氏　　名　TRAN Van Dang

学位(専攻分野)　博士(情報学)

学 位 記 番 号　総研大甲第 2362 号

学位授与の日付　2022 年 9 月 28 日

学位授与の要件　複合科学研究科　情報学専攻
学位規則第6条第1項該当

学 位 論 文 題 目　Programmable View Update Strategies in Relational Databases

論 文 審 査 委 員　主　査　蓮尾 一郎
情報学専攻　教授
加藤 弘之
情報学専攻　助教
対馬 かなえ
情報学専攻　助教
関山 太朗
情報学専攻　助教
胡 振江
北京大学　計算機学院　教授

(Form 3)

Summary of Doctoral Thesis

Name in full: TRAN Van Dang

Title: Programmable View Update Strategies in Relational Databases

Since Codd's seminal work was introduced in 1970, data management has become a leading research area of computer science. In practice, relational database management systems (RDBMS) such as Oracle Database, MySQL, PostgreSQL, and so forth are widely used. In relational databases, data are represented as collections of tables that allow applications to not only query but also update the source data. Due to the separation of applications and backend databases, database administrators usually define logical tables, called views, over other source tables to expose only relevant data computed by these views and hide other confidential data. View update is an important mechanism that allows updates on a view by translating them into the corresponding updates on the source tables. The existing literature has shown the ambiguity of view updates that for a given update on the view, there are potentially many strategies to update the source tables. Due to this ambiguity, it is challenging to automatically determine view update strategies. Many practical database management systems such as PostgreSQL, Oracle, and so forth can automatically deal with very simple updatable views. For complex views, the database administrators need to take the responsibility to define their own update strategies. However, manually writing view update strategies is non-trivial and error-prone.

To address the aforementioned view update problem, we propose an effective language-based approach for making view update strategies programmable. Specifically, our approach allows using the Datalog language, a well-known data query language, to completely write view update strategies. This is in sharp contrast to previous approaches in which the view defining queries are enriched to capture some certain update intentions. To reduce the user's burden in programming view update strategies, we propose algorithms to validate, optimize, debug, and compile the user-written programs to run correctly in off-the-shelf relational database management systems.

First, we design a fragment of non-recursive Datalog for specifying view update strategies. This fragment not only has good properties in theory but is also useful for solving view updates in practical relational database management systems where views are commonly defined in SQL without recursion. Since writing view update strategies is error-prone, we propose a validation algorithm that statically checks the well-behavedness of user-written programs and automatically derives from view update strategies the corresponding view definition to confirm the one expected beforehand. To improve the performance of view update strategy programs, we introduce a new

optimization method by incrementalizing the hand-written programs. We theoretically prove the soundness and completeness of our approach and practically validate the efficiency of the framework implementation by experiments on a benchmark collected from real-world applications. The experiments show that our validation algorithm is feasible for solving many view update strategies and our incrementalization can significantly reduce the running time of view updates in practical relational database management systems.

To enhance the ease of use of Datalog in programming view update strategies, especially for the case that the user-written programs are not valid, we propose an efficient approach to interactively debugging Datalog programs so that the user's burden is reduced. Specifically, we provide a syntax for users to specify properties of non-recursive Datalog programs. We present a counterexample generator that verifies specified properties and generates counterexamples to show unexpected behaviors of user-written programs. We design a debugging engine combined with a dialog-based user interface to assist users in locating bugs in the programs with the generated counterexamples. We have implemented a prototype of our approach and demonstrated its feasibility and efficiency.

We further extend our approach to allow using Datalog for programming view update strategies on knowledge graphs in the RDF (Resource Description Framework) format. Recursion is the key that gives Datalog the capabilities to exploit the graph structure of RDF data. We formulate a view update strategy as the combination of two parts: a recursive part, which implements recursive patterns, and a non-recursive one, which consists of inner update strategies. On the one hand, the recursive Datalog rules of the first part are pre-defined and pre-validated so that their well-behavedness is guaranteed. On the other hand, we allow programmers to manually write the inner update strategies in non-recursive Datalog, which are automatically validated. To guarantee the ease of use of pre-defined recursive programs in constructing a new one, we extend Datalog with a restricted form of higher-order predicate syntax. To improve the performance of the Datalog programs, we propose an algorithm to transform all higher-order predicates into equivalent first-order predicates that can be evaluated efficiently. We show the expressiveness of our proposed approach by implementing several classes of view update strategies for some common recursive patterns of RDF graphs.

We have implemented a framework for our proposed methods. To integrate our framework with a relational database management system such as PostgreSQL, we design a compilation algorithm to transform Datalog-written view update strategies into procedural SQL code. The SQL program consists of all necessary statements for creating an updatable view in the database. The updatable view uses trigger mechanisms to automatically invoke the view update strategy in response to a single request or transaction of view updates. Therefore, programmers can run Datalog programs in a PostgreSQL database via our provided command-line tool or web-based user interface.

Form8 : Separate Sheet
（様式８・別紙１）

Results of the doctoral thesis screening

博士論文審査結果

Name in Full
氏 名　　TRAN Van Dang

Title
論文題目　　Programmable View Update Strategies in Relational Databases

本博士論文は、ビュー更新（View-updating）という挑戦的な問題に対して、関係データベース上のビュー更新戦略を自由に記述できる言語の設計と実現に関するものである。ビュー更新問題が双方向変換と深く関係していることは Foster らの POPL 2005 で示されている。これまで、多くの双方向変換を記述する領域特化言語が提案されてきたが、表現力の低さと利用の難しさは課題として残されている。これに対して、本研究は Datalog という既存言語を用いてビュー更新戦略の新しい記述手法を提案することにより、データベース分野の研究者が容易に利用できるようになった。また、記述の正当性を自動的に判断するアルゴリズムを与えるとともに、実用的なシステム BIRDS を実装した。

本論文は、英語で記述されており、全 6 章から構成されている。

第1章は序論である。研究の背景、先行研究の課題, 研究目的, 主要な貢献など、論文全体の構成を述べている。

第2章は基礎知識の紹介である。関係データベースとその問い合わせ言語である Datalog の基礎知識、ビュー更新問題と双方向変換の基礎知識、および関連研究について議論している。

第3章では、ビュー更新戦略を簡潔に記述できる非再帰的なDatalogのクラスを定義し、ビュー定義の導出およびビュー更新のデータベースに対する伝播の正当性を判断する，健全かつ完全なアルゴリズムを提案するとともに、ビュー更新の伝播を漸進的に実行できる最適化手法を示している。

第4章では、第3章で提案している更新戦略を記述するプログラムをデバッグする手法を与えている。具体的には、プログラムの予期しない動作を明らかにする反例を生成するとともに、バグのある場所をインタラクティブに見つける手法を提案している。

第5章では、第3章の結果を利用して、再帰的な更新戦略を記述できるように、ビュー更新の正当性を判断する手法を議論している。

第6章は論文のまとめと今後の課題である。

審査会において、出願者はビュー更新戦略を記述する言語の設計、実現及び評価について、45 分のスライド発表と 30 分の質疑応答によって行われた。スライドの発表では、既存手法の問題点と新しい解決手法を要領よく説明された。そのあと審査委員との質疑応答を行い、的確な回答がなされた。

以上のように、本論文では、データベース分野で、ビュー更新という未解決問題に対して、表現力の高さと利用しやすさを兼ね備えた更新戦略言語とその実現手法を与え、ビュー更新機能をより効率的に運用できるようになっており、理論だけでなく実践的な観点からも本研究の貢献が大きいものと認められる。なお、本研究の成果はデータベース分野のトップカンファレンス VLDB 2020 の研究論文をはじめ、5 件の査読つき国際会議論文を発表した。また、開発した BIRD システムもウェブで公開し自由にダウンロードできるようになっており、国内外の大学の研究者に利用されている。以上の理由により、審査委員会は、本論文が学位の授与に値すると判断した。