Exercise 4

Matthias Gollwitzer, Jan Schalkamp

May 26, 2013

1 EXERCISE 1

1.1 GOO

Consider the cardinalities of relations R_1 to R_4 and the query graph as given in figures 1.1 and 1.2:

Relations	
Relation	Cardinality
R_1	4
R_2	4
R_3	1.000
R_4	1.000

Figure 1.1: Example relations

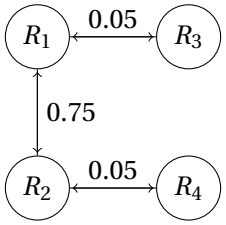

Figure 1.2: Query graph with cardinalities

Figure 1.3 shows the resulting join tree of the GOO algorithm. Labels beside the edges depict the cardinality. The values in brackets depict C_{out}.

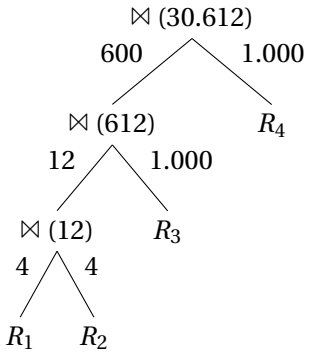

Figure 1.3: GOO join tree

The optimal join tree shown in figure 1.4, however, has lower cost. As one can see, the cost in the GOO generated join tree are 30.612, whereas the optimal join tree's cost are 30.400.

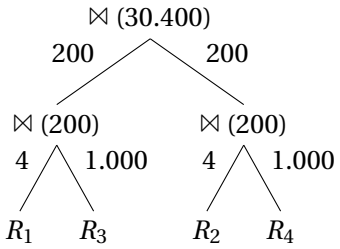

Figure 1.4: Optimal join tree

1.2 IKKBZ

-

2 Exercise 2

Compiling of our even-tinierDB is done via *make* command - just like tinyDB. To run an example of the second exercise, please execute *./bin/homework3*. This will build the query graph for the following query:

```
select v.titel, v.gelesenvon, p.persnr, p.name
from vorlesungen v, professoren p, hoeren h
where   v.titel = p.persnr and
        v.gelesenvon = p.persnr and
        h.vorlnr = v.vorlnr;
```

Figure 2.1: Example query

Important files resp. classes for this exercise are (further explanation via comments in each class):

- **src/compiler/querygraph/Edge**: Represents an edge between 2 nodes. Selectivity estimation is done here in the constructor.
- **src/compiler/querygraph/Node**: Represents a relation within the query graph. Estimated cardinality is done each time ::addSelection() is called.
- **src/compiler/querygraph/QueryGraph**: Wrapper class. Manages all edges/nodes and their correct insertion into the graph.
- **src/compiler/QueryGraphCompiler**: Reads the parsed query and acts accordingly. Relations will be added to QueryGraph as nodes, joins as edges. Self-joins and selections will be added via addSelection in order to estimate the cardinality of a certain node.
- **src/parser/Parser**: Parses the actual query string and returns a Query object.
- **src/parser/Query**: Important data structure, which stores all important informations of the query.
- **homeworks/homework3**: provides means of executing and testing the current homework.