Sprawozdanie 3

Metody inteligencji obliczeniowej – Informatyka Stosowana, WFIIS,
Jakub Salamon, II rok

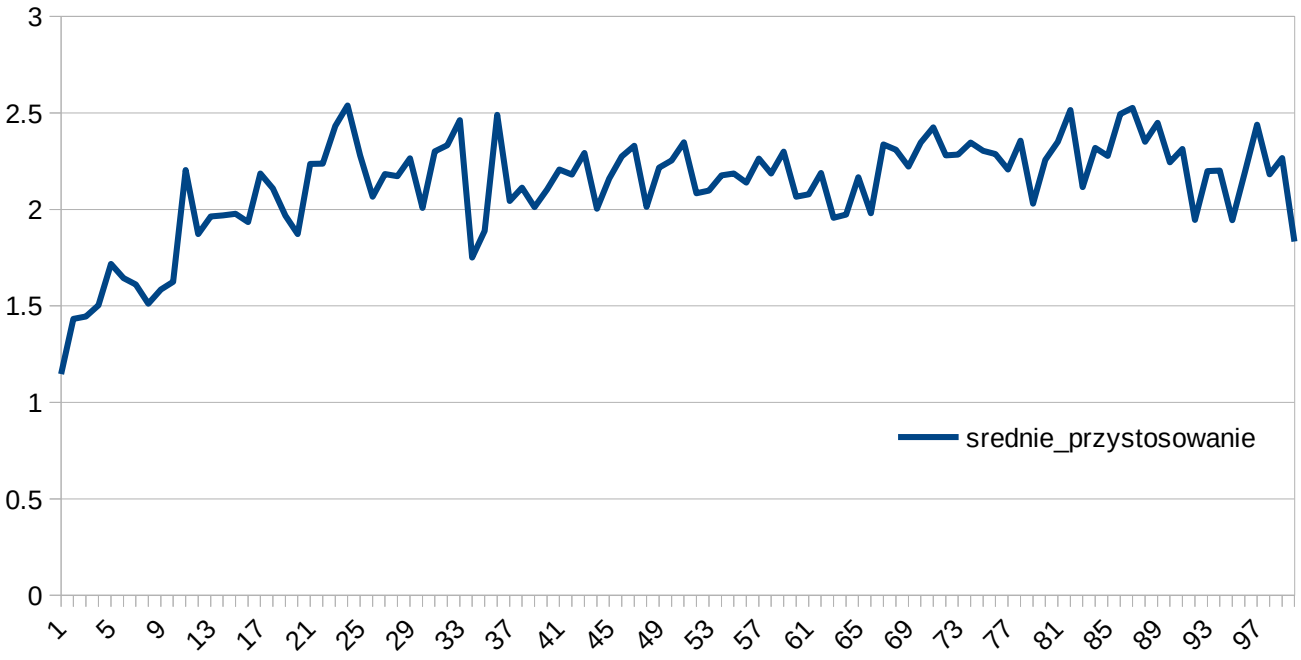
Celem zajęć laboratoryjnych nr 3 było wykorzystanie algorytmu genetycznego do wyznaczenia maksimum funkcji
$f(x) = x \sin(10\pi * x) + 1$, dla x$\in$ [-1, 2]

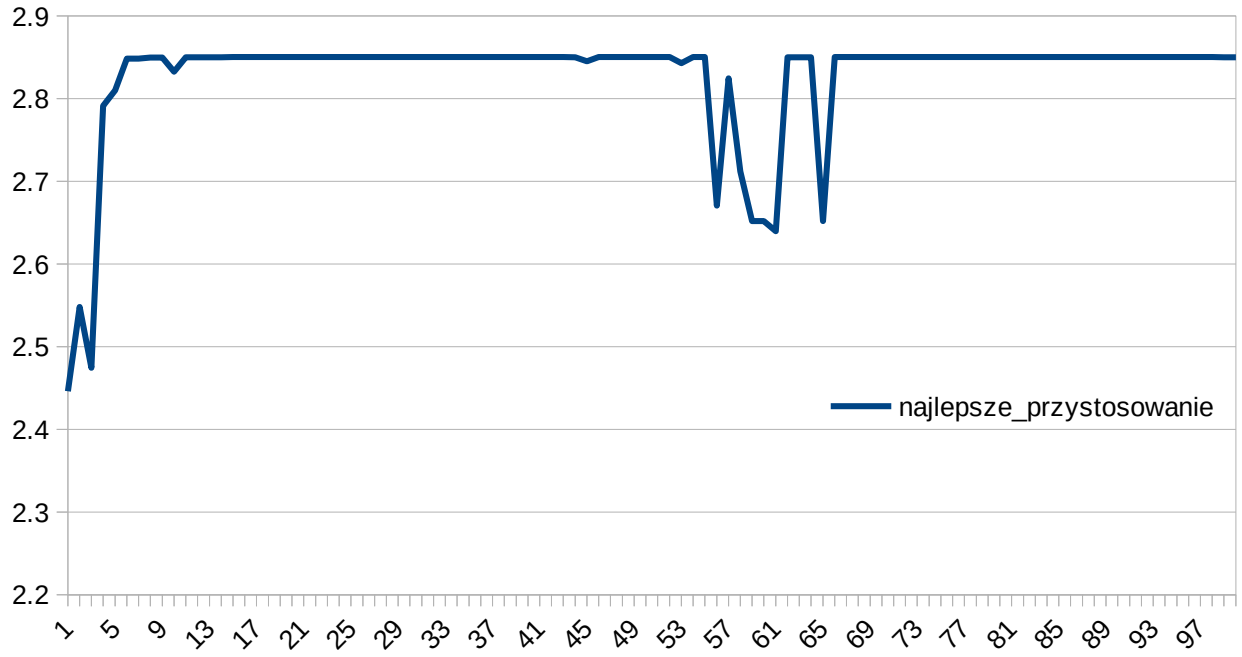
Algorytm genetyczny to rodzaj heurystyki przeszukującej przestrzeń rozwiązań problemu w celu wyszukania najlepszych rozwiązań.

W naszym algorytmie każdy osobnik miał mieć genotyp, który składał z 22 genów będących bitami. Fenotyp takiego genotypu to w takim razie przedział [0, 4194303], więc aby uzyskać rozwiązania algorytmu musieliśmy zrzutować fenotyp na przedział [-1, 2]. Populacja składa się z 20 osobników, a maksymalna liczba ewolucji populacji to 100. Dla każdej epoki obliczamy wartość funkcji przystosowania, następnie dokonujemy selekcji metodą ruletki. Wybieramy punkt w genotypie, którym będziemy krzyżować osobniki. Dzieci tych osobników zastępują rodziców. W każdej z epok osobniki są mutowane. Losowe pmut*N*pop bitów w genotypach są zamieniane na przeciwne z prawdopodobieństwem pmut.

Dla parametrów:
pop = 20 - wielkość populacji
pcross = 0.9 - prawdopodobieństwo krzyżowania
pmut = 0.01 - prawdopodobieństwo mutacji
imax = 100 - maksymalna liczba ewolucji

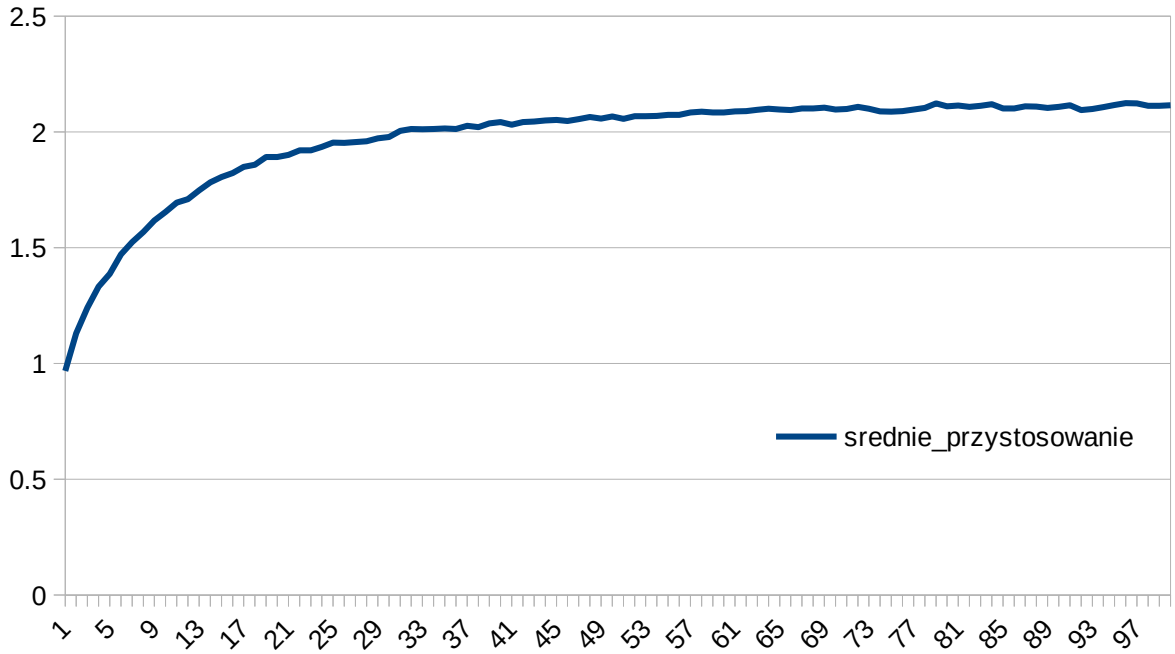
Kodowanie binarne:



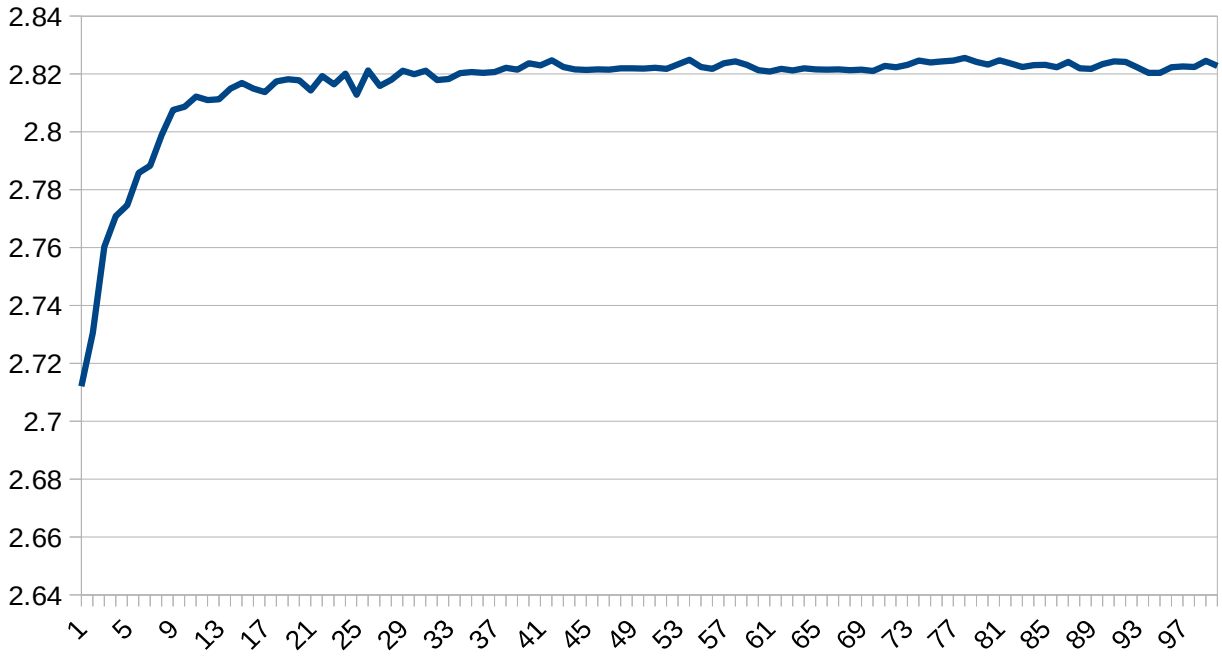


Jak widzimy w prawie każdej kolejnej epoce rośnie wartość średniego przystosowania osobników.

Kodowanie Graya:



Najlepsze przystosowanie:

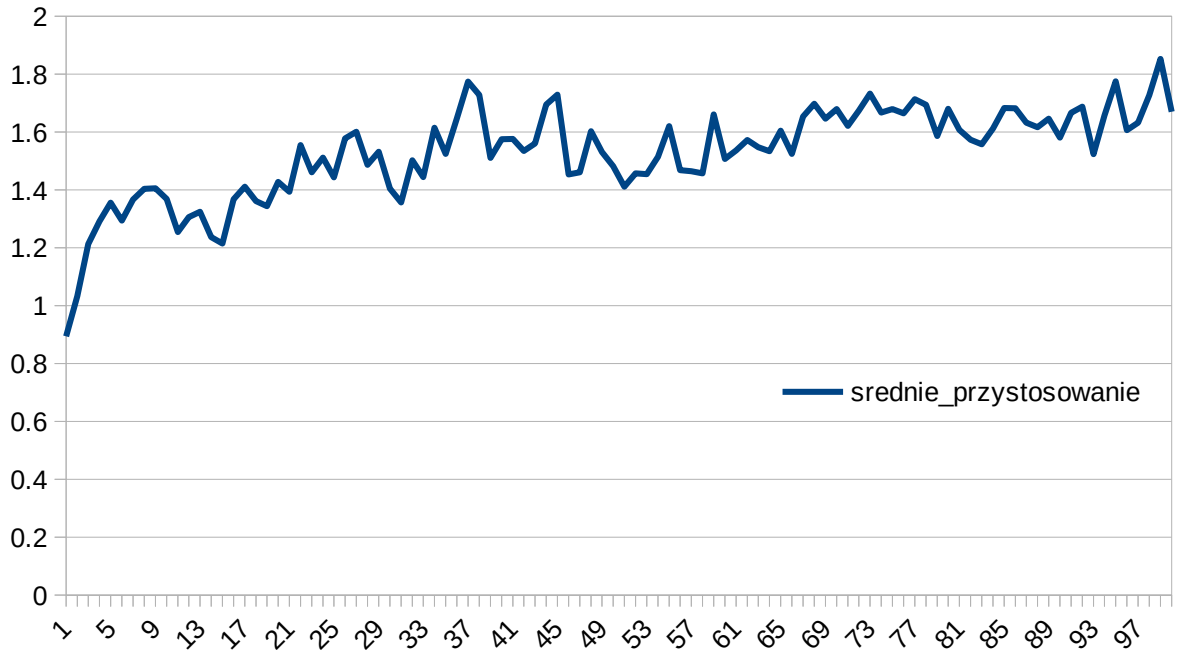


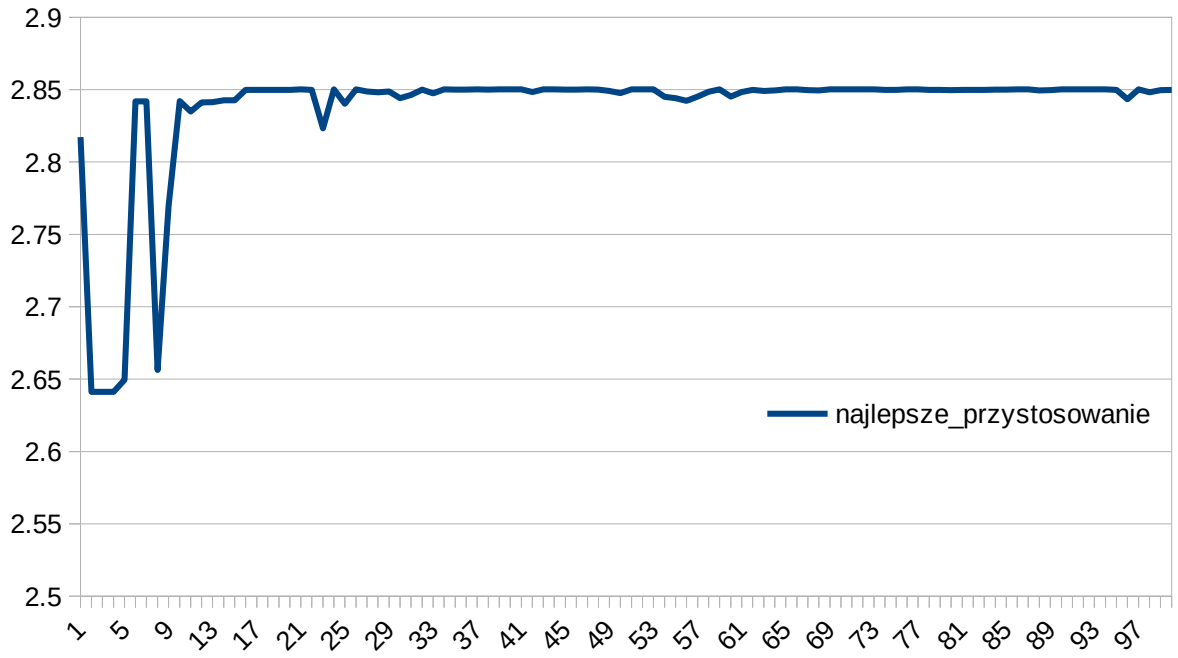
Jak widać na wykresach algorytm dla kodowania Graya jest bardziej stabilny. Wartości nie są tak porozrzucane na wykresach, „łagodniej przechodzimy do rozwiązania”. Można zauważyć, że od pewnej epoki najlepsze przystosowanie jest praktycznie stałe co nie ma miejsca w przypadku kodowania binarnego. Dlatego też do znajdowania maksimum tej funkcji lepiej użyć kodowania Graya. Patrząc na wykresy można wywnioskować, że przy kodowaniu Graya możemy otrzymać satysfakcjonujące nas rozwiązania w mniejszej ilości epok.

Algorytm znalazł maksimum funkcji f(x) dla x = 1.85073181 dla którego wartość
f(x) = 2.85024271774

Test algorytmu dla innych danych

W ramach testowania wpływu zmiany niektórych parametrów na zachowanie algorytmu zmieniłem rozmiar populacji na 100, a prawdopodobieństwo mutacji na 0.04.





Największa wartość funkcji f(x) dla x = 1.85057 wyniosła 2.8502733

Zastosowanie większej populacji poprawiło działanie algorytmu w późniejszych epokach, ale zastosowanie większego prawdopodobieństwa mutacji przyczyniło się większych zachwiań wyników w początkowych epokach. Większa populacja to większa zajętość pamięci. Można wyciągnąć wniosek, że populacja powinna być jak największa, ale musimy zwracać też uwagę na użycie pamięci, bo z mniejszą populacją przecież też osiągniemy zamierzany rezultat. Prawdopodobieństwo mutacji powinno być za to jak najmniejsze.

Ulepszenia, które można wprowadzić do algorytmu

Możemy zastosować inne metody selekcji, np.:

- Strategia elitarna – zachowujemy najlepszego osobnika i wprowadzamy do następnej populacji. Pozwala to na szybsze otrzymanie rozwiązania.
- Selekcja turniejowa – dzielimy populacje na grupy, a wśród nich „rozgrywamy turniej" pomiędzy osobnikami z różnych grup. Do populacji rodziców wybieramy najlepsze osobniki, czyli te które wygrały turniej. Działa ona dużo lepiej niż selekcja metodą ruletki i może być wykorzystywana w zadaniach optymalizacji wielokryterialnej.
- Selekcja rankingowa – nadajemy osobnikom range, wybieramy osobników zgodnie z przypisanymi im rangami. Na podstawie „listy rankingowej" definiujemy funkcję określającą liczbę wybieranych kopii chromosomów w zależności od ich rangi. Nie musimy skalować funkcji przystosowania.

Selekcje turniejowa i rankingowa są odpowiednie zarówno do procesu minimalizacji jaki i maksymalizacji.

Możemy zastosować także inne metody krzyżowania, np.:

- Krzyżowanie wielopunktowe – wybieramy dwa lub więcej punktów krzyżowania, co usprawnia proces krzyżowania, gdy używamy długich chromosomów.
- Krzyżowanie równomierne – losujemy wzorzec, który określa który gen będzie przekopiowany od rodzica 1 a który od rodzica 2.

Możemy także stosować inne metody kodowania chromosomów, szczególnie gdy są one bardzo długie. Skracamy tym czas działania algorytmu, czasami sprawiając tym, że możliwe jest znalezienie rozwiązania problemu. Przykładowe kodowania:

- Kodowanie logarytmiczne – skracamy długość chromosomu przez zastosowanie funkcji wykładniczej. Pierwszy gen to znak liczby wyrażonej za pomocą funkcji wykładniczej, drugi to znak wykładnika, a pozostałe to wykładnik.
- Kodowanie rzeczywiste – geny w chromosomie są wartościami rzeczywistymi. Stosowane jest w przypadku wielowymiarowych przestrzeni potencjalnych rozwiązań. Niestety stosując je odchodzimy od idei algorytmów genetycznych.

Jakub Salamon
Informatyka Stosowana, WFIIS, II rok