|  |  |
| --- | --- |
| 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.

1. **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.

1. **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, losującym punkty krzyżowania.

Mutacja

Program działa na systemach Windows oraz Linux.

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

1. **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.

1. **Tabele wyników**

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.

1. **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.