Wstępny projekt aplikacji

Spis treści

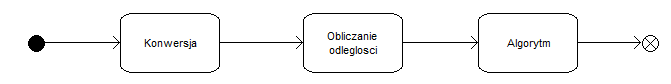
[1 Główne fazy aplikacji 1](#_Toc291215544)

[2 Konwersja 1](#_Toc291215545)

[3 Obliczanie odległości 1](#_Toc291215546)

[3.1 Macierz odległośi 2](#_Toc291215547)

# 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

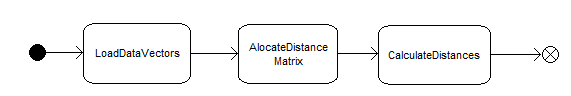
# Konwersja

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.

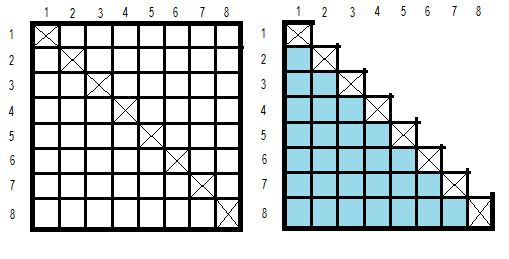


‑ Ogólny przebieg fazy obliczania odległości

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

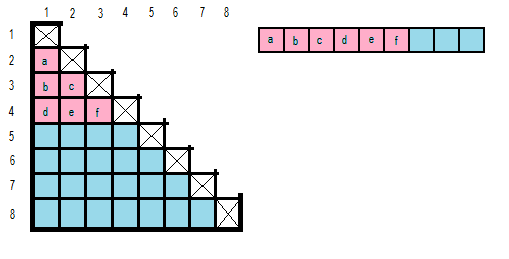
## Macierz odległośi

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

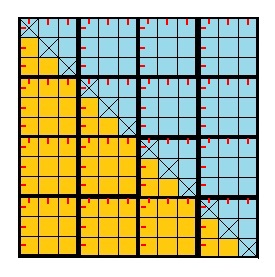


rys ‑ Tłumaczenie współrzędnych macierzy na wektor

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

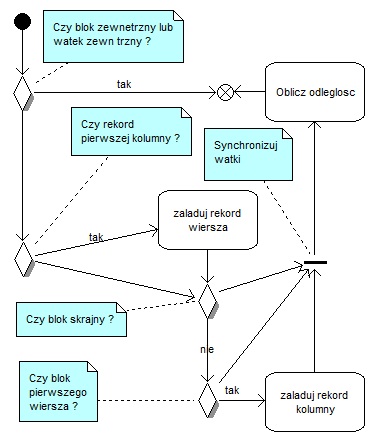
## Obliczenia odległości

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:

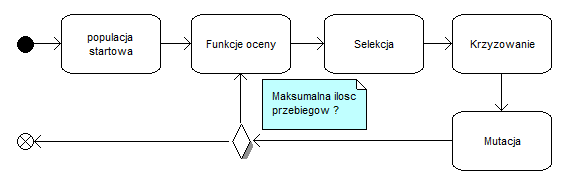
rys ‑ Typy bloków i skrajne wątki

* **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

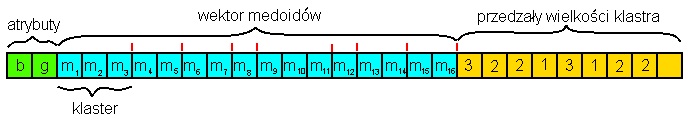
# Algorytm ewolucyjny



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