Dokumentacja projektu z przedmiotu Metody Numeryczne i Symulacja

Temat: Demonstrator działania algorytmu genetycznego na przykładzie rozwiązywania równań nieliniowych

Założenia:

1. Znalezienie pierwiastków trójmianu kwadratowego z dwoma rzeczywistymi pierwiastkami używając zaimplementowanego algorytmu genetycznego

Opis programu:

1. Biblioteki
2. Biblioteka **iostream** została użyta w celu wyświetlenia operacji oraz wyniku działania algorytmu (**cin**, **cout**, **endl**)
3. Biblioteka **random** została użyta w celu generowania liczb losowych (funkcja **pow** oraz generator mt19937 wraz z funkcjami rozkładu izotropowego, zawiera **time()**)
4. Biblioteka **bitset** została użyta w celu operowania na tablicy bitów (**bitset**, **bits**)
5. Biblioteka **iomanip** została użyta w celu ustawienia dokładności wyświetlania liczb typu **float** po przecinku
6. Zmienne globalne (typu **const** określane jako „stałe”)
7. Stała **ZAKRES** wyznacza zakres liczb początkowo losowanych do tabeli argumentów funkcji
8. Stała **LICZEBNOSC\_POPULACJI** określa ilość liczb w populacji argumentów
9. Stała **LICZBA\_CHROMOSOMOW** określa ilość chromosomów (bitów) każdego argumentu
10. Stała tablica **FUNKCJA[]** jest tablicą współczynników wielomianu, którego pierwiastki są szukane
11. Stała **MAKS\_ITERACJI** określa maksymalną liczbę powtórzeń głównej pętli programu
12. Stała **DOKLADNOSC** określa dokładność, z którą są wyszukiwane pierwiastki
13. Stała **ODLEGLOSC\_MIEDZY\_PIERWIASTKAMI** wyznacza minimalną odległość między szukanymi argumentami (ta liczba została ustalona, żeby uniknąć generowania dwóch bardzo bliskich siebie pierwiastków, które są de facto jednym pierwiastkiem funkcji
14. Struktura **Osobnik** definiuje jeden argument funkcji, składa się z:
15. Zmiennej **nrOsobnika** określającej numer argumentu
16. Zmiennej **argument**, w której zapisywana jest dziesiętna reprezentacja argumentu
17. Tablicy zmiennych **reprezentacja\_bitowa**, w której zapisywana jest reprezentacja bitowa danego argumentu
18. Zmiennej **wartosc\_funkcji**, w której zapisywana jest wartość funkcji w danym argumencie
19. Funkcji bezargumentowej **wartoscFunkcji()**, która pozwala obliczyć wartość funkcji w danym argumencie
20. Konstruktor, w którym **reprezentacja\_bitowa** jest wypełniana zerami, aby uniknąć przetwarzania losowych danych w pamięci
21. Tablica typu **Osobnik**, zadeklarowana w ilości **LICZEBNOSC\_POPULACJI**
22. Funkcja **int main()** jest główną funkcją programu, wykonywaną jako pierwsza
23. Za pomocą **setprecision(9)** ustalamy do którego miejsca po przecinku będą się wyświetlać liczby
24. **random\_engine** - deklaracja silnika liczb losowych typu mt19937 z biblioteki <random>
25. **random\_engine.seed(time(NULL))** - podanie ziarna do silnika liczb losowych na podstawie bieżącego czasu
26. **losowaniePopulacji(random\_engine)** - wywołanie funkcji, która losuje pierwszą populację
27. Pętla **for** służy do wyświetlenia pierwszej populacji w konsoli
28. Zmienna **n** typu **int** oznacza ilość wykonanych iteracji głównej pętli programu, inicjalizacja 0 (brak wykonań)
29. Zmienna **warunek** typu **bool** określa warunek zakończenia wykonywania głównej pętli, inicjalizacja **true** (można wykonać następną iterację)
30. **float ARG1, ARG2, WAR1, WAR2** to deklaracja zmiennych zawierających wynik działania programu (w przypadku nieznalezienia miejsc zerowych program wyświetli wartość tych zmiennych, z którą są inicjalizowane, czyli **100.0**).
31. Pętla **while(warunek)** jest główną pętlą programu. jest w niej wywoływana funkcja **Main\_Loop(random\_engine)** oraz **warunek(n, &ARG1, &ARG2, &WAR1, &WAR2)**
32. Po zakończeniu działania pętli **while** zostaje wyświetlony wynik działania programu
33. Funkcja **check(int n, float \*ARG1, float \*ARG2, float \*WAR1, float \*WAR2)** służy do sprawdzania, czy program po wykonaniu iteracji pętli zdołał odnaleźć miejsca zerowe funkcji. Zapisuje do zmiennych **ARG1, ARG2, WAR1, WAR2** odpowiednio argumenty oraz przybliżone wartości znalezionych miejsc zerowego. Przy znalezieniu **Osobnika** z wartością funkcji mniejszą od ustalonej dokładności **DOKLADNOSC**, przypisuje jego argument i wartość odpowiednio do **ARG1** i **WAR1**, następnie sprawdza, czy do **ARG2** i **WAR2** przypisywane były już jakieś wartości (czy **WAR2** był różny od **100.0**). Jeżeli nie były to przypisuje im wartości znalezionego argumentu i wartości, a **ARG1** i **WAR1** są resetowane go wartości **100.0**. W kolejnych iteracjach **check()** przypisuje do **ARG1** i **WAR1** parametry osobnika, którego argument znajduje się odpowiednio daleko od **ARG2**. Po znalezieniu dwóch różnych miejsc zerowych lub wykonaniu maksymalnej liczby iteracji **check()** zwraca **false**, czym przerywa główną pętlę programu
34. Funkcja **losowaniePopulacji()** wypełnia tablicę struktur Populacja losowymi Osobnikami, w pętli:
    1. Numeruje osobniki
    2. Losuje argument z zakresu <-**ZAKRES**, **ZAKRES**>
    3. Wywołuje składową struktury **Osobnik**, aby obliczyć wartość funkcji w danym argumencie
35. Funkcja **floatToByte()** zamienia **argument** danego Osobnika na reprezentację bitową i zapisuje go do tablicy **reprezentacja\_bitowa[]**
36. Funkcja **byteToFloat()** zamienia reprezentację bitową argumentu przekazanego Osobnika na argument w formie **float** i zwraca go (nie zapisuje)
37. Funkcja **swap(Osobnik \*xp, Osobnik \*yp)** służy do zamiany danych dwóch Osobników
38. Funkcja **sortowanie()** sortuje tablicę **Populacja[]** według bliskości wartości funkcji każdego Osobnika do zera. Po wykonaniu funkcji, na początku tablicy **Populacja[]** są Osobniki z wartościami najbliższymi zeru
39. Funkcja **krzyzowanie(int pozycja1, int pozycja2)** funkcja ta zamienia od losowego miejsca chromosomy dwóch Osobników, **pozycja1** i **pozycja2** to numery Osobników, na których są wykonywane działania
40. Funkcja **mutacja()** powoduje, że w trakcie działania programu istnieje mała szansa (5%), że Osobnik zmutuje jeden losowo wybrany chromosom na przeciwny (0->1 lub 1->0)
41. Funkcja **Main\_Loop()** to funkcja, w której są wywoływane i wykonywane główne działania na tablicy **Populacja[]**:
42. Wywoływane są funkcje **mutacja()** oraz **sortowanie()**
43. Zostają zadeklarowane oraz wyzerowane tablice używane w procesie krzyżowania
44. Pierwsza pętla **for** losuje tablicę krzyżowania Osobników (z pierwszej połowy tabeli **Populacja[]**, aktualnie najlepsze Osobniki) z innymi Osobnikami
45. Druga pętla **for** wywołuje krzyżowanie wybranych Osobników
46. Trzecia pętla **for** wywołuje funkcje obliczania argumentu z tablicy bitowej oraz wyznaczania wartości funkcji w argumencie

Dokumentacja sporządzona na podstawie kodu źródłowego zawartego w pliku main.cpp dołączonego do dokumentu.

Wykonali:

Szymon Feliński

Bartłomiej Hojnacki