STUDIA INFORMATICA 2009
Volume 30 Number 2A (83)

Agnieszka NOWAK, Tomasz JACH, Tomasz XIĘSKI
Uniwersytet Śląski, Instytut Informatyki

ANALIZA HIERARCHICZNYCH I NIEHIERARCHICZNYCH ALGORYTMÓW GRUPOWANIA DLA DOKUMENTÓW TEKSTOWYCH

Streszczenie. Autorzy prezentują wybrane metody grupowania dokumentów tekstowych za pomocą ręcznie generowanych słów kluczowych. Dokonano porównania hierarchicznych i niehierarchicznych algorytmów grupowania. Zaprezentowano wyniki obu grup algorytmów, uwzględniając kompletność i dokładność wyszukiwania. Podejścia sprawdzane są dla tego samego zbioru danych (tematów prac licencjackich).

Słowa kluczowe: grupowanie, dane tekstowe, K-medoids, Agnes

ANALYSIS OF HIERARCHICAL AND NON-HIERARCHICAL CLUSTERING ALGORITHMS FOR TEXTUAL DATA

Summary. Authors present selected clustering methods of text documents described by man-made keywords. Comparison of hierarchical and non-hierarchical algorithms is made. The results (both accuracy and completeness is included in the study) are presented for both types of algorithms. The hierarchical and non-hierarchical approaches are tested for the same data set, which consists of topics of undergraduate papers.

Keywords: clustering, textual data, K-medoids, Agnes

1. Wprowadzenie

W artykule poruszono zagadnienie stworzenia systemu wyszukiwania prac licencjackich z wykorzystaniem hierarchicznych i niehierarchicznych metod grupowania danych. Jak powszechnie wiadomo, problem wyboru tematów pracy licencjackiej przez studentów jest złożony, tym bardziej że jest to nierzadko ich pierwsza praca naukowa. Każdy z nich ma swoje za-

interesowania oraz pomysły na pracę, które jednakże bardzo często nie są sprecyzowane w sposób prawidłowy.

W obliczu tych problemów autorzy postanowili stworzyć system wspomagający podjęcie decyzji o temacie pracy licencjackiej na podstawie słów kluczowych, opisujących każdą pracę. Podczas budowania systemu posłużono się rzeczywistymi danymi zebranymi w Instytucie Informatyki Wydziału Informatyki i Nauki o Materiałach Uniwersytetu Śląskiego.

Jak wspomniano wcześniej, każdy ze studentów jest w stanie na podstawie swoich zainteresowań oraz specjalizacji wybrać dziedzinę, w której chciałby prowadzić badania. Jednakże samo określenie konkretnego tematu bardzo często nastręcza już trudności. Znając lub wybierając z listy słowa kluczowe, system jest w stanie, za pomocą algorytmów grupowania omówionych w późniejszych rozdziałach, podać konkretne tematy z danej dziedziny, które zostały już zrealizowane w ramach prac naukowych.

Innowacją systemu jest fakt, że dokonywana jest nie tylko prosta filtracja słów kluczowych, lecz także grupowanie prac na ich podstawie. Możliwa jest zatem sytuacja, w której użytkownik otrzyma w wyniku pracę, która nie zawiera w swoim opisie wejściowego słowa kluczowego, lecz jest bardzo zbliżona w swym opisie do prac zawierających słowa kluczowe wybrane przez użytkownika. Zachowanie to pozwala rozwiązać problem wąskiego zakresu prac opisywanych przez rzadko występujące słowa kluczowe.

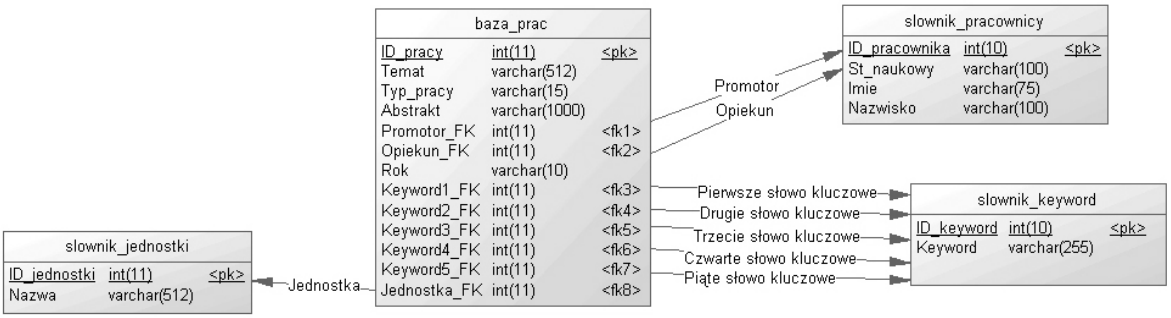
2. Struktura bazy danych

Jako podstawę do budowy bazy danych prac dyplomowych wybrano darmowe oprogramowanie *MySQL 5.0 Community Server* w wersji 5.0.51a. Ze względu na dużą popularność tego serwera bazodanowego możliwe jest wykorzystanie stworzonej bazy również do innych celów badawczych, niż zostały przewidziane przez jej autorów. Zapewnia on także dużą elastyczność, ponieważ dostępne są biblioteki programistyczne, umożliwiające podłączenie się do bazy przy użyciu najpopularniejszych języków programistycznych (jak C++, języki oparte na platformie .NET, Java, PHP) czy też korzystając ze standardów ODBC i JDBC.

Baza danych składa się z czterech głównych tabel: **baza_prac**, **slownik_pracownicy**, **slownik_keyword**, **slownik_jednostki**, co zostało zaprezentowane na rysunku 1. Obejmuje ona swym zakresem prace dyplomowe z okresu pięciu lat. Tabela baza_prac zawiera najważniejsze informacje o każdej pracy takie jak: jej *temat*, *typ* (licencjacka, magisterska), *promotor*, *rok obrony*, *nazwa uczelni* oraz *pięć unikalnych słów kluczowych*, opisujących możliwie najlepiej tematykę w niej ujętą. Dodatkowo zostały przewidziane w strukturze pola *abstrakt* (krótkie streszczenie pracy) oraz *opiekun pracy* (jeśli taki został również wyznaczony) w celu późniejszego wykorzystania, gdy stworzony system zostanie wdrożony na szeroką

skalę. Oczywiście pola *promotor* (i docelowo *opiekun*), *nazwa jednostki* i *słowa kluczowe* opisujące prace są kluczami obcymi, mającymi swoje odwzorowanie w odpowiednich tabelach słownikowych. Upraszcza to znacząco sposób aktualizacji bazy, zapobiega nadmiernej redundancji i problemowi niezgodności danych. Dodatkowo, stworzono dwa widoki pomocnicze realizujące odpowiednie złączenia na wymienionych tabelach w celu prezentacji danych zawartych w bazie w formie czytelnej i zrozumiałej dla użytkownika. Są one dodatkowo wykorzystywane przy procedurach aktualizacji danych.

Dość istotnym problemem było opisanie każdej pracy za pomocą pięciu słów kluczowych, w taki sposób by oddawały istotę danej pracy, a jednocześnie nie powodowały zbytniego rozdrobnienia bazy i nie stanowiły dla użytkownika przeszkody w korzystaniu ze stworzonego programu. Ze względu na dość duże narzucone ograniczenia czasowe, słowa kluczowe zostały stworzone ręcznie przez dwóch ekspertów, jedynie na podstawie tematu pracy i jej streszczenia (jeśli takie było dostępne). W chwili obecnej baza danych zawiera **360 dokumentów** opisanych za pomocą **407 słów kluczowych**. Planowane jest jednak wprowadzenie ujednoliconego standardu opisu każdej z prac (m.in. za pomocą słów kluczowych dobranych przez jej autora), co powinno skutecznie rozwiązać przedstawiony problem.



Rys. 1. Struktura bazy danych
Fig. 1. Structure of the data base

3. Opis algorytmu hierarchicznego

Inspiracją do stworzenia systemu grupującego i wyszukującego prace licencjackie był rozpowszechniony i uznany system SMART Saltona, w którym to dokumenty grupowano na podstawie ich podobieństwa między sobą [9, 11]. Powstałą w ten sposób strukturę, na czele z reprezentantami grup, przeszukiwano w dużo mniejszym czasie w stosunku do przeszukiwania liniowego całego zbioru dokumentów. Podobną ideę wykorzystuje również system Carrot2 [15, 16]. Grupuje on wyniki wyszukiwarek internetowych w tematycznie powiązane grupy, ułatwiając tym samym analizę i interpretację uzyskanych wyników. Oba systemy charakteryzują się tym, iż operują na dużej ilości danych. Jest to jedno z założeń tzw. eksploracji

danych – nauki zajmującej się odkrywaniem nowych, znaczących korelacji, wzorców i trendów przez przeszukiwanie dużych zbiorów danych przechowywanych w repozytoriach, wykorzystując technologie rozpoznawania wzorców, a także narzędzia matematyczne i statystyczne[1]. Jedną z najbardziej rozpowszechnionych technik eksploracji wiedzy jest grupowanie (zwane także analizą skupień).

Analiza skupień dzieli zestaw danych na sensowne grupy (skupienia). Grupuje ona dane wejściowe tylko i wyłącznie na podstawie informacji znalezionych w danych, które określają same obiekty i ich relacje między sobą. Istotne jest zachowanie zależności, która mówi, że obiekty wchodzące w skład jednej grupy powinny być jak najbardziej do siebie podobne oraz jak najbardziej niepodobne w stosunku do obiektów należących do innych grup [12].

Przedstawicielem pierwszej grupy metod analizy skupień, czyli tzw. aglomeracyjnych algorytmów hierarchicznych, jest algorytm Agnes. Ogólna zasada działania algorytmów aglomeracyjnych jest następująca: na początku zakłada się, że każdy z dokumentów jest reprezentantem oddzielnej grupy. Następnie, w kolejnych krokach, algorytm próbuje odnaleźć dokumenty, które są najbardziej do siebie podobne. Po ich odnalezieniu zostaje utworzona nowa grupa dokumentów, składająca się z tych najbardziej podobnych. Krok ten powtarza się, dopóki podobieństwo grup jest wyższe lub równe zadanemu progowi, bądź dopóki nie powstanie jedna grupa dokumentów [1].

Algorytmy aglomeracyjne wymagają do poprawnego działania macierzy podobieństwa. Struktura ta to macierz trójkątna dolna, w której znajduje się tyle samo wierszy co kolumn. Na przekątnej macierzy znajdują się zwykle same jedynki. Zarówno wiersze, jak i kolumny są etykietowane identyfikatorami dokumentów wchodzących w skład bazy danych podlegającej grupowaniu. Na przecięciu *i*-tej kolumny oraz *j*-tego wiersza znajduje się wartość podobieństwa dokumentu *i*-tego do *j*-tego. Im ta wartość jest większa, tym dokumenty bardziej do siebie podobne. Ze względu na to, iż jest to macierz symetryczna (*n x n*, gdzie *n* to liczba dokumentów), wystarczy wypełnić jedynie jej część pod (lub nad) przekątną.

Skuteczność algorytmów hierarchicznych zależy w ogromnym stopniu od wyboru sposobu wypełniania macierzy podobieństwa (czyli od wybranej metody wiązania obiektów w grupy). Trzy najprostsze, a zarazem najpopularniejsze metody badania podobieństwa dwóch obiektów stosowane przy podejściu hierarchicznym to: metoda całkowitego wiązania (ang. *complete linkage clustering*), metoda pojedynczego wiązania (ang. *single-linkage clustering*) oraz metoda średniego wiązania (ang. *average linkage clustering*) [2].

W przypadku metody całkowitego wiązania użyta jest funkcja max. Zdefiniowana jest ona następująco:

$$\max\{d(x,y) : x \in \mathcal{A},\ y \in \mathcal{B}\} \tag{1}$$

[1] Definicja według Gartner Group.

gdzie: x, y – porównywane obiekty, $\mathcal{A}$, $\mathcal{B}$ – skupienia, $d(x, y)$ – wybrana miara niepodobieństwa między dwoma obiektami.

Powoduje to, że tak zbudowane grupy są bardziej jednorodne, albowiem funkcja max bierze pod uwagę maksymalną odległość pomiędzy dwoma obiektami należącymi do grupy.

Przeciwieństwem tej metody jest metoda pojedynczego wiązania. W tym przypadku używana jest funkcja min, która bierze pod uwagę odległość do najbliższego sąsiada. Funkcja min zadana jest wzorem:

$$\min\{d(x, y) : x \in \mathcal{A},\ y \in \mathcal{B}\} \tag{2}$$

gdzie: x, y – porównywane obiekty, $\mathcal{A}$, $\mathcal{B}$ – skupienia, $d(x, y)$ – wybrana miara niepodobieństwa między dwoma obiektami.

Obie powyższe metody mają swoje wady i zalety. Metoda pojedynczego wiązania ma tendencje do tworzenia małej liczby heterogenicznych grup, a w rezultacie – zmniejszenia skuteczności wyszukiwania (bo należy przejrzeć większą część bazy danych). Całkowite wiązanie z kolei prowadzi do dylatacji - liczba grup może okazać się za duża [13].

Metoda średnich wiązań jest elementem pośrednim między dwoma przedstawionymi wcześniej metodami. W metodzie tej odległość między dwoma skupieniami oblicza się jako średnią odległość między wszystkimi parami obiektów należących do dwóch różnych skupień. Metoda ta jest efektywna, gdy obiekty formują naturalnie oddzielone „kępki", ale zdaje także egzamin w przypadku skupień wydłużonych, mających charakter „łańcucha".

Ze względu na przedstawione zalety metoda średnich wiązań została zastosowana w badanym algorytmie hierarchicznym – Agnes [8].

Tak jak zostało wspomniane wcześniej, elementem, który najbardziej wpływa na jakość ostatecznego zgrupowania, jest sposób tworzenia macierzy (nie)podobieństwa, a jedyną zauważalną wadą algorytmu jest jego relatywnie wysoka złożoność obliczeniowa [rzędu $O(n^2)$]. Do zalet z całą pewnością należy zaliczyć odporność na obiekty odległe (skrajnie niepodobne w stosunku do wszystkich innych), jak i fakt, że przeszukiwanie drzewa odbywa się w czasie logarytmicznym – złożoność tego procesu to $O(\log n)$ typowa dla drzew binarnych (jakie tworzy wspomniany algorytm).

W wyniku eksperymentów postanowiono wybierać liczbę klastrów dynamicznie. Zależy ona od wielkości bazy dokumentów i jest ustalana na 10% ilości dokumentów grupowanych. Dzięki temu średnio w grupie znajduje się 10 dokumentów.

4. Opis algorytmu niehierarchicznego

Wykorzystanym algorytmem analizy skupień był również algorytm K-medoids. Należy on do grupy algorytmów niehierarchicznych, które dzielą zbiór danych wejściowych na

k porcji, zwanych skupieniami. Operuje on na pojęciu medoida, czyli obiektu wybranego ze zbioru obiektów tworzących daną grupę, do którego odległość (niepodobieństwo) do wszystkich obiektów tej grupy jest najmniejsza. Wykorzystanie już istniejących obiektów jako reprezentantów grup powoduje, iż lepiej odzwierciedlają one powiązania i zawartość konkretnej grupy (w stosunku do centroidów, na których opiera się pierwowzór omawianego algorytmu – algorytm K-średnich), oraz umożliwiają relatywnie łatwe przeniesienie i implementację tej idei do różnorakich typów danych.

Pierwszym krokiem algorytmu K-medoids jest określenie przez użytkownika liczby skupień K. Następnie wybieranych jest losowo K obiektów (medoidów) ze zbioru danych, jako początkowych reprezentantów przyszłych grup. Skupienia są tworzone przez przypisanie pozostałych obiektów do najbliższych im (w sensie podobieństwa) medoidów. Następnie, medoidy są cyklicznie zastępowane przez obiekty niebędące medoidami, jeżeli tylko poprawi to jakość grupowania (rozumiana jako minimalizacja funkcji kosztu, która mierzy średnie niepodobieństwo między obiektem a medoidem). Algorytm kończy swe działanie, gdy zostanie spełnione określone kryterium zbieżności i podział się ustabilizuje.

Analizując uważnie przedstawiony opis algorytmu, można od razu dostrzec trzy najpoważniejsze wady tegoż algorytmu. Pierwszą z nich jest konieczność odgórnego ustalenia przez użytkownika liczby grup K. Jako iż nie mamy narzuconych żadnych kategorii, do jakich można by zaklasyfikować nasz zbiór prac, oraz nie znamy wewnętrznych powiązań między danymi, właściwe określenie tej liczby jest trudne lub wręcz niemożliwe. Błędne określenie początkowej liczby grup ma niestety bardzo negatywny wpływ na ostateczną jakość zgrupowania. Zbyt mała liczba powoduje, że w danym skupieniu występuje dużo szumu informacyjnego, który może zaciemniać istniejące powiązania między danymi, a podanie zbyt dużej liczby spowoduje nadmierne rozdrobnienie bazy danych i utratę kompletności informacji po przeprowadzeniu procesu wyszukiwania. Jednym z rozwiązań tego problemu jest kilkukrotne uruchomienie algorytmu z różnymi wartościami parametru K, oraz analiza jakości zgrupowania za pomocą odpowiedniego wskaźnika i ostatecznie przedstawienie jedynie najlepszego wyniku końcowego. Tutaj jednak przejawia się druga wada algorytmu. K-medoids charakteryzuje się dość dużą złożonością[2] [rzędu $O(k(n-k)^2)$, gdzie n to liczba obiektów, a k to liczba skupień], co w przypadku zastosowania proponowanego rozwiązania znacząco wydłuży czas działania programu i wpłynie negatywnie na komfort jego użytkowania. Tak wysoka złożoność wynika z konieczności wyznaczenia odległości (niepodobieństwa) aktualnie analizowanego obiektu (medoidu) w stosunku do wszystkich pozostałych obiektów i ewentualnej jego zamiany oraz cykliczności tego procesu [3].

Trzecią poważną wadą algorytmu K-medoids jest losowy sposób wyboru początkowych reprezentantów grup. Ze względu na niedeterministyczność tego procesu algorytm przy każ-

[2] Podana złożoność to złożoność jedynie jednej iteracji algorytmu.

dym kolejnym uruchomieniu generuje różny przydział obiektów do grup. Często zdarza się też, że w wyniku tego procesu przedstawiony ostateczny podział nie jest podziałem optymalnym, dobrze odwzorowującym rzeczywiste powiązania między danymi, a co za tym idzie użytkownik może otrzymać niezadowalające go rezultaty. Podobnie jak w poprzednim przypadku wielokrotne uruchomienie algorytmu i wybór jedynie najlepszego zgrupowania może stanowić częściowe rozwiązanie tego problemu. Innym rozpowszechnionym podejściem jest wybór medoidów jak najmniej podobnych w stosunku do tych już wybranych. Zmniejsza to znacząco prawdopodobieństwo złego sklasyfikowania obiektów do grup [5].

Do zalet algorytmu możemy z całą pewnością zaliczyć, tak jak już to zostało wspomniane wcześniej, wykorzystanie istniejących obiektów do reprezentowania skupień, przez co w niektórych sytuacjach można wykorzystać ich reprezentantów do opisu grup (co ułatwia analizę i interpretację uzyskanych wyników), oraz odporność na wartości izolowane (czyli obiekty skrajnie niepodobnie w stosunku do całej reszty), przez co radzi on sobie lepiej z zaszumianymi danymi niż jego pierwowzór [14].

Innym aspektem, który należy rozważyć przy omawianiu algorytmów analizy skupień, jest wybór odpowiedniej miary podobieństwa dwóch obiektów między sobą. W zależności od typu danych, z jakimi mamy do czynienia (binarnymi, dyskretnymi, ciągłymi), i stosowanej reprezentacji wiedzy należy stosować metrykę podobieństwa, która najlepiej charakteryzuje relacje między dwoma obiektami i najlepiej je rozróżnia [4].

Dla danych typu dyskretnego najczęściej stosowane metryki to odległość Euklidesowa i miejska. Ta pierwsza prezentuje się następująco:

$$euk(O_j, O_h) = \sqrt{\sum_{i=1}^{m} (O_{ji} - O_{hi})^2} \tag{3}$$

gdzie O_j, O_h – porównywane obiekty, m – wymiar przestrzeni cech, O_{ji} – i-ta współrzędna wektora stanowiącego opis obiektu O_j.

Druga z miar jest jeszcze prostsza i została opisana wzorem:

$$msk(O_j, O_h) = \sum_{i=1}^{m} | O_{ji} - O_{hi} | \tag{4}$$

gdzie O_j, O_h – porównywane obiekty, m – wymiar przestrzeni cech, O_{ji} – i-ta współrzędna wektora stanowiącego opis obiektu O_j.

Obydwie miary znacząco rosną w przypadku obiektów skrajnie niepodobnych do siebie względem jakiejś cechy, a co za tym idzie nie sprawdzają się, gdy dwa badane obiekty różnią się między sobą bardzo nieznacznie (np. tylko o jedną mała wartość). Do badania obiektów, których opis został przedstawiony w postaci wektora binarnego, zostały stworzone inne miary. Są to przykładowo odległość Hamminga, współczynnik Jaccarda czy współczynnik Simple Matching Coefficient [2].

We wszystkich prezentowanych wzorach zostanie zastosowana następująca notacja:

- f_{00} – liczba zmiennych, które na danej pozycji wektorów cech porównywanych obiektów osiągają wartość 0.
- f_{01} – liczba zmiennych, które na danej pozycji pierwszego wektora cech porównywanych obiektów mają wartość 0, a dla drugiego wektora wartość 1.
- f_{10} – liczba zmiennych, które na danej pozycji pierwszego wektora cech porównywanych obiektów mają wartość 1, a dla drugiego wektora wartość 0.
- f_{11} – liczba zmiennych, które na danej pozycji wektorów cech porównywanych obiektów osiągają wartość 1.
- O_j, O_h – porównywane obiekty.

I tak, odległość Hamminga definiowana jest wzorem:

$$ham(O_j, O_h) = f_{01} + f_{10} \tag{5}$$

Innymi słowy, odległość Hamminga to liczba cech, którymi różnią się dwa obiekty (wzorce). Współczynnik Simple Matching Coefficient określany jest następująco:

$$smc(O_j, O_h) = \frac{f_{00} + f_{11}}{f_{00} + f_{01} + f_{10} + f_{11}} \tag{6}$$

Natomiast współczynnik Jaccarda dany jest wzorem:

$$jac(O_j, O_h) = \frac{f_{11}}{f_{01} + f_{10} + f_{11}} \tag{7}$$

Jak widać, nie uwzględnia on sytuacji, gdy oba bity wektora binarnego są zerami. W zależności od wybranej reprezentacji danych może to być pożądane bądź nie.

Biorąc pod uwagę wszystkie przedstawione wcześniej aspekty jako sposób reprezentacji wiedzy wybrano reprezentację za pomocą cech nominalnych. Dany dokument reprezentowany był w programie jako wektor zawierający identyfikator pracy, oraz identyfikatory poszczególnych słów kluczowych z odpowiadającego im słownika terminów. Taka reprezentacja pozwoliła na efektywne wykorzystanie dostępnej pamięci oraz pozostała przy tym dość intuicyjna. Jako metrykę podobieństwa wykorzystano po licznych dyskusjach miarę Simple Matching Coefficient (SMC). Podobieństwo danego dokumentu do innego zostało określone zatem jako stosunek liczby cech wspólnych (liczby wspólnych słów kluczowych) do liczby wszystkich cech, jakimi opisane są te dokumenty. Takie podejście do mierzenia podobieństwa obiektów uwzględniało najlepiej (spośród przetestowanych miar) różnice między obiektami zakodowanymi za pomocą opisanej reprezentacji wiedzy [14].

5. Eksperymenty obliczeniowe

Głównym celem przeprowadzonych badań było porównanie hierarchicznych algorytmów analizy skupień (reprezentowanych przez algorytm Agnes) z niehierarchicznymi (których reprezentantem jest K-medoids), oraz analiza ich efektywności na zaprezentowanym specyficznym zestawie danych. W przypadku algorytmu K-medoids liczba grup była ustawiona na stałą wartość równą czterdzieści. Porównanie efektywności obu algorytmów odbędzie się na podstawie współczynników kompletności i dokładności wyszukiwania.

Kompletność rozumiana jest jako stosunek liczby relewantnych, wyszukanych dokumentów, do wszystkich relewantnych do zapytania dokumentów zawartych w bazie danych [9, 11]. Dokument jest relewantny do zadanego zapytania, jeśli zawiera co najmniej jedno podane (przez użytkownika) słowo kluczowe. Dokładność natomiast jest to stosunek liczby wyszukanych, relewantnych dokumentów do wszystkich wyszukanych prac. Warto zauważyć, że w systemach wspomagania decyzji dużo istotniejszym parametrem jest wysoka dokładność odpowiedzi ze względu na to, iż system powinien przede wszystkim odrzucać wszelkie dokumenty nierelewantne do podanego zapytania i nie popełniać żadnych błędów. Pełna kompletność nie jest w tym momencie wymagana [6, 7, 10].

Tabela 1

Kompletność i dokładność wyszukiwania dla przykładowych pytań do systemu

Lp.	Zapytanie	Agnes		K-medoids	
		Kompletność	Dokładność	Kompletność	Dokładność
1	administracja	0,684211	0,838710	0,026316	0,200000
2	zjawiska świetlne	1,000000	0,062500	1,000000	0,058824
3	telefonia komórkowa, telekomunikacja	0,375000	0,214286	0,250000	0,032258
4	Linux, C++	0,500000	0,333333	0,750000	0,103448
5	ACS, systemy mrowiskowe, mrowisko	0,451613	0,311111	0,129032	0,500000
6	systemy operacyjne, Windows, Linux	0,375000	1,000000	0,125000	0,117647
7	bazy danych, hurtownia danych, MySQL, GIS	0,300000	1,000000	0,125000	0,208333
8	Macromedia, Flash, MAMS, grafika komputerowa	0,580645	0,750000	0,645161	0,800000
9	administracja, analiza, urzędy i instytucje, straż pożarna, gmina	0,298611	1,000000	0,152778	0,916667
10	algorytmy ewolucyjne, algorytmy genetyczne, algorytmy, programowanie, C++	0,398230	1,000000	0,250000	0,852941

Tabela 1 prezentuje wyniki przeprowadzonych eksperymentów. Wynika z niej jasno, iż generalnie algorytmy hierarchiczne lepiej sprawują się na dostępnych danych wejściowych aniżeli algorytmy niehierarchiczne. Jak juz zostało wspomniane wcześniej, specyfika danych wejściowych uwydatnia podstawowe wady algorytmu K-medoids, jak konieczność uprzedniego podania liczby grup na jakie chcemy podzielić zbiór danych, czy też spora zależność końcowych wyników procesu grupowania od warunków początkowych. Algorytmy hierarchiczne pozbawione są tych wad, co wyjaśnia znaczne różnice w parametrach kompletności i dokładności.

Aby dodatkowo pokazać, jak bardzo ostateczne wyniki grupowania mogą zależeć od początkowego doboru zestawu reprezentantów (medoidów), wykonano test polegający na trzykrotnym przeprowadzeniu procesu grupowania i wyszukiwania dla tych samych słów kluczowych (*Macromedia*, *Flash*, *MAMS*, *grafika komputerowa*). Wyniki przedstawiono w tabeli 2.

Tabela 2

Efektywność podziału dla K-medoids
a początkowy wybór reprezentantów

Nr przebiegu	Kompletność	Dokładność
1	0,645161	0,800000
2	0,064516	0,048780
3	0,451612	0,451612

Efektywność algorytmu zmierzono również na specjalnie dobranym zestawie danych (tzn. są to takie dane, o których wiadomo, na jaką liczbę grup należy podzielić, oraz znany jest pożądany rozkład dokumentów do grup). Należy tutaj zaznaczyć, iż mamy do czynienia z jednym i tym samym zbiorem danych, jaki został zaprezentowany wcześniej. Tabela 3 przedstawia wybrane dokumenty odpowiednio wyselekcjonowane z całej bazy prac. Mamy zatem 9 prac odnośnie do administracji, 7 prac z dziedziny bioinformatyki, 5 dotyczących baz danych oraz 4 dotyczące programowania w Javie. Naturalnie najlepiej jest je podzielić na *cztery* grupy. Takiego właśnie rozkładu oczekuje się od algorytmu K-medoids. I rzeczywiście, potrafi on podzielić te prace zgodnie z naszymi oczekiwaniami. Niestety, klasyczna wersja algorytmu podczas pierwszego losowego wyboru medoidów może wybrać jako potencjalnych reprezentantów obiekty, które powinny znaleźć się razem w jednej grupie, co zaburza dość znacznie ostateczny podział, mimo że określiliśmy właściwą liczbę grup równą cztery.

Dzieje się tak w przypadku, gdy jako potencjalne medoidy zostaną wybrane tematy opisane następującymi słowami kluczowymi:

- **Medoid1**: (bazy danych, bezpieczeństwo, dydaktyka, kryptografia, szyfrowanie).
- **Medoid2**: (administracja, dane geograficzne, graficzna reprezentacja, specjalistyczne systemy, urzędy i instytucje).

- **Medoid3**: (administracja, dane geograficzne, informacja przestrzenna, służba publiczna, specjalistyczne systemy).
- **Medoid4**: (analiza, bioinformatyka, implementacja, medycyna, programowanie).

W grupie trzeciej pozostanie tylko jeden dokument (ponieważ wszystkie inne dokumenty są bardziej podobne do pozostałych medoidów i zostaną wcielone do innych grup), natomiast do grupy czwartej, zawierającej dokumenty z bioinformatyki, zostaną również dołączone prace z zakresu programowania, ponieważ medoid, mimo iż jest pracą z zakresu bioinformatyki, zawiera w swoim opisie termin "programowanie", tak jak dokumenty odnośnie do języka Java.

Tabela 3

Wyselekcjonowane dane z bazy

Temat pracy	Słowo Kluczowe1	Słowo Kluczowe2	Słowo Kluczowe3	Słowo Kluczowe4	Słowo Kluczowe5
T1	administracja	dane geograficzne	graficzna reprezentacja	służba zdrowia	specjalistyczne systemy
T2	administracja	dane geograficzne	informacja przestrzenna	służba publiczna	specjalistyczne systemy
T3	administracja	dane geograficzne	firmy	graficzna reprezentacja	specjalistyczne systemy
T4	administracja	dane geograficzne	graficzna reprezentacja	motoryzacja	specjalistyczne systemy
T5	administracja	dane geograficzne	graficzna reprezentacja	oświata	specjalistyczne systemy
T6	administracja	dane geograficzne	graficzna reprezentacja	specjalistyczne systemy	straz pozarna
T7	administracja	dane geograficzne	graficzna reprezentacja	kultura	specjalistyczne systemy
T8	administracja	dane geograficzne	graficzna reprezentacja	służba zdrowia	specjalistyczne systemy
T9	administracja	dane geograficzne	graficzna reprezentacja	specjalistyczne systemy	urzędy i instytucje
T10	analiza	bioinformatyka	ilość komórek	laboratorium	medycyna
T11	analiza	bioinformatyka	choroby oczu	medycyna	specjalistyczne systemy
T12	analiza	bioinformatyka	medycyna	systemy ekspertowe	wspomaganie
T13	analiza	bioinformatyka	implementacja	medycyna	programowanie
T14	analiza	bioinformatyka	społeczność	symulacja	sztuczna inteligencja
T15	analiza	bioinformatyka	mikroskop	minerały	rozpoznawanie obrazu
T16	analiza	bioinformatyka	grafika	medycyna	pozycjonowanie ciała
T17	bazy danych	bezpieczeństwo	kryptografia	laboratorium	szyfrowanie
T18	bazy danych	bezpieczeństwo	kryptografia	obliczenia	szyfrowanie
T19	bazy danych	bezpieczeństwo	kontrola dostępu	uwierzytelnienie	wiedza zerowa
T20	bazy danych	bezpieczeństwo	implementacja	kontrola dostępu	role
T21	bazy danych	bezpieczeństwo	dydaktyka	kryptografia	szyfrowanie
T22	Java	lista TABU	programowanie	systemy mrowiskowe	wersja operatorska
T23	Java	lista TABU	programowanie	systemy mrowiskowe	wersja operatorska
T24	Java	lista TABU	programowanie	systemy mrowiskowe	wersja instruktorska
T25	Java	lista TABU	programowanie	systemy mrowiskowe	wersja instruktorska

6. Wnioski

Przeprowadzone badania prowadzą do wniosków, iż niehierarchiczny algorytm grupujący K-medoids dla zaprezentowanego zestawu specyficznych danych tekstowych nie osiąga zadowalających i oczekiwanych wyników, co znajduje odzwierciedlenie podczas analizy poszczególnych kroków algorytmu. Sytuacji nie poprawiają zasugerowane wcześniej zmiany w sposobie inicjalizacji początkowych reprezentantów skupień, mimo że na przedstawionych danych idealnych podział generowany przez algorytm jest akceptowalny i poprawny. Świadczy to przede wszystkim o naturze danych wejściowych, które nie przejawiają tendencji do naturalnego podziału na grupy z wykorzystaniem algorytmów k-optymalizacyjnych.

Porównanie z algorytmem hierarchicznym wykazało jego przewagę w jakości otrzymywanych zgrupowań, co wskazuje, że to algorytmy z tej grupy lepiej sprawdzą się przy zadaniu grupowania w stosunku do tak specyficznych danych.

Kolejnym wnioskiem płynącym z pracy jest fakt, iż bez odpowiedniego standardu opisu każdej pracy dyplomowej, a co za tym idzie, stworzenia minimalnego zbioru słów kluczowych opisujących dokumenty zawarte w bazie, komfort użytkowania systemu znacząco spadnie (gdy użytkownik będzie miał do wyboru zbyt dużą liczbę słów kluczowych, nie będzie mógł podjąć decyzji co do wyboru właściwych), a sam program zamiast wspomagać decyzję może ją utrudnić. W przyszłości zatem należałoby zweryfikować opis zawartych prac pod kątem redukcji słów kluczowych i ich ujednolicenia. Być może głębsza analiza wyników otrzymywanych z systemu pozwoli na stworzenie ogólnych kategorii, do których zostaną sklasyfikowane wstępnie prace dyplomowe, a proces grupowania i wyszukiwania będzie się odbywał nie na całym zbiorze danych, a w konkretnej kategorii.

Warto również dla przedstawianego problemu rozważyć zastosowanie algorytmu grupowania rozmytego. Pozwoliłoby to wyznaczyć przynależność opisanych prac do wszystkich analizowanych grup i uwzględnić prace interdyscyplinarne.

BIBLIOGRAFIA

1. Abonyi J., Feil B.: Cluster Analysis for Data Mining and System Identification. Birkhäuser Verlag AG, Niemcy, 2007.
2. Jain A. K., Dubes R. C.: Algorithms for clustering data. Prentice Hall, New Jersey 1988.
3. Świniarski R. W., Kurgan Ł. A., Cios K. J., Pedrycz W.: Data mining. A Knowledge Discovery Approach. Springer Science+Business Media, LLC., USA, 2007.
4. Myatt G. J.: Making Sense of Data A Practical Guide to Exploratory Data Analysis and Data Mining. John Wiley and Sons, Inc., Hoboken, New Jersey 2007.

5. Kumar V., Tan P. N., Steinbach M.: Introduction to Data Mining. Addison-Wesley, USA, 2006.
6. Nowak A., Wakulicz-Deja A.: Effectiveness comparison of classification rules based on k-means clustering and Salton's method. Monitoring, Security and Rescue Techniques In Multiagent Systems, Advanced In Soft Computing, Springer-Verlag, Berlin 2005.
7. Nowak A., Wakulicz-Deja A.: Analiza efektywności wnioskowania w złożonych bazach wiedzy. Systemy Wspomagania Decyzji, Zakopane 2007.
8. Nowak A., Wakulicz-Deja A., Bachliński S.: Optimization of Speech Recognition by Clustering of Phones. Fundamenta Informaticae, Vol. 72, 2006, s. 283÷293.
9. Salton G.: Automatic Information Organization and Retreival. McGraw-Hill, New York, USA, 1975.
10. Wakulicz-Deja A.: Podstawy systemów wyszukiwania informacji. Analiza metod. Akademicka Oficyna Wydawnicza PLJ, Warszawa 1995.
11. C. J. Van Rijsbergen.: Information Retrieval, 2nd edition. Dept. of Computer Science, University of Glasgow, 1979.
12. Ćwik J., Koronacki J.: Statystyczne systemy uczące się. WNT, Waszawa 2005.
13. Jach T.: Grupowanie jako metoda eksploracji wiedzy w systemach wspomagania decyzji. Analiza algorytmów hierarchicznych. Sosnowiec 2008.
14. Xięski T.: Grupowanie jako metoda eksploracji wiedzy w systemach wspomagania decyzji. Analiza algorytmów niehierarchicznych (k-optymalizacyjnych). Sosnowiec 2008.
15. Osiński S., Weiss D.: Carrot2: An Open Source Framework for Search Results Clustering. 26th European Conference on Information Retrieval (Poster session), Sunderland, United Kingdom, 2004.
16. System Carrot2 - http://project.carrot2.org/.

Recenzent: Dr inż. Michał Kozielski

Wpłynęło do Redakcji 20 stycznia 2009 r.

Abstract

In this paper hierarchical and nonhierarchical algorithms of data mining are discussed. Authors focus on comparing the differences of both approaches on the same data set - the real life example are the topics of the undergraduate papers. Authors designed data base system to store information about papers. SQL tables were populated with data describing each paper. Each of the objects has five distinct keywords generated by human experts. The structure is

shown on figure 1. There are 360 documents described by 407 keywords in test set. Agnes (or AHC) is the hierarchical algorithm used in this paper. It is compared to k-medoids algorithm. Both of them are fine tuned to achieve best results in grouping similar documents into groups. During the experiment phase, completeness and accuracy is measured. Table nr 1 shows the results for different queries. As the conclusion, we state that hierarchical algorithm behaves much better than k-medoids approach.

Adresy

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