Axiomatic System for Order Dependencies

Jaroslaw Szlichta[1,2], Parke Godfrey[1] Jarek Gryz[1], and Calisto Zuzarte[2]

[1] York University, Toronto, Canada
{jszlicht, jarek, godfrey}@cse.yorku.ca
[2] IBM Centre for Advanced Studies, Toronto, Canada
calisto@ca.ibm.com

Abstract. Dependencies play an important role in database theory. We study *order dependencies* (ODs) and *unidirectional* order dependencies (UODs), which describe the relationships among *lexicographical* orderings of sets of tuples. We investigate the *axiomatization* problem for order dependencies.

1 Fundamentals

Understanding the semantics of data is important, both for data quality analysis and knowledge discovery. While the relational data model is *set* based and does not concede the concept of *order*, ordered streams nonetheless play important roles in relational systems. SQL allows one to specify by its order-by clause that the answer "set" be returned in the specified order. Ordered streams are prevalent in query plans between operators to provide efficient evaluation. A query optimizer must reason extensively over *interesting orders* while building efficient query plans [5, 7].

We are interested in *lexicographical ordering*, or *nested sort*, as is provided by SQL's order-by directive. Let $\mathbf{X} = [\mathsf{A} \mid \mathbf{T}]$ be a list of attributes, the attribute A is the *head* of the list, and the list $\mathbf{T}$ the tail. For two tuples s and t, $s \preceq_{\mathbf{X}} t$ *iff* $s_{\mathsf{A}} < t_{\mathsf{A}}$ *or* ($s_{\mathsf{A}} = t_{\mathsf{A}}$ and ($\mathbf{T} = [\,]$ or $s \preceq_{\mathbf{T}} t$)).

Given two lists of attributes $\mathbf{X}$ and $\mathbf{Y}$, $\mathbf{X} \mapsto \mathbf{Y}$ denotes a *unidirectional order dependency* (UOD) [5], read as $\mathbf{X}$ *orders* $\mathbf{Y}$. Table $\mathbf{r}$ *satisfies* $\mathbf{X} \mapsto \mathbf{Y}$ *iff*, for all $s, t \in \mathbf{r}$, $s \preceq_{\mathbf{X}} t$ implies $s \preceq_{\mathbf{Y}} t$. That is, given table $\mathbf{r}$, any list of the table's tuples that satisfies order by $\mathbf{X}$ also satisfies order by $\mathbf{Y}$. Also, $\mathbf{X} \sim \mathbf{Y}$ denotes that $\mathbf{X} \leftrightarrow \mathbf{Y}$. The default *direction* of the order for SQL's order-by is *ascending*. We also consider order-by's that mix asc and desc directives; e.g., order by A asc, B desc. This generalization we simply call *order dependency* (OD) [4, 7].

Example 1. Let $\mathbf{r}$ be a table with attributes A, B, C, D, E, F (Table 1). Note that $[\mathsf{A},\mathsf{B},\mathsf{C}] \mapsto [\mathsf{F},\mathsf{E},\mathsf{D}]$ is satisfied by $\mathbf{r}$ but $[\mathsf{A},\mathsf{B},\mathsf{C}] \mapsto [\mathsf{F},\mathsf{D},\mathsf{E}]$ is falsified by $\mathbf{r}$. Also note $\mathbf{r} \models [\overrightarrow{\mathsf{C}}, \overleftarrow{\mathsf{A}}] \mapsto [\overrightarrow{\mathsf{B}}, \overleftarrow{\mathsf{D}}, \overleftarrow{\mathsf{E}}]$, but $\mathbf{r} \not\models [\overrightarrow{\mathsf{C}}, \overleftarrow{\mathsf{A}}] \mapsto [\overleftarrow{\mathsf{E}}, \overrightarrow{\mathsf{B}}, \overleftarrow{\mathsf{D}}]$.

Functional dependencies (FDs) are to group-by as ODs are to order-by. Given $\mathbf{X} \mapsto \mathbf{Y}$, if one has an SQL query with order by $\mathbf{Y}$, one can rewrite the query with order by $\mathbf{X}$ instead, and meet the intent of the original query. Assume we have a tree index for $\mathbf{X}$. This index can help in a query plan with the semantic

Table 1. Relation instance **r**.

#	A	B	C	D	E	F
s	3	2	0	4	7	9
t	3	2	1	3	8	9

information that $\mathbf{X}$ orders $\mathbf{Y}$. In [5–7], we showed the details how current query optimizers could be extended with ODs to simplify queries with order by in similar ways to how FDs have been shown to be useful in simplifying queries with group by.

2 Axiomatization and Challenges

A key concern in dependency theory is developing the algorithms for testing logical implication. Developing inference rules (axioms) is an approach to show logical implication between dependencies. We investigate the *axiomatization* for ODs. In [5], we studied UODs. We provided a *sound* and *complete* axiomatization for UODs. The inference rules of the axiomatization are shown in Figure 1. The inference rules remain *sound* over ODs. To prove the axiomatization is sound, we show that each of the axiom is sound, which is simple. Proving completeness is much more involved. To prove the axiomatization is complete for UODs, we demonstrate in [5] that for any set of UODs $\mathcal{M}$, a table $\mathbf{t}$ can be constructed that *satisfies* $\mathcal{M}$ and that every OD that is not derivable over $\mathcal{M}$ with axioms presented in Figure 1 is falsified by table $\mathbf{t}$.

1. *Reflexivity.*
$$\mathbf{XY} \mapsto \mathbf{X}$$
2. *Prefix.*
$$\frac{\mathbf{X} \mapsto \mathbf{Y}}{\mathbf{ZX} \mapsto \mathbf{ZY}}$$
3. *Normalization.*
$$\mathbf{WXYXV} \leftrightarrow \mathbf{WXYV}.$$
4. *Transitivity.*
$$\frac{\mathbf{X} \mapsto \mathbf{Y} \quad \mathbf{Y} \mapsto \mathbf{Z}}{\mathbf{X} \mapsto \mathbf{Z}}$$
5. *Suffix.*
$$\frac{\mathbf{X} \mapsto \mathbf{Y}}{\mathbf{X} \leftrightarrow \mathbf{YX}}$$
6. *Chain.*
$$\frac{\mathbf{X} \sim \mathbf{Y}_1 \quad \forall_{i \in [1,n-1]} \mathbf{Y}_i \sim \mathbf{Y}_{i+1} \quad \mathbf{Y}_n \sim \mathbf{Z} \quad \forall_{i \in [1,n]} \mathbf{Y}_i\mathbf{X} \sim \mathbf{Y}_i\mathbf{Z}}{\mathbf{X} \sim \mathbf{Z}}$$

Fig. 1. Axioms for UODs.

In [5], we additionally demonstrate that Armstrongs axiomatization for functional dependencies (FDs) is subsumed within our axiomatization for ODs. Working with ODs is more involved than with FDs because the order of the attributes matters. Thus, we must work with lists of attributes instead of with sets. This necessarily complicates our axioms compared with Armstrongs axioms for FDs. The axioms provide insight into how ODs behave and patterns

for how ODs logically follow from others that are not easily evident reasoning from first principles. The work about ODs, we feel, opens exciting venues for future work to develop a powerful new family of query optimization techniques in database systems.

There is more that can be done, and that we plan to do. We plan to work on extending our work axiomatization for UODs [5] into an complete axiomatization for ODs, which allow the mix of ascending and descending orders. Such an axiomatization might provide insight into how ODs behave, and provide input for useful query rewrites.

If $\mathsf{ABC} \mapsto \mathsf{D}$ holds but not $\mathsf{AB} \mapsto \mathsf{D}$, is ordering by AB useful if we need a stream sorted by D? If the stream is sorted by AB, it may be *nearly sorted* on D. If it were known that every partition of AB is small, each AB-partition could be sorted on-the-fly in main memory, removing the need for an external sort operator. We would like to formalize the concept of *nearly sorted*.

We are working on introducing a framework for discovering *conditional order dependencies*. (Conditional sequential dependencies were proposed in [2].) A conditional order dependency can be represented as a pair $(\mathbf{X} \mapsto \mathbf{Y}, \mathbf{T}_r)$, where $\mathbf{X} \mapsto \mathbf{Y}$, referred to as the embedded OD, and $\mathbf{T}_r$ is a range pattern tableau defining over which rows the dependency applies. It would provide a novel integrity constraint allowing one to express, that an OD $\mathsf{date} \mapsto \mathsf{salary}$ holds for a given $\mathsf{employee\_id}$.

Furthermore, in the process of merging data from various sources, it is often the case that small variations occur. For example, one movie site might report the movie *Gone with the Wind* as having a running time of 222, while site two reports 238 minutes for it. The FD that *movie* $\rightarrow$ *length* would be violated. In [3], they define a metric over *FDs* to allow for such small variations. Likewise, we would like to define metric ODs to generalize both ODs as in this paper and metric FDs. We would like to devise algorithms for determining whether a given metric OD holds for a given relation, and to investigate the use of these as data cleaning techniques as in [1] for *matching dependencies*.

References

1. L. Bertossi, S. Kolahi, and V. Lakshmanan. Data cleaning and query answering with matching dependencies and matching functions. In *ICDT, 268-279*, 2011.
2. L. Golab, H. Karloff, F. Korn, and D. Srivastava. Sequential Dependencies. *PVLDB, 2(1): 574-585*, 2009.
3. N. Koudas, A. Saha, A. Srivastava, and S. Venkatasubramanian. Metric Functional Dependencies. In *ICDE, 1291-1294*, 2009.
4. J. Szlichta, P. Godfrey, and J. Gryz. Chasing Polarized Order Dependencies. In *AMW,168-179*, 2012.
5. J. Szlichta, P. Godfrey, and J. Gryz. Fundamentals of Order Dependencies. *PVLDB, 5(11): 1220-1231*, 2012.
6. J. Szlichta, P. Godfrey, J. Gryz, W. Ma, P. Pawluk, and C. Zuzarte. Queries on Dates: Fast Yet not Blind. In *EDBT, 497-502*, 2011.
7. J. Szlichta, P. Godfrey, J. Gryz, and C. Zuzarte. Expressiveness and Complexity of Order Dependencies. *PVLDB*, 2013.