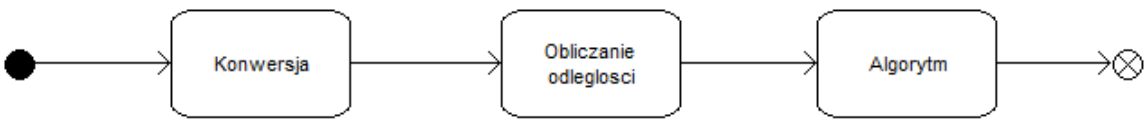
Wstępny projekt aplikacji

Spis treści

1 Główne fazy aplikacji



1-1 Ogólny przebieg aplikacji

Aplikacja dzieli się na trzy fazy:

- Konwersja – dokonuje się tu selekcji i ekstrakcji cech i rekordów bazy i zapisuje się w postaci płaskich wektorów.(2)

- Obliczanie odległości – Tworzona jest i wypełniana matryca odległości
- Algorytm – właściwy algorytm klasteryzacji

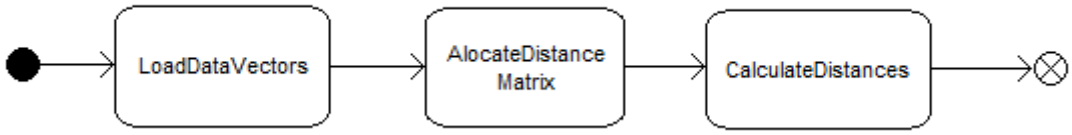
2 Konwersja

Jest to kod w dużej mierze uzależniony od budowy źródła danych. Jego zadaniem jest ekstrakcja wszystkich rekordów i zapisanie ich w postaci wektora wartości. Ta część aplikacji w całości wykonuje się wyłącznie na procesorze.

Faza ta jest odnawialna. Tzn., gdy już raz dokonamy konwersji, otrzymana postać może zostać zapisana i użyta ponownie omijając ten etap.

3 Obliczanie odległości

Korzystając z produktu poprzedniej fazy, tworzymy i wypełniamy macierz odległości.

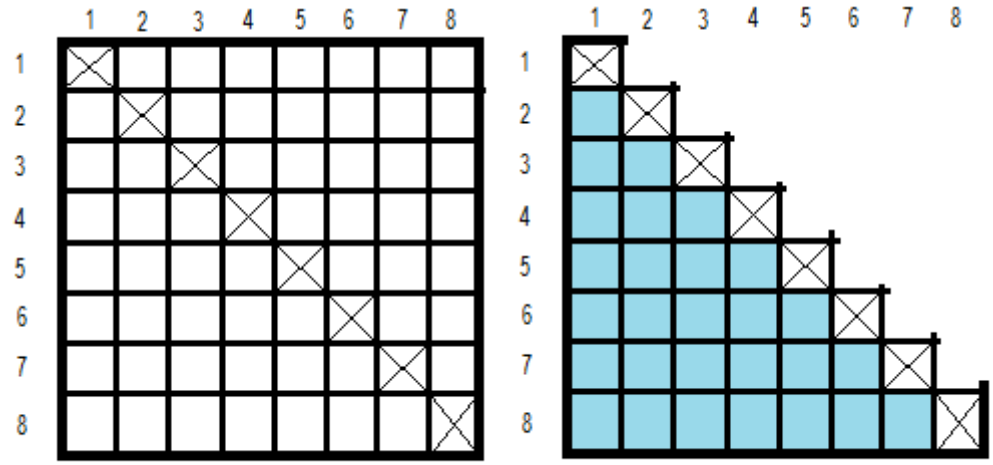


3-1 Ogólny przebieg fazy obliczania odległości

- Load Data Vectors – wczytanie rekordów do globalnej pamięci karty
- Allocate Distance Matrix – Alokuje miejsce dla matrycy w globalnej pamięci karty.
- Calculate Distances – obliczanie odległości

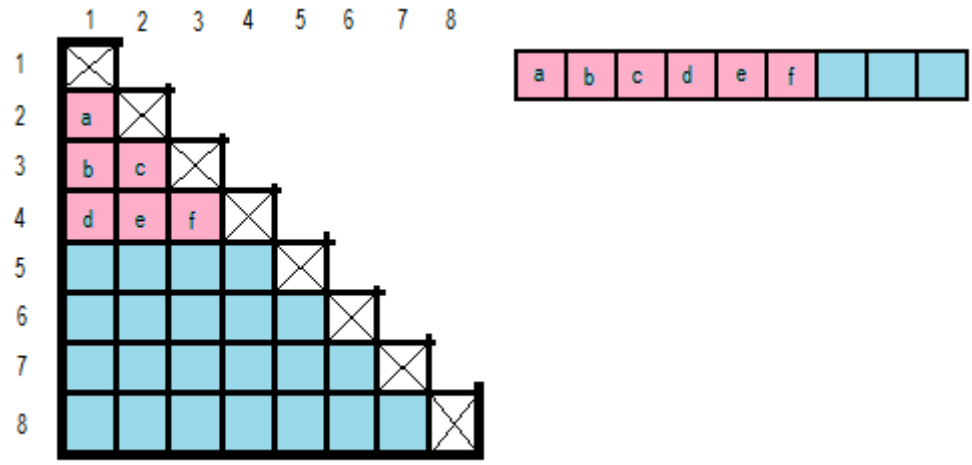
3.1 Macierz odległoŝi

W niej umieszczony zostanie wynik obliczeń i na niej opierać będzie się działanie właściwego algorytmu. Obliczenia dokonywane są tylko raz.



rys 3-2 Macierz odległości i jej część która podlega oblczeniom

Dzięki symetryczności tablicy wystarczy dokonać jedynie $N(N-1)/2$ porównań (gdzie N to liczba rekordów). Najwygodniej jest to jednak zapisać za pomocą wektoru



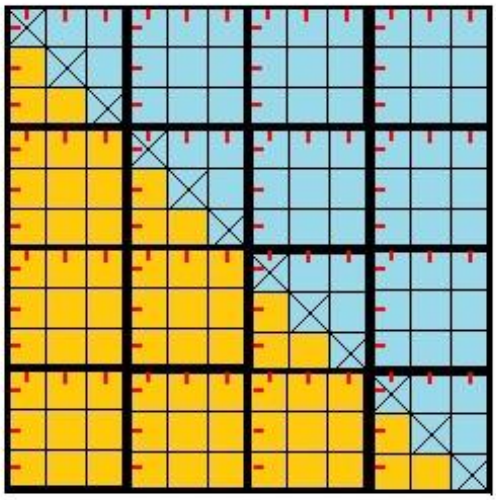
rys 3-3 Tłumaczenie współrzędnych macierzy na wektor

Zakładając indeksowanie od 0 (wiersz, kolumna w macierzy jak i pozycja w wektorze), pozycję w wektorze można uzyskać następującym wzorem:

$$index = \begin{cases} \frac{y(y-1)}{2} + x, dla\ y > x \\ \frac{x(x-1)}{2} + y, dla\ x > y \end{cases}$$

3.2 Obliczenia odległości

Macierz dzielona jest na bloki w dwóch wymiarach. Ilość bloków uzależniona jest od wielkości rekordów i ich ilości jaka może zostać załadowana do pamięci współdzielonej. Blok składa się z macierzy wątków (po jednym na każdą przeliczaną parę).

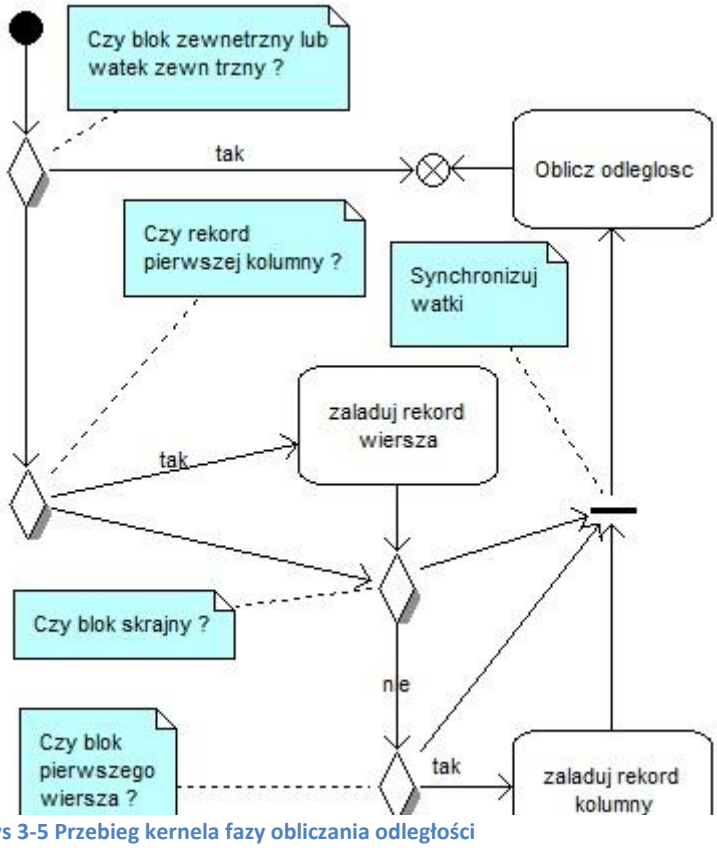


rys 3-4 Typy bloków i skrajne wątki

Bloki można podzielić na trzy rodzaje:

- **Wewnętrzny** – (w pełni pomarańczowy) Tak kolumny jak i wiersze nie powtarzają się. Wszystkie wątki biorą udział w obliczeniach.
- **Skrajne** – (tylko częściowo pomarańczowe) Bloki zawierające przekątną macierzy. Tylko część wątków bierze udział w obliczeniach.
- **Zewnętrzne** – (w pełni niebieskie) Bloki te zawierają wyłącznie wątki, które nie biorą udziału w obliczeniach. Wątki te są zakańczane zaraz po starcie.

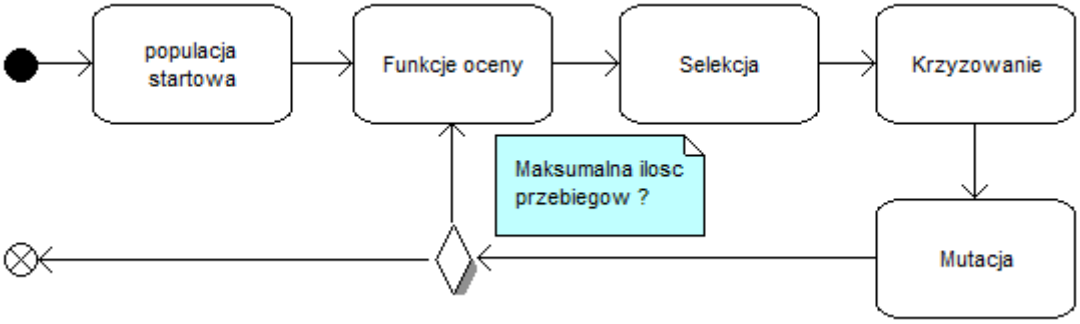
Wątki skrajne – wątki pierwszej kolumny i pierwszego wiersza bloku. Odpowiedzialne są za ładowanie do pamięci współdzielonej rekordów które wezmą udział w obliczeniach. W blokach skrajnych, jako, że rekordy się powtarzają, ładowanie przeprowadzane jest wyłącznie przez wątki



rys 3-5 Przebieg kernela fazy obliczania odległości

pierwszej kolumny. W blokach zewnętrznych nie ładujemy nic.

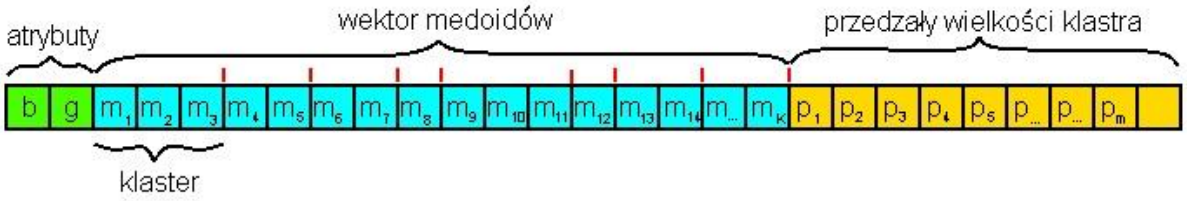
4 Algorytm ewolucyjny



rys 4-1 ogólny przebieg algorytmu ewolucyjnego

4.1 Populacja startowa

Generowanie startowych członków populacji.

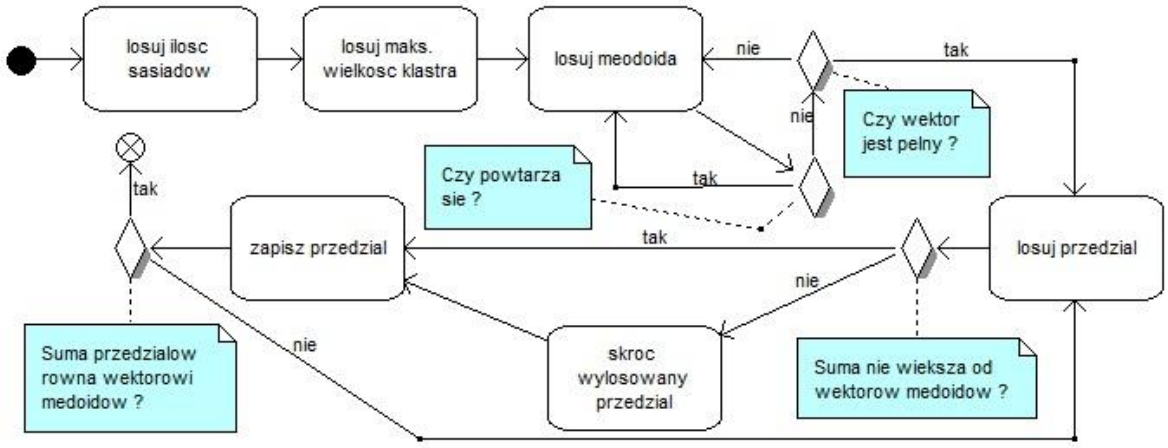


rys 4-2 Budowa pojedynczego rozwiązania.

Chromosom dzieli się na trzy części

- Atrybuty
  - b – ilość sąsiadów brana pod uwagę przy kryterium łączności
  - g – maksymalna wielkość pojedynczego klastra
- Wektor K medoidów
- Wektor klastrów – przechowują wielkości kolejnych klastrów. Suma tych wartości jest równa K

Losowanie odbywa się po jednym wątku na rozwiązanie.



4-3 Generowanie rozwiązań startowych.

4.2 Funkcja oceny

Właściwa funkcja oceny jest podzielona poprzedzona jest fazą funkcji przynależności

4.2.1 Faza przynależności i kryterium gęstości

Funkcja ta sprawdza przynależność rekordu do klastra. Algorytm jest prosty:

- Z wektora medoidów znajdź ten, który jest najbliższy.
- Następnie sprawdź do którego należy on klastra
- Przypisz rekordowi ten klaster.
- Dodaj odległość rekordu do medoida do sumy odległości (kryterium gęstości)

Po jednym wątku na rekord (maks 512 w bloku). Bloki w dwóch wymiarach. Pierwszy to ilość rozwiązań. Drugie to bloki per rozwiązanie, aby wypełnić ilość rekordów wątkami.

Wynikiem jest tablica przypisująca każdemu rekordowi najbliższy medoid

4.2.2 Faza oceny

4.2.2.1 Kryterium Łączności

Podobnie jak w poprzedniej fazie, po jednym wątku na rekord (maks 512 w bloku). Bloki w trzdwóch wymiarach. Pierwszy – ilość rozwiązań. Drugi – ilość bloków per rozwiązanie (aby pokryć wszystkie rekordy wątkami).

- Dla każdego z b sąsiadów
    - Jeśli sąsiad należy do tego samego klastra, zwiększ sumę o $1/b$.
- Sumę dodaj do globalnej dla danego rozwiązania

4.2.2.2 Kryterium Rozłączności

Po jednym bloku na rozwiązanie. Po jednym wątku na medoid.

Oblicza łączną sumę odległości MND (Mutual Neighbour Distance) do każdego medoida nie należącego do tego samego klastra. Następnie wyniki z wszystkich wątków są sumowane globalnie dla rozwiązań.

4.2.2.3 Kryterium poprawności

Po jednym wątku na rozwiązanie.

Ocenia poprawność budowy chromosomu. Obniżając prawdopodobieństwo krzyżowania w przypadku poważnych błędów.

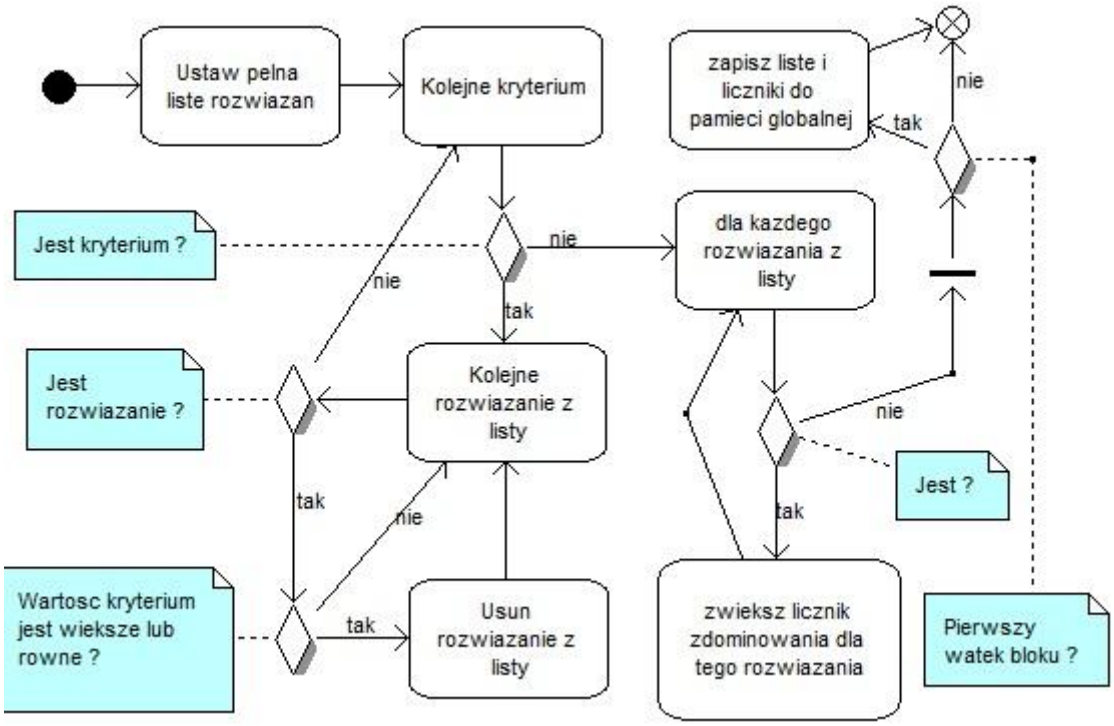
- Powtarzalność medoidów

4.3 Selekcja

Selekcja za pomocą algorytmu NSGA-II

4.3.1 Sortowanie

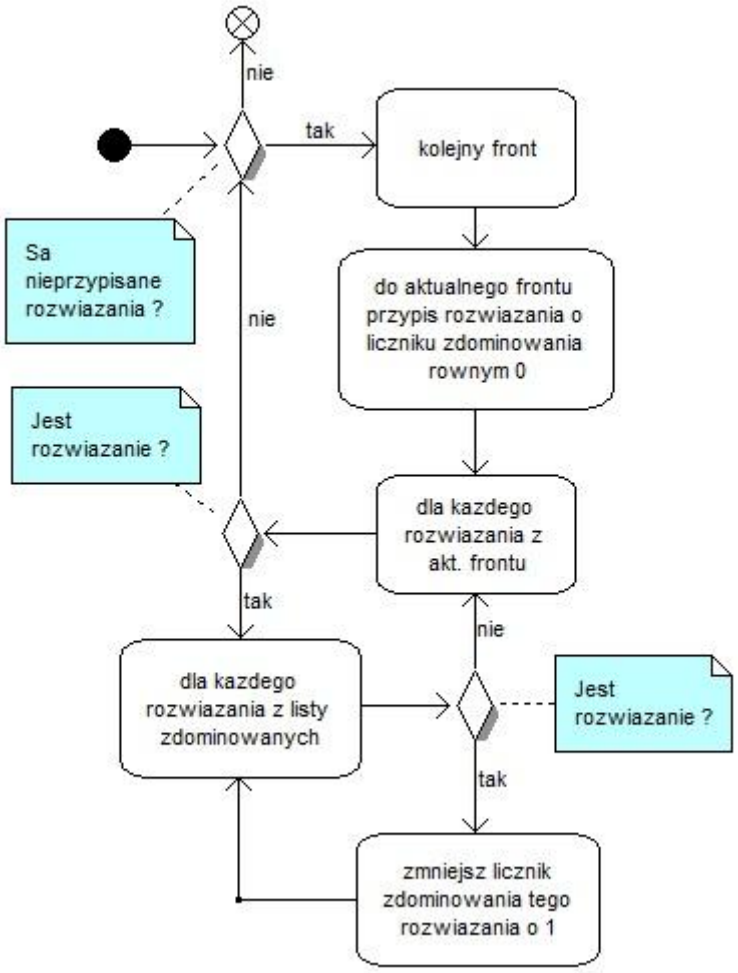
Każdemu rozwiązaniu przypisuje się listę innych rozwiązań, które to dominuje. Oraz liczbę rozwiązań, przez które jest zdominowane.



Po jednym wątku na rozwiązanie. Podział na grupy jedynie dla zwiększenia wydajności i/lub gdy ilość rozwiązań w populacji przekracza 512.

4.3.2 Grupowanie

Podział na fronty według poziomu dominacji.

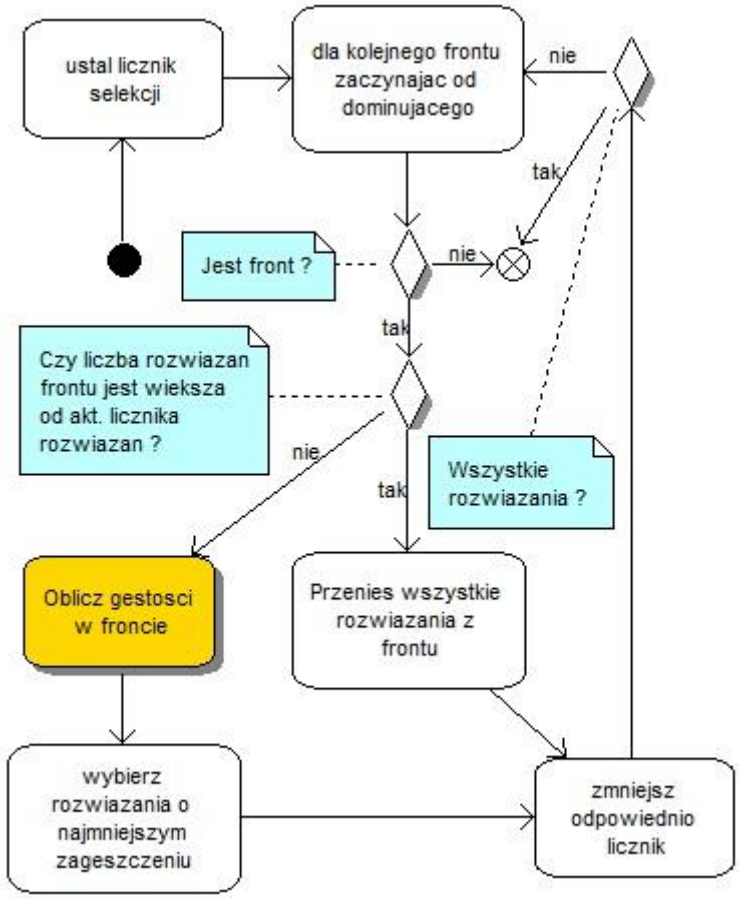


4-4 Grupowanie rozwiązań na fronty według dominacji.

Ta część kodu wykonywana jest w pełni po stronie Hosta na CPU.

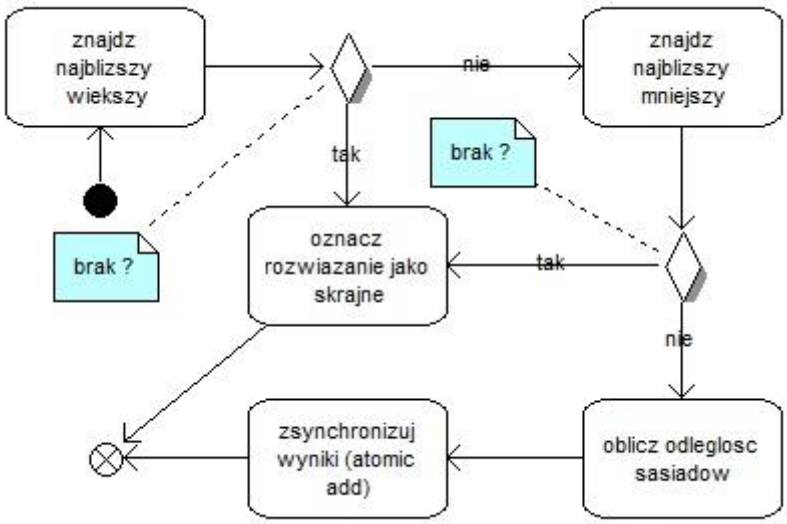
4.3.3 Selekcja

Selekcja według podziału na fronty i gęstości rozwiązań w danym.



4-5 Selekcja rozwiązań

Część w większości wykonywana po stronie CPU. Poza przypadkiem, gdy z frontu wybieranych jest tylko kilka rozwiązań. Wtedy wstępnie oblicza się gęstości rozwiązań.

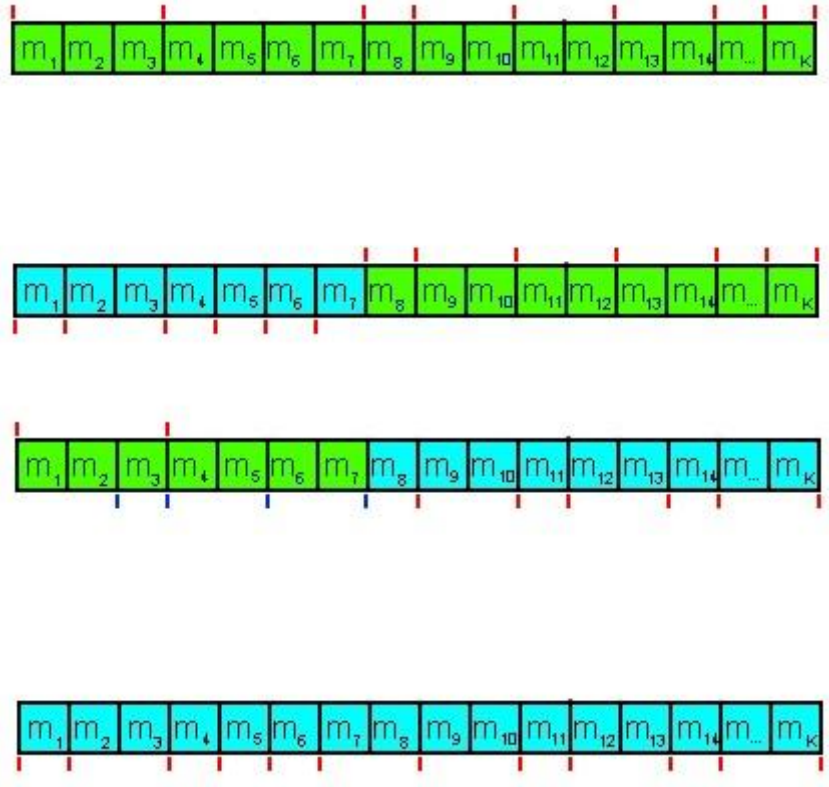


4-6 Obliczanie gęstości rozwiązania.

Wątki w dwóch wymiarach: pierwszy - po jednym na każde rozwiązanie. Drugi – podział na kolejne kryteria w danym rozwiązaniu. Podział na bloki liniowo, jeśli rozwiązań do wyboru jest więcej niż 125 i/lub gdy wyniki operacji nie mieszczą się w pamięci dzielonej bloku.

4.4 Krzyżowanie

W trakcie krzyżowania wymianie podlegają wektor medoidów i grupy w nim zawarte.



4-7 Krzyżowanie dwuch rozwiązań

Atrybuty nie podlegają krzyżowaniu. Jeśli w wynikowym rozwiązaniu grupa jest za duża, to jest ona dzielona

Krzyżowanie odbywa się na GPU, po jednym wątku na każdą parę biorącą udział w losowaniu.

4.5 Mutacja

Mutacji mogą podlegać:

- Wektor medoidów – zmianie ulega jeden medoid
- Wektor grup – zmienia się wielkość losowej grupy
- Wektor atrybutów – zmienia się jeden z atrybutów