Generating Database Schema Test Suites with DOMINO

Abdullah Alsharif
University of Sheffield

Gregory M. Kapfhammer
Allegheny College

Phil McMinn
University of Sheffield

I. INTRODUCTION

Industrial practitioners advocate the testing of relational database schemas because, for instance, omitting the definition of an integrity constraint (i.e., a `PRIMARY KEY` or a `UNIQUE`) in a schema can compromise correctness and increase maintenance costs. For example, forgetting to mark each `username` as `UNIQUE` could lead to incorrect data duplication within a database. Also, different database management systems (DBMSs) often interpret integrity constraints differently (e.g., a `PRIMARY KEY` can accept a `NULL` value once with SQLite but not with other DBMSs) [1].

Systematic schema testing uses integrity constraint coverage criteria [2] that require the creation of an `INSERT` statement to exercise each integrity constraints as true or false (i.e., accepted or rejected by the DBMS). Automatically generating test data has been implemented using Random$^+$ and the Alternating Variable Method (AVM) [2]. However, these techniques for generating test data may be suboptimal, thus requiring improvements to ensure that the generated tests are good at finding faults [3]. In this paper and the accompanying demonstration we introduce the use and benefits of DOMINO (DOMain-specific approach to INtegrity cOnstraint test data generation), a new technique that automatically generates a test suite for a database schema.

II. USING DOMINO FOR AUTOMATED TEST GENERATION

DOMINO uses domain-specific operators to, in comparison to Random$^+$ and two variants of the AVM, rapidly generate test data for schemas [2]. To achieve this performance improvement without compromising the effectiveness of the generated tests, it replicates values generated for an `INSERT` statement (i.e., copying `PRIMARY KEY` values to the related `FOREIGN KEY`), flipping values to `NULL`, and randomizing data where needed. It is integrated into the well-documented and easy-to-use *SchemaAnalyst* tool.

For example, if a test case requires the exercising of a `PRIMARY KEY`, DOMINO will first generate multiple `INSERT`s to prepare the database. To violate the `PRIMARY KEY`, it will *replicate* an existing `PRIMARY KEY` to ensure rejection or it will *randomize* a new value for the `PRIMARY KEY` to guarantee uniqueness and acceptance. Therefore, the benefits of using DOMINO for test generation are:

1) It generates random data rather than using default values, which is better for finding relational schema faults [3].
2) Test suite generation with DOMINO is faster than both of the state-of-the-art approaches based on the AVM. For example, DOMINO is approximately 40 seconds faster than AVM-Random and 10 seconds faster than AVM-Defaults [3].
3) DOMINO's test coverage scores are either equal to or higher than those of tests created by either AVM variant [3].

To use DOMINO, a tester should specify "`dominoRandom`" as the requested test data generator when running either a mutation analysis or test data generation technique provided by *SchemaAnalyst*. To install and use *SchemaAnalyst*, individuals can follow the detailed documentation at the tool's GitHub repository[1].

[1]https://github.com/schemaanalyst/schemaanalyst

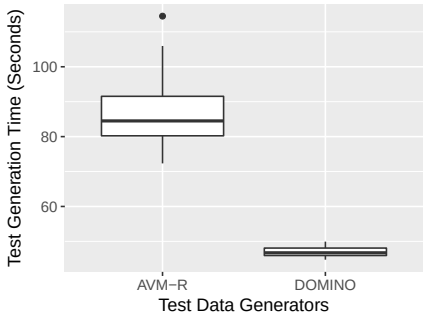

Fig. 1. Test data generation time for the *iTrust* schema.

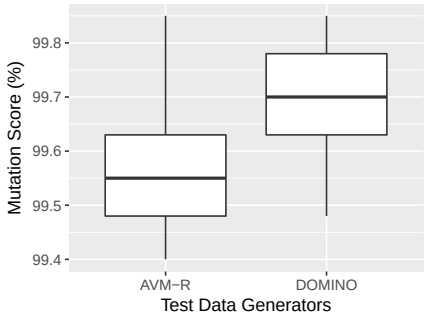

Fig. 2. Mutation score of generated tests for the *iTrust* schema.

III. HIGHLIGHTS OF PRIOR EXPERIMENTAL RESULTS

This section focuses on the illustrative empirical results for the *iTrust* relational schema because it is one of the largest and most complex schemas to which we have applied DOMINO. Figure 1 shows that DOMINO is much faster at generating tests than AVM-R—an AVM that uses random data generation rather than default values—taking nearly half the time. Note that, since both data generators are stochastic, this graph presents results from 30 trials when the PostgreSQL DBMS hosts the schema.

Figure 2 shows the capability of DOMINO at finding synthetic schema faults called mutants, with the same schema and DBMS as in Figure 1. The results show that DOMINO kills more mutants than AVM-R, indicating it is better than this previously implemented method—in this case and also in others. Due to space constraints, refer to our full paper for more results [3].

In conclusion, this paper introduced DOMINO, showing that it is superior to prior methods. It also recommended the use of domain-specific operators that can be beneficial compared to search-based techniques. Overall, this demonstration will help researchers and practitioners learn how to use *SchemaAnalyst* and observe the benefits of using DOMINO to generate schema tests.

REFERENCES

[1] G. M. Kapfhammer, P. McMinn, and C. J. Wright, "Search-based testing of relational schema integrity constraints across multiple database management systems," in *Proc. of ICST*, 2013.

[2] P. McMinn, C. J. Wright, and G. M. Kapfhammer, "The effectiveness of test coverage criteria for relational database schema integrity constraints," *TOSEM*, vol. 25, no. 1, 2015.

[3] A. Alsharif, G. M. Kapfhammer, and P. McMinn, "DOMINO: Fast and effective test data generation for relational database schemas," in *Proc. of ICST*, 2018.