Jakub Kowalczyk, 300238
Adam Stec, 300261

02.12.2020

# PSZT – Podstawy Sztucznej Inteligencji

## Sprawozdanie z projektu pierwszego

### 1. Treść zadania

Naszym zadaniem było napisanie algorytmu genetycznego znajdującego optimum funkcji Ackley'a oraz porównanie działania trzech różnych algorytmów selekcji.

### 2. Interpretacja i założenia

Celem naszego projektu, oprócz napisania algorytmu genetycznego, jest zbadanie jego działania dla różnych parametrów takich jak: wymiar, liczebność populacji, ilość pokoleń, siła mutacji. Dodatkowo za cel postawiliśmy sobie porównać czas wykonania algorytmów dla różnych selekcji.

Wszelkie pomiary oraz wykresy, jakie znajdują się w naszej dokumentacji, to wartości uśrednione z 51 powtórzeń wykonania algorytmu genetycznego dla tych samych parametrów.

We wszystkich przeprowadzonych badaniach prawdopodobieństwo mutacji jest równe 0.2, zmienialiśmy jedynie jej siłę.

### 3. Realizacja

Stworzyliśmy algorytm, który w zależności od podanej przez standardowe wejście wartości, posługuje się odpowiednim algorytmem selekcji. Zastosowaliśmy selekcję ruletkową, turniejową oraz prosty algorytm, który wymyśleliśmy sami. Polega on na wyborze najlepszej części populacji oraz sklonowaniu jej (możliwy jest wybór jaki ułamek populacji zostanie sklonowany – przykładowo dla liczby 2 zostanie sklonowane najlepsze 50% populacji, dla liczby 3 – 33% populacji itd.). Wszystkie algorytmy są w całości zaimplementowane przez nas.

Posłużyliśmy się krzyżowaniem dwupunktowym.

Podczas mutacji liczone jest prawdopodobieństwo jej wystąpienia dla każdego genu. W przypadku wybrania danego genu, jego nowa wartość wyznaczana jest za pomocą rozkładu normalnego.

Program działa na systemach Windows oraz Linux.

Link do repozytorium: https://github.com/radamp11/GeneticAlgorithm

## 4. Podział zadań

Postanowiliśmy podzielić się opracowywaniem metod klasy Algorithm z pliku algorithm.cpp. Jakub zrealizował odpowiednio:

- Mutację
- Selekcję ruletkową
- Inicjalizację
- Zapis do pliku

Adam:

- Selekcję turniejową
- Selekcję bestFraction
- Krzyżowanie

Pozostałe części pracy wykonaliśmy wspólnie. Staraliśmy się zadbać o poprawność i przejrzystość kodu.

## 5. Tabele wyników

A. Wpływ wielowymiarowości problemu na dokładność. Im algorytm jest bliższy osiągniecia optimum globalnego, tym wartość funkcji celu jest bliższa „1”.
Stałe wartości: populacja = 50, sigma mutacji = 5, liczba generacji = 100

**Selekcja Ruletkowa**

| Wymiar | Min | Max | Średnia | Odchylenie | Mediana |
|---|---|---|---|---|---|
| 2 | 0.822079 | 0.997985 | 0.920312 | 0.059244207 | 0.921299 |
| 4 | 0.794493 | 0.981297 | 0.8619311 | 0.045911745 | 0.85073 |
| 10 | 0.552539 | 0.786 | 0.6756775 | 0.05685477 | 0.684525 |

**Selekcja Turniejowa**

| Wymiar | Min | Max | Średnia | Odchylenie | Mediana |
|---|---|---|---|---|---|
| 2 | 0.892132 | 0.998854 | 0.97485014 | 0.02345212 | 0.98244 |
| 4 | 0.903059 | 0.990861 | 0.95683535 | 0.02423385 | 0.959651 |
| 10 | 0.81343 | 0.94152 | 0.87245216 | 0.02608155 | 0.869558 |

**Nasza selekcja**

| Wymiar | Min | Max | Średnia | Odchylenie | Mediana |
|---|---|---|---|---|---|
| 2 | 0.947206 | 0.999252 | 0.98790967 | 0.01039666 | 0.99063 |
| 4 | 0.902874 | 0.996092 | 0.97182241 | 0.0182174 | 0.974524 |
| 10 | 0.846065 | 0.938193 | 0.89071943 | 0.02334293 | 0.893289 |

**Krótki wniosek**: im większy jest wymiar problemu, tym trudniej jest algorytmom znaleźć jego optimum.

B. Teraz badamy wpływ siły mutacji.
Stałe wartości: populacja = 50, wymiar = 4, liczba generacji = 100

**Selekcja Ruletkowa**

| MUTACJA (sigma) | Min | Max | Średnia | Odchylenie | Mediana |
|---|---|---|---|---|---|
| 0.3 | 0.323113 | 0.899867 | 0.64765298 | 0.1237536 | 0.647956 |
| 2 | 0.797345 | 0.966624 | 0.87195424 | 0.03880882 | 0.862557 |
| 5 | 0.744639 | 0.915895 | 0.84936661 | 0.0361098 | 0.855688 |
| 8 | 0.72658 | 0.949786 | 0.84329747 | 0.04688718 | 0.843792 |

**Selekcja Turniejowa**

| MUTACJA (sigma) | Min | Max | Średnia | Odchylenie | Mediana |
|---|---|---|---|---|---|
| 0.3 | 0.402684 | 0.998638 | 0.75322806 | 0.18064298 | 0.773502 |
| 2 | 0.959263 | 0.997873 | 0.98693873 | 0.0081768 | 0.987584 |
| 5 | 0.909361 | 0.996925 | 0.95920957 | 0.02211154 | 0.964362 |
| 8 | 0.863969 | 0.991556 | 0.93107055 | 0.03105407 | 0.925769 |

**Nasza selekcja**

| MUTACJA (sigma) | Min | Max | Średnia | Odchylenie | Mediana |
|---|---|---|---|---|---|
| 0.3 | 0.551286 | 0.999913 | 0.84568629 | 0.13803985 | 0.883736 |
| 2 | 0.978897 | 0.998343 | 0.99202016 | 0.00504255 | 0.99377 |
| 5 | 0.916411 | 0.995144 | 0.97260782 | 0.01774592 | 0.976413 |
| 8 | 0.884391 | 0.994564 | 0.95445545 | 0.0261884 | 0.961143 |

**Krótki wniosek**: Mniejsza siła mutacji dawała lepsze wyniki maksymalne (poza selekcją ruletkową), lecz za mała sprawiała, że algorytm utykał w minimach lokalnych co odkładało się na pogorszeniu jakości. Biorąc pod uwagę średnie wyniki najlepiej wypadła mutacja o sile sigma = 2.

C. Wpływ ilości osobników w populacji.
Stałe wartości: siła mutacji = 5, wymiar = 4, liczba generacji = 100

**Selekcja Ruletkowa**

| OSOBNIKI | Min | Max | Średnia | Odchylenie | Mediana |
|---|---|---|---|---|---|
| 25 | 0.721799 | 0.991056 | 0.80576575 | 0.05030106 | 0.799727 |
| 75 | 0.792184 | 0.968202 | 0.87464657 | 0.04398693 | 0.870901 |
| 200 | 0.852824 | 0.979544 | 0.92686429 | 0.03366389 | 0.92744 |

**Selekcja Turniejowa**

| OSOBNIKI | Min | Max | Średnia | Odchylenie | Mediana |
|---|---|---|---|---|---|
| 25 | 0.835658 | 0.985152 | 0.90830324 | 0.04008244 | 0.899351 |
| 75 | 0.923457 | 0.998764 | 0.9715359 | 0.01492554 | 0.973572 |
| 200 | 0.974006 | 0.998393 | 0.99249882 | 0.00517461 | 0.993734 |

**Nasza selekcja**

| OSOBNIKI | Min | Max | Średnia | Odchylenie | Mediana |
|---|---|---|---|---|---|
| 25 | 0.859859 | 0.985719 | 0.93360633 | 0.03239777 | 0.943456 |
| 75 | 0.952379 | 0.997026 | 0.98190306 | 0.01122138 | 0.985947 |
| 200 | 0.982461 | 0.999231 | 0.99546847 | 0.00299504 | 0.99625 |

**Krótki wniosek**: Zwiększenie liczby osobników w populacji korzystnie przekłada się na dokładność znalezionego optimum, lecz w zamian przedłuża czas działania algorytmu.

D. Wpływ liczby generacji
Stałe wartości: populacja = 50, siła mutacji = 5, wymiar = 4

**Selekcja Ruletkowa**

| GENERACJE | Min | Max | Średnia | Odchylenie | Mediana |
|---|---|---|---|---|---|
| 50 | 0.676922 | 0.893619 | 0.80248937 | 0.05394887 | 0.815934 |
| 200 | 0.786567 | 0.962482 | 0.86711259 | 0.03775734 | 0.860335 |
| 1000 | 0.813673 | 0.987166 | 0.91858396 | 0.04809134 | 0.927382 |

**Selekcja Turniejowa**

| GENERACJE | Min | Max | Średnia | Odchylenie | Mediana |
|---|---|---|---|---|---|
| 50 | 0.838015 | 0.973027 | 0.902443 | 0.03281104 | 0.905192 |
| 200 | 0.952556 | 0.997551 | 0.98231635 | 0.01028967 | 0.985087 |
| 1000 | 0.991846 | 0.999466 | 0.99771067 | 0.00137703 | 0.997869 |

**Nasza selekcja**

| GENERACJE | Min | Max | Średnia | Odchylenie | Mediana |
|---|---|---|---|---|---|
| 50 | 0.851461 | 0.986475 | 0.93551473 | 0.03337148 | 0.94086 |
| 200 | 0.967898 | 0.998201 | 0.98821014 | 0.0065 | 0.988817 |
| 1000 | 0.996458 | 0.999629 | 0.99859845 | 0.00069368 | 0.998722 |

**Krótki wniosek:** Zwiększenie liczby generacji korzystnie przekłada się na dokładność znalezionego optimum, lecz w zamian przedłuża czas działania algorytmu.

E. Badanie czasu działania algorytmu w zależności od zastosowanej selekcji.
Stałe wartości: populacja = 50, liczba generacji = 2000, wymiar = 4,

| Czas wykonania | Selekcja |
|---|---|
| 89.59 sec | ruletkowa |
| 178.41 sec | turniejowa |
| 82.75 sec | nasza selekcja |

Testy wykonane w Visual Studio 2019 w systemie Windows 10.

## 6. Wnioski

We wszystkich wymiarach najlepiej się spisał nasz algorytm selekcji. Ponadto (na podstawie wykresów w oddzielnym pliku) widać, że najszybciej dąży on do wartości pożądanej. Także istotnym wnioskiem jest to, że dokładność znalezionego optimum zależy głównie od siły mutacji. Krzyżowanie nie wprowadza aż tak zauważalnych efektów.

Wykresy załączyliśmy oddzielnie w celu zwiększenia czytelności. Ich celem jest zobrazowanie, w jaki sposób wygląda dążenie do optimum dla różnych algorytmów selekcji oraz wymiarowości. Zauważamy, że nasz algorytm selekcji dąży do optimum najszybciej.

Przeprowadziliśmy również test, który miał na celu sprawdzenie znaczenia początkowej populacji. Gdy zainicjowane wartości wszystkich genów nie różniły się zbytnio (inicjacja w przedziale wartości 20 – 25) wszystkie nasze algorytmy utykały w minimach lokalnych nawet dla sił mutacji przewyższających „sigma = 1.5”. Jak można odczytać z tabel, przy równomiernym rozsianiu osobników, nawet „sigma = 0.3” dawała w pewnych przypadkach zadawalające wyniki. Pokazuje to znaczący wpływ sposobu inicjacji populacji na końcowy efekt.