ALHE – dokumentacja i sprawozdanie
Temat: SK.ALHE.6

Patryk Pankiewicz, Maciej Turski

07.01.2019

1 Temat projektu

Masz 10 kart ponumerowanych od 1 do 10. Znajdź przy uzyciu Algorytmu Ewolucyjnego sposób na podział kart na dwie kupki w taki sposób, że suma kart na pierwszej kupce jest jak najbliższa wartości A, a suma kart na drugiej kupce jest jak najblizsza wartości B. Należy zastosować dodatkowo inny wybrany algorytm i porównać wyniki.

2 Opis funkcjonalny

Do rozwiązania problemu został użyty algorytm ewolucyjny, wykorzystujący krzyżowanie i mutacje. Celem algorytmu jest minimalizacja różnicy. Wartości kart są przechowywane jako list i nie ma potrzeby by była ona kopiowana dla każdego osobnika - jest on reprezentowany jako lista przynależności kart do stosów. Wykorzystane metody zostały wyszczególnione poniżej.

- **Funkcja oceniająca** - musi być dostosowana do celu minimalizacji i zostały zaimplementowane jej nastepujące wersje:
  - suma modułów różnicy wartości kart na stosach
  - suma różnicy wartości kart na stosach podniesiona do potęgi >= 2 (jest ona parametrem algorytmu
- **Selekcja** osobników - zostały stworzone trzy wskazane w czasie wykładów z przedmiotu podejścia:
  - metoda turniejowa
  - metoda progowa
  - metoda proporcjonalna
- **Mutacja** - realizowana poprzez zmianę przynależności karty do stosu
- **Krzyżowanie** - zostały opracowane dwa rozwiązania:
  - dla każdej karty z listy losowany będzie z jednakowym prawdopodobieństwem osobnik z którego zostanie wzięta informacja gdzie ma przynależeć karta
  - wylosowanie indeks lub indeksów względem których przynależność kart zostanie zmieniona na przeciwną

3 Opis interfejsu użytkownika

Aplikacja posiada intefejs konsolowy. Argumenty aplikacji:

- -1 wartość - wartość oczekiwana dla pierwszego stosu, wymagane
- -2 wartość - wartość oczekiwana dla pierwszego stosu, wymagane
- rodzaj metody selekcji, wymagane - jeden argument z poniższych:
  - -r wartość, selekcja proporcjonalna, wartość odpowiada liczbie *beta*, która jest używana przy obliczniu wartości prawdopodobieństwa ze wzoru $exp(-beta * z_i)$, gdzie z_i odpowiada wartości celu i-tego osobnika
  - -t wartość, selekcja turniejowa, wartość odpowiada rozmiarowi szranek
  - -h wartość, selekcja progowa, wartość odpowiada wartości progu
- -m wartość - wartość odpowiada prawdopodobieństwu mutacji, wymagane
- -v wartość - wybór funkcji celu, wartość odpowiada potędze do której zostanie podniesiona różnica wartość kart na obu stosach, wymagane
- rodzaj metody krzyżowania, wymagane - jeden argument z poniższych:
  - -u - wybór każdej właściciela dla każdej karty z osobna
  - -k wartość - wybór k punktów względem których zostanie zmieniona przynależność kart, wartość odpowiada liczbie k
- -s wartość - wartość odpowiada wartości ziarna, jeśli nie podane zostanie ustawione na losową
- -c wartość - wartość odpowiada ilości kart, wymagane
- -a wartość - wartość odpowiada rozmiarowi populacji, wymagane
- -b wartość - wartość odpowiada prawdopodobieństwu mutacji, wymagane
- -i wartość - wartość odpowiada ilości iteracji, wymagane
- -f wartość - wartość odpowiada prefiksowi pliku wynikowego

4 Postać plików wynikowych

Pliki wynikowe są plikami w formacie csv, z seperatorem ",". Nazwa pliku wynikowego jest wartością podaną dla argumentu -f, będącego prefiksem, i argumentów programu, tak by ułatwić identyfikacje plików wynikowych.

Przykład nazwy pliku wynikowego:

```
wynik_-1_100_-2_100_-r_1_-m_0.09_-v_2_-u_-c_10_-a_30_-b_0.10_-i_200.csv
```

Program został wywołany z następującymi argumentami:

```
-1 100 -2 100 -r 1 -m 0.09 -v 2 -u -c 10 -a 30 -b 0.10 -i 200 -f wynik
```

Pozwala to w prosty sposób zidentyfikować wybrane argumenty. Program nie posiada żadnych dodatkowych pilków konfiguracyjny - wszystkie parametry są ustalane za pomocą argumentów.

Dla każdej iteracji w wynikowym pliku csv zapisywane są następujące informacje:

- wartość funkcji celu dla najmniejszego osobnika
- wartość mediany populacji
- wartość średnia populacji
- wartość odchylenia standardowego populacji

5 Opis implementacji

Algorytm został zaimplementowany w języku C++ w standardzie C++14 z użyciem wyłącznie bibliotek dostepnych w standardzie. Program został on zaprojektowny w sposób modularny z wykorzystaniem klas abstrakcyjnych. Główne klasy algorytmu:

- **Mutation** - klasa odpowiedzialna za przeprowadzenie mutacji z zadanym prawdopodobieństwem
- **CrossoverAlgorithm** - klasa abstrakcyjna będąca klasą bazową dla algorytmów krzyżowania

- **ScoringFunction** - klasa, która pozwala na obliczenie zadanej funkcji celu
- **SelectionAlgorithm** - klasa abstrakcyjna będąca klasą bazową dla algorytmów selekcji
- **EvolutionaryAlgorithm** - główna klasa algorytmu
- **CSVFileWriter** - klasa opowiadająca za zapis wyników do pliku csv

Dodatkowo do zautomatyzowania testowania i generacji wykresów z danych wynikowych został użyty język Python. Zostały użyte nastepujące moduły i biblioteki:

- pandas[1], biblioteka umożlwiająca łatwą obsługę plików o rozszerzeniu csv
- numPy[2], biblioteka obliczeniowa, tu użyta do obsługi tablic i prostych algorytmów
- matplotlib[3], biblioteka do generowania wykresów
- subprocess[4], moduł umożlwiający wywołanie plików wykonywalnych

[1]https://pandas.pydata.org/
[2]http://www.numpy.org/
[3]https://matplotlib.org/
[4]https://docs.python.org/3/library/subprocess.html