Jan Ignatowicz

Daniel Pogrebniak

Daniel Wojdak

**Sprawozdanie do projektu z**

**Podstaw sztucznej inteligencji**

1. **Treść zadania**

Celem projektu jest napisanie aplikacji, która umożliwi znalezienie minimum funkcji Rosenbrocka

na przedziale,

za pomocą algorytmu ewolucyjnego.

1. **Podjęte decyzje projektowe**

Pierwszą decyzją, którą podjęliśmy był wybór języka programowania, w którym napiszemy program. Wybraliśmy język Java, ponieważ zależało nam, aby program był w pełni konfigurowalny. Pozwoliło nam to na przetestowanie algorytmu z wieloma różnymi parametrami. Kolejną ważną decyzją był wybór zastosowanego algorytmu. Wybraliśmy algorytm μ + λ, który jest jednym z najpopularniejszych algorytmów ewolucyjnych. W przy tworzeniu potomków korzystaliśmy ze strategii elitarnej, ponieważ wiemy, że funkcja Rosenbrocka ma tylko 1 minimum globalne i żadnych minimów lokalnych. Pozwoliło nam to na zwiększenie efektywności algorytmu. Dalsze decyzje nie były już tak oczywiste, bo zależały w głównej mierze od tego co w danym eksperymencie chcieliśmy sprawdzić. Nasz program pozwala użytkownikowi samodzielnie podawać parametry, więc testowaliśmy go na różnych konfiguracjach. Zachęcamy użytkownika do samodzielnego testowania programu z własnymi parametrami. W dalszej części sprawozdania opiszemy jedynie wyniki otrzymane dla jednego przykładowego zestawu danych.

Wybrane przez nas wartości liczbowe:

* rozmiar populacji: μ = 100
* ilość potomków w każdej iteracji: λ = 200
* początkowa σ dla każdego osobnika: σ = 0,5
* współczynnik interpolacji = 0,5
* procent populacji, który zostanie zmutowany ?przed? reprodukcją = 5%
* algorytm zostanie zatrzymany jeśli przez 10 iteracji nie będzie poprawy wartości funkcji celu najlepszego osobnika o ε = 0,001 lub zostanie przekroczony limit 1000 iteracji

Wartości na czerwono należy prawdopodobnie zwiększyć, a na zielono zmniejszyć.

1. **Pseudokod algorytmu**
2. **Instrukcja do programu**

Po uruchomieniu dobieramy parametry potrzebne do poprawnego wykonywania się algorytmu.

* μ - Rozmiar populacji
* λ – ilość generowanych potomków w każdej iteracji
* rodzaj wyboru – Sposób wybierania osobników po reprodukcji
* min x – Minimalna wartość współrzędnej x dopuszczalna w osobnikach.
* min y – Minimalna wartość współrzędnej y dopuszczalna w osobnikach.
* rodzaj optymalizacji – cel przeszukiwania : minimum w zakresie, maksimum w zakresie, poszukiwanie konkretnej wartości.
* max x – Maksymalna wartość współrzędnej x dopuszczalna w osobnikach
* max y – Maksymalna wartość współrzędnej y dopuszczalna w osobnikach
* cel optymalizacji – wartość do jakiej ma dążyć algorytm po wybraniu w polu rodzaj optymalizacji „do wartości”
* σ x – współczynnik mutacji dla współrzędnej x
* σ y – współczynnik mutacji dla współrzędnej y
* procent mutacji – procent osobników mutowanych w każdej iteracji algorytmu
* ε – minimalna zmiana wartości osobnika uważana za zmianę znaczącą
* max iteracji – maksymalna ilość iteracji algorytmu, jeżeli zostanie przekroczona algorytm zatrzyma działanie
* max bez poprawy – maksymalna dopuszczalna ilość iteracji bez poprawy najlepszego osobnika o co najmniej ε , jeżeli zostanie przekroczona algorytm zatrzyma działanie
* wsp. interpolacji – Współczynnik interpolacji lepszego osobnika względem gorszego
* rodzaj algorytmu – Wybór algorytmu pomiędzy „μ + λ” a „μ, λ”
* ilość wyświetlanych – Ilość najlepszych wyświetlanych osobników w każdej iteracji.

Przyciski:

* Inicjuj – Wprowadzanie ustawionych parametrów do klasy implementującej algorytm
* 1 krok algorytmu – przejście jednej iteracji algorytmu
* Znajdź rozwiązanie – rozpoczęcie wyszukiwania rozwiązania do czasu osiągnięcia warunków końca
* Zatrzymaj rozwiązywanie – zatrzymanie wyszukiwania rozwiązania
* Rozpocznij od nowa – wyczyszczenie aktualnej instancji rozwiązania, powrót do wprowadzania parametrów
* Wyczyść konsole – czyszczenie zawartości konsoli

1. **Opis struktury programu**

Program składa się z klas głównych: Widok, Kontroler i Algorytm, z interfejsu Osobnik i FunkcjaPrzystosowania oraz z klas pomocniczych: FunkcjaRosenbrocka , Main , OsobnikMaxComparator, OsobnikMinComparator, OsobnikDoCeluComparator, ParametryPanel, Punkt , Zakres.

* Algorytm – Klasa implementująca algorytm ewolucyjny. Klasy pomocnicze:
  + FunkcjaRosenbrocka – Jest to klasa zawierająca funkcję dla której poszukujemy rozwiązania. Klasa ta implementuje interfejs FunkcjaPrzystosowania.
  + OsobnikMaxComparator, OsobnikMinComparator, OsobnikDoCeluComparator – Klasy służą do definiowania relacji między osobnikami populacji.
  + Punkt – Klasa implementująca interfejs Osobnik, reprezentuje osobnika populacji.
  + Zakres – Klasa służąca do sprawdzania czy osobnik mieści się w zakresie poszukiwania.
* Widok – Klasa tworząca i zarządzająca graficznym interfejsem użytkownika. Klasy pomocnicze:
  + ParametryPanel – Klasa pomagająca wczytywać parametry pobierane od użytkownika.
* Kontroler – Klasa zarządzająca algorytmem. Pośredniczy między klasą Algorytm a klasą Widok.

1. **Wnioski dotyczące osiągniętych rezultatów**