Zaawansowane technologie w bazach danych
Analiza sieci społecznych w portalu last.fm

Justyna Plewa
Paweł Pierzchała

22 czerwca 2010

1 Cel projektu

Celem projektu jest analiza społeczności tworzących się w portalach internetowych. W analizowanym serwisie wyszukujemy społeczności oraz określamy ich strukturę. Sprawdzamy jak ta struktura zmienia się w czasie.

2 Portal last.fm

Last.fm jest internetową radiostacją, system muzycznych rekomendacji oraz portalem społecznościowym. Każdy z użytkowników ma listę odtwarzanych utworów, którą może aktualizować na „żywo" używając plugin-ów do popularnych odtwarzaczy plików mp3. Gromadzone są również dane o koncertach na których bywa użytkownik. Ponad to serwis udostępnia funkcje typowe dla innych portali społecznościowych takie jak znajomi, galerie, komentarze.

Portal udostępnia publiczne dane użytkowników w formacie XML po przez web-service, którego ograniczeniem jest liczba 5 zapytań na sekundę.

W projekcie analizujemy następujące powiązania między użytkownikami:

- Lista znajomych
- Ulubione utwory – dwaj użytkownicy są powiązani, jeżeli mają taki sam ulubiony utwór
- Koncerty – dwaj użytkownicy są powiązani, jeżeli byli na tym samym koncercie

Informacje o koncertach udostępniane są wraz z ich terminem, który wykorzystujemy do wyodrębniania głównych członków grup.

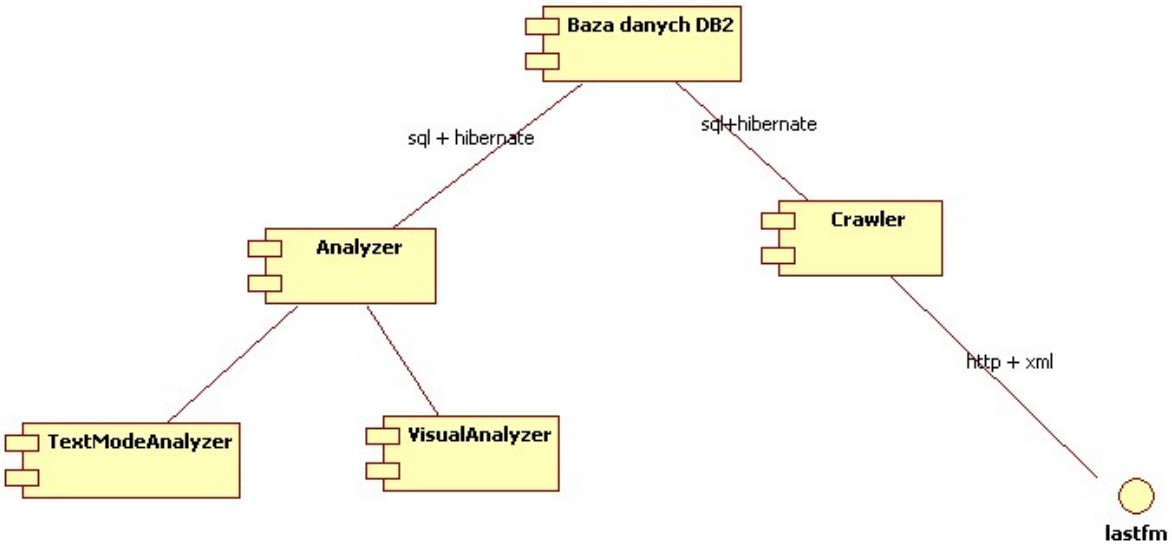
3 Technologie

Projekt został zaimplementowany w języku Java, z użyciem technologii:

- Hibernate – do mapowania danych z bazy DB2
- Jung – wykorzystaliśmy struktury danych, moduł do wizualizacji, alalgorytm klastrujący oraz narzędzia do obliczania miar sieci społecznych
- Baza danych DB2
- last.fm API bindings for Java do pobierania danych z portalu Last.fm

3.1 Architektura

Na załączonej ilustracji widoczne są komponenty projektu oraz komunikacja między nimi.



Rysunek 1: Komponenty projektu.

3.1.1 Komponent Crawler

Komponent Crawler wykorzystuje „last.fm API bindings for Java" do komunikacji z serwisem. Zapewnia on poprawne pobieranie danych po przez protokół HTTP oraz parsowanie plików XML z danymi. Przetworzone dane zapisywane są w bazie danych DB2 znajdującej się na uczelni.

3.1.2 Komponent Baza Danych DB2

Komponent baza danych DB2 umożliwia prosty dostęp danych po przez automatyczne mapowanie klas na rekordy w bazie danych przy pomocy hibernate. Komunikuje się z Crawlerem oraz komponentem Analysis.

3.1.3 Komponent Analysis

Jest to komponent zawierający funkcjonalności niezbędne do przeprowadzenia analiz. Umożliwia generowanie grafów powiązań na podstawie danych z bazy, generowanie raportów oraz analiz.

3.1.4 Komponent VisualAnlyzer

VisualAnalyzer jest graficznym interfejsem do komponentu Analysis. Umożliwia wizualizację sieci oraz sterowanie parametrem klastrowania.

3.1.5 Komponent TextModeAnalyzer

Komponent TextModeAnalyzer jest narzędziem wiersza poleceń które umożliwia generowanie raportów tekstowych z klastrowania.

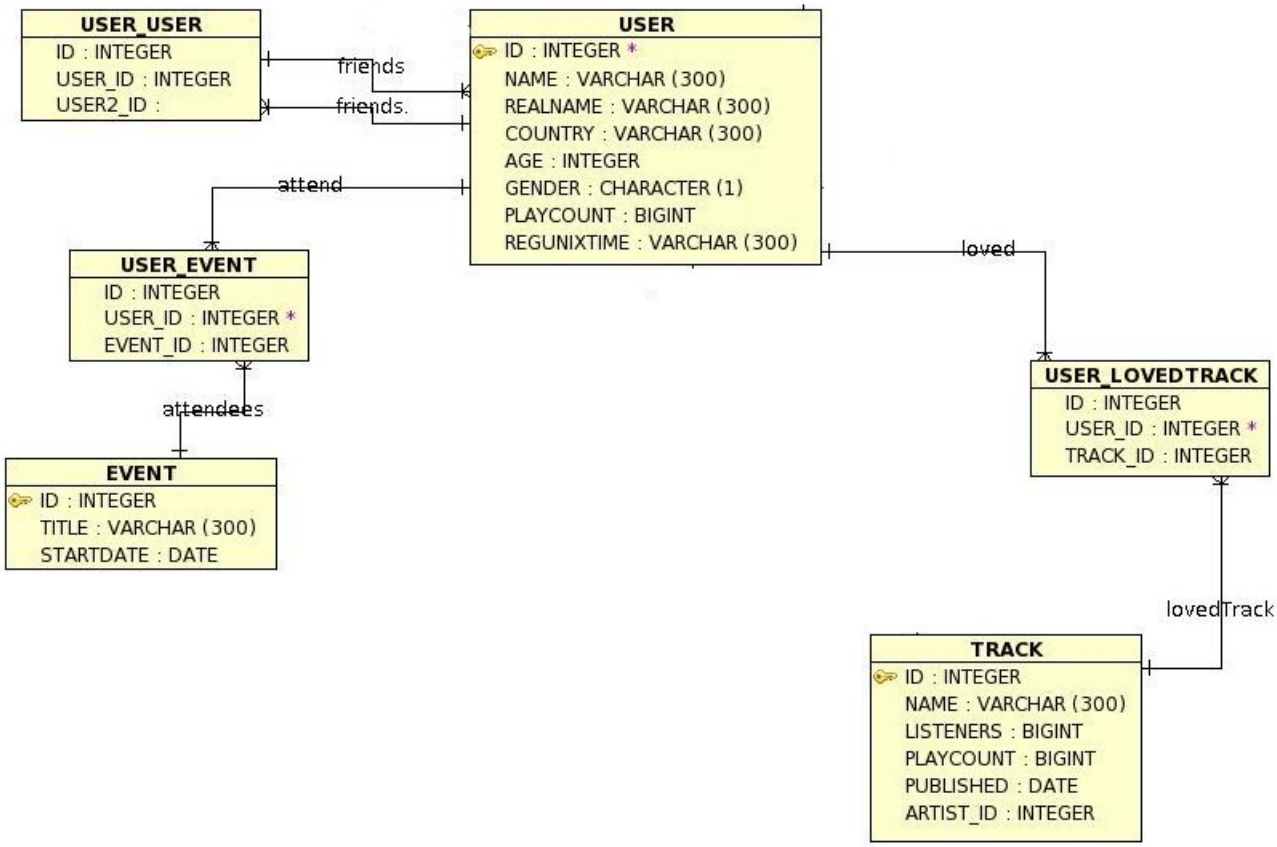
3.2 Pobrane dane

W trakcie semestru pobraliśmy z serwisu last.fm dane o:

- Użytkownikach – 9826 rekordów
- Znajomych – 13700 informacji o powiązaniu
- Utworach – 21 874 rekordów
- Ulubionych utworach użytkowników – 28 004 rekordów
- Koncertach – 31 251 rekordów
- Uczestnikach koncertów – 125 948 powiązań

3.3 Baza danych

Struktura bazy danych użyta w projekcie oddaję strukturę powiązań występujących w serwisie last.fm. Poniżej prezentujemy fragment schematu bazy danych który jest używany w projekcie, nie przedstawiamy na nim tabel z których nie korzystaliśmy (np. tabele na tagi lub shouty).



Rysunek 2: Struktura bazy danych.

3.3.1 Tabela User

Jeden rekord reprezentuje jedenego użytkownika serwisu Last.fm

Nazwa	Typ	Klucz	Opis
ID	Integer	Główny	Unikatowy identyfikator użytkownik
NAME	Varchar(300)		Pseudonim użytkownika
REALNAME	Varchar(300)		Imię i nazwisko
COUNTRY	Varchar(300)		Pochodzenie
AGE	Integer		Wiek użytkownika
GENDER	Character(1)		Płeć M/F
REGUNXTIME	Varchar(300)		Data rejestracji w formacie unixowym

3.3.2 Tabela User_User

Wiąże dwóch użytkowników, reprezentuje powiązanie „Znajomi” z portalu Last.fm.

Nazwa	Typ	Klucz	Opis
ID	Integer	Główny	Unikatowy identyfikator rekordu
USER_ID	Integer	Obcy	Użytkownik pierwszy
USER2_ID	Integer	Obcy	Użytkownik drugi

3.3.3 Tabela Event

Reprezentuje koncert.

Nazwa	Typ	Klucz	Opis
ID	Integer	Główny	Unikatowy identyfikator koncertu
TITLE	Varchar(300)		Nazwa koncertu
STARTDATE	Date		Data rozpoczęcia koncertu

3.3.4 Tabela User_Event

Zawiera informację o uczestnikach koncertu.

Nazwa	Typ	Klucz	Opis
ID	Integer	Główny	Unikatowy identyfikator rekordu
USER_ID	Integer	Obcy	Identyfikator użytkownika
EVENT_ID	Integer	Obcy	Identyfikator koncertu

3.3.5 Tabela Track

Tabela Track przechowuje informacje o utworach z serwisu last.fm.

Nazwa	Typ	Klucz	Opis
ID	Integer	Główny	Unikatowy identyfikator rekordu
NAME	Integer		Nazwa utworu
LISTENERS	BigInt		Liczba użytkowników słuchających utworu
PLAYCOUNT	BigInt		Liczba odtworzeń utworu
PUBLISHED	Date		Data publikacji
ARTIST_ID	Integer	Obcy	Identyfikator autora

3.3.6 Tabela User_LovedTrack

W tej tabeli znajdują się powiązania między użytkownikami a ich ulubionymi utworami.

Nazwa	Typ	Klucz	Opis
ID	Integer	Główny	Unikatowy identyfikator rekordu
USER_ID	Integer	Obcy	Identyfikator użytkownika
TRACK_ID	Integer	Obcy	Identyfikator utworu

3.4 Problemy

Początkowo planowaliśmy pobrać dane przy użyciu skryptów napisanych w języku Ruby, okazało się, że plugin którego chcieliśmy użyć nie umożliwiał nam na pobranie interesujących nas danych. Zrezygnowaliśmy z Rubego, do pobierania danych wykorzystujemy bibliotekę Javy.

W bibliotece „last.fm API bindings for Java" znajdował się błąd który uniemożliwiał pobieranie koncertów użytkownika z przeszłości. Po pobraniu źródeł biblioteki udało nam się ją naprawić.

4 Przeprowadzone analizy

Na podstawie informacji o powiązaniach miedzy użytkownikami tworzony jest nieskierowany graf sieci społecznej. Na tej sieci przeprowadzane jest klastrowanie przy pomocy algorytmu EdgeBetweennessClusterer.

EdgeBetweennessClusterer usuwa zadaną liczbę krawędzi o najwyższej wartości Betweenness (po każdorazowym usunięciu miara obliczana jest na nowo). Klastry w tak zredukowanym grafie wyszukiwane są przy użyciu algorytmu WeakComponentClusterer.

WeakComponentClusterer znajduje wszystkie składowe grafy, będące maksymalnym grafami w których między każdą parą wierzchołków istnieje ścieżka wewnątrz.

Dla każdego wierzchołka grafu obliczane są wartości jego:

- PageRank – *określa ważność węzła na podstawie liczności i ważności węzłów doń prowadzących.*
- Betweenness Centrality – *istotność węzła jest wyznaczana na podstawie liczby najkrótszych ścieżek przechodzących przez dany węzeł.*

Porównujemy wyniki klastrowania różnych sieci powiązań, aby określić ich pokrycie. *Dla dwóch wyników klastrowania A i B, dla każdego klastra z grupy A znajdujemy klaster z grupy B dla którego ich przecięcie jest największe. Określamy stosunek liczebności nowo powstałej grupy i klastra A, który jest pokryciem jednego klastra. Pokrycie jest średnią wszystkich pojedynczych pokryć.*

Użytkowników do generowania grafów wybierani losowo. Dla tak wybranej próby możemy stworzyć grafy odpowiednich powiązań

5 Wyniki eksperymentów

Poniżej prezentujemy wizualizację oraz analizę różnych grafów dla 200 użytkowników. Każdy z powstałych grafów klastrujemy EdgeBetweennessClustererem usuwając $\frac{1}{4}$, $\frac{1}{2}$, $\frac{3}{4}$ krawędzi. Dla każdego z tych przypadków prezentujemy wizualizację sieci, tabelę z wartościami miar tej sieci, histogram PageRank oraz BetweennessCentrality. Wynik w którym grupy są widoczne oraz istnieje niewiele grup jedno użytkownikowych jest opisany szczegółowo.

Graf rysowane są w następujący sposób:

- Koła oznaczają użytkowników
- Ciemne krawędzie oznaczają istniejące połączenia
- Jasne krawędzie to krawędzie usunięte w procesie klastrowania
- Użytkownicy z tej samej grupy oznaczani są takim samym kolorem

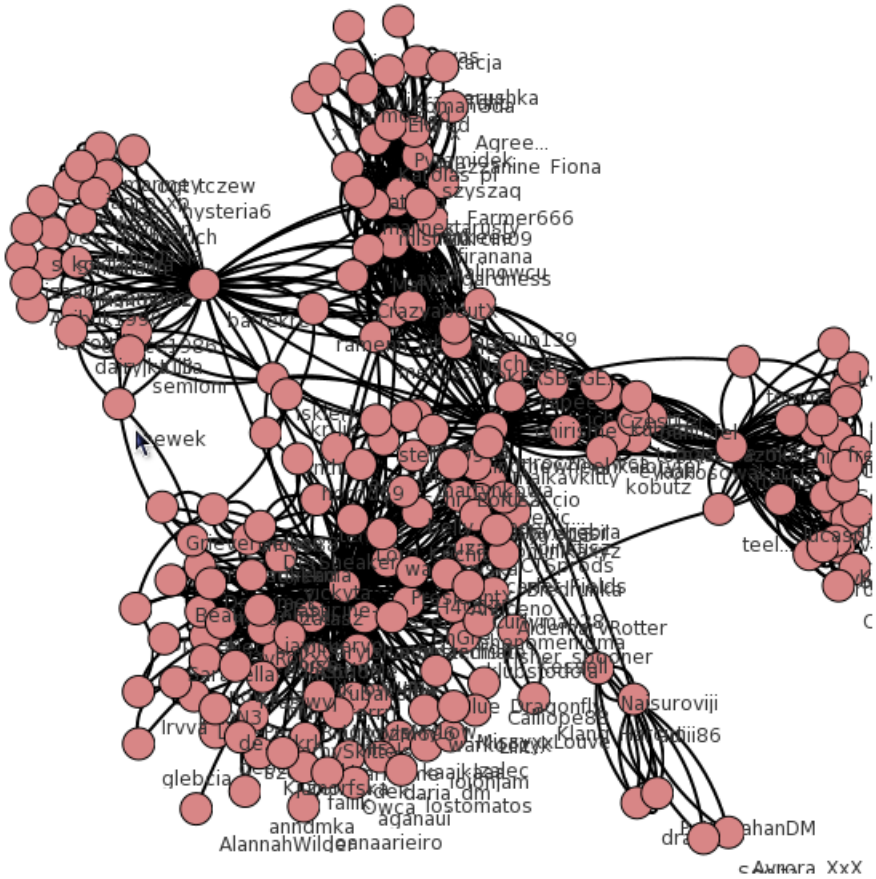
- Liczba kolorów jest ograniczona do 10 aby grupy były łatwo rozróżnialne. W sytuacji kiedy jest więcej niż 10 grup ich kolory mogą się powtarzać, natomiast węzły rozmieszczane są tak aby widoczny był brak połączeń ciemnymi krawędziami

5.1 Klastrowanie Znajomych

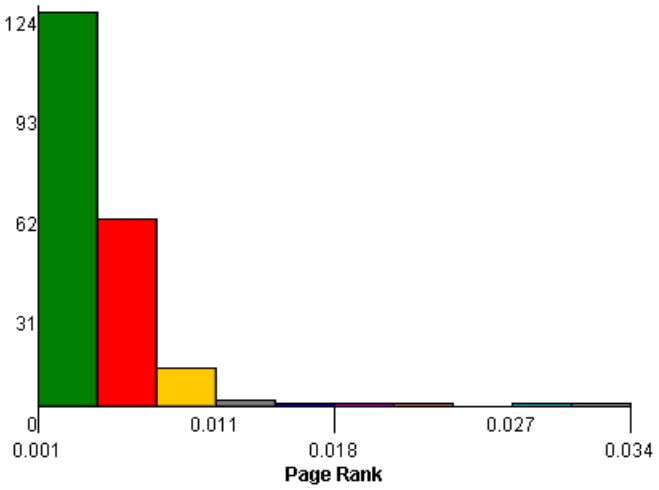
W tym rozdziale przedstawiamy wyniki klastrowania znajomych, grafu w którym dwaj użytkownicy są połączeni jeżeli są znajomymi w serwisie last.fm. Analizowany graf ma 200 węzłów i 1194 krawędzie.

5.2 Bez usuniętych krawędzi

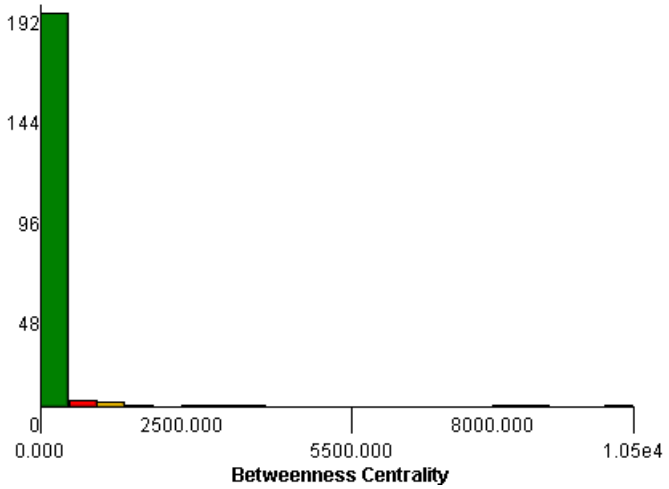
Rysunek poniżej przedstawia sieć znajomych z której nie usunięto jeszcze żadnych krawędzi.



Rysunek 3: Graf znajomych bez usuniętych krawędzi.



Rysunek 4: Histogram PageRank, 0 usniętych krawędzi, rozmiar przedziału 0.004.



Rysunek 5: Histogram Betweenness Centrality, 0 usniętych krawędzi, rozmiar przedziału 500.

Tablica 1: Miary grafu znajomych bez usuniętych krawędzi

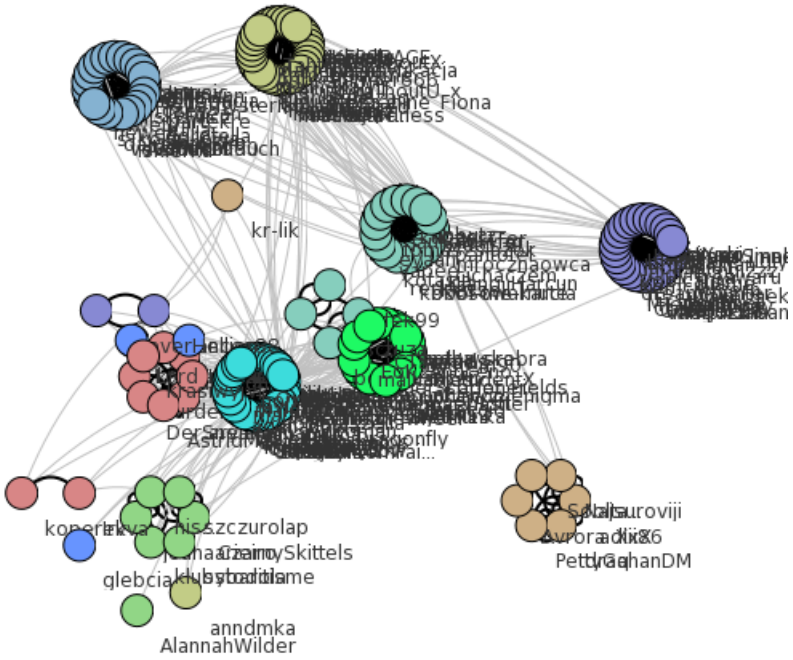
Miara	Średnia	Odchylenie standardowe
Liczba znajomych	5.97	8.76
PageRank	0.0050	0.0076
BetweennessCentrality	371.79	2241.43

Załączone powyżej rysunki przedstawiają normalną strukturę połączen występujacych w serwisie last.fm. W tabeli zgromadzone zostały wartości miar sieci społecznych oraz liczbę znajomych, większość użytkowników ma PR oraz BC z najniższego przedziału ponieważ w nieklas-

trowanym grafie jest niewielu istotnych użytkowników.

5.2.1 Usunięcie $\frac{1}{5}$ krawędzi

Kolejne ilustracje oraz tabele prezentują wynik klastrowania grafu pozbawionego 248 krawędzi.



Rysunek 6: Graf znajomych po usunięc $\frac{1}{5}$ krawędzi, czyli 248 z 1194

- MathHelper – Klasa zawiera metody pomocnicze do obliczania średniej oraz odchylenia standardowego.
- TextModeAnalyzer – Klasa służąca do uruchamiana analiz w trybie tekstowym.
- UserLabeller – Klasa która etykietuje użytkowników w wizualizacji
- VisualAnalyzer – Analizator w trybie graficznym

7.2 Pakiet crawler

Pakiet ten zawiera klasy służące do pobierania danych z serwisu Last.fm.

- Crawler – zawiera informacje o kluczu API last.fm
- EventCrawler – pobiera koncerty oraz ich uczestników
- LovedCrawler – pobiera ulubione utwory użytkowników
- UserCrawler – pobiera znajomych

7.3 Pakiet hiberex

Klasy pakietu hiberex odpowiedzialne są za mapowanie danych z bazy danych.

- AdditionalFunc – klasa zawierająca metody uproszczające zapytania do bazy danych
- SessionFactoryUtil - fabryka sesji, niezbędna do działania hibernate
- Klasy odpowiadające rekordom w bazie danych
  - Shout
  - User
  - Artist
  - Pair
  - Tag
  - Venue
  - Event
  - Playlist
  - Group
  - Track

Literatura

[1] Community structure in social and biological networks" http://www.pnas.org/content/99/12/7821.full.pdf

[2] „JUNG" http://jung.sourceforge.net/