Minimization and Group-By Detection for Nested XQueries

Alin Deutsch Yannis Papakonstantinou Yu Xu
University of California, San Diego
E-mail: {deutsch,yxu,yannis}@cs.ucsd.edu

Abstract

We extend tree pattern queries into Group-by Normal Form Tree Pattern (GNFTP) *queries, which are nested, perform arbitrary joins, and freely mix bag and set semantics. In [1], we describe a subset of XQuery, called* OptXQuery *and provide a normalization algorithm that rewrites any OptXQuery into a GNFTP query. Key logical query optimizations can be solved for GNFTP/OptXQuery. As a proof-of-concept but also for its own importance and value in query optimization, we developed and evaluated a query* minimization *algorithm for GNFTP. The rich features of GNFTP/OptXQuery create key challenges that fundamentally extend the prior work on the problems of minimizing conjunctive queries. An important application of this technique is group-by detection. [1] extends GNFTP into extGNFTP to capture XQueries outside the OptXQuery set. The extGNFTP notation provides the logical plan optimization framework of our XQuery processor.*

1 Introduction

We illustrate a very common case in which redundant navigation across nesting levels cannot be avoided and an efficient implementation requires minimization.

Consider the following query that groups book titles by authors (it is a minor variation of query Q9 from W3C's XMP use case). The **distinct-values( )**function eliminates duplicates, comparing elements by value-based equality.

```
let $doc := document("input.xml")
for $a in distinct-values($doc//book/author)
return ⟨result⟩ { $a,                                        (X1)
                  for $b in $doc//book
                  where some $ba in $b/author satisfies $ba = $a
                  return $b/title }
⟨/result⟩
```

Notice that the **for** loop binding $\$a$ (from now on called the $\$a$ loop) has *set* semantics, all others have *bag* semantics.

The straightforward nested-loop execution of this query is wasteful since the nested loops (the $\$b$ **for** loop and the $\$ba$ **some** loop) are redundant: the $\$a$ loop has already navigated to the corresponding book and author elements. In this case, we say that the redundant navigation appears *across* nested subqueries, where nesting is w.r.t. the **return** clause. Clearly there is a more efficient implementation (inspired by a well-known OQL optimization): eliminate the redundant navigation by scanning books and authors just once and then apply a group-by operation. This plan is expressed by the following GNFTP query. See the companion paper for a visually intuitive GNFTP tree notation. The $\$doc$ variable is defined as above.

```
for $b in $doc//book, $a in $b/author                              (X2)
groupby $a into $L return
⟨result⟩ { $a, for $b′in $L/tuple/b groupby [$b′] return $b′/title }
⟨/result⟩
```

Each group corresponding to a binding of the group-by variable $\$a$ is a list $\$L$ of tuples of bindings for all non-group-by variables of the enclosing **for** (only $\$b$ in this case). To avoid introducing a new data type into the XML data model, we encode tuples as XML elements tagged `tuple`. The binding of variable $\$b$ is the content of a subelement tagged `b`. The XPath expression $\$L/tuple/b$ simulates the projection on the component corresponding to the binding of $\$b$, thus eliminating the need for a fresh navigation to books on the input document. Notice that the **groupby** version is the more efficient one regardless of whether the execution model is based on nested-loop navigation (fewer nested loops) or on set-oriented techniques such as joins of tuples of variable bindings (fewer joins).

The minimization technique we propose in [1] rewrites query (X1) to (X2). In general, our technique detects and eliminates redundant navigation *within* and *across* nested subqueries. As a side-effect, it unifies and generalizes prior solutions for tree pattern minimization and group-by detection. We show in [1] that there is a unique minimal form for any GNFTP query, and that our algorithm is guaranteed to find it in optimal time. The experimental evaluation in [1] shows that the benefit of minimization is significant.

Companion paper for this poster

[1] A. Deutsch and Y. Papakonstantinou and Y. Xu: Minimization and Group-by Detection for Nested XQueries. Available from `http://www.db.ucsd.edu/people/alin`.

Proceedings of the 20th International Conference on Data Engineering (ICDE'04)
1063-6382/04 $ 20.00 © 2004 IEEE