Technical Perspective: Optimizing Tree Patterns for Querying Graph- and Tree-Structured Data

Benny Kimelfeld
Technion – Israel Institute of Technology
bennyk@cs.technion.ac.il

From the early days of databases, practitioners and researchers have pursued techniques for rewriting queries into equivalent ones that are easier to evaluate. The following paper closes a fundamental gap that we have had in our understanding of this challenge in the context of *tree patterns*. Such patterns are common and basic components of query languages for graph and tree data such as SPARQL, Cypher and XQuery. The authors study the question of whether the given tree pattern can be replaced with a smaller one, the question of whether it involves redundant conditions, and most importantly, the relationship between these two questions.

Formally, a tree pattern p is *matched* in a labeled graph G if the nodes of p can be mapped to the nodes of G in a way that all the *constraints* of p are satisfied. A *node constraint* is either a label match (e.g., the label is "person") or *wildcard* (no constraint), and an *edge constraint* is either *child* (the edge is mapped to an edge) or *descendant* (the edge is mapped to a path). Two patterns are *equivalent* if one is matched in a given graph precisely when the other does. The properties in focus are *minimality*—does p have the minimal size among all equivalent patterns? and *redundancy*—does the removal of any node of p (along with the subtree underneath) result in an equivalent pattern?

Tree-pattern minimization was studied by Flesca et al. [2] in 2003, where it was claimed that minimization can be achieved through containment tests among sub-patterns. Moreover, their results imply that determining whether a tree pattern is minimal is an NP-complete problem. It was not before 2008 that Kimelfeld and Sagiv [4] established that Flesca et al. [2] had a gap in their arguments, as their results apply to *nonredundancy* and not *minimality*, and in fact, the case of minimality was still open.

Nevertheless, Kimelfeld and Sagiv believed that the results of Flesca et al. [2] were valid, and formulated a conjecture that the following paper refers to as the *M-NR conjecture*: minimality (M) and nonredundancy (NR) are the same, that is, a tree pattern is minimal if and only if it is nonredundant [4]. The conjecture was supported by various classes of tree patterns where it was proved to hold true [3,4]. Yet, correctness of the conjecture in general remained open. Verifying the conjecture required showing that every nonredundant pattern is minimal (as the other direction is clearly true). Refuting the conjecture required finding a single counterexample. To our surprise, one was indeed found.

The following paper highlights a publication in 2016 ACM PODS, where Czerwinski et al. [1] refuted the M-NR conjecture. Notwithstanding the time it took to find it, their counterexample is fairly simple and small (enough to fit in T-shirts worn by the authors of [1] during the conference); it is a nonredundant tree pattern of 32 nodes, and they show an equivalent one with only 31 nodes. As they further show, not only are the two properties different, the corresponding computational decision problems are fundamentally different (under conventional complexity assumptions), again in contrast to past beliefs [3, 4]: while nonredundancy is NP-complete, minimality is Π_2^p-complete (hence, resides higher in the polynomial hierarchy). In particular, it follows that one needs tools beyond containment tests if minimization of general tree patterns is desired. That and more in what follows.

1. REFERENCES

[1] W. Czerwinski, W. Martens, M. Niewerth, and P. Parys. Minimization of tree pattern queries. In *PODS*, pages 43–54. ACM, 2016.

[2] S. Flesca, F. Furfaro, and E. Masciari. On the minimization of XPath queries. In *VLDB*, pages 153–164, 2003.

[3] S. Flesca, F. Furfaro, and E. Masciari. On the minimization of XPath queries. *J. ACM*, 55(1):2:1–2:46, 2008.

[4] B. Kimelfeld and Y. Sagiv. Revisiting redundancy and minimization in an XPath fragment. In *EDBT*, pages 61–72. ACM, 2008.