Advanced Data Management

Medha Atre

Office: KD-219
atrem@cse.iitk.ac.in

Aug 8, 2016
Groups for the course project are due on **August 22, 2016 18:00 IST**. Instructions on how to submit project groups will be posted soon.

While emailing me always start subject line with **[CS698F]** (with square brackets), else emails may get ignored.
Recap

- BitMat structure.
- Fold and Unfold procedures.
- *Semi-joins.*
- Nice properties of *acyclic* query graphs.
- N-way multi-joins.
Graph data and queries

Data

:Jerry :hasFriend :Larry
:Jerry :hasFriend :Julia
:Larry :actedIn :CurbYourEnthu
:Julia :actedIn :Seinfeld
:Julia :actedIn :Veep
:Julia :actedIn :CurbYourEnthu
:Julia :actedIn :NewAdvOldChristine
:Seinfeld :location :NewYorkCity
:Veep :location :D.C.
:CurbYourEnthu :location :LosAngeles
:NewAdvOldChristine :location :Jersey

SPARQL

SELECT ?friend ?sitcom WHERE {
  :Jerry :hasFriend ?friend .
}

Eqv. SQL query

SELECT t1.o, t2.o from rdf as t1, rdf as t2, rdf as t3 WHERE t1.s=":Jerry" and t1.p=":hasFriend" and t2.p=":actedIn" and t3.p=":location" and t3.o=":NewYorkCity" and t1.o=t2.s and t2.o=t3.s

Graphical Representation
fold($BM_{tp}$, $RetainDimension$) procedure is nothing but projection of distinct values from the given dimension of BitMat, e.g., in the triple pattern (?friend :actedIn ?sitcom) if $BM_{tp}$ is an O-S BitMat, then ?sitcom is in the “row” dimension of the BitMat.

$$fold(BM_{tp}, dim_{?j}) \equiv \pi_{?j}(BM_{tp})$$
For every unset bit in the MaskBitArray, $\text{unfold}(BM_{tp}, \text{MaskBitArray}, RetainDimension)$ clears all the bits corresponding to that position of the RetainDimension.

\[
\text{unfold}(BM_{tp}, \beta_{?j}, \text{dim}_{?j}) \equiv \{ t \mid t \in BM_{tp}, t.?j \in \beta_{?j} \}
\]

$t$ is a triple in $BM_{tp}$ that matches $tp$. $\beta_{?j}$ is the MaskBitArray containing bindings of $?j$ to be retained. $\text{dim}_{?j}$ is the dimension of $BM_{tp}$ that represents $?j$, and $t.?j$ is a binding of $?j$ in triple $t$. In short, unfold keeps only those triples whose respective bindings of $?j$ are set to 1 in $\beta_{?j}$, and removes all other.
Semi-join and clustered-semi-join

- $tp_2 \bowtie ?j tp_1 = \pi_{attr(tp_2)}(tp_2 \bowtie ?j tp_1)$ is a semi-join [Bernstein1981, Ullman1989].
- A clustered-semi-join between $(tp_1, tp_2, ... tp_n)$ over $?j$ is similar to $n$-way semi-join.
- Semi-joins are achieved through the fold and unfold primitives of BitMat.

```
\begin{align*}
\text{tp1} & :\text{Jerry} :\text{hasFriend} ?\text{friend} \\
\text{tp2} & :?\text{friend} :\text{actedIn} ?\text{sitcom} \\
\text{tp3} & :?\text{sitcom} :\text{location} :\text{NewYorkCity} \\
\end{align*}
```

`tp2` left with all the triples

```
\begin{align*}
\text{tp2} & \bowtie (?\text{friend}) \text{tp1} \\
\text{tp2} & \bowtie (?\text{sitcom}) \text{tp3} \\
\text{tp2} & \bowtie (?\text{friend}) \text{tp3} \\
\end{align*}
```

Now `tp2` left with only one triple

```
\begin{align*}
\text{tp2} & \bowtie (?\text{sitcom}) \text{tp3} \\
\text{tp2} & \bowtie (?\text{friend}) \text{tp3} \\
\text{tp3} & \leftarrow \text{left with the original one triple}
\end{align*}
```
If the Graph of Tables (GoT) is acyclic (tree), then the tuples in each table can be reduced to a minimal by traversing the GoT in a bottom-up followed by top-down fashion, performing a semi-join at each table node [Bernstein1981, Ullman1989].

- A table has minimal tuples for a query, if every tuple contributes to at least one final result, none of the tuples gets eliminated in the final result generation.

- If the Graph of Triple Patterns (GoT) is acyclic, the Graph of Join-variables (GoJ) is acyclic too, and vice versa (Lemma 3.2 in [Atre2015]).
Pattern Query Processing

- Choose the \textit{least selective} join variable (jvar) as the root of the GoJ tree, so that more selective jvars are leaves\(^1\), and do a bottom-up and top-down pass on GoJ with \textit{clustered-semi-joins} at each jvar.
  - This leaves a \textit{minimal} set of triples in the BitMat associated with each triple pattern.

- Do \textit{n-way multi-join} to join all the triple patterns to produce the final results.

---

\(^1\) Any jvar can be chosen as the root, but this \textit{anti-greedy} selection favors query performance.
N-way multi-joins

Reuse the same vmap over and over
Contemporary Systems

- RDF-3X [Neumann2010]
- gStore [Zou2011]
- TripleBit [Yuan2013]
- Virtuoso
- MonetDB
- Neo4j
RDF-3X

- Assumes the graph as a 3-column table.
- Creates all 6-way indexes – PSO, POS, SPO, SOP, OPS, OSP.
- Index compression using *delta-encoding*.
- Indexes are created as compressed B+ trees.
- Creates a *pipelined left-deep* join operator tree.
- *Sideways-information-passing* during scans and merge-joins.
- Aggressive *selectivity estimation* for all possible single edge patterns.