wiedza o problemie nie jest potrzebna,
- możliwość optymalizacji wielokryterialnej,
- duży wybór postaci algorytmu,
- łatwa współpraca z innymi technikami (heurystyki inicjalizacyjne, przeszukiwanie lokalne),
- naturalna paralelność algorytmu.

Wady algorytmów ewolucyjnych

- heurystyczny charakter techniki (nie daje pewności osiągnięcia optimum w określonym czasie),
- czasochłonność obliczeń (łagodzona przez gwałtowny postęp techniczny),
- często nieefektywna w końcowej fazie przeszukiwań,
- określenie właściwych wartości parametrów nie jest łatwe.

Algorytm genetyczny

Model procesu ewolucyjnego do rozwiązywania zagadnień, których rozwiązanie jest sformalizowane w postaci ciągu binarnego, przyjęto nazywać algorytmem genetycznym. Ogólnie biorąc, algorytm genetyczny jest iteracyjną procedurą poprawiania rozwiązań zawartych w zbiorze najlepszych rozwiązań danego zagadnienia, realizowaną przez zbiór pseudogenetycznych operatorów.

- Chromosom jest ciągiem genów (elementów) o ustalonej długości; standardowo jest to ciąg binarny.
- Populację stanowi zbiór chromosomów o zadanej liczebności.

Algorytm genetyczny

Podstawowymi elementami algorytmu genetycznego są:

- genetyczna reprezentacja rozwiązania problemu,
- sposób generowania populacji początkowej,
- postać funkcji dopasowania (ang. fitness function), oceniającej dopasowanie rozwiązań do otoczenia,
- sposób doboru rodziców,
- stosowane operatory genetyczne,
- sposób selekcji następnego pokolenia,
- wartości parametrów:
  -rozmiar populacji *P*,
  -prawdopodobieństwo zastosowania operatorów krzyżowania *pc*,
  -prawdopodobieństwo zastosowania operatorów mutacji *pm*,
  -warunek zatrzymania.

Algorytm genetyczny

```
BEGIN
t := 0;
inicjalizacja populacji początkowej P(t);
REPEAT
    badanie dopasowania P(t);
    t := t+1;
    P(t) = { };
    WHILE  (nie zakończona selekcja P(t)) DO
    BEGIN
        wybór dwóch osobników zgodnie z ich wartością funkcji
        dopasowania;
        rekombinacja osobników w celu utworzenia dwóch potomków;
        mutacja potomków;
        dodanie potomków do populacji P(t);
    END;
UNTIL (spełniony warunek zatrzymania)
END.
```

Algorytm genetyczny - podstawy

Reprezentacja

- Dostosowana do problemu.
- Powinna kodować jak najwięcej informacji specyficznych dla problemu.
- Może wymagać specjalizowanych operatorów.

Funkcja dopasowania

- Dopasowanie osobnika może być obliczone na podstawie genotypu.
- Najczęściej jest to wprost funkcja celu.
- Często funkcja zawiera ograniczenia nałożone na zmienne np. w postaci funkcji kary.

Algorytm genetyczny - podstawy

Reprezentacja

- Potencjalne rozwiązanie problemu może być reprezentowane przez ciąg parametrów (np. wymiary przęseł w przypadku projektu mostu). Parametry te (geny) są łączone tworząc ciąg wartości (chromosom). Przyjmuje się, że alfabet binarny jest najlepszy do tworzenia takiego ciągu. Jeśli np. problemem jest maksymalizacja wartości funkcji dwóch zmiennych F(x,y), reprezentacją każdej zmiennej może być 10-bitowa liczba binarna. Tak więc chromosom będzie się składał z 2 genów i liczył 20 cyfr binarnych.

Algorytm genetyczny - podstawy

Reprezentacja liczby rzeczywistej

d – dokładność obliczeń (miejsca po przecinku),

min, *max* – dolna, górna granica przedziału,

m – długość chromosomu (ile bitów),

x_b – wartość dziesiętna liczby binarnej.

Znajdujemy najmniejszą liczbę całkowitą *m* taką, że:

$(max - min)*10^d <= 2^m - 1$

Wartość rzeczywista liczby *x* z przedziału [*min*, *max*]:

$x = min + x_b*(max - min)/(2^m - 1)$ Udowodnić!

d=1 => *m*=5	4	3	2	1	0	dziesiętna
binarna	1	0	1	1	0	22
min	0	1	-3	-1	-4	
max	3	4	0	2	-1	
x	2,1	3,1	-0,9	1,1	-1,9	

Algorytm genetyczny - podstawy

Reprezentacja binarna

- Podstawowym problemem przy takiej reprezentacji jest bardzo możliwa duża odległość między reprezentacją a rzeczywistą wartością.

  Np. przypuśćmy, że optimum wynosi 32 i używamy 6-bitowego chromosomu, tak więc reprezentacją 32 jest **100000**. Najbliższą optimum jest liczba 31, lecz jej postać to **011111**. Z kolei liczba binarna **000000** reprezentująca 0 różni się tylko 1 bitem od wartości kodującej optimum.

Algorytm genetyczny - przykład

- Maksymalizacja f(x)=2*x dla 0<=x<=31
- Założenia:
  - rozmiar populacji *P* = 4,
  - krzyżówka jednopunktowa,
  - dobór rodziców zgodnie z funkcją dopasowania (ruletka),
  - prawd. zastosowania operatora krzyżowania *pc* = 0,8,
  - prawd. zastosowania operatora mutacji typu flip *pm* =0,05,
  - selekcja następnego pokolenia – 50% najlepszych dzieci ,
  - warunek zatrzymania.

Algorytm genetyczny - przykład

- Maksymalizacja f(x)=2x dla 0<=x<=31
- Krok 0 – wylosowanie populacji początkowej.
- Pętla 1 – krok 1: obliczenie funkcji dopasowania osobników.

osobnik	4	3	2	1	0	x_i	$FD(x_i)$
1	1	1	0	0	0	24	48
2	0	0	1	0	1	5	10
3	1	0	1	1	0	22	44
4	0	1	1	0	0	12	24
suma						63	126
średnia						15,75	31,50

- Pętla 1 – krok 2: wybór rodziców zgodnie z funkcją dopasowania.

0	x_i	$FD(x_i)$	Prawd.	Dystr.
0	24	48	0,381	0,381
1	5	10	0,079	0,460
0	22	44	0,349	0,810
0	12	24	0,190	1,000
	63	126	1,000	

L.losowa	1. rodzic	L.losowa	2. rodzic
0,258	1	0,462	3
0,593	3	0,030	1
0,984	4	0,340	1
0,704	3	0,963	4

Algorytm genetyczny - przykład

- Maksymalizacja f(x)=2x dla 0<=x<=31
- Pętla 1 – krok 3: krzyżówka.

L.losowa	Rodzice										
0,642	1	1	1	0	0	0	3	1	0	1	1	0
0,117	3	1	0	1	1	0	1	1	1	0	0	0
0,987	4	0	1	1	0	0	1	1	1	0	0	0
0,694	3	1	0	1	1	0	4	0	1	1	0	0
	Dzieci										
	1	1	1	1	1	0	2	1	0	0	0	0
	3	1	0	1	0	0	4	1	1	0	1	0
	5	0	1	1	0	0	6	1	1	0	0	0
	7	1	1	1	0	0	8	0	0	1	1	0

Algorytm genetyczny - przykład

- Maksymalizacja f(x)=2x dla 0<=x<=31
- Pętla 1 – krok 4: mutacja.

osobnik	4	3	2	1	0	x_i	FD(x_i)	L.losowa	L.losowa	L.losowa	L.losowa	L.losowa
1	1	1	1	1	0	30	60	0,848	0,409	0,962	0,407	0,118
2	1	0	0	0	0	16	32	0,158	0,543	0,472	0,141	0,074
3	1	0	1	0	0	20	40	0,596	0,328	0,284	0,368	0,136
4	1	1	0	1	0	26	52	0,824	0,260	0,116	0,333	0,821
5	0	1	1	0	0	12	24	0,843	0,600	0,344	0,695	0,617
6	1	1	0	0	0	24	48	0,883	0,724	0,922	0,320	0,713
7	1	1	1	0	0	28	56	0,715	0,281	0,790	0,924	0,547
8	0	0	1	1	0	6	12	0,015	0,646	0,399	0,318	0,918
suma						162	324					
średnia						20,25	40,50					

osobnik	4	3	2	1	0	x_i	FD(x_i)
1	1	1	1	1	0	30	60
2	1	0	0	0	0	16	32
3	1	0	1	0	0	20	40
4	1	1	0	1	0	26	52
5	0	1	1	0	0	12	24
6	1	1	0	0	0	24	48
7	1	1	1	0	0	28	56
8	1	0	1	1	0	22	44
suma						178	356
średnia						22,25	44,50

Algorytm genetyczny - przykład

- Maksymalizacja f(x)=2x dla 0<=x<=31
- Pętla 1 – krok 5: selekcja następnego pokolenia.

osobnik	4	3	2	1	0	x_i	$FD(x_i)$
1	1	1	1	1	0	30	60
7	1	1	1	0	0	28	56
4	1	1	0	1	0	26	52
6	1	1	0	0	0	24	48
8	1	0	1	1	0	2	
3	1	0	1	0	0	2	
2	1	0	0	0	0	1	
5	0	1	1	0	0	1	
suma						17	
średnia						22,	

osobnik	4	3	2	1	0	x_i	$FD(x_i)$
1	1	1	0	0	0	24	48
2	0	0	1	0	1	5	10
3	1	0	1	1	0	22	44
4	0	1	1	0	0	12	24
suma						63	126
średnia						15,75	31,50

osobnik	4	3	2	1	0	x_i	$FD(x_i)$
1	1	1	1	1	0	30	60
7	1	1	1	0	0	28	56
4	1	1	0	1	0	26	52
6	1	1	0	0	0	24	48
suma						108	216
średnia						27,00	54,00