Magiczny Kwadrat rozwiązywany
Metodą algorytmu genetycznego

Wykonali:
Bartłomiej Jankowski
Michał Górski

1. Treść zadania:

Zadanie polega na wygenerowaniu kwadratu NxN, w którym elementami są liczby (1, 2, ..., n^2) ułożone tak, aby w każdym wierszu, kolumnie i przekątnej suma wartości liczb była taka sama i wynosiła n*(1+n^2)/2. Zastosować do rozwiązania zadania algorytm genetyczny. WE: liczba N, maksymalna liczba iteracji. WY: wygenerowany kwadrat.

1. Przyjęte założenia, doprecyzowanie treści:

Początkowe rozmieszczenie elementów w kwadracie jest losowe. Znajdywanie magicznego kwadratu polega na zastosowaniu algorytmu genetycznego do zbioru wylosowanych kwadratów jako osobników.

2. Podział odpowiedzialności w zespole:

Bartłomiej Jankowski: projektowanie architektury, redukcja złożoności obliczeniowej programu, tworzenie dokumentacji, wnioski
Michał Górski: napisanie funkcji realizujących architekturę, dobór parametrów rozwiązujących problem, testy

3. Zwięzły opis algorytmu/architektury i uzasadnienie sposobu realizacji:

Algorytm

Osobnik to klasa Subject w której przechowywane są macierze NxN. Populacja to lista osobników klasy Population. Osobniki są oceniane na podstawie funkcji celu, która jest wartości bezwzględną róznicy sum elementów dla (kolumn, wierszy, diagonali) z wartością wymaganą (N*(N^2+1)/2).
Kroki postępowania:

1. Wygeneruj początkową populację o n-krotnie większej liczebności od standardowej i dokonaj oceny osobników
2. Wybierz μ najlepszych osobników z populacji początkowej i dodaj do zbioru R (R - rodzice)
3. Dopóki nie przekroczono dopuszczalnej liczby iteracji lub nie znaleziono magicznego kwadratu wykonuj kroki: (4), (5), (6)

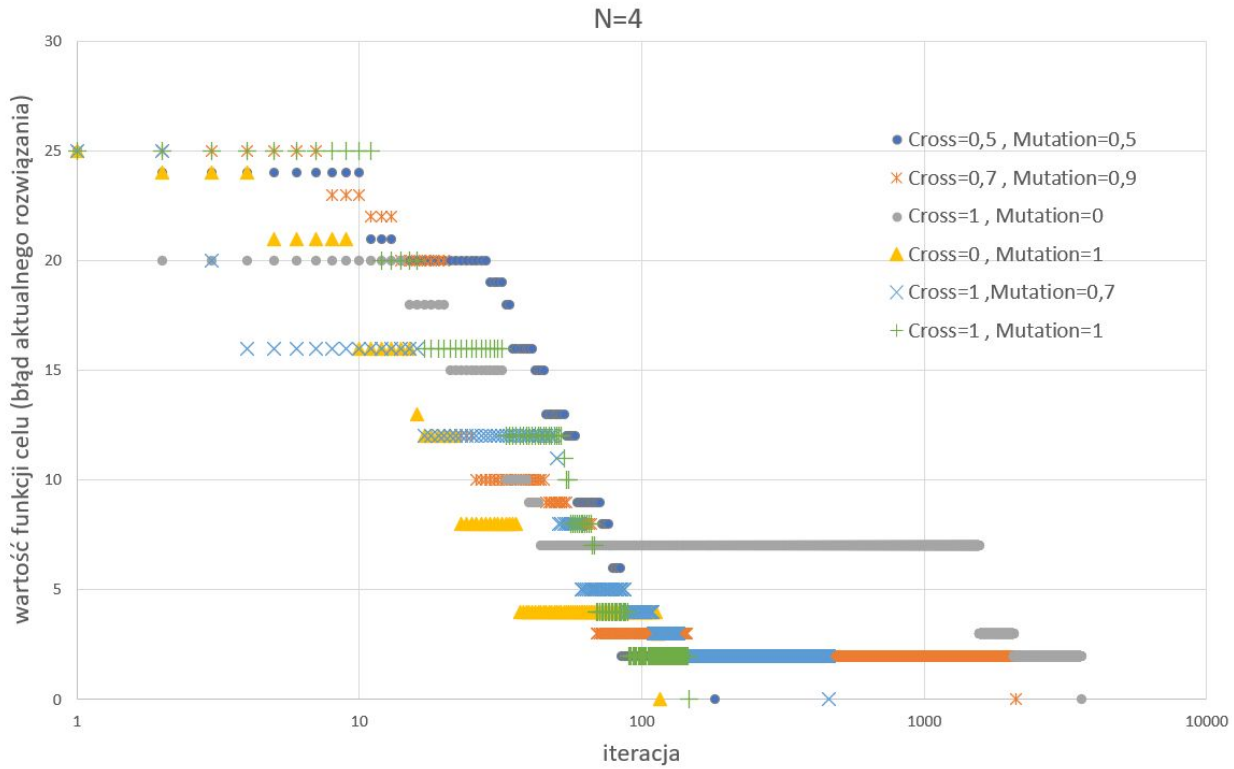
4. Stwórz μ potomstwa poprzez krzyżowanie i mutacje i dokonaj oceny potomstwa
5. Ze zbioru rodziców i potomstwa wybierz μ najlepszych, niepowtarzających się osobników
6. Jeśli błąd nie zmienia się od K iteracji, wylosuj na nowo połowę rodziców
7. Zwróć wynik: magiczny kwadrat, liczbę wykonanych iteracji, błąd zwróconego rozwiązania, informację (znaleziono [przy błędzie = 0] lub nie znaleziono [przy błędzie > 0] rozwiązania)

4. Raport z przeprowadzonych eksperymentów:

Na rys.1 zamieszczono wykres przedstawiający liczbę wymaganych iteracji do znalezienia magicznego kwadratu dla różnych wartości parametru N.

Rys. 1 Wykres przedstawiający liczbę wymaganych iteracji do znalezienia magicznego kwadratu dla różnych parametrów N

Dla kwadratu o rozmiarze N=4 zbadano różne kombinacje współczynników krzyżowania i mutacji. Zadbano o to aby w każdym przypadku algorytmy zaczynały działanie od tej samej populacji. Wyniki przedstawiono na Rys. 2. Z wykresu tego wynika, że w tym przypadku sama mutacja radziła sobie znacznie lepiej od krzyżowania. Przyczyn takiego zachowania można się dopatrywać w fakcie, że przy początkowo dobrym rozwiązaniu mutacja wprowdza mniej chaosu dlatego szybciej sobie radzi z rozwiązaniami o małym błędzie początkowym.



Rys.2 Wykres przedstawiający aktualny błąd rozwiązania w kolejnych iteracjach przy różnych współczynnikach krzyżowania(Cross) i mutacji(Mutation) dla N=4.

5. Wnioski:

Tempo uzyskania wyników bardzo zależy od początkowo wylosowanej macierzy. Na rys.1 można zauważyć, że dla np. N = 6 Liczba iteracji waha się między bardzo małymi wartościami (około 500) i sięga do wysokich (około 17000). Jednak generalna obserwacja pokazuje, że wzrost liczby iteracji zależy od parametru N i jest silnie nieliniowy. Dla N=7 program czasem nie zamykał się w 50000 iteracjach co nie jest pokazane na wykresie. Możliwą opcją poprawiającą działanie programu, byłoby dynamiczne sterowanie rozmiarem populacji i parametrami wyznaczającymi częstotliwość zachodzenia krzyżowania i mutacji.

6. Zwięzła instrukcja użytkowania programu:

Początkowo należy pobrać z pliki: Genetic.py, MagicSquare.py, Algoritm.py, config.txt z https://github.com/michalaws2k18/PSZT_pr1/tree/master/final_code . (katalog final_code z głównego repozytorium). Następnie w folderze z pobranymi plikami należy uruchomić terminal i wpisać komendę “ python MagicSquare.py N max_iter”, gdzie N - rozmiar kwadratu, max_iter - maksymalna liczba iteracji. Program zwróci wyniki po obliczeniach. W pliku config.txt można zmienić użyte współczynniki.