**Testowanie algorytmu**

1. **Strojenie.**

Na potrzeby strojenia algorytmu wybraliśmy dwie funkcje (funkcja kwadratowa i funkcja Griewank dim = 1) i na ich przykładzie chcemy pokazać jak zmienia się jego działanie w zależności od zmiennych uczenia się oraz współczynników wagowych.

Zgodnie z założeniami teoretycznymi projektu współczynniki wagowe (tabela dwóch liczb) określają wpływ na wynik algorytmu kolejno: dotychczasowej prędkości cząstki i różnicy jej położenia i położenia najlepszych punktów. Innymi słowy wysoka waga(pierwsza składowa) oznacza mniej dynamiczne zmiany prędkości cząstki. Wagi sumują się do 1.

Współczynnik uczenia decyduje o tym, która cząstka będzie miała większy wpływ na nową prędkość – czy położenie najlepszej cząstki z roju czy najlepsze położenie z historii danej cząstki.

Metodyka strojenia:

*Badaliśmy zachowanie algorytmu dla dwóch różnych funkcji zmieniając ilość cząstek (count) oraz współczynniki uczenia się(lVariable) i dla każdej pary wykonaliśmy zestaw testów na całej przestrzeni wag (weight 0,01 do 0,99) wykonując test po 100 razy dla każdej wagi i licząc średnią wartość wyników.*

Wszystkie wyniki są zestawione w folderze „TestingResults/tuning” odpowiednio nazwane odpowiednimi zmiennymi. Wartość podawana w nazwie pliku to współczynnik uczenia się (odpowiada za eksploatację, odpowiadający za eksploatację jest odwrotnie proporcjonalny)

Analiza:

* Im większy współczynnik uczenia się(a mniejszy wpływ poprzedniej prędkości cząstki) tym większy wpływ na wynik mają wagi (wpływ przeważnie negatywny, odchylenia od ekstremów).
* Obie funkcje osiągają w wartościach brzegowych wag wyniki nieakceptowane.
* W przypadku funkcji kwadratowej dużo lepsze wyniki można osiągnąć zdając się na eksplorację, nie eksploatację (brak ekstremów lokalnych), osiąganą przez zwiększanie wag, przy funkcji Griewank nie jest to zauważalne (wiele ekstremów lokalnych).
* Dla count = 5 statystycznie obie funkcje dają lepsze wyniki dla wyższych learnich variables(przewaga wpływu lidera) oraz wag powyżej 0,5 (łagodne zmiany prędkości)
* Dla count = 20 wyniki są druzgocąco lepsze (zgodnie z oczekiwaniami)
* Dla funkcji Griewank widać mniejszą zależność od współczynnika uczenia się, przy wyższych weight przeważających w stronę utrzymywania dotychczasowej prędkości widać znaczne pogorszenie(duża ilość cząstek zwiększa szanse na to, że któraś trafia blisko minimum)
* Dla funkcji kwadratowej widać niezmiennie znacznie większy wpływ współczynnika uczenia się, najlepsze wyniki osiągamy dla jak najwyższego współczynnika i jak najniższych wag (wpływ na prędkości mają przede wszystkim najlepsze położenia lidera)

Wnioski:

1. Count – warto, żeby było duże
2. Współczynnik uczenia się – trzeba uzależnić od funkcji – jeśli spodziewamy się niewielu minimów lokalnych może być wysokie(wpływ lidera). W wybranych funkcjach najlepiej sprawdzało się ok. 0.7
3. Wagi – trzeba uzależnić od funkcji – jeśli spodziewamy się wielu minimów lokalnych – musi być odpowiednio mniejsze (ok.0,3), jeśli brak minimów lokalnych warto przekroczyć 0,5 (najlepsze wyniki w okolicach 0.5 – 0.7)
4. **Wyniki dla wybranych funkcji**

Wykresy znajdują się w folderze „TestingResults/tuning”.

* 1. De Jong’s function

Parametry – współczynnik uczenia 0.85, waga 0.6

* 1. Rosenbrock’s Valley (tylko dwuwymiarowa)

Parametry – współczynnik uczenia 0.1, waga 0.5

* 1. Easom Function (tylko dwuwymiarowa)

Parametry - współczynnik uczenia 0.1, waga 0.9 (najlepsze wyniki)

* 1. Griewank’s function

Parametry - współczynnik uczenia 0.8, waga 0.3

1. **Wnioski**

Algorytm radzi sobie zgodnie z oczekiwaniami. Jego skuteczność w bardzo dużym stopniu zależy od dobrych współczynników, a te są bardzo różne zależnie od charakteru funkcji celu. Wskazane jest dokładne i szczegółowe strojenie dla konkretnych funkcji, według opisanych zależności. Trudno jest wyznaczyć arbitralnie jakie parametry powinien przyjmować tak by zwracać dobre wyniki.