Wstępny projekt aplikacji

Spis treści

[1 Główne fazy aplikacji 1](#_Toc291793448)

[2 Konwersja 2](#_Toc291793449)

[3 Obliczanie odległości 2](#_Toc291793450)

[3.1 Macierz odległośi 2](#_Toc291793451)

[3.2 Obliczenia odległości 3](#_Toc291793452)

[4 Algorytm ewolucyjny 5](#_Toc291793453)

[4.1 Populacja startowa 5](#_Toc291793454)

[4.2 Funkcja oceny 6](#_Toc291793455)

[4.2.1 Faza przynależności i kryterium gęstości 6](#_Toc291793456)

[4.2.2 Faza oceny 6](#_Toc291793457)

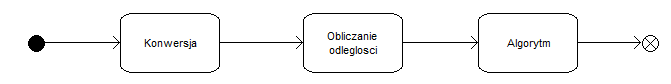
[4.3 Selekcja 7](#_Toc291793458)

[4.3.1 Sortowanie 7](#_Toc291793459)

[4.3.2 Grupowanie 7](#_Toc291793460)

[4.3.3 Selekcja 7](#_Toc291793461)

# Główne fazy aplikacji



‑ 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

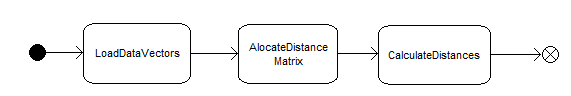
# Konwersja

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.

# Obliczanie odległości

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

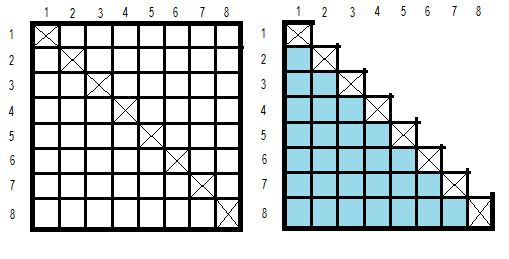


‑ Ogólny przebieg fazy obliczania odległości

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

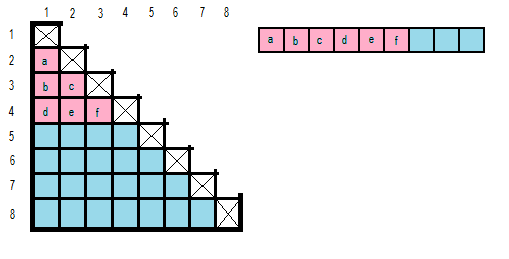
## Macierz odległośi

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



rys ‑ Macierz odległości i jej część która podlega oblczeniom

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

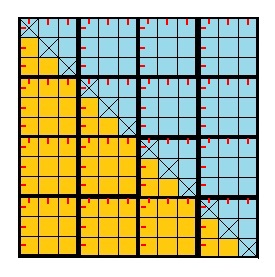


rys ‑ Tłumaczenie współrzędnych macierzy na wektor

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

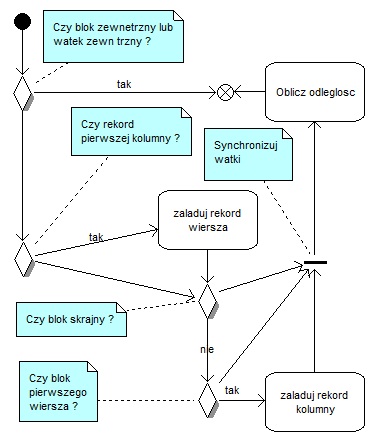
## Obliczenia odległości

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).

Bloki można podzielić na trzy rodzaje:

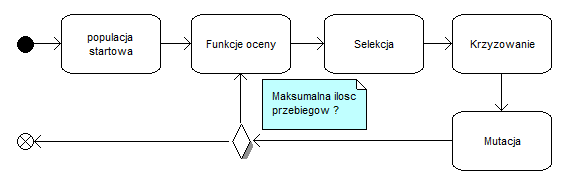
rys ‑ Typy bloków i skrajne wątki

* **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 pierwszej kolumny. W blokach zewnętrznych nie ładujemy nic.

rys ‑ Przebieg kernela fazy obliczania odległości

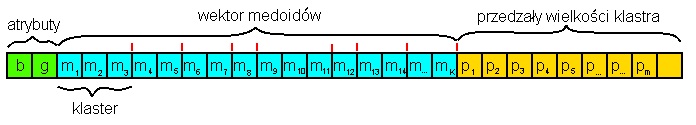
# Algorytm ewolucyjny



rys ‑ ogólny przebieg algorytmu ewolucyjnego

## Populacja startowa

Generowanie startowych członków populacji.

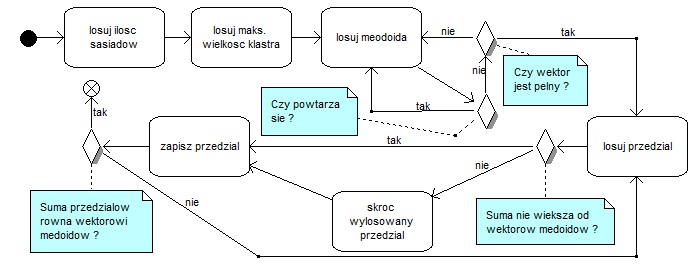


rys ‑ Budowa pojedynczego rozwiązania.

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.



‑ Generowanie rozwiązań startowych.

## Funkcja oceny

Właściwa funkcja oceny jest podzielona poprzedzona jest fazą funkcji przynależności

### 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

### Faza oceny

#### 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 sąsiadów
  + Jeśli sąsiad należy do tego samego klastra, zwiększ sumę o .
* Sumę dodaj do globalnej dla danego rozwiązania

#### 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.

#### Kryterium poprawności

Po jednym wątku na rozwiązanie.

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

* Powtarzalność medoidów

## Selekcja

### Sortowanie

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

### Grupowanie

Podział na fronty według poziomu dominacji.

### Selekcja

Selekcja według podziału na fronty i gęstości rozwiązań w danym.

## Krzyżowanie

## Mutacja