Declarative Query Tuning and Optimization using Answer Set Programming

Yuliya Lierler[1] and Philip Cannata[1,2]

[1] University of Texas at Austin, TX 78712
[2] Oracle, Austin TX 78727

1 Query Optimization

In [2, Chapter 14], Lewis describes the basics behind the procedure used by the ORACLE query optimizer for computing the optimal join order for a sample query. Given a set T of n pairs *table:cost*, a *join order* is a full binary tree such that (i) T is the set of its leaves, (ii) the right child of an inner node is a leaf, and (iii) each inner node is a pair *join tag:cost* where *join tag* is either nested-loop (*nl*), sort-merge (*sm*), or hash (*ha*), and *cost* is defined recursively using its children's costs and its join tag. Requirement (ii) ensures that a join order is a left-deep tree [5]. The *join order cost* is the sum of the costs of its inner nodes. The *optimal* join order is an order with the minimal cost. Figure 1 (a) illustrates a sample join order for a set {*c:2517*, *p:631*, *gp:127*, *ggp:64*} of pairs, which expresses that the tables *gp* and *p* are joined first, then the resulting table is joined with *c*, and at last the table *ggp* is joined. An *nl* join is used for each of the three joins. The join order cost is $360 = 347 + 12 + 1$.

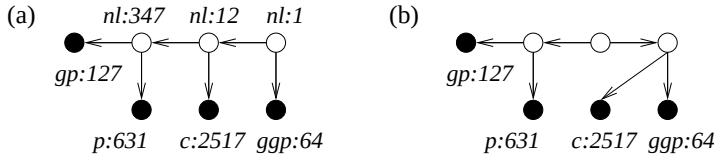

Fig. 1. (a) optimal join order 11 in [2, Chapter 14]; (b) bushy tree that is not left-deep.

In this paper we show how answer set programming (ASP) can be used to generate optimal join orders. ASP is a declarative programming paradigm oriented towards difficult combinatorial search problems [4]. The main advantage of a declarative ASP approach is that a programmer, instead of designing special-purpose search procedures, builds a constraint logic program that encodes the problem and utilizes an answer set solver for finding the solutions.

2 Computing Join Order using Answer Set Programming

A common methodology to solve a problem in ASP is to design two main parts of a program: GENERATE and TEST [3]. The former defines a larger collection

of answer sets that could be seen as potential solutions. The latter consists of constraints that eliminate the answer sets that do not correspond to solutions. Often a third part of the program, DEFINE, is also necessary to express auxiliary concepts that are used to encode the constraints.

In an ASP approach to computing an optimal join order, the goal is to encode the problem as a logic program so that the answer sets of the program correspond to the join orders. Once a join order J is computed by an answer set solver CLINGCON[3] [1], J is then used to express an additional constraint to instruct the solver to find a join order with a smaller cost than the cost of J. As a result of multiple invocations of CLINGCON on the original program with the additional constraints, an optimal join order is computed.

Due to the lack of space, we demonstrate only a few GENERATE, DEFINE, and TEST rules from the logic program for finding join orders.[4] GENERATE rule

```
1{parentLeftRight(P,C1,C2):node(C1;C2)}1 :- innerNode(P).
```

states that each inner node of the full binary tree has two children. The following rule from DEFINE encodes the cost formula for *sm* join [2, Chapter 14]:

```
cost(sm,P)$==tSort(L)+tCost(R)+tSort(R) :- parentLeftRight(P,L,R).
```

The constraint from TEST

```
:- joinTag(J,P), cost(J,P)$>cost(J1,P), J!=J1, join(J;J1).
```

forbids solutions where an inner node of the tree is assigned a join tag whose cost is not minimal.

We also note that our logic program with a simple modification allows lifting the requirement *(ii)* of the join order definition; thereby permitting answer sets that correspond to *bushy tree* join orderings [5] such as, the one illustrated in Figure 1 (b). Comparing this approach to the state of the art large join query optimization algorithms is the topic of our future work.

References

1. Gebser, M., Ostrowski, M., Schaub, T.: Constraint answer set solving. In: Proceedings of 25th International Conference on Logic Programming (ICLP'09) (2009)
2. Lewis, J.: Cost-Based Oracle Fundamentals. Apress (2005)
3. Lifschitz, V.: Answer set programming and plan generation. Artificial Intelligence (2002)
4. Marek, V., Truszczyński, M.: Stable models and an alternative logic programming paradigm. In: The Logic Programming Paradigm: a 25-Year Perspective (1999)
5. Steinbrunn, M., Moerkotte, G., Kemper, A.: Heuristic and randomized optimization for the join ordering problem. The VLDB Journal (1997)

[3] http://www.cs.uni-potsdam.de/clingcon/

[4] The complete logic program: http://www.cs.utexas.edu/users/tag/query .