

Algorytmy genetyczne

Michał Bereta

Paweł Jarosz (część teoretyczna)

Wprowadzenie

Dana jest funkcja f , jednej lub wielu zmiennych. Należy określić wartości zmiennych, dla których funkcja f osiąga ekstrema tj. wartość minimalną lub maksymalną. Zadanie maksymalizacji może być w prosty sposób zamienione na zadanie minimalizacji i odwrotnie. Numeryczne metody wyznaczania ekstremum funkcji, powinny działać szybko i przy jak najmniejszej ilości pamięci [7].

1.1 Optymalizacja lokalna i globalna

Optymalizacja to wyznaczanie optymalnego (tj. najkorzystniejszego, najlepszego), ze względu na wybrane kryteria rozwiązania danego problemu, przy użyciu metod matematycznych (numerycznych).

Przyjęto, że R oznacza zbiór wszystkich rozwiązań rozpatrywanego problemu. Punkt x^* w przestrzeni R jest minimum globalnym, jeśli dla każdego x należącego do R $f(x^*) \leq f(x)$.

$$x = x^* \Leftrightarrow \forall x \in R \quad f(x^*) \leq f(x)$$

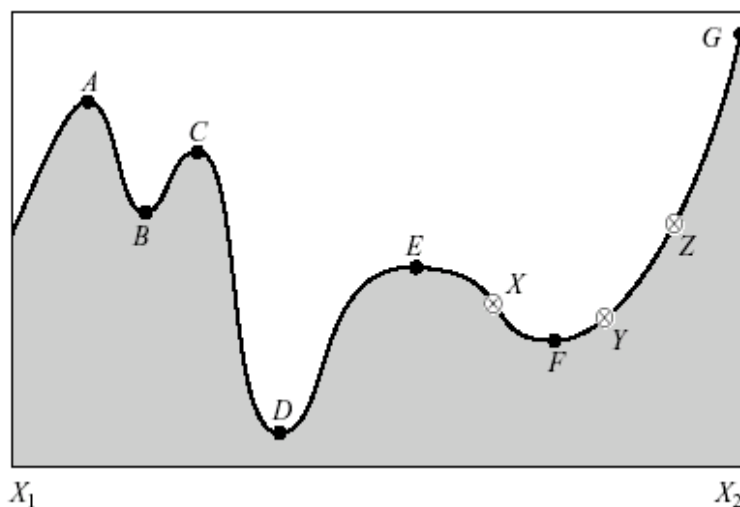
W takim przypadku (najbardziej idealnym), x^* jest optimum globalnym. Częściej formułowanym problemem (łatwiejszym do rozwiązania) jest znalezienie takiego x_i , dla którego $f(x_i) < f(x_i+t)$, gdzie: $t > 0$, x_i+t są punktami z najbliższego sąsiedztwa x_i .

$$x = x_i \Leftrightarrow f(x_i) \leq f(x_i + t), \quad t \neq 0$$

Tak sformułowany problem określa minimum lokalne. Klasyczne metody optymalizacji znajdują zazwyczaj minima lokalne. Na rysunku 1 pokazano optima lokalne i globalne przykładowej funkcji jednej zmiennej, w przedziale $[X_1, X_2]$.

Maksima istnieją w punktach A, C, E, G, przy czym A, C, E są maksimami lokalnymi. W punkcie G jest maksimum globalne. Natomiast w punktach B, D i F znajdują się minima: B i F to minima lokalne. W punkcie D jest minimum globalne.

Korzystając z klasycznych metod optymalizacji (np. metod gradientowych) i szukając minimum dla wartości początkowych w punktach X, Y lub Z – rozwiązaniem jest zawsze punkt F, który jest minimum lokalnym [7].



Rysunek 1. Przykład funkcji z oznaczonymi ekstremami [7]

W wielu praktycznych zastosowaniach pojawiają się trudności przy korzystaniu z tradycyjnych metod optymalizacji. Większość ma zasięg lokalny. Ich sposób działania zależy od istnienia pochodnych i nie są one dostatecznie odporne na nieciągłości, rozległe wielomodalności lub występowanie zakłóceń w przeszukiwanej przestrzeni [4].

1.2 Algorytmy genetyczne – podstawowe pojęcia

Michalewicz w [4] pisze:

„Pomysł leżący u podstaw algorytmów genetycznych polega na zrobieniu tego samego, co robi natura. Weźmy dla przykładu króliki. W każdej chwili mamy jakąś populację królików. Niektóre z nich są szybsze i sprytniejsze od innych. Szybsze i sprytniejsze króliki mają większą szansę umknienia przed lisem. Wobec tego więcej ich przeżywa, aby zrobić to, co króliki robią najlepiej, a mianowicie nowe króliki. Oczywiście, niektóre z wolniejszych i głupszych królików też przeżyją, bo mają szczęście. Ta ocalała populacja królików wydaje potomstwo. Jest ono dobrą mieszaniną materiału genetycznego. Niektóre wolne króliki krzyżują się z szybkimi, niektóre szybkie z szybkimi, niektóre sprytnie z głupimi itd. A do tego natura dorzuca od czasu do czasu „dziką kartę”, wprowadzając mutację do genetycznego materiału królików. Urodzone w wyniku takiego procesu króliki będą (średnio) szybsze i sprytniejsze niż te z początkowej populacji, ponieważ szybsi i sprytniejsi rodzice umknęli przed lisami.”

Ten obrazowy przykład przedstawia analogię pomiędzy algorytmem genetycznym a naturą. W algorytmach genetycznych stosuje się pojęcia zapożyczone z genetyki naturalnej. Istnieje *populacja* rozwiązań. Każde rozwiązanie jest nazywane *osobnikiem*. Na *populacji* dokonujemy *selekcji* (słabe osobniki giną), *krzyżowania* (czyli tworzenia nowych osobników) i *mutacji* (dzika karta). W terminologii algorytmów genetycznych używa się następujących pojęć [6]:

- *Populacja* – zbiór osobników o określonej liczebności.
- *Osobniki* populacji w algorytmach genetycznych to zakodowane w postaci chromosomów zbiory parametrów zadania, czyli *rozwiązania*, określane też jako *punkty przestrzeni poszukiwań*.
- *Chromosomy* – inaczej *łańcuchy* lub *ciągi kodowe* – to uporządkowane *ciągi genów*.
- *Gen* – nazywany też *cecha*, *znakiem*, *detektorem* – jest to pojedynczy element genotypu, w szczególności chromosomu.
- *Genotyp*, czyli *struktura* - to zespół chromosomów danego osobnika. Osobnikami populacji mogą być genotypy albo pojedyncze chromosomy.
- *Fenotyp* - zestaw wartości odpowiadający danemu genotypowi, czyli *zdekodowana struktura*. Jest to zbiór parametrów zadania (rozwiązanie, punkt przestrzeni poszukiwań).
- *Allel* to wartość danego genu (określana też jako *wartość cechy* lub *wariant cechy*).
- *Locus* to położenia danego genu w *łańcuchu* czyli w chromosomie.
- *Funkcja przystosowania* (ang. *fitness function*) nazywana też *funkcją dopasowania* lub *funkcją oceny*. Stanowi ona miarę przystosowania (dopasowania) danego osobnika w populacji.

Funkcja przystosowania pozwala ocenić stopień przystosowania poszczególnych osobników w populacji. Na tej podstawie wybiera się osobniki najlepiej przystosowane (o największej wartości funkcji przystosowania). Jest to zgodne z ewolucyjną zasadą przetrwania „najsilniejszych” (najlepiej przystosowanych). Ma ona duży wpływ na działanie algorytmów genetycznych i musi być odpowiednio zdefiniowana.

- *Generacja* – to kolejna iteracja w algorytmie genetycznym.
- *Pokolenie* (nowe pokolenie lub pokolenie potomków) – to nowoutworzona populacja osobników.

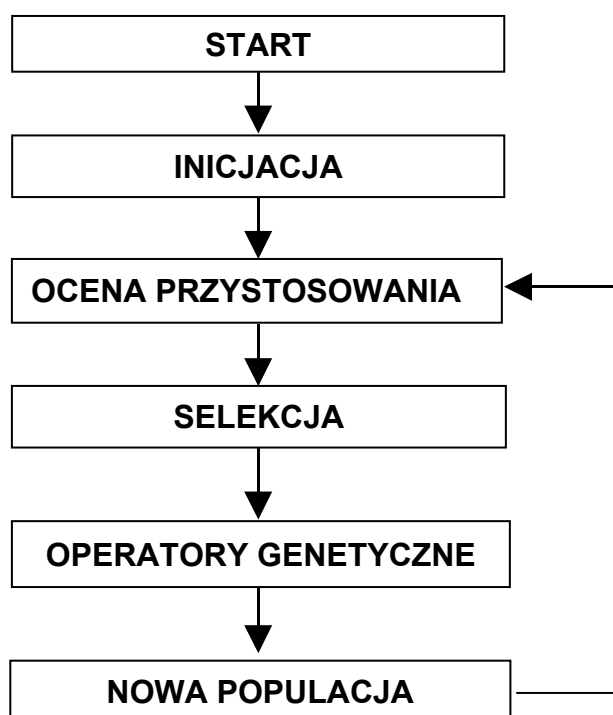
W algorytmie genetycznym, w każdej jego iteracji, jest oceniane przystosowanie każdego osobnika danej populacji za pomocą funkcji przystosowania. Na tej podstawie tworzona jest nowa populacja osobników, którzy stanowią zbiór potencjalnych rozwiązań problemu, np. zadania optymalizacji.

1.3 Klasyczny algorytm genetyczny

Mechanizm działania klasycznego (elementarnego, prostego) algorytmu genetycznego jest prosty. Polega na kopiowaniu ciągów i wymianie podciągów. Na rysunku 2 pokazano schemat działania algorytmu genetycznego, które składa się z następujących kroków [6]:

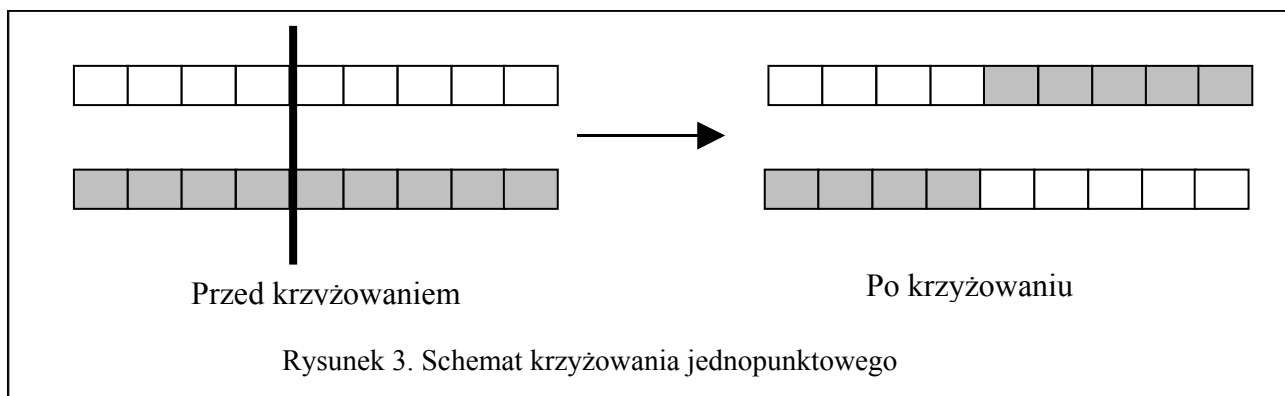
- **Inicjacja** – utworzenie populacji początkowej, poprzez losowy wybór ustalonej liczby chromosomów.
- **Ocena przystosowania** – obliczenie wartości funkcji przystosowania dla każdego chromosomu.
- **Selekcja chromosomów** – wybór chromosomów, które biorą udział w tworzeniu nowej populacji.
- **Zastosowanie operatorów genetycznych** – na grupie chromosomów, wybranej drogą selekcji działają operatory genetyczne. W klasycznym algorytmie genetycznym są operatory krzyżowania i mutacji. Przy czym, krzyżowanie powinno zachodzić znacznie częściej niż mutacja.
- **Utworzenie nowej populacji** – Chromosomy otrzymane jako rezultat działania operatorów genetycznych na chromosomy tymczasowej populacji wchodzi w skład nowej populacji. W klasycznym algorytmie genetycznym cała poprzednia populacja chromosomów jest zastępowana przez tak samo liczną nową populację potomków.
- **Wyprowadzenie „najlepszego” chromosomu** – najlepszym rozwiązaniem jest chromosom o największej wartości funkcji przystosowania.

Nową populację tworzą chromosomy powstałe w wyniku selekcji i działania operatorów genetycznych. Nowa populacja zastępuje w całości starą, i staje się bieżącą w kolejnej generacji (iteracji) algorytmu genetycznego.



Rysunek 2. Schemat algorytmu genetycznego

Charakterystyczny dla klasycznego algorytmu genetycznego jest sposób kodowania. Chromosomy są zakodowane binarnie - geny mogą przyjmować tylko wartości 0 i 1. Selekcji dokonuje się z wykorzystaniem *metody ruletki*. Krzyżowanie jest jednopunktowe (rysunek 3) [4].



Metoda ruletki (koła ruletki) – jedna z najbardziej podstawowych metod selekcji. Polega na utworzeniu koła ruletki z polami odpowiadającymi poszczególnym chromosomom. Wielkość pól jest proporcjonalna do wartości funkcji przystosowania. Proces selekcji oparty jest na obrocie ruletką tyle razy ile osobników jest w populacji i na wyborze za każdym razem jednego chromosomu do nowej populacji. Pewne chromosomy są wybierane więcej niż jeden raz, niektóre dokładnie raz, a niektóre wcale [4].

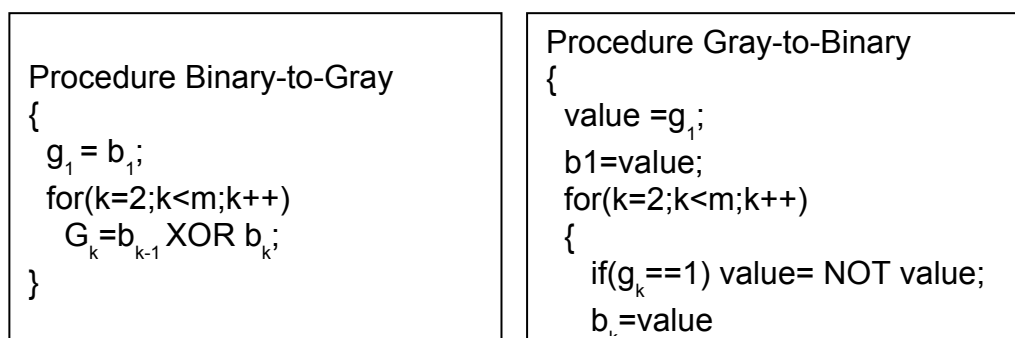
Krzyżowanie jednopunktowe – krzyżowanie zachodzi z pewnym prawdopodobieństwem p_c ; dla każdego osobnika losuje się liczbę i sprawdza, czy zachodzi krzyżowanie. Następnie dobiera się wybrane chromosomy losowo w pary. Losuje się liczbę określającą miejsce krzyżowania i wymienia kod

Mutacja – również zachodzi z pewnym zadanym prawdopodobieństwem. Dla każdego osobnika losuje się liczbę sprawdzając czy będzie on podlegał mutacji. Jeśli tak to losuje się gen, który będzie zmutowany i dokonuje się mutacji, czyli zamiany wartości genu na przeciwny [4]. Mutacja w klasycznym algorytmie genetycznym odgrywa drugorzędną rolę. Częstość mutacji potrzebna do uzyskania dobrych wyników w empirycznych badaniach nad algorytmami genetycznymi jest rzędu jeden do tysiąca skopiowanych bitów. W naturalnych populacjach częstość jest równie mała – lub nawet mniejsza [3].

1.4 Metody kodowania

Kodowanie binarne przyjęto w klasycznym algorytmie genetycznym. Oznacza to, że allele wszystkich genów w chromosomie są równe 0 lub 1. Michalewicz w [4] pisze: „*Reprezentacja rozwiązań w postaci łańcuchów binarnych zdominowała badania nad algorytmami genetycznymi. Kodowanie to ułatwia także analizę teoretyczną i umożliwia wprowadzenie eleganckich operatorów genetycznych.* Stosuje się różne modyfikacje tego sposobu kodowania.

Kod Gray’a jest jedną z alternatyw dla prostego kodowania binarnego. Ma on tę zaletę, że chromosomy odpowiadające dwóm kolejnym liczbom całkowitym różnią się tylko jednym bitem. Na rysunku 4 przedstawiono procedury zamiany kodu binarnego na kod Gray’a i na odwrot.



Rysunek 4. Procedury zmiany kodu binarnego na kod Gray’a i odwrotnie [4]

Kodowanie logarytmiczne jest kolejnym przykładem modyfikacji kodowania binarnego. Ten sposób kodowania jest stosowany celem zmniejszenia długości chromosomów w algorytmie genetycznym. W kodowaniu logarytmicznym pierwszy bit (□) ciągu kodowego oznacza znak funkcji wykładniczej, drugi bit (□) oznacza znak wykładnika funkcji, a pozostałe bity (bin) są reprezentacją wykładnika funkcji [3].

Wprowadzenie zapisu zmiennopozycyjnego można uznać za kolejną modyfikację metod kodowania. Każdy chromosom jest zapisany jako wektor liczb zmiennopozycyjnych. Dokładność takiego podejścia jest zwykle znacznie wyższa niż przy kodowaniu binarnym. Reprezentacja zmiennopozycyjna może objąć całkiem duże lub nawet nieznane dziedziny. W przypadku kodowania zmiennopozycyjnego znacznie łatwiej jest zaprojektować specjalistyczne narzędzia ułatwiające postępowanie w przypadku nie trywialnych ograniczeń [4]. Goldberg w [3] pisze:

„W pewnym sensie wybór kodu dla zadania rozwiązywanego przy użyciu algorytmu genetycznego nie stanowi żadnego problemu, gdyż programista jest ograniczony głównie swoją własną wyobraźnią.(...) Patrząc z innego punktu widzenia, ta swoboda wyboru stanowi wątpliwe błogosławieństwo dla niedoświadczonego użytkownika; widok całych szeregów możliwych sposobów kodowania może zarówno dodawać otuchy, jak i oszalać”.

Jedną z podstawowych zasad przy wyborze kodu jest *zasada minimalnego alfabetu*. Mówi ona [3], że należy wybrać najmniejszy alfabet, w którym rozpatrywane zadanie wyraża się w sposób naturalny. Wynika stąd, że można przyjąć struktury inne niż wektory tj. listy, drzewa, macierze, permutacje. Należy pamiętać, że przy wyborze metody kodowania ważne jest, aby zakodowane osobniki były możliwie podobne do rzeczywistych rozwiązań oraz, że do wybranej metody należy zaprojektować odpowiednio specjalistyczne operatory.

1.5 Metody selekcji

W algorytmach genetycznych wyróżniamy etap selekcji, który z bieżącej populacji osobników wybiera te o największej wartości funkcji przystosowania do populacji rodzicielskiej.

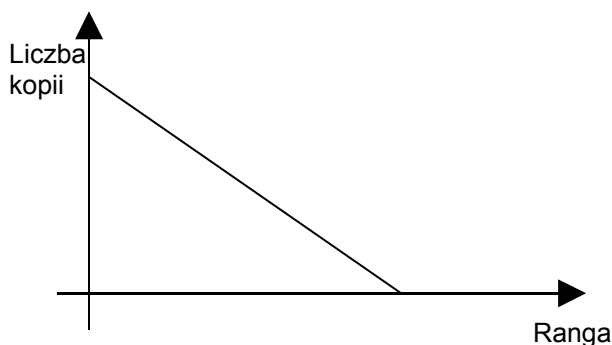
Selekcję metodą koła ruletki omówiono przy okazji klasycznego algorytmu genetycznego (rozdz.1.3). Jest to metoda prosta i daje w miarę zadowalające wyniki. Ma jednak wady.

- Jedną z nich jest możliwość stosowania metody ruletki tylko do jednej klasy zadań, tzn. tylko do maksymalizacji (lub tylko do minimalizacji). Takie ograniczenie jest niewątpliwie jej wadą [6].
- Bardziej istotną wadą metody ruletki jest zbyt wczesne eliminowanie z populacji osobników o bardzo małej wartości funkcji przystosowania. Może to prowadzić do zbyt wczesnej zbieżności algorytmu [2].

Dlatego stosuje się inne metody selekcji takie jak: selekcja turniejowa, rankingowa i strategia elitarna.

Selekcja turniejowa polega na podzieleniu populacji na podgrupy k -elementowe (k to rozmiar turnieju – zwykle 2 lub 3) i wyborze z każdej podgrupy osobnika o najlepszym przystosowaniu. Można to zrobić poprzez wybór losowy lub wybór deterministyczny. Wtedy wyboru dokonuje się z prawdopodobieństwem równym 1. Metoda turniejowa nadaje się zarówno do problemów maksymalizacji jak i minimalizacji.

W selekcji rankingowej osobniki populacji są ustawiane kolejno w zależności od funkcji przystosowania – tzn. od najlepiej do najgorzej przystosowanego. Każdemu osobnikowi przyporządkowana jest liczba zwana rangą i oznaczająca jego pozycję na liście. Liczba kopii danego osobnika wprowadzonych do nowej populacji jest ustalana na podstawie wcześniej zdefiniowanej funkcji, zależnej od rangi (Rysunek 5) [2].



Rysunek 5. Przykład funkcji określającej zależność liczby osobników w nowej populacji od wartości rangi [2]

W *strategii elitarnej* nacisk położony jest na zachowanie w kolejnych iteracjach najlepiej przystosowanych osobników. W klasycznym algorytmie genetycznym możliwa jest sytuacja, w której do nowej populacji nie wejdą najlepsze osobniki. Model elitarny ma za zadanie do tego nie dopuścić [4].

Wymienione powyżej metody selekcji są pewnymi podstawowymi kanonami. Michalewicz w [4] opisał wiele modyfikacji podstawowych metod selekcji. Ich celem była poprawa selekcji dla konkretnego zadania.

Metody selekcji można podzielić na *statyczne* i *dynamiczne*. Statyczna selekcja oznacza, że prawdopodobieństwa wyboru są stałe dla wszystkich pokoleń, w selekcji dynamicznej natomiast takiego wymagania nie ma.

Inny podział metod selekcji wyróżnia metody *wygaszające* i *zachowujące*. W selekcji zachowującej są niezerowe prawdopodobieństwa wyboru dla każdego osobnika. Natomiast w selekcji wygaszającej nie istnieją. Selekcje *wygaszające* można podzielić na *lewe* i *prawe*. W selekcji wygaszającej lewej nie dopuszcza się najlepszych osobników do reprodukcji, aby uniknąć przedwczesnej zbieżności. W prawej tego takiej zasady nie ma.

Niektóre metody selekcji można nazwać jako *wylączne*, co oznacza, że rodzice mogą reprodukować tylko w jednym pokoleniu –tak więc czas życia osobnika jest ograniczony do jednego pokolenia. Można wyróżnić selekcje *pokoleniowe* i *w locie*. W selekcji pokoleniowej zbiór rodziców jest stały dopóki nie zostanie utworzona nowa populacja i dopiero wtedy następuje zamiana. Natomiast w selekcji w locie potomek wymienia rodzica natychmiast po utworzeniu [4].

Skalowanie funkcji celu stosuje się, aby zapobiec przedwczesnej zbieżności algorytmu genetycznego. Przedwczesna zbieżność algorytmu polega na tym, że najlepsze ale jeszcze nie optymalne chromosomy dominują w populacji.

Ponadto w końcowej fazie algorytmu, w przypadku, gdy populacja zachowuje znaczną różnorodność, średnia wartość przystosowania niewiele odbiega od maksymalnej. Skalowanie funkcji przystosowania może wówczas zapobiec takiej sytuacji, że osobniki przeciętne i najlepsze otrzymują prawie taką samą liczbę potomków w następnych generacjach, co jest zjawiskiem niepożądanym.

Skalowanie funkcji przystosowania chroni populację przed dominacją nieoptymalnego chromosomu. Skalowanie polega na odpowiednim przekształceniu funkcji przystosowania. Podstawowe metody skalowania to: skalowanie liniowe, skalowanie \square -obcinające typu sigma i skalowanie potęgą [3][6].

- *Skalowanie liniowe* (ang. *linear scalling*) polega na przekształceniu funkcji przystosowania f do postaci f' poprzez przekształcenie liniowe według wzoru:

$$f' = af + b$$

Współczynniki a i b powinny: zachować niezmienną wartość średniego przystosowania oraz ustalić maksymalną wartość wyskalowanego przystosowania na poziomie określonej krotności średniego przystosowania. Te dwa warunki łącznie zapewniają, że przeciętny osobnik populacji otrzymuje przeciętnie jednego potomka, a najlepszy tyłu – ile wynosi wspomniana krotność.

Skalowanie liniowe działa dobrze, za wyjątkiem, gdy funkcja f' przyjmuje wartości ujemne.

- *Skalowanie \square -obcinające typu sigma* (ang. *sigma truncation*). Przekształcenie funkcji przystosowania f do postaci f' dokonuje się według następującej zależności:

$$f' = \bar{f} + (f - c \cdot \sigma)$$

gdzie: \bar{f} jest średnią wartością funkcji przystosowania w populacji, c jest małą liczbą naturalną (zwykle 1 do 5) [6], σ jest odchyleniem standardowym w populacji. Wartości ujemne f' przyjmuje się jako równe zero.

- *Skalowanie potęgą* (ang. *power law scaling*), polega na podniesieniu pierwotnej funkcji przystosowania f do pewnej ustalonej potęgi k :

$$f' = f^k$$

Wartość k jest liczbą bliska 1 i zależy od rozpatrywanego problemu. Może wymagać zmiany w kolejnych generacjach – aby doprowadzić do „rozciągnięcia” lub „ściągnięcia” zakresu, zależnie od potrzeby [3], [6].

1.6 Operatory genetyczne

Następnym po etapie selekcji, jest etap nazywany też *ewolucją*. Polega on stosowaniu operatorów genetycznych, tzn. krzyżowania i mutacji, które dokonują rekombinacji genów w chromosomach.

1.6.1 Operator krzyżowania

Operacja krzyżowania polega na wymianie fragmentów łańcuchów dwóch chromosomów rodzicielskich. Krzyżowanie jest kluczowym operatorem w algorytmach genetycznych, stanowiącym o ich sile i efektywności. Mutacja odgrywa znacznie mniejszą rolę [6].

Ideą operatorów krzyżowania jest wymiana kodu genetycznego pomiędzy osobnikami, tak jak to się dzieje w naturze. Stworzono wiele teorii i rodzajów krzyżowań, które stosowane są do różnych rodzajów zadań i są zależne od sposobu kodowania. Dla potrzeb klasycznego algorytmu genetycznego opisano operator krzyżowania jednopunktowego.

Stosuje się również inne rodzaje krzyżowania: dwupunktowe (ang. *two-point crossover*), wielopunktowe (ang. *multi-point crossover*), równomierne (ang. *uniform crossover*) [6]. Krzyżowanie wielopunktowe można stosować, gdy chromosomy są zakodowane zarówno jako wektory binarne jak i zmiennopozycyjne. Dwie inne popularne metody krzyżowania: *krzyżowanie arytmetyczne* i *heurystyczne*.

Krzyżowanie arytmetyczne. Jest to dwuargumentowy operator, zdefiniowany jako liniowa kombinacja dwóch wektorów. Jeżeli do krzyżowania zostały wybrane wektory (chromosomy) x_1 i x_2 , to potomkowie są wyznaczani następująco:

$$\begin{aligned}x_1' &= a x_1 + (1-a) x_2 \\x_2' &= a x_2 + (1-a) x_1.\end{aligned}$$

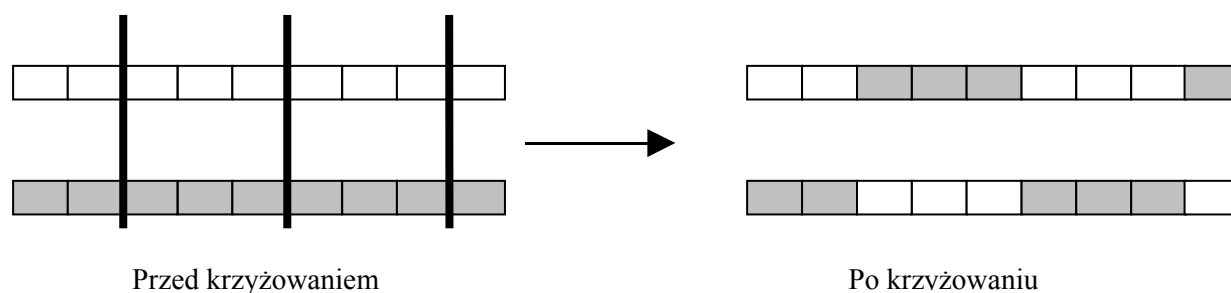
gdzie: a jest wartością przedziału $[0,1]$, co gwarantuje, że potomkami są rozwiązania dopuszczalne.

Krzyżowanie heurystyczne. Jest to operator do kierunku poszukiwań używa wartości funkcji celu. Tworzy tylko jednego potomka i może w ogóle nie utworzyć potomka [4]. Nowy osobnik jest określany następująco:

$$ch_3 = r(ch_2 - ch_1) + ch_2$$

gdzie: r jest wartością losową z przedziału $[0,1]$ i $F(ch_2) \square F(ch_1)$ [6]

Na rysunku 6 pokazano schemat krzyżowania wielopunktowego dla trzech punktów.



Rysunek 6. Schemat działania krzyżowania trzy punktowego

1.6.2 Operator mutacji

Mutacja polega na wprowadzeniu do istniejących zakodowanych chromosomów, pewnych losowych zmian. Mutacja tworzy nowego osobnika na bazie jednego i tylko jednego rodzica. Jest wiele metod tworzenia nowych osobników u operatora mutacji.

Podstawową formę mutacji można zapisać następująco:

$$x' = m(x)$$

gdzie: x jest chromosomem rodzica, m funkcją mutacji, x' chromosomem potomka [3].

W przypadku chromosomów zakodowanych binarnie (klasyczny algorytm genetyczny) nie ma problemu ze stosowaniem mutacji (po prostu zamieniamy wartość genu na przeciwny).

W przypadku wartości zmiennopozycyjnych nie jest to już takie oczywiste. Poniżej przedstawiono kilka sposobów implementacji operatora mutacji.

Mutacja równomierna. Operator ten działa na jednym rodzicu x tworzy potomka x' . Operator wybiera losowo pozycję genu, który zostanie mutowany $k \in (1, \dots, q)$. Mutowany wektor $x = (x_1, \dots, x_k, \dots, x_q)$ przekształca się w nowy $x' = (x_1, \dots, x_k', \dots, x_q)$, gdzie x_k' przyjmują wartość losową z dozwolonego przedziału dla tej wartości. Operator ten odgrywa szczególnie ważną rolę we wczesnych fazach procesu ewolucyjnego [4].

Mutacja brzegowa. Działa na podobnych zasadach, co mutacja równomierna. Natomiast x_k' może z jednakowym prawdopodobieństwem przyjąć wartości brzegów dozwolonego przedziału. Operator ten jest przydatny dla zadań optymalizacji, gdy poszukiwane rozwiązanie leży blisko brzegu dopuszczalnej przestrzeni poszukiwań [4].

Mutacja nierównomierna. Jest stosowana głównie do dokładnego dostrojenia się systemu. Jeżeli do mutacji wybrano element x_k rodzica x , to x_k' może wynosić:

$$x_k' = \begin{cases} x_k + \Delta(t, P(k) - x_k) & \text{jeżeli losową liczbą jest } 0 \\ x_k - \Delta(t, x_k - L(k)) & \text{jeżeli losową liczbą jest } 1 \end{cases}$$

Funkcja $\Delta(t, y)$ przyjmuje wartości z przedziału $[0, y]$ – prawdopodobieństwo, że ta wartość jest bliska 0, gdy t wzrasta (t jest numerem pokolenia). $P(k)$ jest wartością prawego brzegu przedziału, natomiast $L(k)$ lewego.

Δ zdefiniowane jest w następujący sposób:

$$\Delta(t, y) = y(1 - r^{(1-t/T)b})$$

gdzie: r jest liczbą losową z przedziału $[0, 1]$, T jest najwyższym numerem pokolenia, b jest parametrem systemu określającym stopień niejednorodności [4].

Operacja mutacja jest niezbędna, ponieważ operacje krzyżowania mogą okazać się zbyt nadgorliwe i wyeliminować potencjalnie obiecujący materiał genetyczny. Sama w sobie mutacja jest błędzeniem przypadkowym w przestrzeni ciągów kodowych. Stosowana oszczędnie jako dodatek do pozostałych operatorów, stanowi swego rodzaju polisę na wypadek utraty ważnych składników rozwiązania.

1.6.2 Operator inwersji

Operator inwersji. Jest to operator, którego nie można zakwalifikować ani jako operator krzyżowania ani jako operator mutacji. Operuje on na istniejącym chromosomie i wszystkie geny pochodzą z istniejącego chromosomu – nie ma żadnej „dzikiej karty”.

Operacja inwersji jest głównym mechanizmem odpowiedzialnym za rekonfigurację kodu. Podczas inwersji chromosom ulega przecięciu w dwóch wybranych punktach, a następnie środkowy jego odcinek ulega odwróceniu i połączeniu z dwoma pozostałymi. Operator inwersji działa przeciwko wstrzymującemu przeszukiwaniu brakowi różnorodności [6].

Program Evol

Program Evol jest prostym narzędziem do wizualizacji i analizy algorytmu genetycznego zastosowanego do rozwiązania problemu optymalizacji funkcji jednowymiarowej. Umożliwia szukanie zarówno minimum jak i maksimum. Obecnie zaimplementowane elementy:

- wszystkie elementy **klasycznego** algorytmu genetycznego, tj. kodowanie binarne, selekcja za pomocą metody koła ruletki, krzyżowanie jednopunktowe, mutacja jednopunktowa.
- dodatkowa metoda selekcji – metoda rankingowa
- prezentacja ewolucji populacji rozwiązań na wykresie funkcji
- prezentacja najlepszych, średnich oraz najgorszych rozwiązań w formie wykresu
- dziesięć funkcji jednowymiarowych wielomodalnych, tj. o wielu minimach (maksimach)
 - $y = \sin(2*x) + \log(x^2)$ w przedziale [1 ; 15]
 - $y = \sin(2*x) + (\cos(4*x))^3$ w przedziale [-1 ; 1]
 - $y = x^2 + (\cos(4*x))^3$ w przedziale [-1.5 ; 2.5]
 - $y = \log(x^3) - \log(x^5) + (\cos(4*x))^3$ w przedziale [5 ; 12]
 - $y = x + \sin(3*\cos(5*x))$ w przedziale [0 ; 2]
 - $y = x^{1/3} + \sin(5*x)$ w przedziale [1.5 ; 5]
 - $y = 0.5*x + \sin(x^3)$ w przedziale [0 ; 4]
 - $y = x^3 - x^2 + \sin(4*x) - \cos(15*x)$ w przedziale [-1 ; 1]
 - $y = -x^2 - x + \cos(10*x) - \sin(50*x)$ w przedziale [-1 ; 1]
 - $y = -x + \sin(5*x) - \cos(13*x)$ w przedziale [-1 ; 1]

Poniżej przedstawiony jest główny panel programu umożliwiający ustawienie szeregu parametrów algorytmu.

Populacja Binarna

Parametry

Populacja

Rozmiar populacji: 20

Liczba bitów na wymiar: 15

Selekcja (turniejowa)

Ruletka (v) Rozmiar turnieju: 3

Krzyżowanie (jednopunktowe)

Prawdopodobieństwo krzyżowania: 0.7

Mutacja (jednopunktowa)

Prawdopodobieństwo mutacji: 0.1

Ewolucja

Liczba iteracji: 300

Szukaj minimum

Create

Start

Włącz opóźnienie

Funkcja

funkcja 1 (v)

Minimum: 1.0

Maximum: 15.0

Rysuj

Funkcję i populację

Statystyki

Najlepszy w ogóle

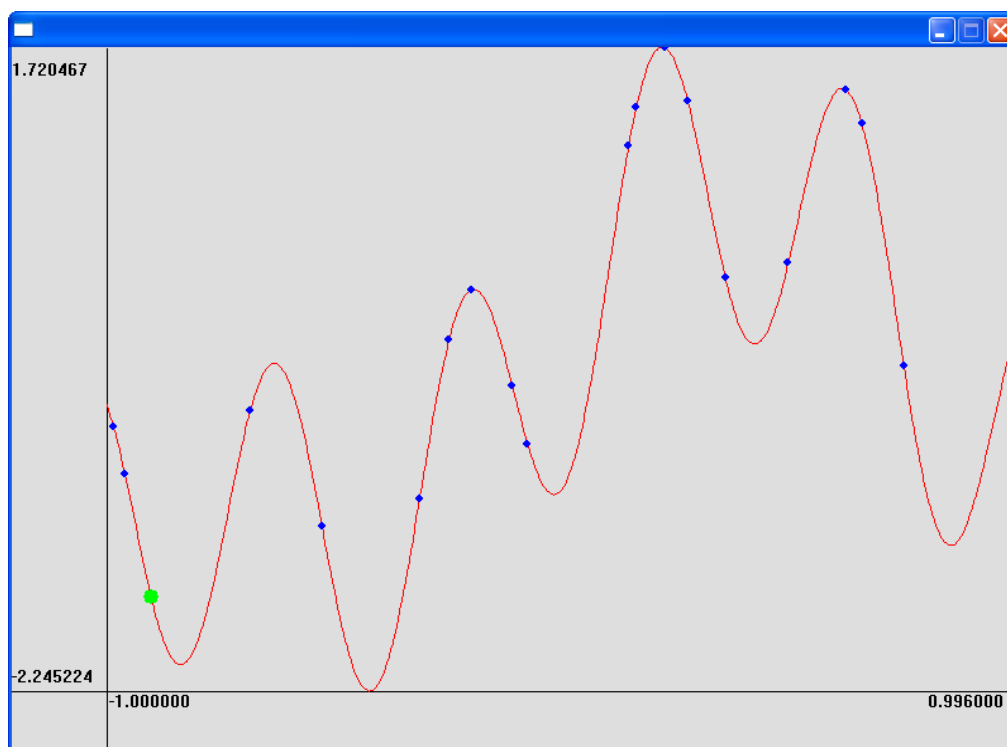
Najlepszy w generacji

Najgorszy w generacji

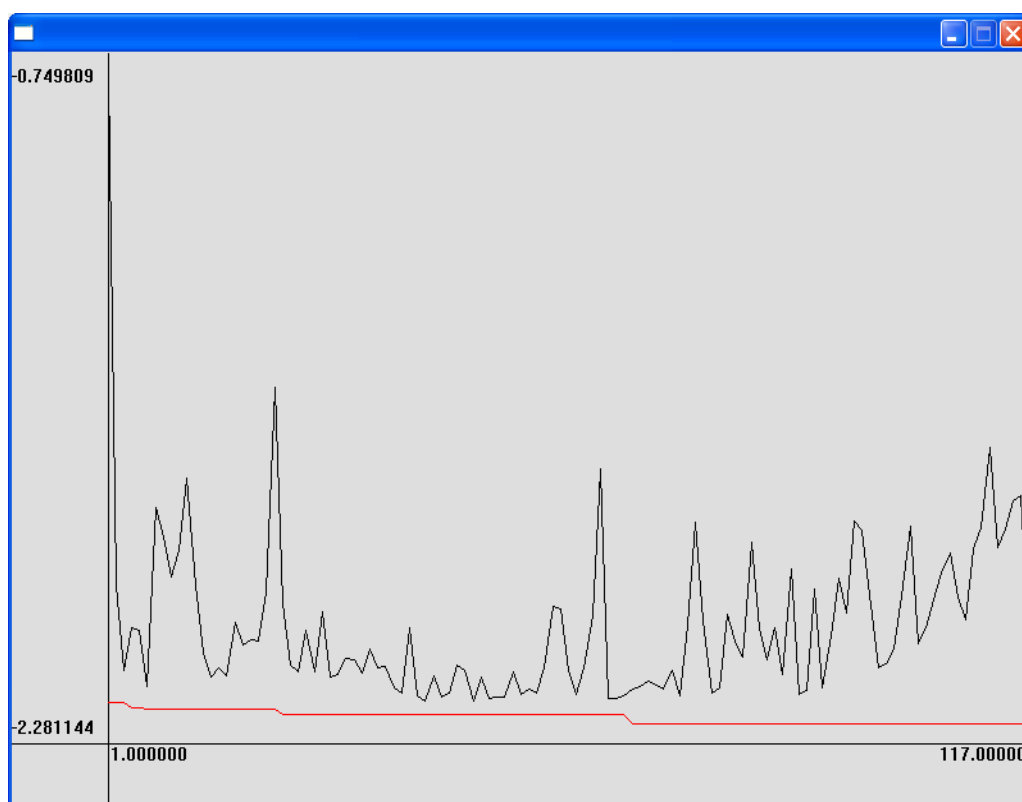
Średni w generacji

Zapisz wykres

Główny panel programu Evol.



Wykres funkcji oraz populacji. Na zielono zaznaczony aktualny najlepszy osobnik.



Okno wykresu zmiany jakości osobników w kolejnych iteracjach.

Zadania do wykonania

Pracując z funkcjami wskazanymi przez prowadzącego przeprowadź symulacje potrzebne do opracowania odpowiedzi na poniższe pytania, jak również ich uzasadnienia i wizualizacji.

1. Sprawdź jaki wpływ na działanie algorytmu mają wartości parametrów:
 - wielkość populacji (4, 10, 20, 40, 100)
 - prawdopodobieństwo krzyżowania (0.7, 1, 0.5, 0.1)
 - prawdopodobieństwo mutacji (0.1, 0.5, 0.7)
 - wybrana metoda selekcji
 - liczba iteracji
2. Jaki wpływ na dokładność znalezionego rozwiązania ma liczba bitów, którymi opisany jest jeden osobnik? Podaj dokładny wzór na maksymalną dokładność znalezionego rozwiązania w zależności od liczby bitów przypadających w danym osobniku na kodowanie jednego wymiaru.
3. Jak przebiega ewolucja przy wyłączonej mutacji (prawdopodobieństwo równe 0)? Czy populacja równie łatwo zawsze znajduje rozwiązanie?
4. Jak przebiega ewolucja przy wyłączonym krzyżowaniu (prawdopodobieństwo równe 0)? Czy populacja równie łatwo zawsze znajduje rozwiązanie?
5. Co się stanie jeśli wyłączona zostanie zarówno mutacja jak i krzyżowanie (działa tylko selekcja)? Od czego zależy wtedy znalezione rozwiązanie?
6. Jaki wpływ na końcowy wynik i ewolucję ma rozmiar turnieju w przypadku selekcji turniejowej? Rozważ przypadek, gdy rozmiar turnieju jest większy niż rozmiar populacji (losowanie ze zwracaniem).
7. Co można odczytać z wykresów uzyskanych z programu **Evol**? Zwróć szczególną uwagę na wykres średniej wartości optymalizowanej funkcji w kolejnych iteracjach (generacjach).

Wnioski i spostrzeżenia powinny być zobrazowane uzyskanymi wykresami jak i wykresami wykonanymi samodzielnie. Jako że efekty uzyskane przez tego typu algorytmy w dużej mierze zależą od początkowej losowej populacji, przeprowadź obliczenia dla tych samych ustawień kilkakrotnie i wnioski oprzyj na uśrednionych wynikach (co najmniej 5 przebiegów dla każdego ustawienia). Przedstaw te uśrednione wyniki w postaci tabel oraz wykonanych własnoręcznie wykresów. Przeprowadź obliczenia dla minimalizacji jak i maksymalizacji funkcji.

Przykładowe pytania do zaliczenia laboratorium:

1. Określ zasady działania algorytmu genetycznego.
2. Czym charakteryzuje się klasyczny algorytm genetyczny?
3. Wyjaśnij zasady selekcji metodą koła ruletki
4. Na czym polega krzyżowanie jednopunktowe?
5. Jakie są najważniejsze parametry algorytmu genetycznego wpływające na jego działanie?
6. Wyjaśnij pojęcia: chromosom, gen, populacja.
7. Podaj różnice pomiędzy kodowaniem binarnym a rzeczywistym.

Literatura:

- [1] <http://www.shef.ac.uk/acse/research/ecrg/gat.html> pakiet *Genetic Algorithm Toolbox (GA-Toolbox)*.
- [2] Arabas J., *Wykłady z algorytmów ewolucyjnych*, WNT, Warszawa 2001
- [3] Goldberg D.E., *Algorytmy genetyczne i ich zastosowania*, WNT, Warszawa 1998
- [4] Michalewicz Z., *Algorytmy genetyczne+struktury danych=programy ewolucyjne*, WNT, Warszawa 1999
- [5] Osowski S., *Sieci neuronowe w ujęciu algorytmicznym*, WNT, Warszawa 1997
- [6] Rutkowska D., Piliński M., Rutkowski L., *Sieci neuronowe, algorytmy genetyczne i systemy rozmyte*, Wydawnictwo Naukowe PWN, Warszawa 1999
- [7] Press W.H., Teukolsky S.A., Vetterling W.T., Flannery B.P., *Numerical Recipes*, Cambridge University Press, Cambridge 1992