19.09.2016

HDiSED

Projekt zasadniczy

Raport z implementacji

Autorzy:

Michał Urbański

Łukasz Janas

Wstęp:

Częścią projektu zasadniczego było zaimplementowanie jednego z rozwiązań algorytmicznych zawartych w tłumaczonym artykule. Do tego celu wybraliśmy fragment z rozdziału o powiązanych pracach, dotyczący wyznaczania punktacji jakości planu zapytania do bazy danych. Fragment ten znajduje się na 14 stronie oryginalnej wersji artykułu, w 3 akapicie.

Wersja oryginalna:

According to [24], the quality of a query plan is determined as follows. Each source receives information quality (IQ) scores for each criterion considered relevant, which are then combined in an IQ-vector where each component corresponds to a different criterion. Users can specify their preferences of the selected criteria by assigning weights to the components of the IQ-vector, hence obtaining a weighting vector. This weighting vector is used in turn by multi-attribute decision-making (MADM) methods for ranking the data sources participating in the universal relation. These methods range from the simple scaling and summing of the scores (SAW) to complex formulas based on concordance and discordance matrices. The quality model is independent of the MADM method chosen, as long as it supports user weighting and IQ-scores. Given IQ-vectors of sources, the goal is to obtain the IQ-vector of a plan containing the sources. Plans are described as trees of joins between the sources: leaves are sources whereas inner nodes are joins. IQ-scores are computed for each inner node bottom-up and the overall quality of the plan is given by the IQ-score of the root of the tree.

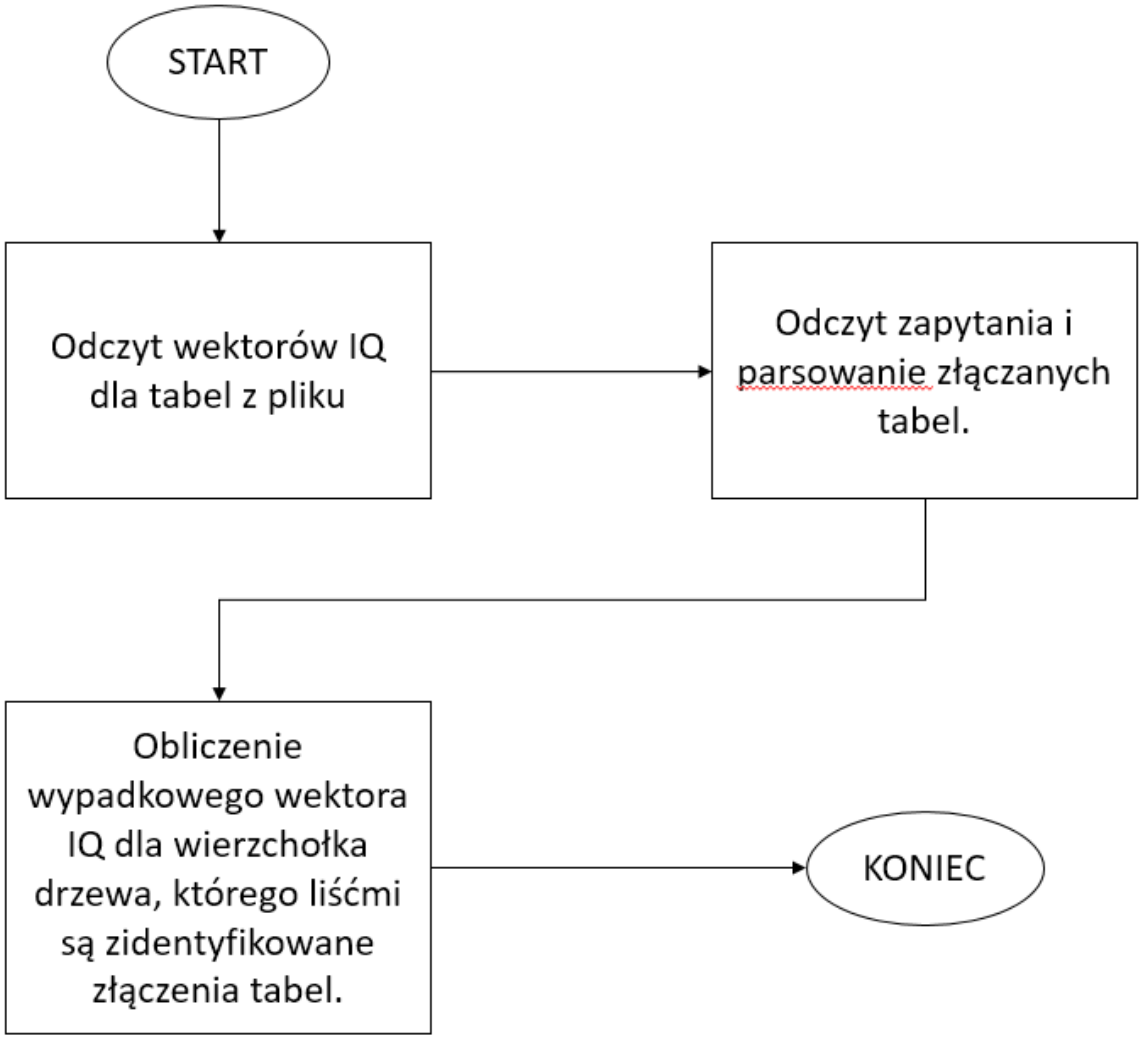
Nasze tłumaczenie:

Zgodnie z [24], jakość zapytania jest określana następująco. Każde źródło otrzymuje punktację jakości informacji (IQ – information quality) dla każdego kryterium uznawanego za istotne. Punktacje te są następnie formowane w wektor, gdzie każdy element odpowiada innemu kryterium. Użytkownicy mogą wyrazić swoje preferencje dotyczące danego kryterium poprzez nadawanie wag poszczególnym elementom wektora, tworząc wektor ważony. Wektor ten jest wykorzystywany z kolei przez wieloatrybutowe metody podejmowania decyzji (MADM multi-attribute decision-making) w celu utworzenia rankingu źródeł danych wchodzących w skład uniwersalnej relacji. Metodami tymi może być zarówno proste skalowanie i sumowanie punktacji jak i złożone wzory oparte o macierze zgodności. Model jakości nie zależy od wybranych metod MADM, jeżeli korzysta z wag użytkownika i punktacji jakości IQ. Celem jest, biorąc pod uwagę wektory IQ źródeł, uzyskanie wektora IQ planu zapytania zawierającego źródła. Plan zapytania może być rozumiany jako drzewo połączeń między źródłami: liśćmi są źródła danych, natomiast pozostałymi węzłami są złączenia. Punktacje jakości informacji są obliczane dla każdego wewnętrznego węzła, idąc od liści ku korzeniowi. Ogólna jakość planu zapytania dana jest w postaci punktacji jakości korzenia tego drzewa.

Dokonaliśmy wyboru tego właśnie fragmentu, ze względu na jego przejrzystość i zwięzłość.

Zadaniem stworzonego przez nas programu jest obliczanie wektora jakości informacji (information quality IQ vector) dla planu zapytania w języku SQL na podstawie wektorów jakości przyporządkowanym złączanym w zapytaniu tabelom.

Schemat blokowy programu:



Wektor IQ dla danego węzła drzewa jest obliczany jako średnia arytmetyczna wektorów IQ węzłów podrzędnych.

Format danych wejściowych:

Plik z deklaracjami tabel i wektorami IQ:

Poszczególne punktacje IQ powinny zawierać się w przedziale <0 ; 100>.

<nazwa_tabeli>, wiarygodność, celowość, reputacja, weryfikowalność

Przykład:

```
temperatura,20,30,10,60
pielegniarka,20,30,10,60
```

Plik z zapytaniem:

W tym pliku powinno znajdować się zapytanie w języku SQL

Przykład:

```
SELECT *
FROM temperatura, pielegniarka, pacjent
WHERE
temperatura.id_piel = pielegniarka.id
AND pacjent.id_temp = temperatura.id
```

Ścieżki do plików powinny zostać podane jako parametry wywołania programu (dostarczony w formacie .jar). Ścieżka do pliku z tabelami jest pierwszym argumentem, ścieżka do pliku z zapytaniem jest drugim argumentem.

Przykładowe wyniki:

```
TABLES:

temperatura: p = 20, c = 30, r = 10, w = 60
pielegniarka: p = 20, c = 30, r = 10, w = 60
lozko: p = 50, c = 35, r = 25, w = 45
pacjent: p = 1, c = 1, r = 1, w = 1
oddzial: p = 90, c = 90, r = 55, w = 78

QUERY:

SELECT *
FROM temperatura, pielegniarka, pacjent
WHERE
temperatura.id_piel = pielegniarka.id
AND pacjent.id_temp = temperatura.id

JOINS:

- temperatura <-> pielegniarka
- pacjent <-> temperatura
```

```
QUERY INFORMATION QUALITY VECTOR:


believability: 15

objectivity: 22

reputation: 7

verifiability: 45
```

Wnioski:

Podczas implementacji programu nie natrafiliśmy na żadne znaczące utrudnienia. Najciekawszą według nas częścią implementacji jest przetwarzanie zapytania w celu wydobycia złączanych w jego ramach tabel. Do tego celu posłużyły nam wyrażenia regularne.