WSI - Algorytm genetyczny

Jan Szymczak

Marzec 2024

1 Cel i opis eksperymentów

1.1 Opis problemu

Zadaniem było stworzenie implementacji algorytmu genetycznego wykorzystującego mutację, krzyżowanie jednopunktowe, selekcję ruletkową oraz sukcesję generacyjną. Następnie algorytm ten wykorzystywany był w celu optymalizacji problemu polegającego na znalezieniu najlepszego parkingu wielkości $10x10$, w którym każde pole mogło być miejscem parkingowym lub drogą. Miejsca parkingowe były reprezentowane przez 1, drogi przez 0. Do każdego miejsca parkingowego musiał być dojazd: z zewnątrz (spoza parkingu, tzn., że miejsce znajdowało się na granicy) lub poprzez drogę, która prowadzi na zewnątrz. Ponadto, na początku rozpoczęcia testowania każdej populacji, sprawdzałem poprawność osobników. Teoretycznie, po każdej iteracji populacja powinna być sprawdzana, czy nie zawiera osobników z samymi zerami. Zrezygnowałem jednak z tego, ponieważ przy "normalnych" (to znaczy o sensownych wielkościach) populacjach, szansa na to, że z niezerowej populacji po mutacjach (która zresztą ma bardzo małe prawdopodobieństwo zajścia) i krzyżowaniu, powstanie populacja w pełni zerowa jest praktycznie niemożliwa, zwłaszcza, że przyjęta selekcja ruletkowa nie "przepuszcza" zerowych osobników (ich prawdopodobieństwo selekcji jest równe 0). Takie założenie przyspieszy znacznie działanie programu.

1.2 Wpływ parametrów

Działanie algorytmu było uzależnione od kilku parametrów:

- **Wielkość populacji początkowej:** większa populacja początkowa daje większą szansę na istnienie dobrego osobnika już na samym początku. Z drugiej strony zwiększa czas obliczeń oraz, z uwagi na wykorzystywanie selekcji ruletkowej, powoduje "wyrównywanie się" szans wszystkich osobników, ponieważ suma funkcji celu wszystkich osobników zwiększa się, co zmniejsza różnice pomiędzy osobnikami.
- **Ilość iteracji:** przeprowadzenie symulacji dla większej liczby pokoleń zwiększa szansę na znalezienie dobrego osobnika, ale oczywiście znacząco wydłuża działanie programu.
- **Wartości hiperparametrów**
  - **Prawdopodobieństwo krzyżowania:** krzyżowanie jest najważniejszą operacją w przypadku algorytmu genetycznego. Wartość tego hiperparametru powinna być wysoka i jest dalej nazywana jako *pc*.
  - **Prawdopodobieństwo mutacji:** mutacja nie jest tak ważna w przypadku algorytmu genetycznego. Wartość tego hiperparametru zatem powinna być niska, bliska zeru. Nie powinno jednak rezygnować się z mutacji całkowicie, mutacja umożliwia wyjście algorytmu z minimum lokalnego, co z kolei może doprowadzić do znalezienia lepszego rozwiązania w przyszłości. Hiperparametr ten nazywany jest dalej jako *pm*.

2 Wyniki

Symulacje przeprowadzałem na niewielkich populacjach składających się z 50 osobników, aby selekcja miała wpływ na działanie programu. Na początku testowałem różne wartości hiperparametrów, aby odnaleźć ich odpowiednie wartości i wykorzystać je w celu szukania dobrych osobników.

2.1 Prawdopodobieństwo krzyżowania

Przeprowadziłem symulację populacji 50 osobników przez 2000 pokoleń dla *pc* z przedziału $\langle 0.0; 1.0 \rangle$ ze skokiem równym 0.02. Tak przedstawiają się wykresy średniej wartości wszystkich funkcji celu wraz z odchyleniem standardowym:

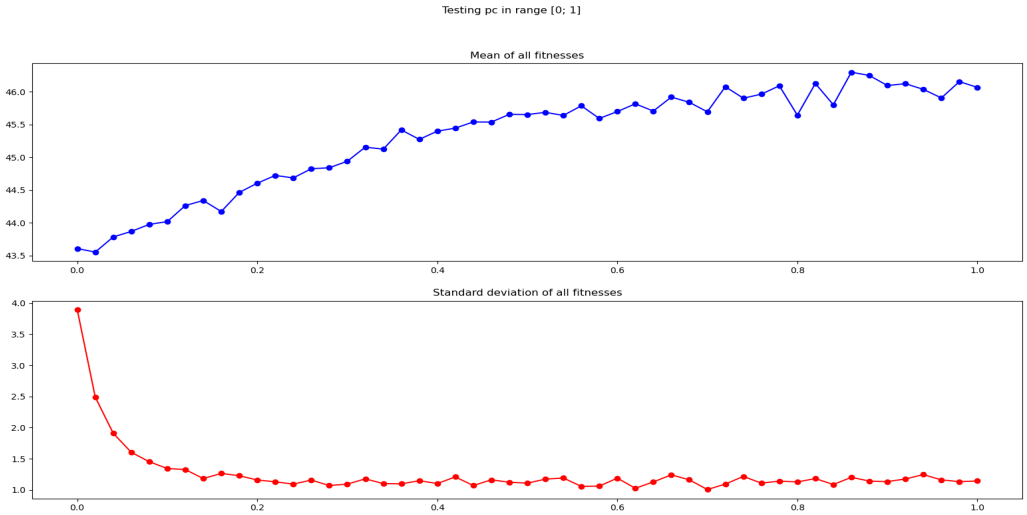

Figure 1: Średnia (na górze) oraz odchylenie standardowe wartości funkcji celu dla całych populacji.

Wyraźnie widać, jak średnia wyniku całej populacji stopniowo rośnie wraz ze wzrostem wartości parametru. Najlepsze wartości osiągane są w okolicach wartości 0.84, od tego momentu widać zahamowanie wzrostu, a nawet spadek wartości. Odchylenie standardowe bardzo szybko spada i przyjmuje w przybliżeniu stałą wartość już przy wartości około 0.1. Przy niższych wartościach wyniki po prostu były bardzo losowe, tak naprawdę głównie zależne od populacji początkowej.
Tak wyglądają wyniki janw przypadku, gdy bierzemy pod uwagę jedynie listę najlepszych znalezionych do tego momentu rozwiązań:

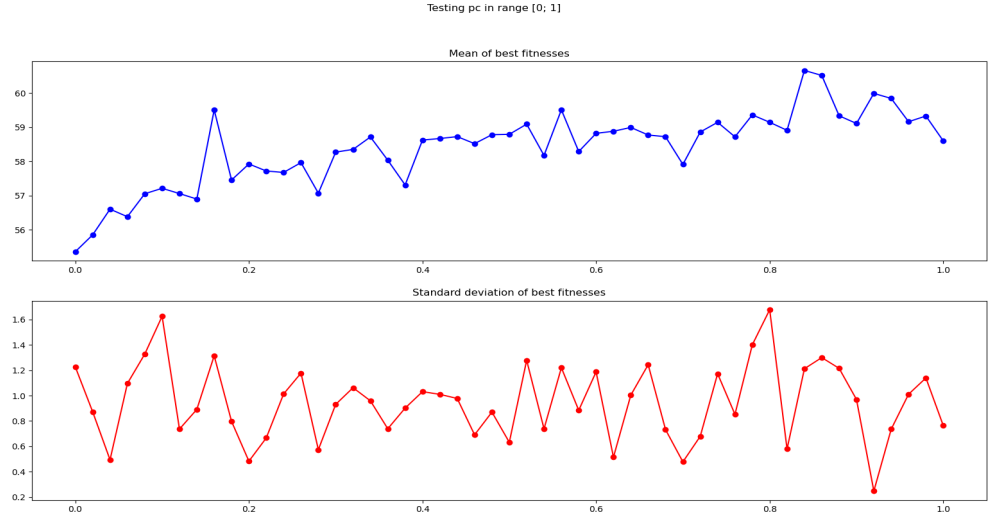

Figure 2: Średnia oraz odchylenie standardowe najlepszych wartości funkcji celu.

Zgodnie z oczekiwaniami, od razu można odrzucić wszystkie wartości *pc* poniżej 0.5, tutaj średnie obu danych są wyraźnie najniższe. Najlepsze wyniki zdają się dawać wartości parametru powyżej 0.8, jednak widać zauważalny spadek dla wartości bliskich jedności. Widać też tutaj losowość, dla parametru równego 0.16 wynik jest zauważalnie wyróżniający się, nawet przy "złym" parametrze algorytm czasem odnajduje dobre rozwiązanie. Odchylenie standardowe mocno waha się w trakcie testowania. Jest to spowodwane np. tym, że w niektórych symulacjach najlepszy osobnik znajdowany był bardzo szybko, przez co, w przypadku brania pod uwagę tylko najlepszych wyników, większość z nich osiągała tą samą wartość i odchylenie malało. Wysokie odchylenie wskazywało na "późniejsze" znalezienie najlepszego osobnika. W celu testowania parametru *pm* przyjąłem za tym *pc* równe 0.84.

2.2 Prawdopodobieństwo mutacji

Symulacja została przeprowadzona z takimi samymi parametrami jak powyżej, oczywiście z tą różnicą, że *pm* zmieniało się w podanym wyżej przedziale, a *pc* miało stałą wartość. Ponownie, tak wygląda wykres średniej ze wszystkich wartości funkcji celu wraz z ich odchyleniem standardowym:
Wyraźnie widać, że najlepsze wyniki są dla mutacji bliskiej zeru, potem bardzo szybko stają się gorsze. Co ciekawe, dla szansy mutacji bliskiej jedności wyniki nagle rosną, jest to spowodowane tym, że dla nich

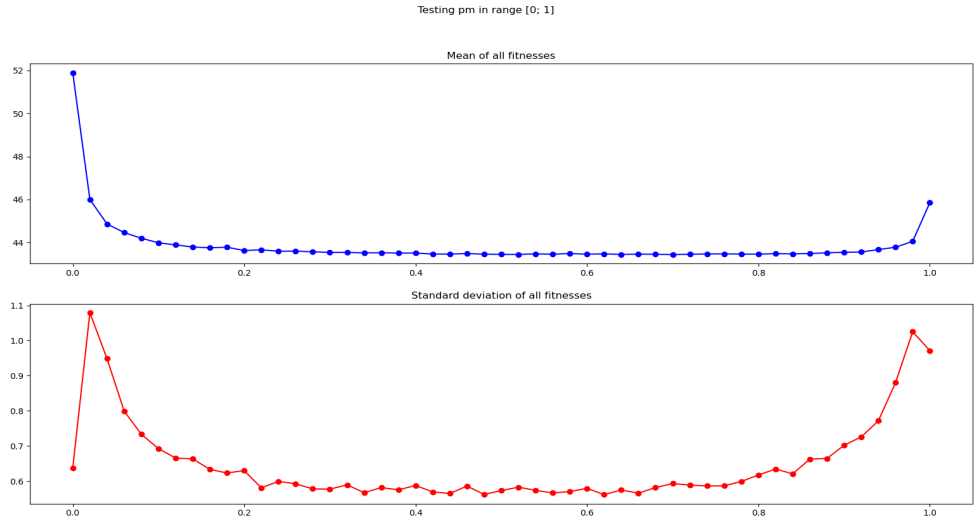

Figure 3: Średnia (na górze) oraz odchylenie standardowe wartości funkcji celu dla całych populacji.

losowość jest tak duża, że czasem możemy uzyskać wyniki lepsze, czasem gorsze. Dla nich też standardowe odchylenie znacząco rośnie. Tak wyglądają wykresy biorące pod uwagę tylko najlepsze osobniki:

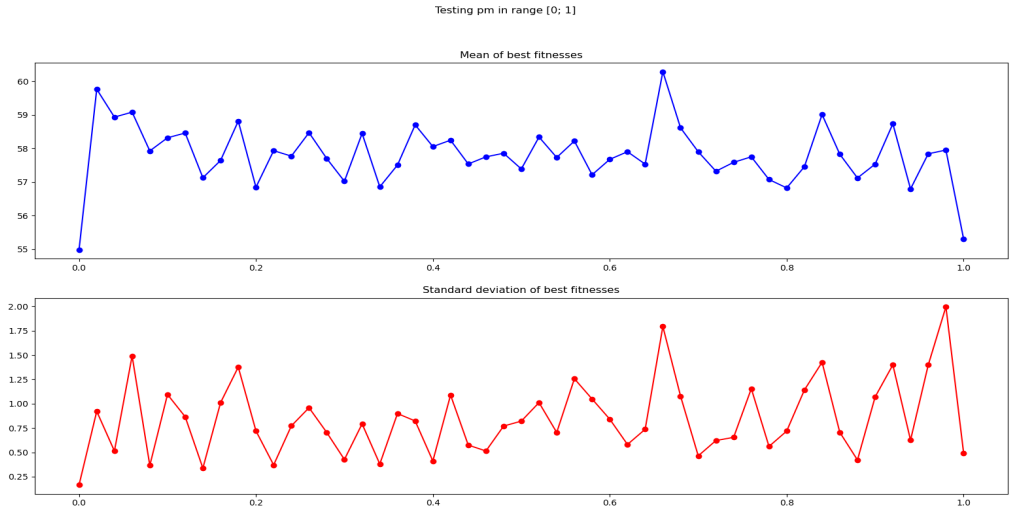

Figure 4: Średnia oraz odchylenie standardowe najlepszych wartości funkcji celu.

W tym przypadku, nie licząc losowego "wyskoku" dla wartości 0.66, ponownie widać, że najlepsze wartości są dla niskiego pm i maleją wraz ze wzrostem parametru. Odchylenie standardowe ponownie mocno się waha i jest uzależnione od tego jak szybko najlepszy osobnik zostanie znaleziony.

2.3 Ostateczne testy

Po przeprowadzeniu tych eksperymentów, ostateczne testy przeprowadziłem dla hiperparametrów o wartości $pc = 0.84$ i $pm = 0.02$ dla 100 populacji o wielkości 50 i maksymalnych iteracji 2000.
Tak przedstawia się wykres średniej z wartości funkcji celu dla wszystkich osobników dla 100 populacji:

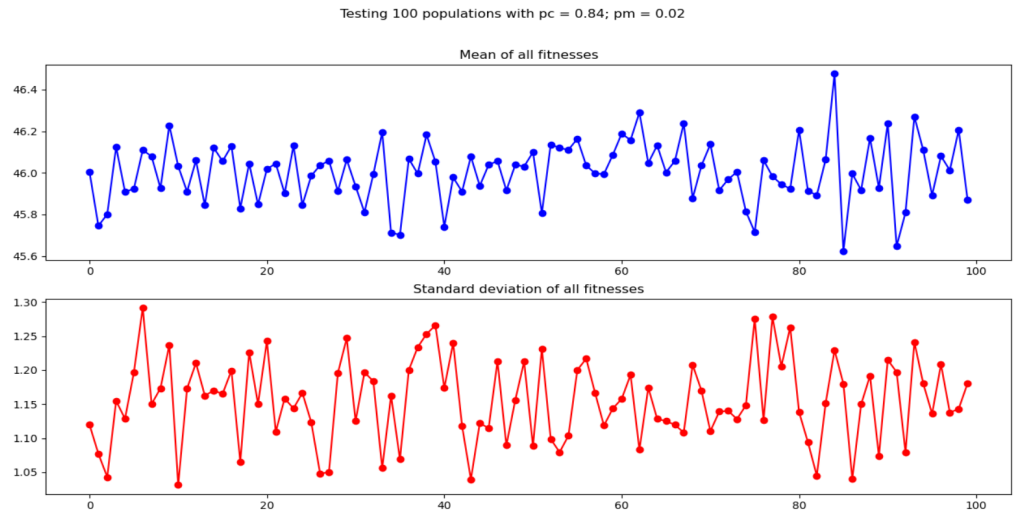

Figure 5: Średnia oraz odchylenie standardowe wartości funkcji celu dla całych populacji.

Wszystkie populacje dały bardzo podobne wyniki, ich średnia ogólnie nie była wysoka, zakładając najlepszy możliwy parking jako wartość 73. Odchylenia standardowe jednak nie są wysokiie, najlepszy osobnik z większości symulacji posiadał funkcje celu o wartości 61.
Wykres przedstawiający zmiany najlepszych osobników:

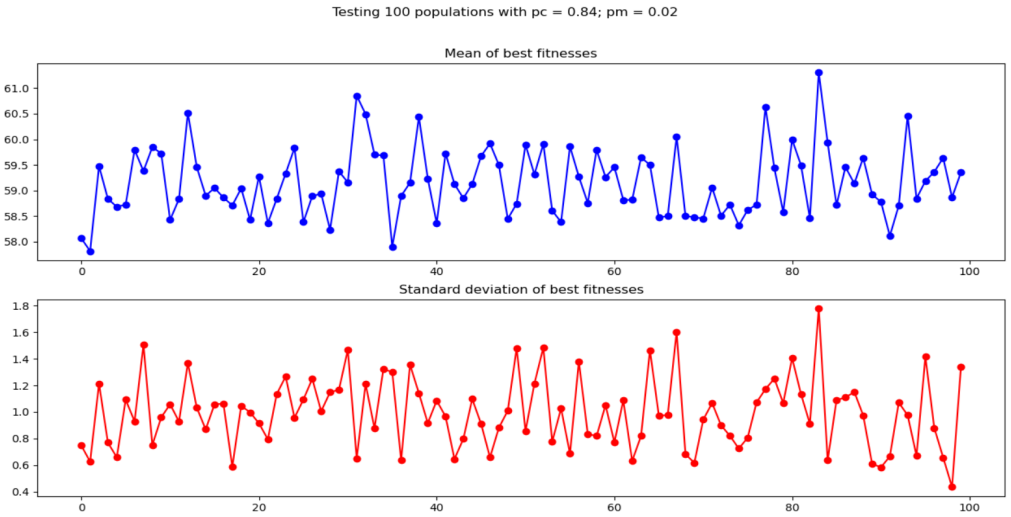

Figure 6: Średnia oraz odchylenie standardowe najlepszych wartości funkcji celu.

Tak jak wyżej, każda symulacja przyniosła zbliżone efekty. Pozytywnym faktem jest występowanie populacji, gdzie średnia najlepszych osobników przekracza wartość 60, co jest już dobrym wynikiem.

Przebieg wartości najlepszego osobnika jak dotąd dla najlepszej populacji z przetestowanych: Widać, że

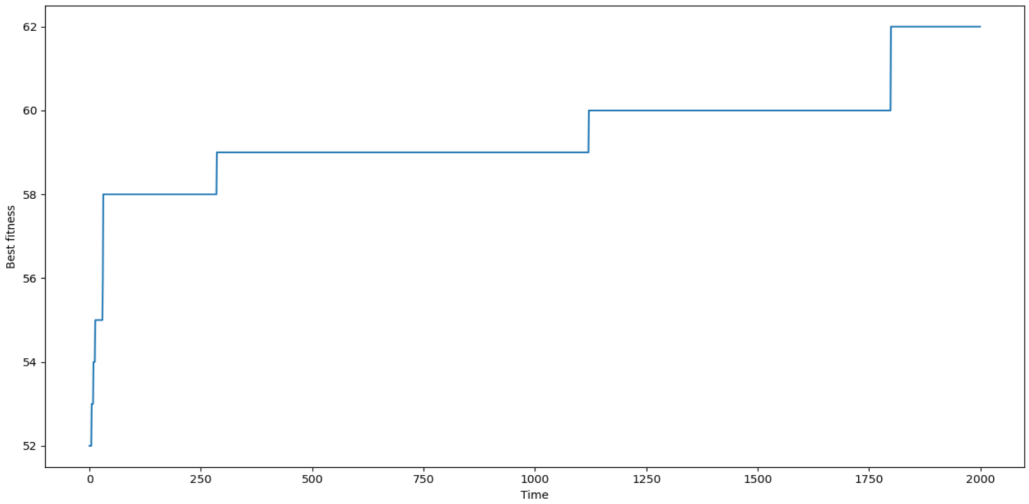

Figure 7: Wartość najlepszego osobnika w czasie

na początku wartość ta szybko rośnie, po osiągnięciu 58 zmiany następują już po większej liczbie iteracji. Przeprowadzałem także testy dla większej liczby iteracji, np. 5000 lub 10000, jednak rzadko przynosiły one lepsze efekty, a znacznie zwiększały czas testów. W zdecydowanej większości przypadków, po 2000 iteracji nie zostały znajdowane lepsze osobniki, stąd testy przeprowadzałem do 2000 iteracji.
Na koniec heatmapa przedstawiająca jednego z najlepszych osobników jakie udało mi się uzyskać o wartości 64:

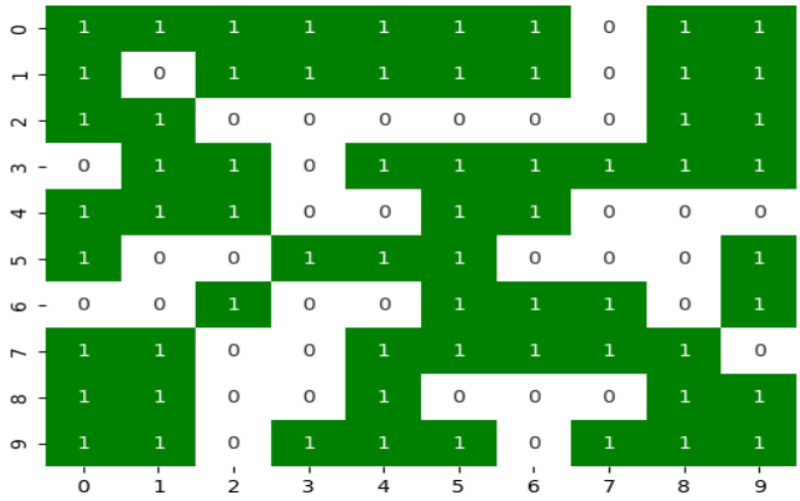

Figure 8: Osobnik o wartości 64.

3 Wnioski

Krzyżowanie jest główną siłą w algorytmie genetycznym, mutacja często jest niepożądana, ale w małych ilościach pozwala na wychodzenie z minimum lokalnych. W przypadku selekcji ruletkowej sprawdzają się małe populacje, aby różnice pomiędzy osobnikami były widoczne. Algorytm dobrze znajduje silnych osobników, jednak ma problemy ze znalezieniem najlepszego, przynajmniej dla omawianego problemu. Osobniki o wartości powyżej 65 były zdecydowaną rzadkością, a najlepszy możliwy ma wartość aż 73, więc w celu znalezienia jego prawdopodobnie należałoby użyć lepszego algorytmu.