Optymalizacja wielokryterialna – Ćwiczenie 5.

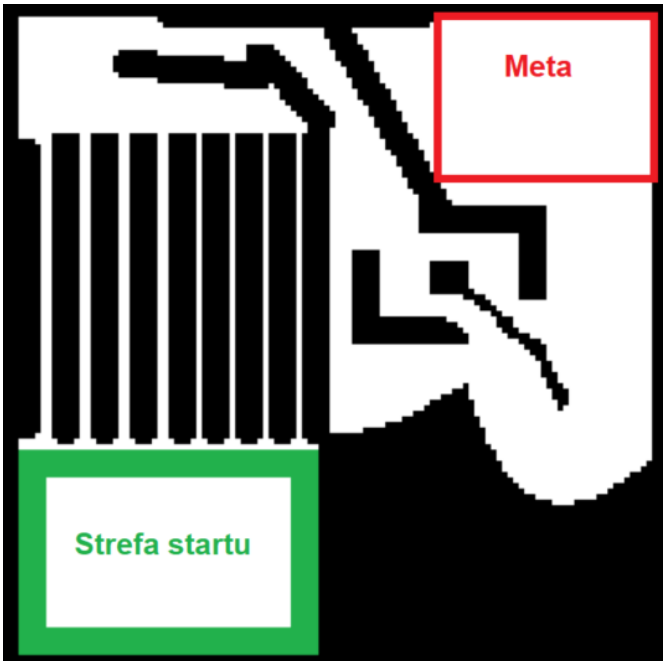
Projekt: Optymalizacja czasu przejazdu oraz uszkodzeń powstałych podczas przejazdu roju robotów przez przeszkodę

1. Wstęp

W pracy zaproponowano sposób na sterowanie rojem robotów minimalizując czas trwania przejazdu oraz prawdopodobieństwo uszkodzenia robota mogące powstać na skutek zbyt bliskich odległości podczas poruszania się. Badany problem można przedstawić następująco:

„Rój robotów ma za zadanie pokonanie przeszkody w jak najkrótszym czasie. Podczas przejazdu roboty mogą zwiększyć prawdopodobieństwo uszkodzenia proporcjonalnie do odległości, która występuje między robotami.”

W pracy założono, że rzut izometryczny mapy, z przeszkodami którą muszą pokonać roboty jest znany. Przykład takiej mapy przedstawiony został na rysunku 1.



Rysunek 1. Przykładowa mapa z przeszkodami.

Założono także że liczba robotów w roju jest znana oraz niezmienna podczas trwania przejazdu, prędkość maksymalna robota jest znana a także położenie robota jest określone jako punkt.

2. Rozwiązanie problemu

W celu rozwiązania problemu zaproponowano połączenie algorytmu wyznaczania trasy ASTAR wraz z algorytmem inteligencji roju PSO. Etapy działania algorytmu przedstawione zostały poniżej:

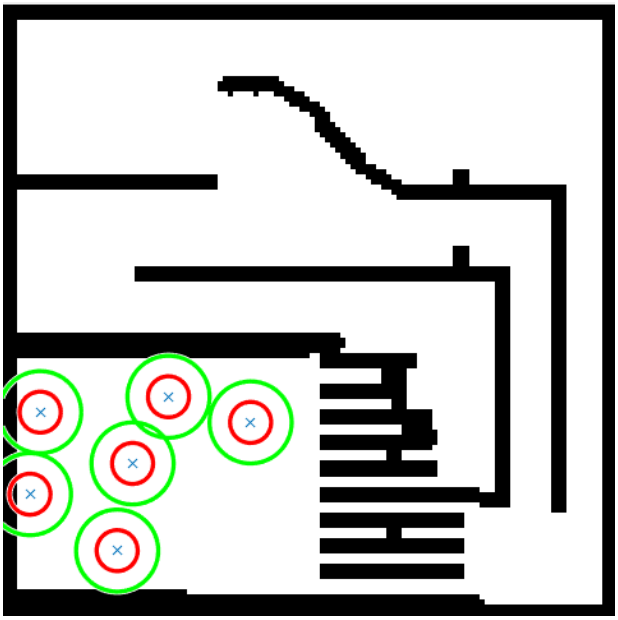
1. Znając liczbę robotów w roju oraz obszar początkowy roboty są losowo rozmieszczane na obszarze strefy startu oraz zadane zostają ich parametry.
2. Na podstawie pozycji startowej wybrany zostaje lider roju.
3. Pozostałe roboty w roju optymalizują swoje położenie tak aby podążać jak najbliżej lidera jednocześnie zachowując odpowiednie odległości w celu uniknięcia uszkodzeń.

2.1 Inicjalizacja programu.

Na początku użytkownik musi zadać odpowiednie parametry w celu uruchomienia programu. Wczytana musi zostać mapa, wybrana liczba robotów w roju oraz ich prędkość. Prędkość robota informuje program o maksymalnej odległości, którą robot może pokonać w trakcie jednej iteracji programu. Wybrany musi zostać także parametr obszaru szczególnej ostrożności(„*caution_distance*”) po przekroczeniu, którego funkcja kary za odległości będzie naliczana.

Wartość maksymalnej prędkości jest także wartością minimalnej odległości, która może dzielić roboty. Zabieg ten ma na celu zapobiegnięcie zderzeniu się robotów oraz dodaniu marginesu pozwalającego na bezpieczeństwo od uszkodzeń „krok do przodu”.

Następnie należy wybrać współrzędne obszaru startowego, który przyjęto jako prostokąt. Wpisując wartości granic X oraz Y obszaru startowego algorytm wylosuje współrzędne startowe dla robota tak aby odległości między nimi były większe niż maksymalna prędkość.



Rysunek 2. Wylosowane współrzędne początkowe robotów w obszarze startowym. Niebiski krzyżyk oznacza położenie robota natomiast czerwony oraz zielony okrąg to odpowiednio granice maksymalnej prędkości oraz obszaru szczególnej ostrożności.

Należy także wybrać punkt końcowy trasy, czyli współrzędne do których powinien dążyć rój robotów.

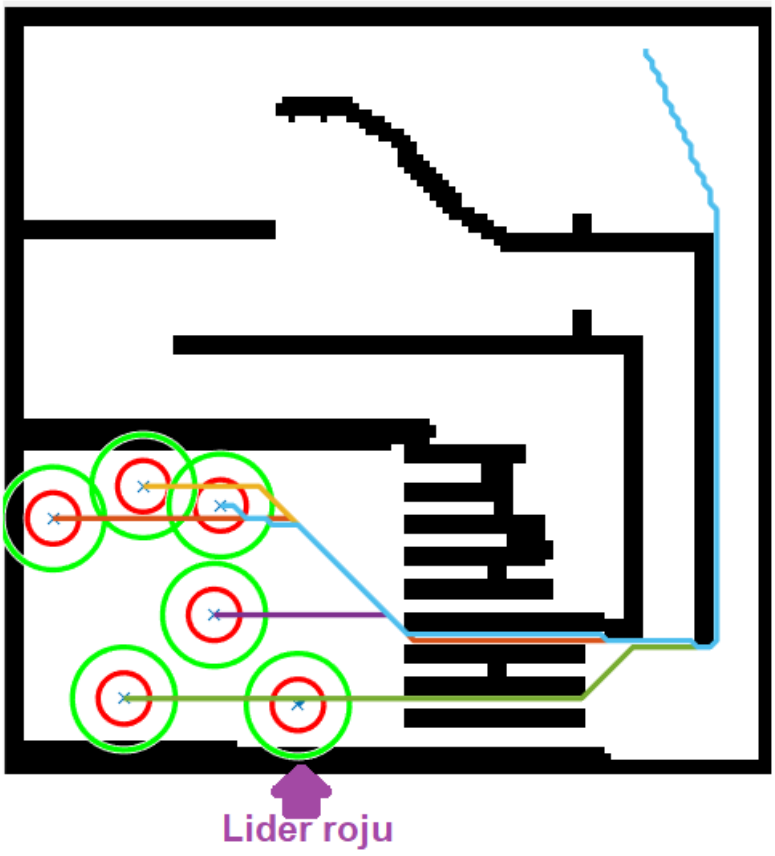
Kolejnym krokiem jest ustalenie parametrów algorytmu PSO. Wybór tych parametrów należy do użytkownika oraz w zależności od badanego problemu może poprawić/pogorszyć wyniki. Parametry te wpływają także na złożoność obliczeniową algorytmu. Wykorzystany algorytm PSO został opisany w rozdziale 3.

2.2 Wybór lidera roju

Wykorzystując początkowe położenia robotów obliczone zostają trasy dla każdego robota do wybranego punktu końcowego z wykorzystaniem algorytmu ASTAR opisanego w rozdziale 3. Punkt z najkrótszą trasą zostaje wybrany liderem roju oraz może poruszać się swobodnie bez potrzeby brania pod uwagę odległości między pozostałymi robotami.

Pozostałe roboty w roju podczas estymacji trasy biorą pod uwagę tylko pozycję lidera oraz podążają za nim dostosowując swoją pozycję tak aby zoptymalizować odległości między sobą, liderem oraz pozostałymi robotami w roju.

Na podstawie długości trasy także wyznaczany jest priorytet ruchu. Estymacja kroku robota ustalony jest w kolejności: Lider -> pozostałe roboty w kolejności od robotów z najkrótszą trasą do robotów z najdłuższą trasą.

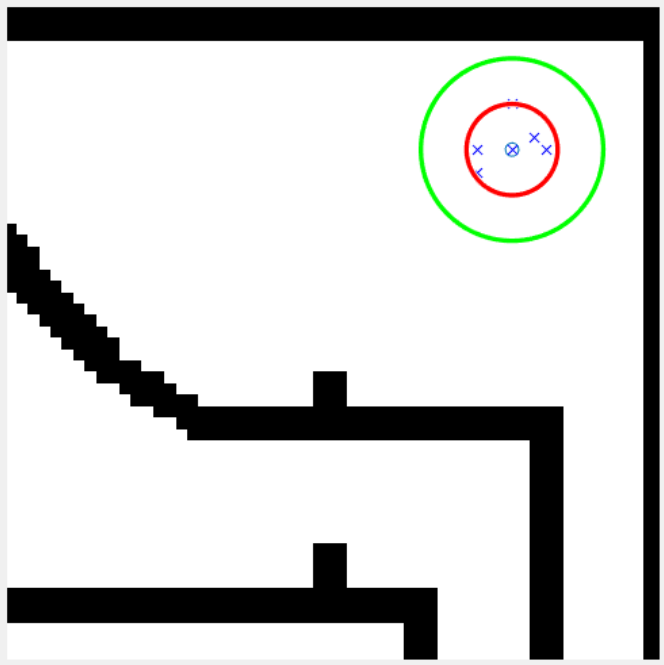


Rysunek 3. Roboty oraz wyznaczone dla nich trasy z zaznaczonym wybranym liderem.

2.3 Wybór optymalnego ruchu oraz aktualizacja położenia.

Każdy członek roju, poza liderem który porusza się swobodnie, musi znaleźć punkt w zasięgu swojej prędkości tak aby minimalizować długość swojej trasy do obecnej pozycji lidera oraz maksymalizować swoją odległość od pozostałych robotów, jeśli znajdują się w jego strefie szczególnej ostrożności. Aby wybrać najlepsze położenie wykorzystano algorytm PSO.

Z wykorzystaniem inteligencji roju szuka się najlepszego położenia w obrębie strefy ruchu robota, biorąc pod uwagę granicę mapy, przeszkody oraz inne roboty. Następnie po znalezieniu najlepszego położenia, pozycja robota jest natychmiastowo aktualizowana w celu pozwolenia pozostałym osobnikom w roju na swobodny ruch z uwzględnieniem zmiany, która nastąpi w trakcie aktualnego kroku.



Rysunek 4. Rój cząstek poszukujący najlepszej pozycji w zasięgu możliwego ruchu robota.

3. Algorytmy wykorzystane w programie

W celu rozwiązania problemu połączono ze sobą dwa algorytmy, algorytm ASTAR pozwalający na znajdywanie najkrótszej trasy oraz algorytm PSO w celu optymalizacji kroku robota.

3.1 Algorytm PSO [1][2]

Algorytm optymalizacji roju cząstek (PSO) jest algorytmem stochastycznym. Algorytm ten symuluje zachowania społeczne zwierząt, w tym owadów, stad, ptaków i ryb. Roje te są zgodne ze wspólnym sposobem znajdowania pożywienia, a każdy członek roju stale zmienia wzorzec wyszukiwania zgodnie z doświadczeń własnych oraz innych członków.

Główna idea projektowa algorytmu PSO jest ściśle powiązana do dwóch badań: Jednym z nich jest algorytm ewolucyjny, tak jak algorytm ewolucyjny. PSO używa również trybu „swarm”, który umożliwia jednoczesne przeszukiwanie dużego obszaru w przestrzeni rozwiązań

zoptymalizowanej funkcji celu. Drugi to sztuczne życie (ang. artificial life), czyli bada sztuczne systemy z cechami życiowymi.
W PSO cząstki mogą aktualizować swoje położenia i prędkości zgodnie ze zmianą otoczenia, czyli spełnione są wymagania sąsiedztwa i jakości. Dodatkowo, rój w PSO nie ogranicza swojego ruchu, ale nieustannie poszukuje optymalnego rozwiązania w przestrzeni. Cząsteczki w PSO mogą zachować stabilny ruch podczas przeszukiwania przestrzeni, zmieniając tryb ruchu, aby dostosować się do zmiany w środowisku.

1. Strategia selekcji elity

Głównym celem strategii selekcji elity jest znalezienie cząstek, które mają doskonałą zdolność konwergencji, adaptacyjność i dystrybucję oraz zatrzymanie tych cząstek, które mają dobrą wydajność.

Następnie elitarne cząstki będą mieć możliwość kierowania zachowaniem wyszukiwania i ewolucją pozostałych cząstek.

Aby rozpocząć nasz proces, cząstki w całej populacji P są ułożone w porządku malejącym zgodnie z tymi wartościami przystosowania, które są obliczane za pomocą równania 1. Natomiast kolekcja Psort jest tworzona w celu reprezentacji posortowanej populacji. Po takim ułożeniu, do elitarnej populacji S dodawana jest cząstka, która ma największą wartość dopasowania. Jej wartość dopasowania jest zapisywana jako fb. Następnie pozostałe cząstki w posortowanym zbiorze Psort, w którym różnice między ich wartością dopasowania i fb, które są mniejsze niż „threshold" e, są uważane za kandydujące cząstki elitarne. Następnie odległości Euklidesa określone przez równanie 2 są obliczane między każdą kandydującą cząstką elitarną a każdą cząstką w elitarnym zbiorze S. Kandydująca cząstka,

w której odległość euklidesowa jest większa niż „threshold" r zostanie dodana do elitarnej populacji S.

$$|X(f)| = \sqrt{X_R^2(f_k) + X_I^2(f_k)}.$$

Równanie (1)

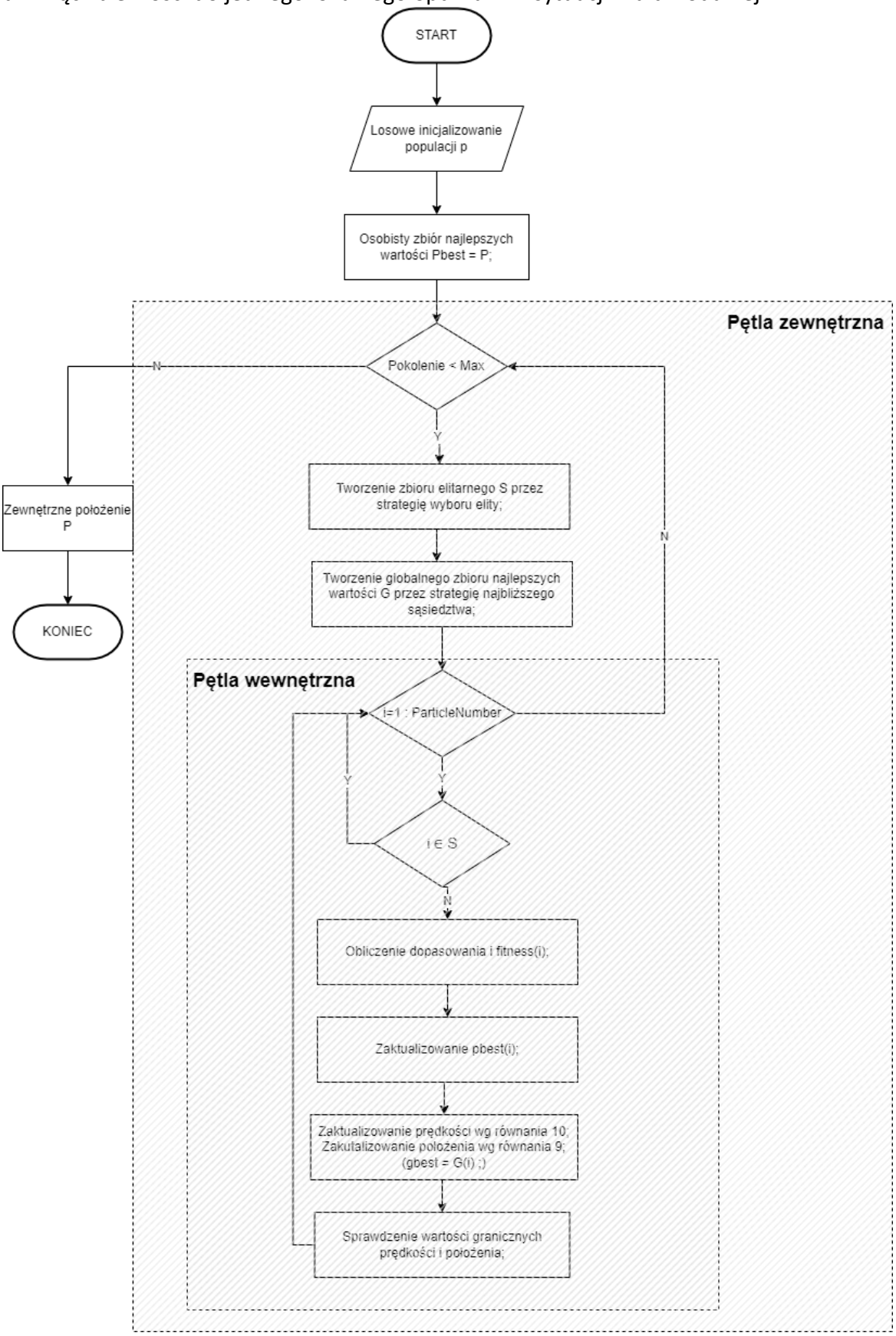
$$Distance(x_i, x_j) = \sqrt{\sum_{d=1}^{m} (x_{i,d} - x_{j,d})^2} \, (i \neq jandi, j \in [1, N]).$$

Równanie (2)

2. Strategia najbliższego sąsiedztwa

Strategia bliskiego sąsiedztwa opiera się na koncepcji niszy w biologii.
W ewolucji biologicznej organizmy zwykle żyją i rozmnażają się z własnym gatunkiem i tworzą liczne subpopulacje. Stan ten zwiększa różnorodność populacji i poprawia zdolność lokalnych poszukiwań.
W przeciwieństwie do klasycznej metody PSO, w której wszystkie cząstki zbiegają się do jednej najlepszej globalnej pozycji, strategia bliskiego sąsiedztwa pozwala cząstkom na ciągłe

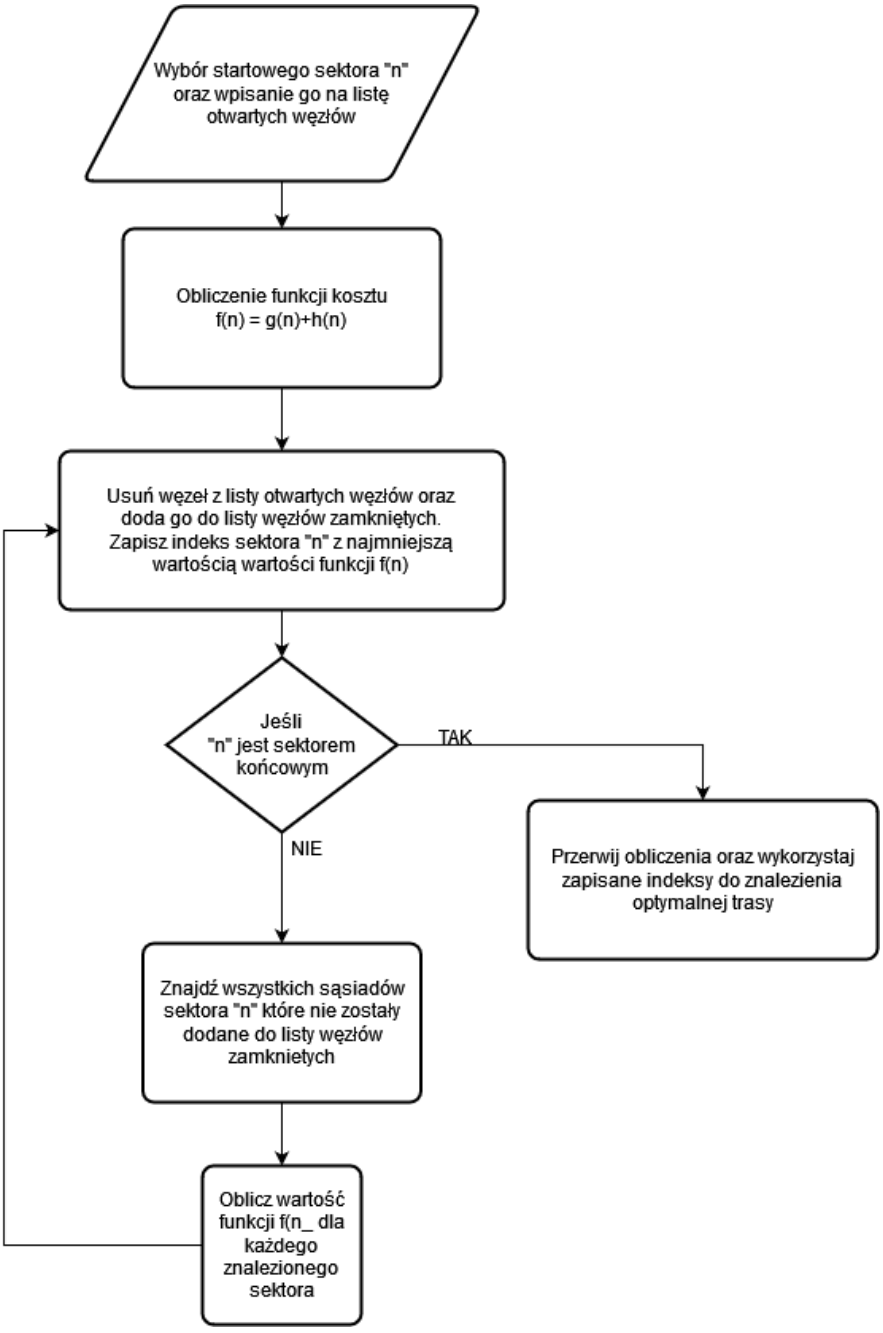
przemieszczanie się do najlepszej pozycji w ich sąsiedztwie. Ta strategia pomaga metodzie PSO uniknąć zbieżności do jednego lokalnego optimum w sytuacji multimodalnej.



Rysunek 5. Flowchart algorytmu PSO.

3.2Algorytm A* [3]

Algorytm A* wykorzystywany jest do znalezienia najkrótszej trasy robota, którą musi pokonać. Algorytm wyszukuje najlepszą trasę przy wykorzystaniu wgranego przez użytkownika obrazu mapy. Flowchart algorytmu został przedstawiony na rysunku 6.



*Rysunek 6. Flowchart algorytmu A**

4. Wyniki

Wyniki działania algorytmu zostały przedstawione na wideo załączonym do projektu.

5. Źródła

[1]. Wang Yifan et. Al., „A High-Precision and Wideband Fundamental Frequency Measurement Method for Synchronous Sampling Used in the Power Analyzer”, Frontiers in Energy Research 9 , 2021.

[2]. Dongshu Wang, Dapei Tan1, Lei Liu, „Particle swarm optimization algorithm: an overview”, Springer-Verlag Berlin Heidelberg 2017

[3]. Zidane, Issa & Ibrahim, Khalil. (2018). Wavefront and A-Star Algorithms for Mobile Robot Path Planning. 69-80. 10.1007/978-3-319-64861-3_7.