Keyword Searching and Browsing in Databases using BANKS

Gaurav Bhalotia, Arvind Hulgeri, Charuta Nakhe, Soumen Chakrabarti, S. Sudarshan

Presenter: Monir Hajiaghayi
Discussion Leader: Ben Vandervalk
Motivation

- Web search engines are very successful
  - Simple and intuitive keyword query interface
- Database querying using keywords is desirable
  - Query languages, e.g., SQL/QBE, are not appropriate for casual users
  - Form interfaces cumbersome, give limited views
- Examples of keyword queries on databases
  - e-store database: “camcorder panasonic”
  - Book store: “sudarshan databases”
- Differences from IR/Web Search
  - Normalization splits related data across multiple tuples
  - Answer to a query is a set of (closely) connected tuples that match all given keywords
Basic Model

- Database: modeled as a graph
  - Nodes = tuples
  - Edges = references between tuples
    - foreign key, inclusion dependencies, etc.
  - Edges are directed

BANKS: Keyword search…
MultiQuery Optimization
Charuta
S. Sudarshan
Prasan Roy

S. Sudarshan
Prasan Roy
Charuta

Aug 2002
VLDB 2002 DEMO
Answer Model

- Rooted, directed tree connecting keyword nodes
  - May include internal nodes that contain no keywords
  - Root node has special significance
    - May be restricted to relations representing entities
    - Avoid relations representing relationships, e.g. “writes”
- Multiple answers may exist
  - Ranked by proximity + prestige

Eg> “Sudarshan Roy”
Discussion Question

• In the 90's, a board game came out called "Tribond". It was a trivia game where players had to find the "common link" between three items.
  – Example: What do "Bering", "Black", and "Baltic" have in common? (Answer: They are seas.)

• This is essentially what BANKS and DISCOVER systems do, but in the context of a relational database.

• What are the practical applications for finding the "common link" between a set of keywords in a database?
Relevance Calculation

• Proximity
  – Forward edges: foreign key → primary key
  – Weight of forward edge is based on schema
    • E.g. “cites” link weight greater than “writes” link weight
  – May need backward edges to form answer tree
    • Weight of backward edge $u \rightarrow v \propto$ indegree of $u$

• Node prestige based on indegree
Discussion Question

• On WebCT, many of you commented that the assignment of forward/reverse edge weights was complicated and ad hoc.
  – What criteria should be used for assigning edge weights? Is there a good way to assign the weights automatically? Or should the weights be assigned manually, based on the particular schema?
Searching for the Best Answers

- We have to use not just the tree with the highest relevance score but also those with high scores.

- Answers have to be generated incrementally so that the user are provided with the ‘best’ answers at the beginning.
Backward Expanding Search

- Incrementally computes search results
- **Start at leaf nodes** each containing a query keyword
- **Run concurrent single source shortest path algorithm** from each such node
  - Output a node whenever it is on the intersection of the sets of nodes reached from each keyword
- **Answer trees may not be generated in relevance order**
  - Insert answers to a small buffer (heap)
  - Output highest ranked answer from buffer to user when buffer is full
Searching for Best Answers

- Model (Query : Roy Sudarshan)
Browsing through BANKS

- BANKS system provides
  - A rich interface to browse data stored in a relational database
  - Automatically generates browsable views of database relations and query results
  - Schema browsing and data browsing
  - A hyperlink to the referenced tuple
Example of Browsing in BANKS

<table>
<thead>
<tr>
<th>SROLLNO</th>
<th>SNAME</th>
<th>FEMAIL</th>
<th>TITLE</th>
<th>DABBR</th>
</tr>
</thead>
<tbody>
<tr>
<td>90417401</td>
<td>Nand Kumar Singh</td>
<td><a href="mailto:sudhakar@aero.iit">sudhakar@aero.iit</a></td>
<td>Get column info and Drop column</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sort in Ascending order and these</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sort in Descending order</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Group by</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Group by prefix</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Join (FACULTY)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SELECT</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FROM STUDENTS, THESIS</td>
<td></td>
</tr>
<tr>
<td>91401702</td>
<td>N. Shama Rao</td>
<td><a href="mailto:mujumdar@aero.iitb.ernet.in">mujumdar@aero.iitb.ernet.in</a></td>
<td>THROUGH THICKNESS ELASTIC CONSTANTS AND STRENGTHS OF ADVANCED FIBRE COMPOSITES</td>
<td></td>
</tr>
<tr>
<td>91409005</td>
<td>Mini N Balu</td>
<td><a href="mailto:sys@math.iitb.ernet.in">sys@math.iitb.ernet.in</a></td>
<td>Some Preservation Results in Mathematical Theory of Reliability</td>
<td></td>
</tr>
</tbody>
</table>

3/30/2009
DISCOVER: Keyword Search in Relational Databases

- **Vagelis Hristidis**
  University of California, San Diego

- **Yannis Papakonstantinou**
  University of California, San Diego
Motivation

• Currently, information discovery in databases requires:
  – Knowledge of schema
  – Knowledge of a query language (eg: SQL)
  – Knowledge of the role of the keywords

• DISCOVER eliminates these requirements
Keyword Query Semantics
(definition of “document” in databases)

Keywords are:
• in same tuple
• in same relation
• in tuples connected through primary-foreign key relationships

Score of result:
• distance of keywords within a tuple
• distance between keywords in terms of primary-foreign key connections
• IR-style score of result tree
Result of Keyword Query

Result is tree $T$ of tuples where:

• each edge corresponds to a primary-foreign key relationship

• every keyword contained in a tuple of $T$ (total)

• no tuple of $T$ is redundant (minimal)
Discussion Question

• In BANKS/DISCOVER "search hits" are not documents but rather trees of connected tuples.
• The BANKS paper shows an example result for the keyword search “soumen sunita”:

Are these results easy for the user to understand? How could the results be displayed/navigated so that the system is more intuitive for the user?
Example - Schema

Subset of TPC-H schema

ORDERS \(\xleftarrow{n:1}\) CUSTOMER \(\xrightarrow{n:1}\) NATION
Example - Data

<table>
<thead>
<tr>
<th>ORDERKEY</th>
<th>CUSTKEY</th>
<th>TOTALPRICE</th>
<th>CLERK</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000105</td>
<td>12312</td>
<td>$5,000</td>
<td>John Smith</td>
</tr>
<tr>
<td>1000111</td>
<td>12312</td>
<td>$3,000</td>
<td>Mike Miller</td>
</tr>
<tr>
<td>1000125</td>
<td>10001</td>
<td>$7,000</td>
<td>Mike Miller</td>
</tr>
<tr>
<td>1000110</td>
<td>10002</td>
<td>$8,000</td>
<td>Keith Brown</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CUSTOMER</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTKEY</td>
</tr>
<tr>
<td>12312</td>
</tr>
<tr>
<td>10001</td>
</tr>
<tr>
<td>10013</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>NATIONKEY</td>
</tr>
<tr>
<td>01</td>
</tr>
</tbody>
</table>
Example – Keyword Query

**Smith**  **Miller**

<table>
<thead>
<tr>
<th>ORDERKEY</th>
<th>CUSTKEY</th>
<th>TOTALPRICE</th>
<th>CLERK</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000105</td>
<td>12312</td>
<td>$5,000</td>
<td>Smith</td>
</tr>
<tr>
<td>1000111</td>
<td>12312</td>
<td>$3,000</td>
<td>Miller</td>
</tr>
<tr>
<td>1000125</td>
<td>10001</td>
<td>$7,000</td>
<td>Miller</td>
</tr>
<tr>
<td>1000110</td>
<td>10002</td>
<td>$8,000</td>
<td>Keith Brown</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CUSTKEY</th>
<th>NAME</th>
<th>NATIONKEY</th>
</tr>
</thead>
<tbody>
<tr>
<td>12312</td>
<td>Brad Lou</td>
<td>01</td>
</tr>
<tr>
<td>10001</td>
<td>George Walters</td>
<td>01</td>
</tr>
<tr>
<td>10013</td>
<td>John Roberts</td>
<td>01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>NATIONKEY</td>
</tr>
<tr>
<td>01</td>
</tr>
</tbody>
</table>
Example – Keyword Query

Query: “Smith, Miller”

<table>
<thead>
<tr>
<th>ORDERKEY</th>
<th>CUSTKEY</th>
<th>TOTALPRICE</th>
<th>CLERK</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000105</td>
<td>12312</td>
<td>$5,000</td>
<td>Smith</td>
</tr>
<tr>
<td>1000111</td>
<td>12312</td>
<td>$3,000</td>
<td>Miller</td>
</tr>
<tr>
<td>1000125</td>
<td>10001</td>
<td>$7,000</td>
<td>Miller</td>
</tr>
<tr>
<td>1000110</td>
<td>10002</td>
<td>$8,000</td>
<td>Keith Brown</td>
</tr>
</tbody>
</table>

Results:

<table>
<thead>
<tr>
<th>Size</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>o₁ ← c₁ → o₂</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CUSTKEY</th>
<th>NAME</th>
<th>NATIONKEY</th>
</tr>
</thead>
<tbody>
<tr>
<td>12312</td>
<td>Brad Lou</td>
<td>01</td>
</tr>
<tr>
<td>10001</td>
<td>George Walters</td>
<td>01</td>
</tr>
<tr>
<td>10013</td>
<td>John Roberts</td>
<td>01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>NATIONKEY</td>
</tr>
<tr>
<td>01</td>
</tr>
</tbody>
</table>
Example – Keyword Query

**Smith**  **Miller**

<table>
<thead>
<tr>
<th>ORDERKEY</th>
<th>CUSTKEY</th>
<th>TOTALPRICE</th>
<th>CLERK</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000105</td>
<td>12312</td>
<td>$5,000</td>
<td>Smith</td>
</tr>
<tr>
<td>1000111</td>
<td>12312</td>
<td>$3,000</td>
<td>Miller</td>
</tr>
<tr>
<td>1000125</td>
<td>10001</td>
<td>$7,000</td>
<td>Miller</td>
</tr>
<tr>
<td>1000110</td>
<td>10002</td>
<td>$8,000</td>
<td>Keith Brown</td>
</tr>
</tbody>
</table>

**Size** | **Result**
---|---
2 | \( o_1 \leftarrow c_1 \rightarrow o_2 \)
4 | \( o_1 \leftarrow c_1 \leftarrow n_1 \rightarrow c_2 \rightarrow o_3 \)

**NATION**

<table>
<thead>
<tr>
<th>NATIONKEY</th>
<th>NAME</th>
<th>REGIONKEY</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>USA</td>
<td>N.America</td>
</tr>
</tbody>
</table>
Architecture

Keywords

Tuple Sets

Database Schema

Candidate Networks

Master Index

Candidate Network Generator

Plan Generator

Execution Plan

Joining Networks of tuples

SQL queries

Database

"Smith","Miller"

ORDERS$^{Smith} =$\{0_1\}

ORDERS$^{Miller} =$\{0_2,0_3\}

ORDERS$^{Smith}$$\bowtie$$CUSTOMER$$\bowtie$$ORDERS^{Miller}$$,

ORDERS$^{Smith}$$\bowtie$$CUSTOMER$$\bowtie$$NATION$$\bowtie$$

CUSTOMER$$\bowtie$$ORDERS^{Miller}

T_1 \leftarrow ORDERS^{Smith}$$\bowtie$$CUSTOMER

C_1 \leftarrow T_1$$\bowtie$$ORDERS^{Miller}

C_2 \leftarrow T_1$$\bowtie$$NATION$$\bowtie$$CUSTOMER$$\bowtie$$ORDERS^{Miller}

ORDERS$^{Smith}$, CUSTOMERS WHERE ... SELECT * FROM T_1, ORDERS^{Miller} WHERE ... SELECT * FROM T_1, NATION, CUSTOMERS, ORDERS^{Miller} WHERE ...
Architecture

User

"Smith", "Miller"

ORDERS^Smith = \{o_1\}
ORDERS^Miller = \{o_2, o_3\}

ORDER$Smith \bowtie$CUSTOMER \bowtie$ORDERS^Miller$
ORDER$Smith \bowtie$CUSTOMER \bowtie$NATION \bowtie$CUSTOMER \bowtie$ORDERS^Miller$

T_1 \leftarrow ORDER$Smith \bowtie$CUSTOMER
C_1 \leftarrow T_1 \bowtie$ORDERS^Miller
C_2 \leftarrow T_1 \bowtie$NATION \bowtie$CUSTOMER \bowtie$ORDERS^Miller

CREATE TABLE T1 AS SELECT * FROM
ORDERS^Smith, CUSTOMERS WHERE ...
SELECT * FROM T_1, ORDERS^Miller WHERE ...
SELECT * FROM T_1, NATION, CUSTOMERS, ORDERS^Miller WHERE ...
Candidate Networks Generator - Definition

- **Candidate Network** is a connected graph of tuple sets, where:
  - each edge has corresponding edge in schema graph
  - each keyword contained in at least one tuple set
  - there are no redundant tuple sets (with no keyword or not helping connect other keyword relations)
Candidate Network - Example

ORDERSMiller

CUSTOMERNATION

n:1

ORDERSMiller

n:1

ORDERSSmith

n:1

n:1
Candidate Network - Example

\[ \text{CN1: } O^\text{Smith} \leftarrow C \rightarrow O^\text{Miller} \quad \text{size}=2 \]
Candidate Network - Example

CN1: $O^{Smith} \leftarrow C \rightarrow O^{Miller}$  size=2

CN2: $O^{Smith} \leftarrow C \leftarrow N \rightarrow C \rightarrow O^{Miller}$  size=4
Candidate Network - Example

ORDERS
Smith

ORDERS
Miller

CUSTOMER

NATION

CN3: O_{Smith} ← C → O_{Miller} ← C

size=3
Candidate Network - Example

**CN4:**  
\[ \text{Smith} \rightarrow \text{O} \rightarrow \text{C} \rightarrow \text{O} \rightarrow \text{Smith} \]

\[ \text{Miller} \rightarrow \text{O} \rightarrow \text{C} \rightarrow \text{O} \rightarrow \text{Miller} \]

\[ \text{size} = 4 \]

\[ c_1 \rightarrow o \rightarrow c_2 \]

\[ c_1 \equiv c_2 \], because primary to foreign key from CUSTOMER to ORDERS

**Pruning Condition:** \( R^K \rightarrow S \leftarrow R^L \)
Architecture

User

Keywords

Master Index

Tuple Sets

Candidate Network Generator

Database Schema

Candidate Networks

Plan Generator

Execution Plan

Plan Execution

Joining Networks of tuples

SQL queries

Database
Execution Plan

- Each CN corresponds to a SQL statement
- CN1: $O^{Smith} \leftarrow C \rightarrow O^{Miller}$

CN2: $O^{Smith} \leftarrow C \leftarrow N \rightarrow C \rightarrow O^{Miller}$

- Execution Plan
  CN1 $\leftarrow O^{Smith} \triangleright\triangleleft C \triangleright\triangleleft O^{Miller}$
  CN2 $\leftarrow O^{Smith} \triangleright\triangleleft C \triangleright\triangleleft N \triangleright\triangleleft C \triangleright\triangleleft O^{Miller}$
Reuse Common Subexpressions - Example

• Execution Plan
  CN1 ← O^{Smith} ▷◁ C ▷◁ O^{Miller}
  CN2 ← O^{Smith} ▷◁ C ▷◁ N ▷◁ C ▷◁ O^{Miller}

• Optimized Execution Plan
  Temp ← O^{Smith} ▷◁ C
  CN1 ← Temp ▷◁ O^{Miller}
  CN2 ← Temp ▷◁ N ▷◁ C ▷◁ O^{Miller}
Discussion Question

• BANKS and DISCOVER share the same goal of enabling keyword searches on relational databases. What are the key differences between the BANKS approach and the DISCOVER approach?

• If you wanted to add keyword search to your database, which system would you rather use?
Thank You
Any Question??