Politechnika Wrocławska

Wydział Informatyki i Zarządzania
kierunek studiów: wpisz właściwy
specjalność: wpisz właściwą

Praca dyplomowa - magisterska

Algorytmy ewolucyjne w zadaniu klasteryzacji

Jarosław Wojtasik

słowa kluczowe:
1 linia
2 linia
3 linia

krótkie streszczenie:
1 linia
2 linia
3 linia
4 linia
5 linia
6 linia

Promotor:	.....	.....	.....
	imię i nazwisko	*ocena*	*podpis*

*Do celów archiwalnych pracę dyplomową zakwalifikowano do:**
a) kategorii A (akta wieczyste)
b) kategorii BE 50 (po 50 latach podlegające ekspertyzie)
** niepotrzebne skreślić*

Pieczątka instytutu, w którym student wykonywał pracę

Wrocław 2011

Zawartość

1 Wstęp

2 Wprowadzenie w problematykę pracy

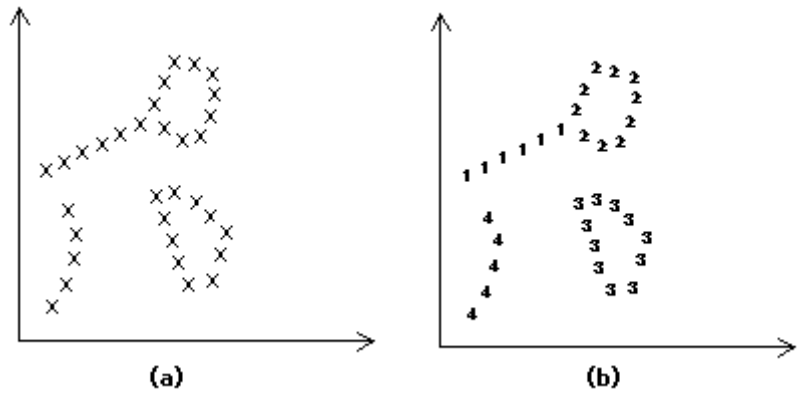
Rozdział ten zawiera teoretyczne wprowadzenie do tematu, którego dotyczy niniejsza praca. Opisano w nim pokrótce Klasteryzację oraz algorytmy ewolucyjne.

2.1 Klasteryzacja

Klasy lub, koncepcyjnie mające sens, grupy obiektów, współdzielących pewne cechy, odgrywają ważną rolę w sposobie, w jaki ludzie analizują i opisują świat. Faktycznie, ludzie posiadają dobre umiejętności w podziale obiektów na grupy (klasteryzacja) oraz w przypisywaniu poszczególnych obiektów do tych grup (klasyfikacja). Dla przykładu, nawet stosunkowo małe dzieci mogą szybko określić obiekty na zdjęciu jako budynki, samochody, ludzie, zwierzęta, rośliny, itp. W kontekście rozumienia danych, klastry są potencjalnymi klasami a analiza klastrów to badanie technik automatycznego wyszukiwania tych klas [1].

Analiza klastrów grupuje obiekty danych bazując jedynie na informacjach znalezionych w danych opisujących obiekty i ich relacje. Celem jest, aby obiekty znajdujące się w tej samej grupie były sobie podobne (lub powiązane), a różniły się (lub były nie powiązane) z obiektami w innych grupach. Im większe jest podobieństwo wewnątrz grupy, a różnica pomiędzy grupami, tym lepsza lub wyraźniejsza jest klasteryzacja [1].

Przykład klasteryzacji przedstawiono na rys 1. Celem jest opracowanie automatycznego algorytmu, który odkryje naturalne ugrupowania (rys 1b) w nie oznaczonych danych (rys 1a). Z rys 1 wynika, że klastry mogą różnić się pod względem kształtu, wielkości oraz gęstości. Obecność szumu w danych czyni wykrywanie klastrów jeszcze trudniejszym. Idealny klaster może zostać zdefiniowany jako zbiór punktów, zwarty i odizolowany. Jednakże w rzeczywistości, klaster to subiektywny byt będący w oku patrzącego, którego znaczenie i interpretacja wymaga wiedzy na temat domeny. Gdy jednak ludzie są doskonali w wyszukiwaniu klastrów w dwóch i prawdopodobnie w trzech wymiarach, potrzebujemy zautomatyzowanych algorytmów dla danych wielowymiarowych. To i niewiadoma ilość klastrów stały się wyzwaniem, które wynikło tysiącami algorytmów, które zostały opublikowane i nadal są.



rys 1 - Przykład zbioru danych i wyniku jego klasteryzacji

2.1.1 Reprezentacja rekordu

Rekord może mierzyć tak fizyczny obiekt (np. krzesło) jak i abstrakcyjny (np. styl pisania). Jak wspomniano powyżej, każdy obiekt reprezentowany jest, jako wektor, gdzie każdy wymiar jest pojedynczą cechą. Cechy te można podzielić na ilościowe i jakościowe. Dla przykładu, jeśli waga i kolor były by dwiema użytymi cechami, wtedy (20, black) jest reprezentacją czarnego obiektu o wadze 20 jednostek. Cechy można podzielić na następujące typy.

(1) Cechy ilościowe
   A. Wartości ciągłe (np. waga)
   B. Wartości dyskretne (np. liczba komputerów)
   C. Wartości przedziału (np. czas przebiegu wydarzenia)

(2) Cechy jakościowe
   A. Nominalne lub nieposortowane (np. kolor)
   B. Porządkowy (np. ranga wojskowa, temperatura („gorąco”, „zimno”), czy intensywność głośności („cicho”, „głośno”)

Można także używać cech ustrukturyzowanych, reprezentowanych przez drzewo, gdzie rodzic jest generalizacją swoich dzieci. Dla przykładu, węzeł rodzic może być generalizacją dla dzieci oznaczonych, jako „samochody”, „autobusy”, „ciężarówki” i „motory”. Dalej, węzeł „samochody” może być generalizacją samochodów typu „Toyota”, „Ford”, „Mercedes”

Ważne jest, aby wyizolować tylko te najbardziej wartościowe, opisowe i dyskryminujące cechy z zestawu wejściowego, a następnie indywidualnie poddać kolejnym przetwarzaniom analizy. Techniki wyboru cech wyznaczają podzbiory istniejących cech. Natomiast techniki ekstrakcji obliczają nowe cechy z już istniejących. W obu przypadkach celem jest polepszenie jakości klasyfikacji i/lub efektywności obliczeniowej. Dobór cech to bardzo obszerny temat, jednakże w klasteryzacji często zmuszeni jesteśmy na dokonywanie wyborów „na wyczucie” a nie rzadko jest to proces prób i błędów, gdzie wybierane są różne zestawy cech obiektów i wybór jest oceniany po zakończeniu procesu klasteryzacji.

2.1.2 Miara bliskości

Jako, że podobieństwo jest podstawą definicji klastra, pomiar podobieństwa dwóch rekordów z tej samej przestrzeni cech, jest bardzo ważne dla większości procedur klasteryzacji. Z powodu różnorodności typów wartości cech i ich skalowania, miara (lub miary) podobieństwa, muszą być dobierane bardzo ostrożnie. Najbardziej powszechną praktyką jest obliczanie *miary niepodobieństwa* pomiędzy dwoma rekordami używając miary odległości zdefiniowanej na przestrzeni cech. Najczęściej w przypadku cech o wartościach ciągłych stosowana jest odległość Euklidesowa.

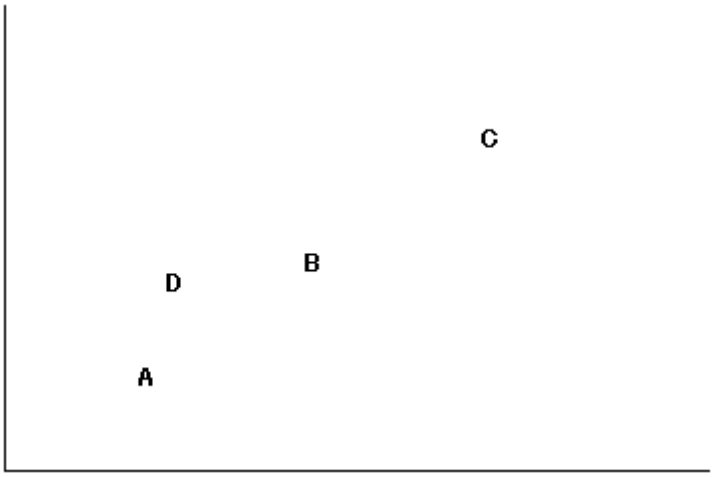
$$d(\mathbf{x}_i, \mathbf{x}_j) = \left(\sum_{k=1}^{d} (\mathbf{x}_{i,k} - \mathbf{x}_{j,k})^p \right)^{1/p} = \|\mathbf{x}_i - \mathbf{x}_j\|_p$$

Odległość Euklidesowa jest intuicyjnie używane do obliczeń odległości obiektów w dwu i trzy wymiarowych przestrzeniach. Sprawuje się doskonale tak z wyizolowanymi jak i złożonymi klastrami. Wadą używania tej miary jest fakt, że szybko cechy o mniejszej skali

stracą na wartości zdominowane przez cechy o większej skali. Rozwiązaniem tego problemu jest normalizacja używanych wartości lub stosowanie wag.

Obliczanie odległości pomiędzy rekordami, których część lub wszystkie cechy nie są wartościami ciągłymi, staje się o wiele trudniejsze. W ekstremalnych przypadkach wartość odległości jest określana binarnie. I tutaj jednak istnieje wiele prac, które starają się rozwiązać ten problem.

Rekordy mogą być także reprezentowane używając stringów czy struktur drzewiastych. Wymaga to wtedy syntaktycznego lub statystycznego podejścia do miary odległości. Ale są one zdecydowanie mniej przydatne.



rys 2 - D i B są bardziej podobne niż B i C

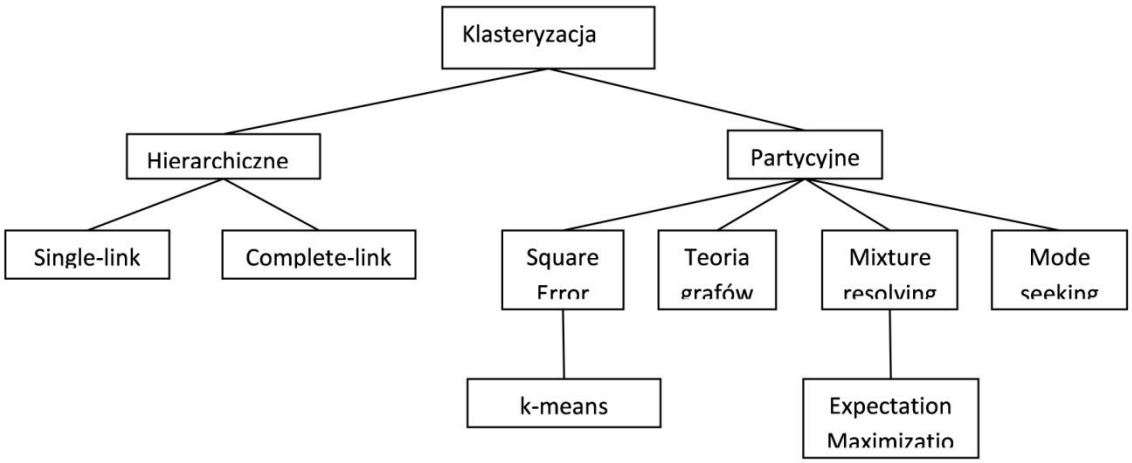
Są także miary odległości, który pod uwagę biorą otoczenie lub sąsiednie punkty, zwanych kontekstem. [2; 3; 4]. Najważniejszym przykładem tych metod jest mutual neighbour distance (MND) zaproponowane przez [2].

$$MND(\mathbf{x}_i, \mathbf{x}_j) = NN(\mathbf{x}_i, \mathbf{x}_j) + NN(\mathbf{x}_j, \mathbf{x}_i)$$

Gdzie NN($\mathbf{x}_i$, $\mathbf{x}_j$) jest numerem sąsiada $\mathbf{x}_j$ w odniesieniu do punktu $\mathbf{x}_i$. Tzn. funkcja ta określa, którym z kolei najbliższym sąsiadem $\mathbf{x}_i$ jest $\mathbf{x}_j$. Na **Błąd! Nie można odnaleźć źródła odwołania.Błąd! Nie można odnaleźć źródła odwołania.** A i D są ja bliższymi sąsiadami. NN(A,D) = NN(D,A) = 1. Zatem MND(A,D) = 2. Jednakże MND(C,B) = 4. Miara ta nie jest metryczna, może ulec zmianie, gdy tylko pojawią się nowe obiekty w sąsiedztwie i nadal potrzebuje standardowej funkcji odległości. Ale z powodzeniem została zaimplementowana w aplikacjach takich jak [5].

2.1.3 Techniki klasteryzacji

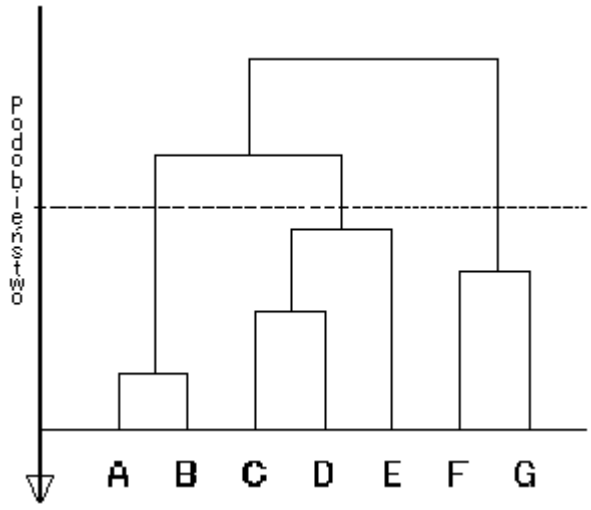
Na podstawie dyskusji w [6] można wprowadzić następujący podział technik klasteryzacji:



rys 3 - Techniki klasteryzacji

2.1.3.1 Hierarchiczne algorytmy klasteryzacji

Zadaniem algorytmów hierarchicznych jest wygenerowanie drzewa reprezentującego zagnieżdżone grupowanie rekordów. Tak jak to przedstawiono na rys 2-4. Następnie drzewo to może zostać przecięte na dowolnym poziomie dając nam różne rozwiązania klasteryzacji tego samego zbioru danych.



rys 2-4 Drzewo uzyskane za pomocą algorytmu single-link

Większość algorytmów hierarchicznych to warianty single-link [7; 6], complete-link [8] oraz minimum-variance [9; 10]. Z tych najpopularniejszymi algorytmami są single-link oraz complete-link, a różnią się jedynie charakterystyką „podobieństwa" pomiędzy klastrami. W single-link jest to minimum z odległości wszystkich par rekordów z tych klastrów (jeden rekord z jednego klastra, drugi z drugiego). Przypadku complete-link jest to maksimum. W obu przypadkach klastry są scalane używając kryterium minimalnej odległości.

2.1.3.2 Algorytmy partycyjne

Zamiast całej struktury klastrów, algorytmy partycyjne uzyskują pojedyncze partycje danych. Zyskują one przewagę przy dużych zbiorach, gdzie tworzenie drzew obliczeniowo jest bardzo wymagające. Jednakże największym problemem algorytmów partycyjnych jest dobór odpowiedniej liczby klastrów. Ponieważ ich liczna na ogół jest wymagana jeszcze przed rozpoczęciem algorytmu. Istnieje wiele propozycji aby znaleźć najlepsze rozwiązanie na ten problem [11]. Ale w praktyce najlepszy wynik uzyskuje się poprzez wykonanie algorytmu kilka razy i wybranie najlepsze rezultatu.

Squared error

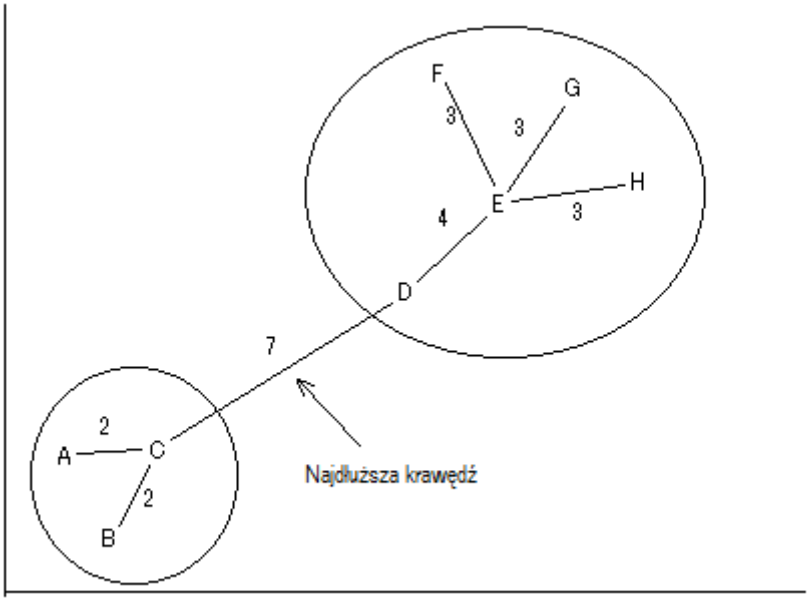
Najbardziej intuicyjna i najczęściej stosowana partycyjna technika klasteryzacji, która dobrze radzi sobie z klastrami wyizolowanymi i zwartymi. Jest to nic innego jak łączna suma odległości rekordów do centroidów klastrów, do których przynależą. Squared error samo w sobie nie jest technika ale kryterium oceniającym jakość rozwiązania. Dla klasteryzacji L zbioru rekordów H (zawierającego K klastrów) obliczamy za pomocą wzoru [12]:

$$e^2(L,H) = \sum_{j=1}^{K} \sum_{i=1}^{n_j} \left\| x_i^{(j)} - c_j \right\|^2 \tag{2-1}$$

Gdzie $x_i^{(j)}$ jest i-tym rekordem należącym do j-tego klastra, a c_j jest centroidem tego klastra. Najprostszym i bardzo szeroko stosowanym algorytmem, korzystającym z tego kryterium, jest k-mens [13]. Rozpoczyna się on z losowym, startowym zestawem partycji, a następnie realokuje rekordy przypisując je do innych klastrów, bazując na podobieństwie rekordu do centroida, dopóki nie zostanie spełnione kryterium (np. nie było kolejnych przesunięć rekordów pomiędzy partycjami lub kryterium Squared-error przestało się znacząco zmniejszać). K-means zyskał swoją popularność dzięki łatwości implementacji oraz złożoności czasowej (O(n), gdzie n to liczba rekordów). Ma on jednak swoje słabe strony. Jest on wrażliwy na dobór startowych partycji i może ostatecznie wyniknąć lokalnym minimum. Powstało wiele wariantów [14] tego algorytmu. Niektóre wstępnie próbują dokonać wyboru partycji startowych, aby zwiększyć swoją szansę w odnalezieniu globalnego minimum. Innym wariantem jest łączenie lub dzielenie wynikowych partycji. Jeśli różnorodność wewnątrz klastra jest zbyt duża (przekracza odp. wskaźnik), jest on dzielony na dwie osobne partycje. Jeśli natomiast centroidy dwóch klastrów są odpowiednio bliskie ich partycje są łączone. Technika ta pozwala na uzyskanie optymalnego zestawu klastrów zaczynając od całkiem innego zbioru startowego. Technika ta jest stosowana w ISODATA [15]. Inne warianty k-means tworzone są poprzez zmianę funkcji kryterium. Np. algorytmy dynamiczne klasteryzacji (które dopuszczają reprezentacje klastra inną niż centroid) zaproponowane w [16; 17; 18].

Teorie grafów

Najlepiej znanym algorytmem używającym tego podejścia jest algorytm bazujący na skonstruowaniu z danych Minimalnego Drzewa Rozpiętego (eng. Minimal Spanning Tree - MST) [19], a następnie na usuwaniu krawędzi o największej długości w celu sformułowania klastrów. Przykład na **Błąd! Nie można odnaleźć źródła odwołania.** ukazuje MST uzyskane z 8 dwu-wymiarowych punktów. Poprzez zerwania połączenia CD (krawędź z największą długością Euklidesową), uzyskujemy dwa klastry ({A, B, C} oraz {D, E, F, G, H}). Kolejny podział może zostać dokonany na krawędzi DE itd.



rys 5 - Przykład użycia minimalnego drzewa rozpiętego o klasteryzacji.

Podejście hierarchiczne także odnosi się do grafów. Klastry Single-link są podgrafami minimalnego drzewa rozpinającego [20; 21]. Klastry complete-link są maksymalnie kompletnymi subgrafami [22] i w pracach [23; 24] uznawane są za definicję klastra. Inne podejście używające grafów w nie-hierarchicznych strukturach nakładających się klastrów prezentuje [25].

D*BSC*AN

Oparty na gęstości algorytm klasteryzacji, który sam potrafi określić ilość klastrów. Punkty w przestrzeni o małej gęstości są traktowane jako zakłócenia i ignorowanie. Oznacza to, że DBSCAN produkuje Klasteryzację niekompletną.

2.1.3.3 Klasteryzacja jednoznaczna, Nakładająca się lub Rozmyta.

Opisane do tej pory metody klasteryzacji są jednoznaczne, ponieważ przypisują pojedynczy obiekt wyłącznie do jednego klastra. Istnieje jednak wiele sytuacji, gdzie bardzo rozsądne staje się przypisanie pojedynczego punktu danych do więcej niż jednego klastra. W tym celu używa się klasteryzacji nakładającej się, lub niejednoznacznej. Także stosowane w przypadkach obiektów, których podobieństwo do dwóch klastrów jest równe. Zamiast

decydować, które podobieństwo jest większe, można przydzielić ten obiekt równolegle do obu klastrów [1].

W klasteryzacji rozmytej, przynależność obiektu do klastra jest opisana za pomocą wagi w przedziale od 0 (kompletnie nie należy) do 1 (całkowicie należy). Innymi słowy klastry traktowane są jako zbiory rozmyte. Podobnie, probabilistyczne techniki klasteryzacji, obliczają prawdopodobieństwo z jakim każdy punkt należy do każdego klastra. Tak aby suma tych prawdopodobieństw wynosiła 1. Jako, że wagi członkowstwa i prawdopodobieństwa dla każdego obiektu sumują się do 1, klasteryzacja rozmyta czy probabilistyczna, nie adresują prawdziwych sytuacji multiklasowości. Techniki te jedynie unikają arbitralności przypisywania obiektu do jednej klasy, gdy może on być blisko do kilku z nich. W praktyce, klasteryzacja rozmyta czy probabilistyczna jest przekształcana na Klasteryzację jednoznaczną, poprzez przypisywanie obiektów do klastrów, dla których waga lub prawdopodobieństwo przynależności są największe [1].

2.1.3.4 Klasteryzacja pełna i częściowa

Klasteryzacja pełna, przypisuje każdy obiekt do pewnego klastra. Gdy natomiast klasteryzacja częściowa pozwala, aby niektóre obiekty pozostały nie przypisane. Może to mieć zastosowanie w przypadku, gdy dane nie posiadają dobrze zdefiniowanych grup. Bardzo często obiekty w zbiorze danych mogą reprezentować zakłócenia, przypadki krawędziowe, czy nie interesujące nas „tło". Są jednak przypadki, gdy każdy obiekt musi zostać poddany klasteryzacji bez wyjątku [1].

2.1.4 Różne typy klastrów

Klasteryzacja ma na celu wynalezienia użytecznych grup obiektów (klastrów), przy czym użyteczność jest zdefiniowane przez cele analizy danych. Według [1] można wyróżnić kilka typów klastrów:

2.1.4.1 Dobrze rozdzielone

Klaster jest zbiorem obiektów, w którym każdy obiekt jest bliższy (lub bardziej podobny) do innych obiektów z tego klastra niż do jakiegokolwiek obiektu z innych klastrów. Czasami używa się wartości progowej, aby obiekty były do siebie odpowiednio bliskie (lub podobne). Ta idealistyczna definicja klastra jest spełniana jedynie gdy dane zawierają naturalne klastry odpowiednio oddalone od siebie.

2.1.4.2 Klastry prototypów

Klaster jest zbiorem obiektów, z których każdy jest bliżej (bardziej podobny) do prototypu definiującego dany klaster niż do prototypu jakiegokolwiek innego klastra. Dla danych z ciągłymi atrybutami, prototypem zwykle jest centroid (np. średnia wszystkich punktów w klastrze). Gdy dane zawierają atrybuty kategoryczne., prototypem może być medoid (np. punkt danych, który najlepiej reprezentuje klaster).

2.1.4.3 Klastry Grafów

Gdy dane reprezentowane są za pomocą grafu. Gdzie węzły są obiektami a krawędzie są relacjami pomiędzy tymi obiektami. Wtedy klaster definiujemy jako grupę obiektów posiadających p[omiędzy sobą połączenie.

2.1.4.4 Klastry gęstościowe

Klastrem jest obszar gęsto umieszczonych obiektów otoczony obszarem o małej gęstości. Taka definicja klastra najczęściej stosowana jest, gdy w rolę wchodzi duża nieregularność klastrów, oraz zakłócenia.

2.1.4.5 Współdzielenie Własności

W klastrze tym znajdują się obiekty, które współdzielą jakąś właściwość. Choćby fakt, że obiekty te są najbliżej położone wspólnego centroida czy medoida.

2.2 Algorytmy Ewolucyjne

2.3 ?Popularne algorytmy klasteryzacji?

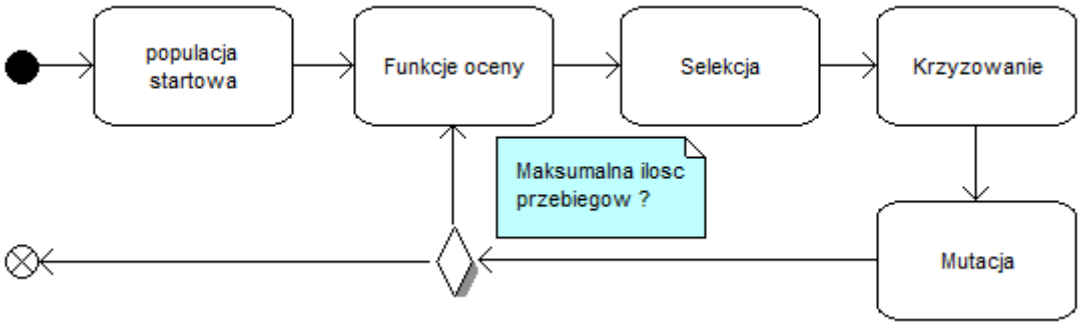
2.4 ?CUDA?

3 Opis rozwiązania

W tym przedstawiono szczegółowo propozycję rozwiązania jak i opis aplikacji za pomocą której doświadczalnie sprawdzono wyniki tegoż rozwiązania.

3.1 Ogólny schemat algorytmu

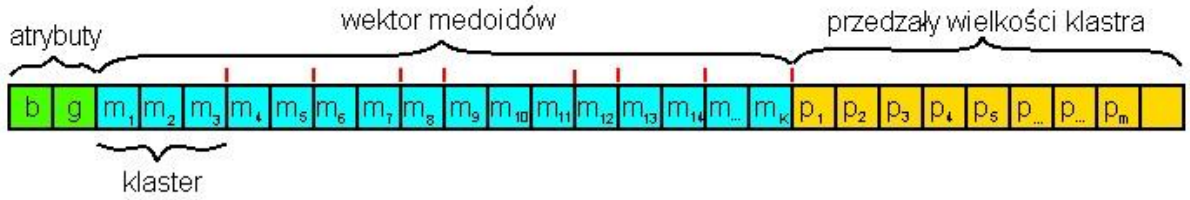
Na rysunku rys 6 - Schemat algorytmu.rys 6 przedstawiono ogólny algorytm ewolucyjny zastosowany w rozwiązaniu. W dalszej części pracy znajduje się dokładny opis przedstawionych tu etapów.



rys 6 - Schemat algorytmu.

3.2 Kodowanie osobników

Kodowanie osobników populacji jest bardzo ważne tak dla wyników algorytmu jak i wydajności samej aplikacji, która go implementuje. Wybrano w miarę proste rozwiązanie, które zaspokoi oba te wymagania.



rys 7 - Logiczna budowa osobnika

Najważniejszym elementem osobnika jest wektor medoidów. Jest on stałej wielkości, określonej przed rozpoczęciem algorytmu. Każdy medoid służy do opisu klastra. Przy czym pojedynczy klaster może zostać opisany za pomocą jednego lub więcej medoidów.

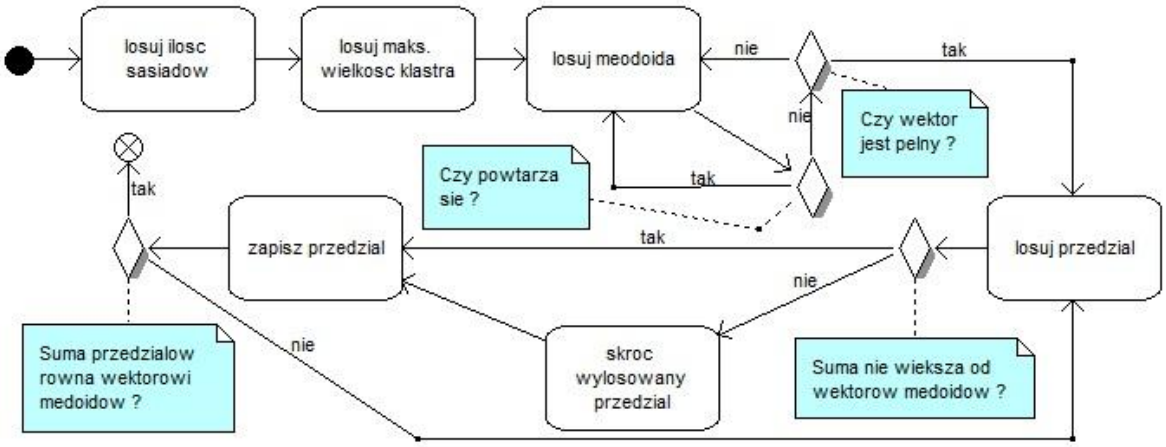
Drugim ważnym elementem jest opis kolejnych klastrów. Opisuje on ilość medoidów przypisanych do danego klastra. Przydział medoidów następuje w tej samej kolejności jak opisy klastrów. Sumaryczna wielkość wszystkich klastrów nie może przekraczać wielkości wektora medoidów.

Pozostałym elementem są atrybuty opisujące zachowanie danego osobnika. Są to:

- Ilość sąsiadów brana pod uwagę w kryterium łączności

- Maksymalna wielkość pojedynczego klastra.

Przy czym drugi atrybut ma bezpośredni wpływ na maksymalną ilość medoidów opisujących pojedynczy klaster, oraz pośredni wpływ na ilość klastrów w danym rozwiązaniu.



rys 8 - Generowanie populacji startowej.

Taka budowa nosi ze sobą ryzyko generowania błędnych rozwiązań w postaci powtarzających się medoidów tak w różnych klastrach jak i w jednym klastrze. Środki przeciwdziałania są częścią algorytmu i zostały opisane w dalszej części pracy.

3.3 Ocena osobników

W rozwiązaniu zastosowano wielokryterialny algorytm ewolucyjny (dokładniej opisany we wprowadzeniu do tematyki pracy). Osobniki populacji są oceniane w czterech niezależnych kategoriach nazwanych odpowiednio: Gęstość, Łączność, Rozłączność i Błędność. Ten rozdział kolejno je opisuje.

3.3.1 Gęstość

Jest to średnia odległość wszystkich obiektów klastra do ich najbliższego medoida, należącego do tego klastra. Zastosowano tutaj standardowo euklidesową miarę odległości punktów danych. Miara ta promuje duże zagęszczenie wokół medoidów. Im mniejsza wartość tym lepszy wynik.

3.3.2 Łączność

Każdy obiekt zbioru danych ma określonych „sąsiadów”. Jest to lista obiektów znajdujących się najbliżej danego obiektu. Dla każdego obiektu określa się ilu z sąsiadów przynależy do tego samego klastra. I Określa to wartością z przedziału 0 (żadne z sąsiadów nie przynależy do tego samego klastra) do 1 (wszyscy sąsiedzi przynależą do tego samego klastra). Następnie wartości te są sumowane. Miara ta promuje dobrą rozdzielność pomiędzy klastrami. Im większa wartość tym lepszy wynik.

3.3.3 Rozłączność

Miara ta określa jak daleko od siebie są umieszczone medoidy opisujące różne klastry. Z taką jedynie różnicą, że jako miarę bliskości zastosowano tutaj MND(Mutual Neighbour Distance

– opisane we wprowadzeniu). Miary te są sumowane dla wszystkich klastrów. Miara ta promuje rozdzielność medoidów opisujących różne klastry jak i większe skupisko medoidów opisujących ten sam klaster. Im większa wartość tym lepszy wynik.

3.3.4 Błędność

W rozwiązaniu zdecydowano, że błędne osobniki nie będą poprawiane, lecz wezmą udział selekcji. Jednakże są one określane miarą zawartych błędów, aby zmniejszyć prawdopodobieństwo ich użycia do krzyżowania. Oceniane są następujące błędy:

- Powtarzalność medoidów – im więcej powtórzeń, tym większy błąd
- Puste klastry – klastry które nie zawierają nic poza medoidami, które je opisują.

Miara ta promuje bezbłędne osobniki. Im mniejsza wartość tym lepszy wynik.

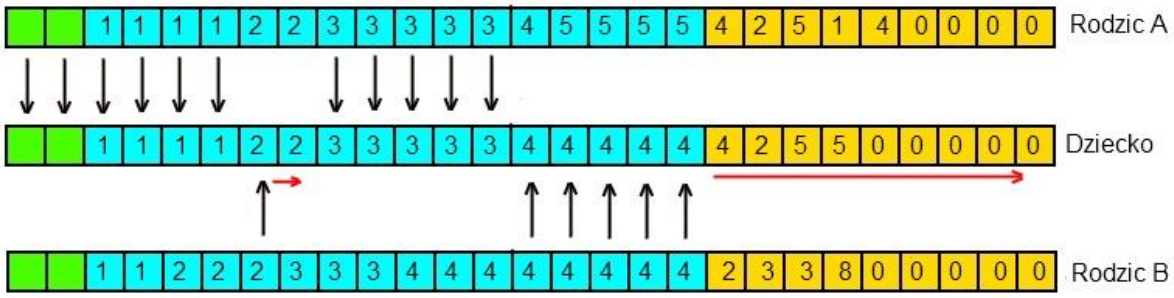
4 Operatory genetyczne

4.1.1 Selekcja

W rozwiązaniu selekcji dokonuje się za pomocą algorytmu NSGA-II opisanego wcześniej we wstępie do pracy. Wybierana jest połowa osobników z puli populacji. Kolejno dodawane do listy krzyżowania według tego jak zostały umieszczone we frontach.

4.1.2 Krzyżowanie

Krzyżowanie odbywa się poprzez wymianę klastrów i ich medoidów. Jeden z rodziców wybierany jest jako dominujący. Z niego kopiowane są atrybuty (sąsiedzi i wielkość klastra). Następnie do dziecka kopiowane są klastry nieparzyste z rodzica dominującego i klastry parzyste z drugiego rodzica. Przy czym jeśli się nakładają, pierwszeństwo mają klastry o mniejszym indeksie.



rys 9 - Operacja krzyżowania.

4.1.3 Mutacja

W trakcie generowania tablicy rozrodczej (zestawianie rodziców w pary) losowany jest współczynnik mutacji <0, 100>, zależnie od wartości.

Atrybuty:

- factor > 50
  - zmiana maksymalnego rozmiaru klastra
- factor > 70
  - zmień ilość sąsiadów
- factor > 90
  - zmień maksymalną klastra i ilość sąsiadów.

Medoidy:

- Factor > 20
  - zmień 1 medoid
- factor > 50
  - zmień 2 medoidy
- factor > 80
  - zmień 3 medoidy

- Factor > 95
  - zmień 4 medoidy

4.2 ?Optymalizacja dla CUDA?

5 Wyniki Badań

5.1 Iris

5.2 Wine

5.3 Cancer

5.4 Porównanie wydajnościowe wyników uzyskanych na CUDA

5.5 Wnioski

6 Podsumowanie

7 Literatura

1. **Tan, Pandg-Ning, Steinbach, Michael i Kumar, Vipin.** Cluster Analisys: Basic Concepts and Algorithms. *Introduction to data mining.* brak miejsca : Pearson Addison Wesley, 2006, 8.

2. *Agglomerative clustering using the concept of mutual nearest neighborhood.* **GOWDA, K. C. i G., KRISHNA.** 105–112, 1977, Pattern Recogn.

3. *Clustering Using a Similarity Measure Based on Shared Near Neighbors.* **JARVIS, R. A. i PATRICK, E. A.** 1025–1034, 1973, IEEE Trans. Comput.

4. *Automated construction of calsifications: conceptual clustering versus numerical taxonomy.* **MICHALSKI, R., STEPP, R. E. i DIDAY, E.** brak miejsca : IEEE, 5 Sept 1983, IEEE Trans. Pattern Anal. Mach. Intell., strony 396–409.

5. *Symbolic clustering using a new dissimilarity measure.* **GOWDA, K. C. AND DIDAY, E.** 368–378, 1992, IEEE Trans. Syst. Man Cybern, Tom 22.

6. *Algorithms for Clustering Data.* **JAIN, A. K. i DUBES, R. C.** Upper Saddle River, NJ : Prentice-Hall, 1988.

7. *Numerical Taxonomy.* **SNEATH, P. H. A. i SOKAL, R. R.** London : Freeman, 1973.

8. *Step-wise clustering procedures.* **KING, B.** brak miejsca : J. Am. Stat. Assoc., 1967.

9. *Hierarchical grouping to optimize an objective function.* **WARD, J. H. JR.** brak miejsca : J. Am. Stat. Assoc., 1963.

10. *A survey of recent advances in hierarchical clustering algorithms which use cluster centers.* **MURTAGH, F.** brak miejsca : Comput. J., 1984.

11. *How many clusters are best?—an experiment.* **DUBES, R. C.** 1987, Pattern Recogn.

12. *Data clustering: a review.* **Jain, A. K., Murty, M. N. i Flynn, P. J.** 3, brak miejsca : ACM, 1999, Comput. Surveys, Tom 31, strony 264-323.

13. *Some methods for classification and analysis of multivariate observations.* **MCQUEEN, J.** 1967. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability.

14. *Cluster Analysis for Applications.* **ANDERBERG, M. R.** New York, NY : Academic Press, Inc., 1973.

15. **BALL, G. H. i HALL, D. J.** *ISODATA, a novel method of data analysis and classification.* Tech. Rep.. Stanford University,. Stanford, CA. : brak nazwiska, 1956.

16. *The dynamic cluster method in non-hierarchical clustering.* **DIDAY, E.** 1973, J. Comput. Inf. Sci., Tom 2, strony 61–88.

17. *Clustering criterion and multi-variate normal mixture.* **SYMON, M. J.** Biometrics, Tom 77, strony 35–43.

18. *A self-organizing network for hyperellipsoidal clustering (HEC).* **MAO, J. AND JAIN, A. K.** 1996, IEEE Trans. Neural Netw., Tom 7, strony 16–29.

19. *Graph-theoretical methods for detecting and describing gestalt clusters.* **ZAHN, C. T.** 1971, IEEE Trans. Comput., Tomy C-20 (Apr.),, strony 68–86.

20. *Minimum spanning trees and single-linkage cluster analysis.* **GOWER, J. C. i ROSS, G. J. S.** 1969, Appl. Stat., Tom 18, strony 54–64.

21. *Semantic clustering of index terms.* **GOTLIEB, G. C. i KUMAR, S.** 1968, J. ACM, Tom 15, strony 493–513.

22. *A graphtheoretic approach to goodness-of-fit in complete-link hierarchical clustering.* **BACKER, F. B. i HUBERT, L. J.** 1976, J. Am. Stat. Assoc., Tom 71, strony 870–878.

23. *An analysis of some graph theoretical clustering techniques.* **AUGUSTSON, J. G. i MINKER, J.** 4 Oct 1970, J. ACM, Tom 17, strony 571–588.

24. *A comparison of the stability characteristics of some graph theoretic clustering methods.* **RAGHAVAN, V. V. i YU, C. T.** 1981, IEEE Trans. Pattern Anal. Mach. Intell., Tom 3, strony 393–402.

25. *A stratificational overlapping cluster scheme.* **OZAWA, K.** 1985, Pattern Recogn., Tom 18, strony 279–286.

26. *Comparisions Between Data Clustering Algorithms.* **Abbas, Osama Abu.** 2008, The Informational Arab Journal of Information Technology.

27. *A Survey of Evolutionary Algorithms for Clustering.* **Hruschka, Eduardo Raul, i inni.** 2009, IEEE transactions on Systems.

28. *Data clustering: 50 years beyond K-means.* **Jain, Anil K.** brak miejsca : Elsevier, 2009, Pattern Recognition Letters.

29. **www.merriam-webster.com.**

30. *NOCEA: A rule-based evolutionary algorithm for efficient and effective.* **Sarafis, Ioannis A., Trinder, Phil W. i Zalzala, Ali M.S.** brak miejsca : Elsevier, 2006, Science Direct.

31. *A Survey of Evolutionary Algorithms for Clustering.* **Hruschka, Eduardo Raul, i inni.**

32. *A genetic algorithm with gene rearrangement for K-means clustering.* **Changa, Dong-Xia, Zhang, Xian-Da i Zheng, Chang-Wen.** brak miejsca : Elsevier, 2008, Pattern Recognition.

33. *An Evolutionary Approach to Multiobjective Clustering.* **Handl, Julia i Knowles, Joshua.** brak miejsca : IEEE, 2007, IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION.

34. *High-dimensional data clustering.* **Bouveyron, C., Girard, S. i Schmid, C.** brak miejsca : Elsevier, 2007, Science Direct.

35. *Asurvey of kernel and spectral methods for clustering.* **Filippone, Maurizio, i inni.** brak miejsca : Elsevier, 2008, Science Direct.

36. *Apartitional clustering algorithm validated by a clustering tendency.* brak miejsca : Elsevier, 2006, Pattern recognition.

37. *Clustering data with measurement errors.* **Kumar, Mahesh i Patel, Nitin R.** brak miejsca : ELSEVIER, 2007, Computational Statistics & Data Analysis.

38. *Dynamic Clustering Using Multi-objective Evolutionary Algorithm.* **Chen, Enhong i Wang, Feng.** brak miejsca : Springer, 2005, Computational Intelligence and Security.

39. *A Scatter Search Algorithm for the Automatic Clustering Problem.* **Abdule-Wahab, Rasha S., i inni.** brak miejsca : Springer, 2006, Advances in Data Mining.

40. *An artificial bee colony approach for clustering.* **Zhang, Changsheng, Ouyang, Dantong i Ning, Jiaxu.** brak miejsca : Elsevier, 2010, Expert Systems with Applications.

41. *A method of relational fuzzy clustering based on producing feature vectors using FastMap.* **Brouwer, Roelof Kars.** brak miejsca : Elsevier, 2009, Information Sciences.

42. *Parallelizing Evolutionary Algorithms for Clustering Data.* **Kwedlo, Wojciech.** brak miejsca : Springer, 2006, Parallel Processing and Applied Mathematics.

43. *Performance evaluation of density-based clustering methods.* **Aliguliyev, Ramiz M.** brak miejsca : Elesevier, 2009, Information Sciences.

44. *Parallel Evolutionary Algorithms.* **Osmera, Pavel, Lacko, Bronislav i Petr, Masopust.** brak miejsca : IEEE, 2003, Computational Intelligent in Robtics and Automation.

45. *Multi-objective Optimization Technique Based on Co-evolutionary Interactions in Multi-agent System.* **Dreżewski, Rafał i Siwik, Leszek.** brak miejsca : Springer, 2007, Applications of Evolutionary Computing.

46. *An overview of evolutionary algorithms: practical issues and common pitfalls.* **Whitley, Darrell.** brak miejsca : Elsevier, 2001, Information and Software Technology.

47. *Data Mining: Introductory and Advanced Topics.* **Dunham, M. H.** brak miejsca : Prentice-Hall, 2003.

48. *Advances in Knowledge Discovery and Data Mining.* **Fayyad, U., i inni.** 1996, AAAI Press/The MIT Press.

49. *Data Mining: Concepts and Techniques.* **Han, J. i Kamber, M.** brak miejsca : Kaufmann Publishers, 2000.

50. *Survey of clustering algorithms.* **Xu, R. i Wunsch, D. I.I.** 16, brak miejsca : IEEE Trans, 3 2005, Neural Networks.

51. *Clustering Analysis.* **DIDAY, E. i SIMON, J. C.** Secaucus, NJ : Springer-Verlag, 1976, Digital Pattern Recognition, strony 47–94.

52. *Cluster analysis and related issues.* **DUBES, R. C.** River Edge, NJ : World Scientific Publishing Co, 1993, Handbook of Pattern Recognition & Computer Vision, strony 3–32.

53. *Fuzzy sets.* **ZADEH, L. A.** 1965, Inf. Control, Tom 8.

54. *A new approach to clustering.* **RUSPINI, E. H.** 1969, Inf. Control, strony 22–32.

55. **BEZDEK, J. C.** *Pattern Recognition With Fuzzy Objective Function Algorithms.* New York, NY. : Plenum Press, 1981.

56. *Generalized fuzzy C-shells clustering and detection of circular and elliptic boundaries.* **DAVE, R. N.** 1992, Pattern Recogn, Tom 25, strony 713–722.

57. *Cluster Analysis: A Survey.* **DURAN, B. S. i ODELL, P. L.** New York, NY : brak nazwiska, 1974, Springer-Verlag.

58. *A recent advance in data analysis: Clustering objects into classes characterized by conjunctive concepts.* **MICHALSKI, R., STEPP, R. E. i DIDAY, E.** [red.] Eds. L. Kanal and A. Rosenfeld. Amsterdam, The Netherlands : North-Holland Publishing Co., 1981, In Progress in Pattern Recognition.

59. *Introduction to the Theory of Neural Computation.* **HERTZ, J., KROGH, A. i PALMER, R. G.** brak miejsca : Addison-Wesley Longman Publ. Co., Inc.,, 1991, Santa Fe Institute Studies in the Sciences of Complexity lecture notes.

60. *Artificial Neural Networks and Pattern Recognition: Old and New Connections.* **SETHI, I. i JAIN, A. K., Eds.** New York, NY. : Elsevier Science Inc., 1991.

61. *Neural networks and pattern recognition.* **JAIN, A. K. i MAO, J.** [red.] J. M. Zurada, R. J. Marks i C. J. Robinson. 1994, In Computational Intelligence: Imitating Life, strony 194-212.

A. Wartości parametrów algorytmów

B. Opis aplikacji

C. Formaty plików

7.1 Dane wejściowe

Pliki wejściowe nie mają określonego formatu. Rolą loadera przygotowanego dla każdej bazy jest wczytanie zawartości pliku i załadowanie do dalszej części programu

7.2 Dane pośrednie

„$źródło_distances.data”

Pliki pośrednie używane są do zapisu wyników z obliczeń dystansów pomiędzy obiektami bazy oraz najbliższych sąsiadów dla każdego obiektu. Tworzone są osobne pliki dla każdego źródła, właściwy format (wielkości zapisanych zbiorów danych) określane są na podstawie informacji o źródle.

Jeśli dla danego źródła istnieje już plik pośredni, faza obliczania odległości i wynajdywania sąsiadów jest pomijana.

7.3 Dane wyjściowe

Dla każdego źródła generowane są dwa pliki wyjściowe, odpowiednio:

„$źródło_reportsFile.txt” oraz „$źródło_reportsFileXls.txt”

W pierwszym zapisywane są wyniki każdego testu w postaci (przykład z bazy cancer):

```
---------------------------------
 medoids: 33 clusters: 9 neighbours: 1
 popSize: 256 steps: 1002
 Results:
 BDI:  0.214720 / 3.500959 / 69.163811
 DI:  0.003059 / 0.089595 / 0.549440
 Rand:  0.377717 / 0.555570 / 0.659984
 Time:  96.000000 / 102.600000 / 110.000000
```

Przy czym trzy ostatnie linijki zostały zapisane w formacie wartość minimalna/ średnia/ maksymalna.

W drugim pliku zapisywane są te same dane tyle że w postaci bardziej zrozumiałej dla programu excel (inny przykład z bazy cancer):

```
33, 1, 29, 265, 502, 8.210702, 11.473364, 41.749393, 0.002508,
0.017673, 0.048257, 0.373941, 0.378550, 0.386287, 125.000000,
131.800000, 155.000000
```

Czyli odpowiednio:

- Medoidy
- Klater (maksymalna)
- Sąsiedzi (maksymalna)
- Populacja

- Kroki
- BDI – min
- BDI – mean
- BDI – max
- DI – min
- DI – mean
- DI – max
- RAND – min
- RAND – mean
- RAND - max

D. Parametry maszyny testowej