Akademia Górniczo-Hutnicza
Wydział EAIIB
Informatyka

SZUKANIE MIEJSC ZEROWYCH FUNKCJI WIELU ZMIENNYCH ZA POMOCĄ ALGORYTMÓW GENETYCZNYCH

Autorzy:
Kamil Dawidów
Jakub Freisler
Beata Giełbaga

Spis treści

1 Zarys problemu

Wybrany przez nas temat dotyczy poszukiwania miejsc zerowych funkcji wielu zmiennych za pomocą algorytmu genetycznego, dlatego w pierwszej kolejności zajmiemy się rozwinięciem czym jest algorytm genetyczny.

1.1 Algorytm ewolucyjny - algorytm genetyczny

Cechą wspólną algorytmów ewolucyjnych jest wykorzystanie schematu działania wzorowanego na teorii doboru naturalnego i ewolucji. Jest to więc kolejna – po sieciach neuronowych – technik inspirowana analogiami biologicznymi. Główne pole zastosowań algorytmów ewolucyjnych to problemy optymalizacji – jak wiemy, wiele problemów praktycznych można przedstawić w języku problemów optymalizacyjnych; większość z nich można w związku z tym próbować rozwiązać ewolucyjnie. **Algorytm genetyczny** – rodzaj algorytmu przeszukującego przestrzeń alternatywnych rozwiązań problemu w celu wyszukania rozwiązań najlepszych. Postawiony problem definiuje środowisko, w którym istnieje pewna populacja osobników. Każdy z osobników ma przypisany pewien zbiór informacji stanowiących jego genotyp, a będących podstawą do utworzenia fenotypu. Fenotyp to zbiór cech podlegających ocenie przez funkcję oceny. Innymi słowy - genotyp opisuje proponowane rozwiązanie problemu, a funkcja przystosowania ocenia, jak dobre jest to rozwiązanie.

Genotyp składa się z chromosomów, gdzie zakodowany jest fenotyp i ewentualnie pewne informacje pomocnicze dla algorytmu genetycznego. Chromosom składa się z genów.

Działanie algorytmu genetycznego zawiera następujące kroki:

1. inicjalizacja - utworzenie populacji i losowe nadanie wartości genom jej osobnikom
2. ocena osobników - obliczanie wartości funkcji oceny dla każdego osobnika
3. selekcja - wybór osobników o najlepszych genach, najlepiej przystosowane osobniki biorą udział w reprodukcji następnego pokolenia
4. ewolucja - zastosowanie operatorów genetycznych, krzyżowanie, czyli wymiana genów pomiędzy dwoma rozwiązaniami funkcji oraz mutacja, czyli wprowadzanie drobnych losowych zmian do genu
5. nowa populacja - osobniki z ewolucji tworzą nowe pokolenie, jeśli nie znaleziona dostatecznie dobrego rozwiązania, algorytm wraca do punktu drugiego. W przeciwnym wypadku wybierany jest najlepszy osobnik z populacji i jego genotyp to uzyskany wynik

Wszystkie te elementy zostaną szczegółowo omówione w dalszych rozdziałach.

2 Adaptacja algorytmu

2.1 Losowanie populacji

Na tym etapie należy wybrać początkowy zakres populacji. Jeśli populacja będzie zawierała zbyt mało osobników to algorytm może zatrzymać się w jakimś minimum lokalnym. Z drugiej strony zbyt duża liczebność populacji spowalnia działanie algorytmu. Liczba osobników jest ustalana przez użytkownika. Po określeniu wielkości populacji, należy utworzyć wszystkich osobników. Ważne jest aby pierwsze pokolenie było jak najbardziej różnorodne. W związku z czym ich geny, czyli w tym przypadku lista list wartości funkcji, są generowane losowo z zakresu również podanego przez użytkownika. Użytkownik ma możliwość ustalenia parametrów populacji dzięki czemu może badać skuteczność algorytmu w odniesieniu do różnych parametrów.

2.2 Funkcja oceny

Funkcja oceny to miara jakości dowolnego osobnika w populacji. Dla naszego problemu miarą jakości będzie po prostu wartość funkcji dla zadanych parametrów.

2.3 Metoda selekcji

W tym miejscu tworzona jest nowa populacja. Jest to podstawa selekcji naturalnej w genetyce, ponieważ eliminujemy "słabe" osobniki, a z "dobrych" tworzymy nowe pokolenie, które daje lepsze rozwiązania. Aby uniknąć spadku różnorodności genotypu populacji każdego osobnika krzyżujemy losowo z innymi 5 razy. ProbabilityOfSurvival ustalone jest na 0.2, nowe pokolenie będzie posiadać 5 razy więcej osobników. Następnie dla tych osobników wyliczamy wartość funkcji przystosowania. Do nowego pokolenia wybieramy 0.2 osobników z najlepszymi wynikami.

2.4 Operatory kodowania

2.4.1 Krzyżowanie

Zadaniem krzyżowania jest wymiana "materiału genetycznego" pomiędzy dwoma osobnikami w populacji. Polega ono na połączeniu niektórych genów, wybranych od rodziców, w jeden nowy genotyp. Genotyp dziecka tworzony jest przez przydzielenie kolejnych bitów losowo od jednego z rodziców, których genotyp został zapisany w postaci binarnej. W wyniku krzyżowania powstaje tylko jeden potomek.

2.4.2 Mutacja

Mutacja podobnie jak krzyżowanie ma za zadanie wprowadzić zmiany do genotypu losowo wybranych osobników. Dzięki czemu zostanie zachowana różnorodność następnego pokolenia. Każdy

osobnik ma 10% szansy na to że ulegnie mutacji. Geny osobnik, który ulega mutacji, są losowo mnożone przez +-(1 + współczynnik mutacji).

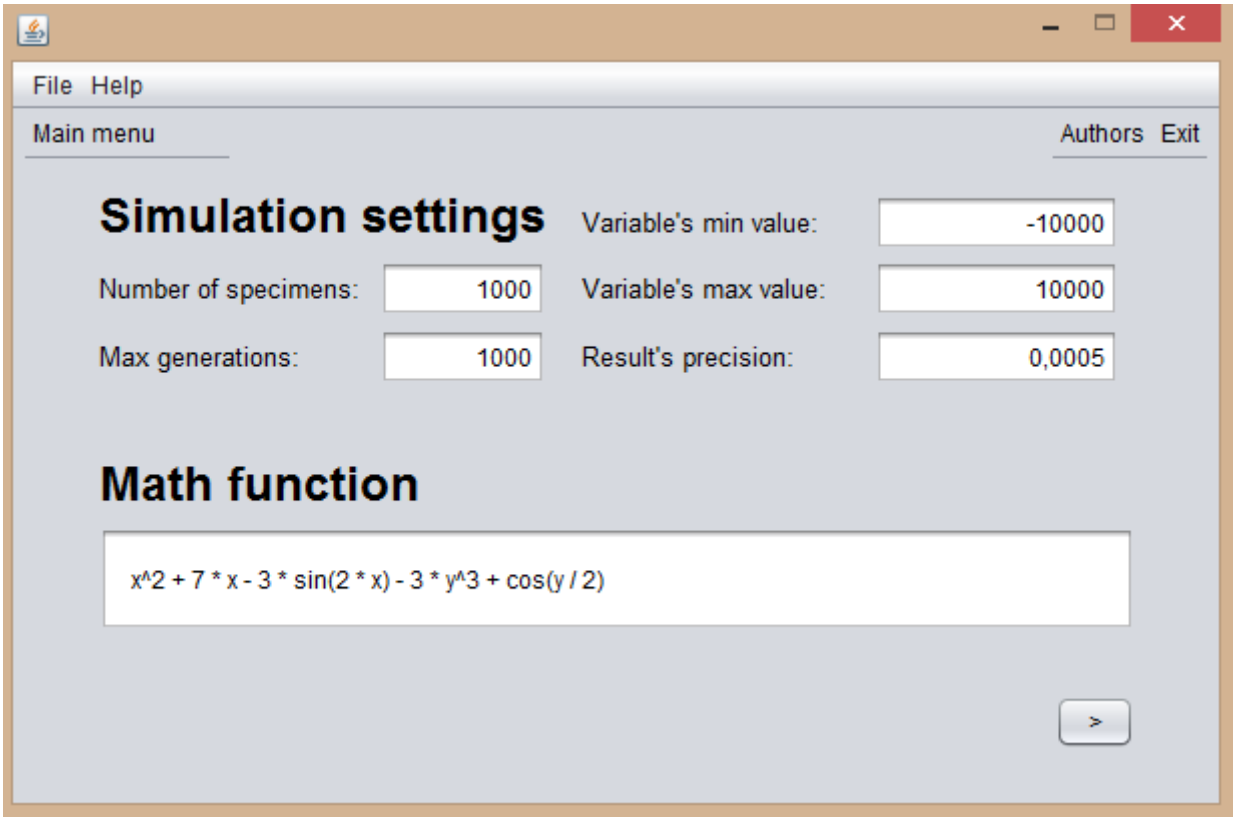
3 Program

Stworzony przez nas program został napisany w JAVIE. Do odczytywania funkcji podanej przez użytkownika została użyta biblioteka exp4j. Do sporządzenia wykresu postępu skorzystaliśmy z biblioteki JFreeChart.

Program służy do wyszukania miejsca zerowego funkcji wielu zmiennych oraz przedstawienia postępu poszukiwania na wykresie. Program umożliwia użytkownikowi zmianę parametrów, co pozwala skutecznie wpływać na przebieg rozwiązania, a także badać sam proces rozwiązywania pod kątem różnych argumentów. Poniżej został opisany interfejs programu, możliwe parametry wejściowe, przebieg pojedynczej iteracji algorytmu, a także zaprezentowany został przykład generowanego wykresu.

3.1 Interfejs programu

DO UZUPEŁNIENIA!!



3.2 Parametry wejściowe programu

Ustawienia, które użytkownik może zmieniać aby otrzymać zadowalający go wynik:

Number of specimens
Ustala ile osobników ma być w pokoleniach. Im więcej osobników, tym dłużej będzie trwać wyszukiwanie miejsca zerowego, jednak uzyskane wyniki powinny być lepsze.

Max generations
Ustala jaka ma być maksymalna liczba pokoleń, dzięki czemu unikniemy zapętleniu się program. Podanie za małej ilości pokoleń może skutkować, tym, że program nie znajdzie miejsca zerowego po osiągnięciu maksymalnego pokolenia. Program zakończy działanie jeśli znajdzie wartość funkcji mniejszą od podanego przez użytkownika epsilona.

Variable's min/max value
Ustala górną i dolną granice szukanych wartości. Podanie złego zakresu może spowodować, że program nie znajdzie miejsca zerowego w żądanym zakresie.

Result's precision
Ustala dokładność z jaką szukamy miejsca zerowego.

3.3 Wprowadzenie funkcji

W polu "Math function" użytkownik może wprowadzić funkcje wieluzmiennych, dla której ma być znalezione miejsce zerowe. Została do tego użyta biblioteka exp4j, która za pomocą algorytmu Shunting-yardv Dijkstra tłumaczy wyrażenia z notacji infiksowej na odwrotną notacje polską (postfiksową) i oblicza wynik za pomocą prostej kolejki lifo.

Dzięki tej bibliotece program może wyliczać miejsce zerowe funkcji trygonometrycznych, cyklometrycznych, logarytmicznych, jak również funkcje zawierające stała π oraz e

Lista wspieranych operatorów:

- dodawanie: 2 + 2
- odejmowanie: 2 - 2
- mnożenie: 2 * 2
- dzielenie: 2 / 2
- potęgowanie: 2^2
- dzielenie modulo: 2 % 2

Lista wspieranych funkcji:

- wartość bezwzględna: abs
- arcus cosinus: acos
- arcus sinus: asin
- arcus tangens: atan
- pierwiastek sześcienny: cbrt
- najmniejsza liczba całkowita, większa lub równa: ceil
- cosinus: cos
- cosinus hiperboliczny: cosh
- potęgowanie e: exp
- największa liczba całkowita mniejsza lub równa: floor
- logarytm naturalny: log
- logarytm o podstawie 2: log2
- logarytm o podstawie 10: log10
- sinus: sin
- sinus hiperboliczny: sinh
- pierwiastek kwadratowy: sqrt
- tangens: tan
- tangens hiperboliczny: tanh

3.4 Pojedyncza iteracja

3.5 Wykres