STUDIA INFORMATICA 2012
Volume 33 Number 2A (105)

Agnieszka NOWAK-BRZEZIŃSKA, Tomasz XIĘSKI
Uniwersytet Śląski, Instytut Informatyki

GĘSTOŚCIOWA METODA GRUPOWANIA I WIZUALIZACJI DANYCH ZŁOŻONYCH

Streszczenie. Artykuł dokonuje przeglądu dotychczas stosowanych rozwiązań implementacyjnych w zakresie grupowania dużych wolumenów danych oraz opisuje problematykę doboru parametrów startowych dla algorytmu gęstościowego DBSCAN. Ponadto stanowi on wprowadzenie w tematykę wizualizacji struktury złożonych skupień, wykorzystując w tym celu algorytm oparty na idei gęstości – OPTICS.

Słowa kluczowe: wizualizacja skupień, grupowanie, OPTICS, DBSCAN

DENSITY-BASED METHOD FOR CLUSTERING AND VISUALIZATION OF COMPLEX DATA

Summary. This work reviews currently used implementation solutions for clustering large volumes of data, and describes the problem of choosing proper initial values for the density-based DBSCAN algorithm. Furthermore it should be also treated as an introduction to the topic of visualization of complex clusters using another density-based algorithm – OPTICS.

Keywords: cluster visualization, clustering, OPTICS, DBSCAN

1. Wprowadzenie

Wybór algorytmu analizy skupień do zadania grupowania dużych wolumenów danych złożonych nie jest zadaniem łatwym. Algorytm taki powinien charakteryzować się możliwie najmniejszą złożonością obliczeniową, wysoką odpornością na występowanie w danych szumu informacyjnego czy pozwalać na odkrywanie skupień o dowolnych kształtach. Jak wykazano w pracach [7, 9], większość z tych wymogów spełnia algorytm gęstościowy DBSCAN (ang. *Density-Based Spatial Clustering of Applications with Noise*). Jego siła tkwi w natural-

nie pojmowanej definicji skupienia jako gęsto upakowanego obszaru, zawierającego podobne obiekty. Niestety, choć intuicyjnie rzecz biorąc założenie to wydaje się logiczne i pozwalające na jasne określenie granic grup, to jednak w praktyce może ono stanowić jedną z dwóch głównych wad algorytmu – nie jest on w stanie wykryć hierarchii skupień, jeśli taka występuje w danych źródłowych. Ponadto rezultat działania tego algorytmu (jakość utworzonych skupień) zależy od właściwego doboru parametrów startowych (*Eps* – rozważanego promienia sąsiedztwa i *MinPts* – liczby obiektów w grupie). Sposób doboru tych parametrów (zaprezentowany m.in. w [8]), polegający na wykonywaniu co najmniej kilku zgrupowań i wyborze tych parametrów, dla których wyniki okazały się najlepsze, jest czasochłonny, przez co nie zawsze możliwy do zastosowania, szczególnie w kontekście analizy dużych, złożonych zbiorów. Dlatego też autorzy postanowili zbadać i zaimplementować inny algorytm, oparty na idei gęstości – OPTICS (ang. *Ordering Points To Identify the Clustering Structure*). W przeciwieństwie do swojego pierwowzoru nie generuje on zgrupowania obiektów, a jedynie ich określone uporządkowanie na podstawie tzw. odległości osiągalnej. Niemniej na podstawie wygenerowanego uporządkowania można w bardzo prosty i, co ważniejsze, dość szybki sposób dokonać podziału zbioru danych na grupy (dla dowolnie ustalonego promienia sąsiedztwa). Dodatkowo uzyskane uporządkowanie obiektów pozwala na określenie i wizualizację wewnętrznej struktury i powiązań danych.

2. Podejścia stosowane do grupowania dużych wolumenów danych

Ilość gromadzonych danych nieustannie rośnie, a co za tym idzie określenie „duże zbiory danych" również zmienia swoje znaczenie – jeszcze niedawno dotyczyło ono kilkunastu tysięcy próbek, a dziś przetwarzane są miliony czy nawet miliardy danych. W literaturze przedmiotu [3, 5] spotykanych jest sześć głównych podejść do problematyki grupowania dużych wolumenów danych złożonych.

Pierwsze z nich – próbkowanie danych – polega na wyborze losowej próbki obiektów ze zbioru danych i zastosowaniu danego algorytmu analizy skupień tylko na wybranej próbce. Następnie możliwe jest przyporządkowanie pozostałych obiektów do już utworzonych reprezentatywnych skupień. Stanowi ono esencję działania popularnego, niehierarchicznego algorytmu grupowania danych CLARA (ang. *Clustering for Large Applications*). Niestety operowanie wyłącznie na wybranych próbkach zamiast na pełnym zbiorze danych najczęściej skutkuje podziałem dalekim od optymalnego. To właśnie dobór rozmiaru czy liczności próbek ma największe znaczenie w jakości uzyskanego rezultatu [6]. Należy również pamiętać, że metoda ta generuje różne podziały w zależności od wybranych próbek.

Kolejną metodyką wykorzystywaną w celu zmniejszenia zajętości pamięci jest dyskretyzacja danych (głównie odnosząca się do danych ilościowych, ale możliwe jest aplikowanie jej w stosunku do danych jakościowych). Generalnie wykorzystywane są dwa typy dyskretyzacji:

- statyczna – gdzie zestaw reguł i klas, na jakie należy podzielić zbiór wartości, jest ściśle określony,
- dynamiczna – gdzie algorytm grupowania jest aplikowany tylko w odniesieniu do jednego atrybutu, a przedziały jego wartości są wyznaczane na podstawie utworzonych grup.

Największym problemem przy dyskretyzacji danych jest wyznaczenie optymalnej liczby przedziałów. Nawet w przypadku zastosowania dyskretyzacji dynamicznej wygenerowany podział może być daleki od optymalnego.

Niektóre publikacje naukowe [3, 5] wykorzystują metodę „dziel i zwyciężaj". Zbiór danych przechowywany jest w większej pamięci pomocniczej (np. na dysku twardym) i jest on dzielony na wiele mniejszych porcji. Każda z tych porcji jest oddzielnie poddawana procesowi analizy skupień. Na końcu tej metody wyświetlane są zagregowane wyniki ze wszystkich zgrupowań. Algorytm CURE (ang. *Clustering Using REpresentatives*) jest popularnym przedstawicielem podejścia „dziel i zwyciężaj". Operuje on na losowo wybranej próbce danych, która jest dodatkowo dzielona na mniejsze porcje. Następnie każda porcja zostaje poddana działaniu algorytmu hierarchicznego celem wyznaczenia reprezentantów przyszłych skupień. Pozostałe obiekty z całego zbioru danych są przyporządkowywane do skupień na podstawie stopnia ich korelacji z reprezentantami. Niestety metoda „dziel i zwyciężaj" nie jest możliwa do zastosowania w każdym przypadku: niektóre algorytmy (np. *Agglomerative Hierarchical Clustering*) wymagają, by operować na pełnym zbiorze danych. Ponadto w zależności od tego na ile i na jakie porcje zostanie podzielony zbiór danych, wygenerowane zgrupowanie może być lepszej lub gorszej jakości. Nie bez znaczenia jest też homogeniczność danych – pożądana jest ich wysoka jednorodność.

Innym rozwiązaniem problemu grupowania dużych, złożonych wolumenów danych jest wykorzystanie algorytmu inkrementalnego (o ile taka implementacja jest dostępna). Podstawowym założeniem, wykorzystywanym w tym podejściu, jest możliwość analizy każdego obiektu ze zbioru niezależnie od pozostałych. W pamięci głównej są przechowywani zazwyczaj jedynie reprezentanci skupień, natomiast każdy obiekt jest korelowany z istniejącymi reprezentantami celem określenia jego przynależności do danej grupy. BIRCH (ang. *Balanced Iterative Reducing and Clustering Using Hierarchies*) jest znanym przyrostowo-hierarchicznym algorytmem, często wykorzystywanym przy zadaniu grupowania. Daje on dobre wyniki w przypadku odkrywania skupień sferycznych o podobnej wielkości, niemniej taka sytuacja (w kontekście analizy danych złożonych) rzadko ma miejsce. Jest to również przykład algorytmu, w którym kolejność analizy obiektów ze zbioru ma duży wpływ na koń-

cowy rezultat (ang. *order-dependent*). Niestety dość duża grupa algorytmów przyrostowych charakteryzuje się tą cechą [5].

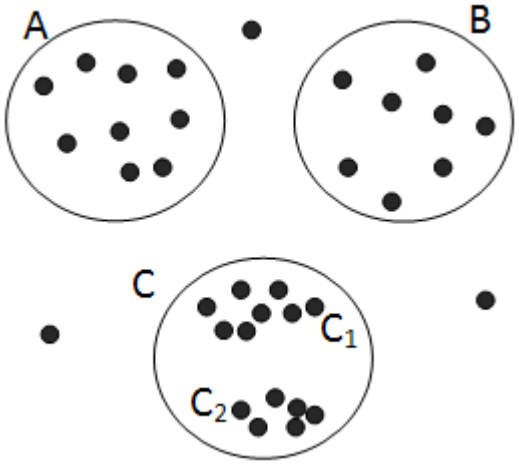
Jeżeli głównym problemem jest wysoka złożoność obliczeniowa algorytmu, to można dokonać jego zrównoleglenia i tutaj wyróżniane są dwa sposoby: przez wykorzystanie wielu procesorów CPU (najczęściej wielordzeniowych) lub też przez skorzystanie z mocy procesorów kart graficznych (GPU), które są wyspecjalizowane w wykonywaniu operacji zmiennoprzecinkowych. Można również dokonać dekompozycji obiektowej między wiele komputerów. Oczywiście implikuje to konieczność określenia i przydziału zadań poszczególnym jednostkom obliczeniowym. Może to uwzględniać podział zbioru danych na części (podobnie jak w metodzie „dziel i zwyciężaj"), tak by każda jednostka dokonywała grupowania swojej porcji, lub przydział zadań, wynikający z budowy samego algorytmu analizy skupień. Niestety zrównoleglenie algorytmu nie zawsze jest możliwe. Jeżeli w tym celu wykorzystywany jest główny procesor komputera, to oczywiście największy wpływ na przyspieszenie obliczeń ma liczba dostępnych procesorów (i rdzeni) oraz częstość i sposób komunikacji między wątkami. Wykorzystując jednostki obliczeniowe zawarte na karcie graficznej można uzyskać dużo większe przyspieszenie (m.in. w publikacji [4] opisano przyspieszenia rzędu 10 – 200-krotne), natomiast dużo bardziej skomplikowany jest proces optymalizacji kodu pod konkretną rodzinę kart graficznych niż pod dany procesor. Ponadto jeśli algorytm wymaga częstej wymiany danych z procesorem głównym, to jakiekolwiek korzyści z jego zrównoleglenia są najczęściej niwelowane bądź mocno ograniczane. Dodatkowo rzadko kiedy (w kontekście danych złożonych) da się wprowadzić cały zbiór danych do pamięci karty graficznej. Warto również wspomnieć, że różne karty graficzne (różnych firm) mają swoje unikalne biblioteki programistyczne, przez co istnieje tu dość duże przywiązanie do konkretnej platformy sprzętowej [4]. Jednak sytuację tę stara się poprawić standard OpenCL (ang. *Open Computing Language*).

3. Motywacja i zasada działania algorytmu OPTICS

Podczas procesu grupowania danych złożonych, oprócz problemów związanych z przetwarzaniem dużej ilości danych, należy wziąć pod uwagę również ich wewnętrzną strukturę. Powiązania występujące w wielowymiarowych danych rzeczywistych mogą tworzyć hierarchie i mieć niejednorodną naturę. Zastosowane przez autorów podejście do grupowania danych odnośnie do telefonii komórkowej, z wykorzystaniem gęstościowego algorytmu DBSCAN (opisane w [8, 9]), uwypukliło dwa znaczące problemy tego algorytmu.

Pierwszym problemem jest dobór właściwych parametrów startowych (Eps, MinPts). Zbyt duże wartości promienia sąsiedztwa (Eps) mogą skutkować tym, że dwa naturalnie wy-

stępujące, mniejsze skupienia zostaną niepoprawnie zinterpretowane przez algorytm jako jedno duże. Małe wartości tego parametru nie sprawdzą się w sytuacji, gdy zbiór danych będzie charakteryzował się niskim zagęszczeniem obiektów. Drugi parametr startowy (MinPts) ma największy wpływ na liczbę obiektów w grupach, a co za tym idzie na rozmiar grupy obiektów izolowanych – im jest on większy, tym potencjalnie więcej obiektów może zostać zaklasyfikowanych jako szum informacyjny (jeśli naturalne skupienia są mało liczne). Najprostszym podejściem do rozwiązania tego problemu jest wykonanie wielu zgrupowań parametrów startowych z różnymi wartościami, ocena ich jakości oraz wybór najlepszego z dostępnych. Niestety już samo wykonywanie zgrupowań jest zadaniem dość czasochłonnym.



Rys. 1. Występowanie hierarchii skupień
Fig. 1. The presence of a hierarchy of clusters

Kolejną wadą algorytmu DBSCAN (bezpośrednio związaną ze sposobem jego działania) jest niemożność wykrywania hierarchii skupień. Sytuacja taka została przedstawiona na rysunku 1. Dla zaprezentowanego zbioru danych nie jest możliwe wykrycie wszystkich czterech skupień (oznaczonych jako A, B, C_1, C_2) przy użyciu stałej wartości parametru sąsiedztwa (zagęszczenia obiektów). Algorytm DBSCAN wygenerowałby podział składający się z grup A, B, C lub grup C1, C2 w zależności od doboru wartości parametrów startowych. W drugim przypadku rzeczywiste skupienia A i B zostałyby potraktowane jako szum informacyjny. Wykorzystując ten fakt, można by było zmodyfikować algorytm DBSCAN tak, by tworzył w tym samym czasie skupienia o różnych zagęszczeniach (tj. wykorzystywałby różne wartości promienia sąsiedztwa). Jednakże by otrzymać spójny rezultat, należałoby zachować określony porządek, w jakim obiekty byłyby przetwarzane. Zawsze należałoby wybierać najpierw obiekt, który jest gęstościowo osiągalny przy najmniejszej wartości promienia sąsiedztwa Eps, by skupienia charakteryzujące się największym zagęszczeniem były wykrywane jako pierwsze. Algorytm OPTICS działa na podobnej zasadzie (biorąc pod uwagę wszystkie możliwe wartości promienia sąsiedztwa do pewnej ustalonej granicy), z tą różnicą, że do konkretnych grup nie są przypisywane identyfikatory przynależności obiektów. Zamiast tego zapa-

miętywany jest porządek, w jakim obiekty są przetwarzane, oraz informacja, która może posłużyć do określenia przynależności poszczególnych obiektów do grup. Informacja ta składa się z dwóch parametrów (wyliczanych dla każdego obiektu ze zbioru) – odległości wewnętrznej (ang. *core-distance*) oraz odległości osiągalnej (ang. *reachability-distance*).

Odległość wewnętrzna (dla obiektu *p* ze zbioru danych) jest najmniejszą odległością pomiędzy *p* i obiektem w jego *Eps-sąsiedztwie*, takim że *p* zostałby zaklasyfikowany jako obiekt wewnętrzny. W przeciwnym wypadku (tj. gdy *p* nie może zostać uznany za obiekt wewnętrzny) odległość wewnętrzna jest nieokreślona[1]. *Odległość osiągalna* dla obiektu *p* (względem obiektu *q*) jest to najmniejsza odległość, taka że *p* jest gęstościowo osiągalny z *q*, jeżeli *q* jest obiektem wewnętrznym. W takim przypadku odległość osiągalna *p* nie może być mniejsza niż *odległość wewnętrzna* dla *q*, ponieważ dla mniejszych wartości *p* nie jest gęstościowo osiągalny z *q*. Natomiast jeżeli *q* nie jest obiektem wewnętrznym, odległość osiągalna dla obiektu *p* (względem *q*) jest nieokreślona.

Biorąc pod uwagę przedstawione definicje, można określić uproszczony schemat działania algorytmu OPTICS. Jest on następujący:

1. Wybierz kolejny obiekt ze zbioru danych.
2. Wyznacz *Eps-sąsiedztwo* dla aktualnie analizowanego obiektu.
3. Wylicz odległość wewnętrzną dla analizowanego obiektu.
4. Jeżeli odległość wewnętrzna jest niezdefiniowana, przejdź do kroku pierwszego.
5. Wylicz odległości osiągalne dla obiektów znajdujących się w sąsiedztwie analizowanego (względem tego obiektu), a następnie posortuj te obiekty rosnąco według ich odległości osiągalnych.
6. Kontynuuj proces (od punktu pierwszego), dopóki nie zostaną przeanalizowane wszystkie obiekty ze zbioru danych.
7. Wypisz kolejność, w jakiej obiekty ze zbioru danych były przetwarzane, wraz z wartościami odległości wewnętrznej i osiągalnej.

Na podstawie przedstawionego schematu można zauważyć duży stopień podobieństwa do algorytmu DBSCAN. Dla obu metod najważniejszym elementem jest wyznaczenie *Eps-sąsiedztwa*. Na jego podstawie algorytm DBSCAN określa przynależność obiektu do danego skupienia, natomiast OPTICS generuje określone uporządkowanie obiektów. Jak zostanie wykazane w następnej sekcji, uporządkowanie to może być wykorzystane nie tylko do przydziału obiektów ze zbioru danych do grup, ale również do wizualizacji ich struktury. Biorąc pod uwagę przedstawione podobieństwo, można stwierdzić, że czas działania algorytmu OPTICS (a w konsekwencji również jego złożoność obliczeniowa) nie różni się drastycznie

[1] Szczegółowe definicje *Eps-sąsiedztwa*, gęstościowej osiągalności oraz związane z nimi terminy zostały opisane w pracach [1, 2, 10].

od czasu działania swojego pierwowzoru. Eksperymenty przeprowadzone w publikacji [1] wykazały, że czas działania algorytmu OPTICS jest równy 1,6-krotności czasu działania algorytmu DBSCAN. Nie jest to zaskakujący rezultat, ponieważ, jak zaznaczono wcześniej, złożoność obliczeniowa obu metod jest silnie uzależniona od procesu wyznaczania *Eps-sąsiedztwa*, który musi być przeprowadzony dla każdego obiektu ze zbioru danych wejściowych. Jeżeli nie jest wykorzystywane żadne indeksowanie danych, to – by uzyskać odpowiedź na zapytanie odnośnie do sąsiedztwa danego obiektu – należałoby dokonać przeglądu zupełnego całej bazy danych. W takim przypadku złożoność obliczeniowa dla algorytmu OPTICS wynosiłaby $O(n^2)$. Sytuacja ulega znaczącej poprawie, jeśli zostanie użyte indeksowanie oparte na strukturach drzewiastych (ang. *tree-based spatial index*), gdyż wówczas średnia złożoność obliczeniowa całego algorytmu maleje do $O(n \log n)$ [1].

4. Grupowanie wykorzystujące algorytm OPTICS

Jak już zostało wspomniane wcześniej, sam algorytm OPTICS bezpośrednio nie generuje podziału na grupy, jednakże wyniki jego działania mogą być w łatwy sposób użyte do stworzenia takiego podziału. Mając określone uporządkowanie zbioru danych, można wygenerować zgrupowanie (dla dowolnego promienia sąsiedztwa $Ei \leq Eps$) przez analizę tego uporządkowania i przypisanie identyfikatorów grup na podstawie wartości odległości wewnętrznych i osiągalnych. Poniższy kod przedstawia procedurę *WydobądźSkupienia*, realizującą opisane zadanie:

```
WydobądźSkupienia(UporządkowaneObiekty, Ei, MinPts):
    skupienieId = SZUM
    foreach obj in UporządkowaneObiekty:
       if obj.odległość_osiągalna > Ei then
          if obj.odległość_wewnętrzna <= Ei then
             skupienieId = skupienieId + 1
             obj.skupienieId = skupienieId
          else
             obj.skupienieId = SZUM
      else
         obj.skupienieId = skupienieId
```

Najpierw sprawdzany jest warunek, czy osiągalna odległość aktualnie analizowanego obiektu jest większa niż ustalona przez użytkownika wartość promienia sąsiedztwa (oznaczona jako *Ei*), dla której należy wygenerować zgrupowanie. W takim przypadku obiekt ten nie jest gęstościowo osiągalny z żadnego innego obiektu umieszczonego przed nim w stworzonym wcześniej uporządkowaniu. Intuicyjnie rzecz ujmując, jeśli analizowany obiekt byłby gęstościowo osiągalny z poprzedzającego go w uporządkowaniu obiektu, jego odległość osiągalna miałaby co najwyżej wartość *Ei*. Jeżeli zatem odległość osiągalna jest większa od *Ei*, sprawdzana jest następnie odległość wewnętrzna tego obiektu i tworzone jest nowe sku-

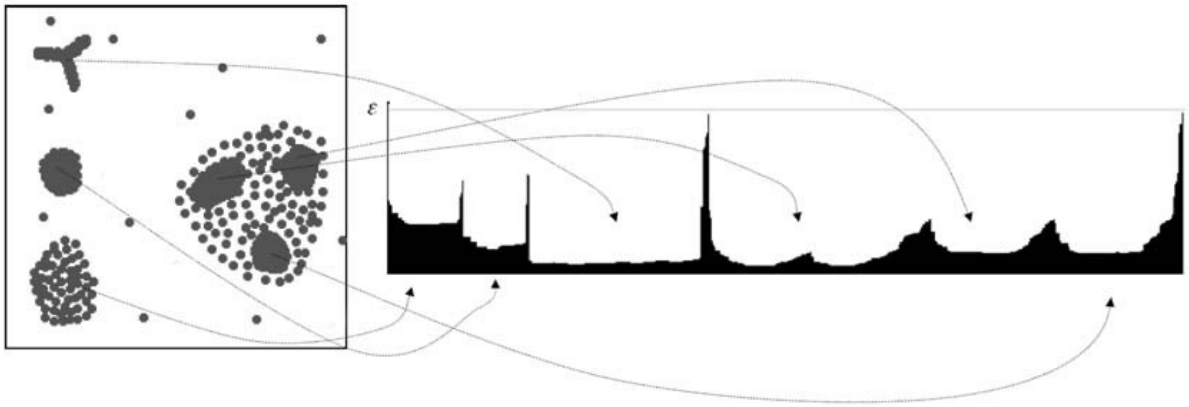
pienie, jeśli analizowany obiekt jest obiektem wewnętrznym. W przeciwnym przypadku dany obiekt jest klasyfikowany jako szum informacyjny (reprezentowany fizycznie przez przypisanie wartości zero jako identyfikatora skupienia). Natomiast w sytuacji, gdy odległość osiągalna dla analizowanego obiektu jest mniejsza niż *Ei*, to jest on przypisywany do aktualnego skupienia (ponieważ jest on wówczas gęstościowo osiągalny z obiektu poprzedzającego go w wygenerowanym uporządkowaniu).

Podział na grupy, wygenerowany przy użyciu przedstawionej procedury, jest niemalże nierozróżnialny w stosunku do działania algorytmu DBSCAN (dla tych samych wartości parametrów Eps i MinPts). Jedynie bardzo niewielka liczba obiektów krańcowych może być pominięta przez algorytm *WydobądźSkupienia*. Dzieje się tak w przypadku, gdy zostały one poddane analizie przez algorytm OPTICS, jeszcze zanim został znaleziony obiekt wewnętrzny, właściwy dla tego skupienia. Jednak, jak zapewniają autorzy samego algorytmu w pracy [1], jest to na tyle rzadko występująca sytuacja, że można pominąć etap naprawczy (tj. przydzielenia obiektów krańcowych do właściwego skupienia) bez znacznego pogorszenia jakości skupień.

5. Wizualizacja struktury zbioru danych

Wizualizacja struktury zbioru danych może być wykorzystana, w przypadku gdy analityk chce poznać ogólny zarys danych na wysokim poziomie abstrakcji. Wówczas zwraca się uwagę na występowanie hierarchii czy poziom spójności powstałych grup. Istotny może być również fakt, czy występują pewne dominujące grupy (o bardzo dużej liczności w stosunku do całego zbioru) czy też struktura ta ma charakter jednorodny. Bardzo często tylko na podstawie wygenerowanego podziału na grupy (zwłaszcza w kontekście dużych, złożonych zbiorów) analityk może nie być w stanie wyjaśnić bądź poprawnie zinterpretować otrzymanych rezultatów. Dlatego też coraz większy nacisk kładzie się właśnie na narzędzia wizualizacyjne, mające wspomagać procesy analizy i interpretacji wyników grupowania danych.

Aby zobaczyć szczegółową strukturę zbioru danych, należy stworzyć wykres wartości odległości osiągalnej dla każdego obiektu, zgodnie z uporządkowaniem wygenerowanym przez algorytm OPTICS. Taki wykres słupkowy nosi nazwę wykresu osiągalności (ang. *reachability plot*). Przykładowy wykres osiągalności dla sztucznie wygenerowanych danych dwuwymiarowych został przedstawiony na rysunku 2. Skupienia na wykresie reprezentowane są przez doliny. Im dana dolina jest węższa, tym mniej obiektów wchodzi w skład danego skupienia, natomiast im mniejsza jest wartość odległości osiągalnej, tym skupienie jest bardziej zagęszczone (bardziej spójne).



Rys. 2. Przykładowy wykres osiągalności
Fig. 2. Example of a reachability plot

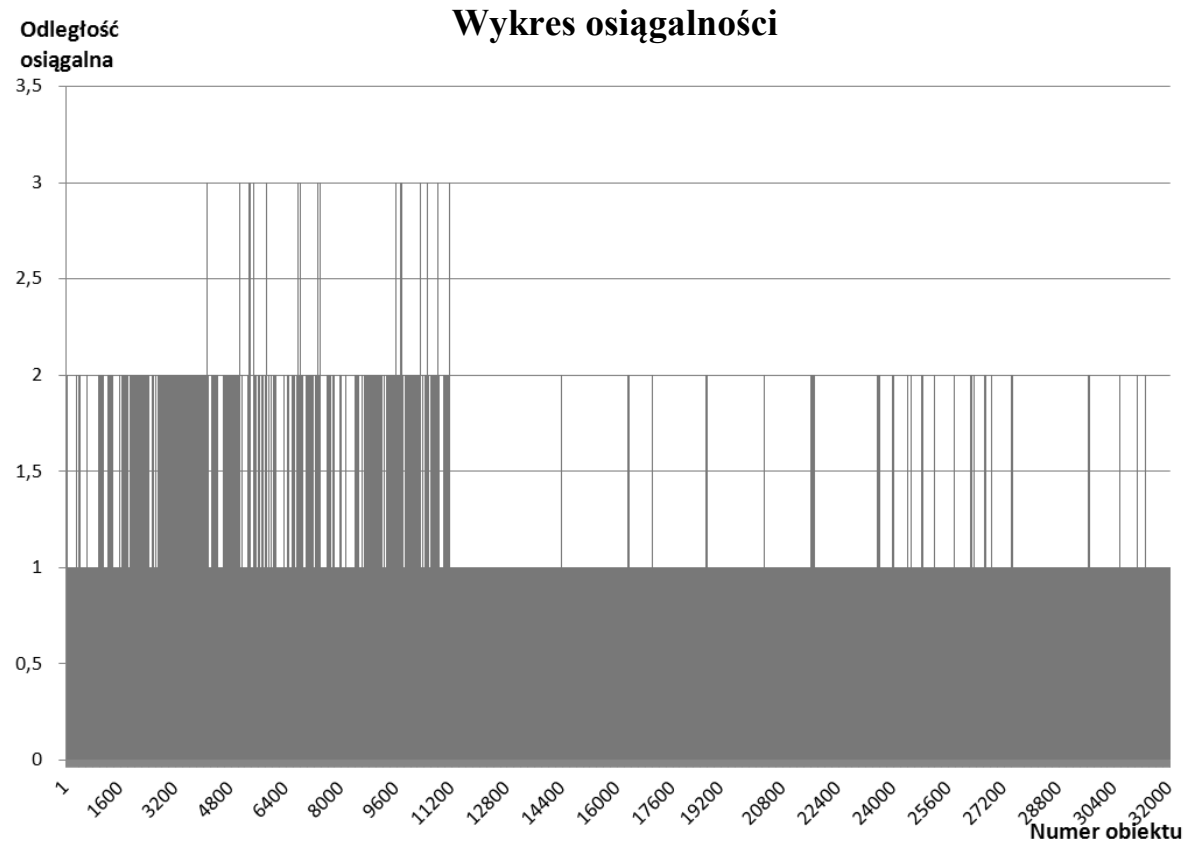
Na podstawie wykresu osiągalności można w łatwy sposób wykryć istnienie hierarchii (zawierania się) skupień. Hierarchia grup jest widoczna na wykresie (rys. 2) jako szereg małych dolin zawarty w jednej głębszej. Podczas identyfikowania przypadków potencjalnego zawierania się skupień należy zwrócić uwagę na to, by doliny były bardzo płytkie. Niska wartość odległości osiągalnej mówi o tym, że kolejny obiekt znajduje się bardzo blisko poprzedniego w ustalonym porządku, co dla płytkich dolin implikuje, że dwa skupienia są rozmieszone bardzo blisko siebie. Jeśli dodatkowo te płytkie doliny znajdują się w obrębie innej, bardzo głębokiej, to z dużą dozą prawdopodobieństwa mamy do czynienia z sytuacją, gdy szereg małych, spójnych skupień jest zawarty w jednym większym (o dużo mniejszym zagęszczeniu). Niestety dla rzeczywistych zbiorów danych złożonych identyfikacja hierarchii skupień na podstawie wykresu osiągalności jest już dużo trudniejsza. Rysunek 3 przedstawia wykres osiągalności dla rzeczywistego zbioru danych[2] złożonych odnośnie do urządzeń nadawczo-odbiorczych telefonii komórkowej. Wykres ten jest jednak „okrojony" do pierwszych 32 000 obiektów (uporządkowanych zgodnie z rezultatem działania algorytmu OPTICS) ze względu na czytelność oraz ograniczenia oprogramowania MS Excel, które nie umożliwia wizualizacji większej liczby danych na wykresie dwuwymiarowym w jednej serii.

Z przedstawionego na rysunku 3 wykresie wynika, iż już w początkowych jego obszarach (między obiektami o numerach od 8000 do ok. 10 000) daje się zauważyć występowanie hierarchii skupień – kilka mniejszych skupień jest zwartych w jednym większym. Zależności te nie są jednak tak łatwo identyfikowalne jak dla sztucznego zbioru danych (pokazanego na rys. 2).

Ponadto mimo znacznego ograniczenia liczby obiektów przedstawionych na wykresie – cały zbiór danych liczy 143 tysięcy obiektów – jego ogólna czytelność nie jest na wysokim poziomie. Dlatego też należy opracować inne metody tworzenia wykresu osiągalności, aby

[2] Zbiór danych poddawany analizie pozostaje niezmienny i został szczegółowo opisany w publikacji [8].

móc zaprezentować cały zbiór danych w formie zrozumiałej i czytelnej dla człowieka. Pewne prace w tym zakresie zostały już poczynione przez autorów samego algorytmu OPTICS – używają oni specyficznego typu wykresu kołowego, w którym każda wartość odległości osiągalnej ma przyporządkowany pewien określony odcień koloru. Rozpoznawanie rozkładu skupień i struktury zbioru odbywa się na podstawie identyfikacji jaśniejszych i ciemniejszych obszarów na wykresie. Szczegółowe informacje na ten temat znajdują się w m.in. w publikacji [1].



Rys. 3. Wykres osiągalności dla danych rzeczywistych
Fig. 3. Reachability plot for real-world data

Na czytelność wykresu osiągalności duży wpływ ma również użyta miara podobieństwa (odległości) obiektów. W tym przypadku miara podobieństwa była zdefiniowana jako liczba cech wspólnych. Implikuje to fakt, że jeżeli położone blisko siebie obiekty różnią się tylko wartościami dwóch cech, na przedstawionym wykresie jest to widoczne jako relatywnie duża zmiana. Z tego też powodu należy w przyszłości wykonać również testy przy użyciu innych miar odległości (które lepiej rozróżniają obiekty między sobą i mają większy zakres możliwych do przyjmowania wartości).

6. Podsumowanie

Celem niniejszego artykułu było przedstawienie dotychczas stosowanych rozwiązań implementacyjnych w zakresie grupowania dużych wolumenów danych, ze szczególnym uwzględnieniem ich wad i problemów. Kolejnym, istotnym celem artykułu były analiza i omówienie algorytmu OPTICS jako metody mającej zastosowanie przy grupowaniu oraz wizualizacji struktury dużych zbiorów danych złożonych. Szczególnym aspektem analizy było również porównanie algorytmu gęstościowego DBSCAN z algorytmem OPTICS. Porównanie to wykazało, że analizowane algorytmy mają wiele podobieństw, mimo że mogą być wykorzystywane w zupełnie różnych celach. Rozważania teoretyczne zostały poparte zastosowaniem algorytmu OPTICS do rzeczywistego zbioru danych odnośnie do telefonii komórkowej, w celu wizualizacji i interpretacji struktury tego zbioru.

BIBLIOGRAFIA

1. Ankerst M., Breunig M. M., Kriegel H. P., Sander J.: Optics: Ordering points to identify the clustering structure. SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data, Philadelphia, USA 1999.
2. Ester M., Kriegel H. P., Sander J., Xu X.: A density-based algorithm for discovering clusters in large spatial databases with noise. Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining, USA 1996.
3. Jain A. K., Murty M. N., Flynn P. J.: Data Clustering: A Review. ACM Computing Surveys, Vol. 31, No. 3, USA 1999.
4. Böhm Ch., Noll R., Plant C., Wackersreuther B.: Density-based Clustering using Graphics Processors. Proceeding of the 18th ACM conference on Information and knowledge management, USA 2009.
5. Berry M. W., Browne M.: Lecture notes in Data Mining. World Scientific Publishing Co. Pte. Ltd., Singapur 2009.
6. Tufféry S.: Data Mining and Statistics for Decision Making. Wiley & Sons Ltd., UK 2011.
7. Nowak-Brzezińska A., Xięski T.: Grupowanie danych złożonych. Studia Informatica, Vol. 32, No. 2A(96), Wydawnictwo Politechniki Śląskiej, Gliwice 2011, s. 391÷402.
8. Wakulicz-Deja A., Nowak-Brzezińska A., Xięski T.: Efficiency of complex data clustering. Lecture Notes in Artifitial Intelligence. Proceedings of 6th International Conference on Rough Sets and Knowledge Technology, RSKT 2011, Canada 2011.

9. Xięski T.: Grupowanie danych złożonych, [w:] Wakulicz-Deja A. (red.): Systemy wspomagania decyzji. Wydawnictwo Uniwersytetu Śląskiego, Katowice 2011.
10. Nowak-Brzezińska A., Jach T., Xięski T.: Wybór algorytmu grupowania a efektywność wyszukiwania dokumentów. Studia Informatica, Vol. 31, No. 2A(89), Wydawnictwo Politechniki Śląskiej, Gliwice 2010, s. 147÷162.

Wpłynęło do Redakcji 7 stycznia 2012 r.

Abstract

In this paper the topic of clustering and visualization of the data structure is discussed. Authors review currently found in literature algorithmic solutions that deal with clustering large volumes of data, focusing on their disadvantages and problems. What is more the authors introduce and analyze a density-based algorithms called OPTICS (*Ordering Points To Identify the Clustering Structure*) as a method for clustering a real-world data set about the functioning of transceivers of a cellular phone operator located in Poland. This algorithm is also presented as an relatively easy way for visualization of the data inner structure, relationships and hierarchies. The whole analysis is performed as a comparison to the well-known and described DBSCAN algorithm.

Adresy

Agnieszka NOWAK-BRZEZIŃSKA: Uniwersytet Śląski, Instytut Informatyki,
ul. Będzińska 39, 41-200 Sosnowiec, Polska, agnieszka.nowak@us.edu.pl.
Tomasz XIĘSKI: Uniwersytet Śląski, Instytut Informatyki,
ul. Będzińska 39, 41-200 Sosnowiec, Polska, tomasz.xieski@us.edu.pl.