Finding Logic Bugs in Spatial Database Engines via Affine Equivalent Inputs

Wenjing Deng, East China Normal University, China; Qiuyang Mang, The Chinese University of Hong Kong, Shenzhen, China
Chengyu Zhang, ETH Zurich, Switzerland; Manuel Rigger, National University of Singapore, Singapore

Problem

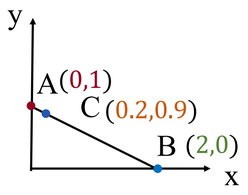
What is SDBMS?

The tool aims to store, manipulate, and retrieve spatial data.

Logic Bugs in SDBMSs: Silent but Dangerous

SDBMSs (e.g., PostGIS, MySQL) **compute wrong spatial results** (e.g., "Does this line cover this point?").
Bugs **do not crash the system**—they silently corrupt data, making them hard to detect.

A Real Bug in PostGIS (#968)




```
1 CREATE TABLE t1 (g geometry);
2 CREATE TABLE t2 (g geometry);
3 INSERT INTO t1 (g) VALUES ('LINESTRING(0 1,2 0)');
4 INSERT INTO t2 (g) VALUES ('POINT(0.2 0.9)');
5 SELECT COUNT(*) FROM t1 JOIN t2 ON ST_Covers(t1.g,t2.g);
-- {0} 🐞 {1} ✔
```

Fig 1. AB covers C. The retrieved value from PostGIS should be 1 instead of 0.

Why Detecting Logic Bugs Automatically is Challenging?

The lack of **ground truth results**.

Current Methodologies Are Inadequate

📌 **Differential testing:** Generate the query, pass it to different systems, and consider the equivalence of their outputs as the expected result
- Fails for features unique to one SDBMS
- Misses bugs in the shared third-party libraries
- False alarms caused by intentional implementation variations among developers

📌 **Ternary Logic Partitioning (TLP):** *Partition the original query into three subqueries, where the union of their results equals the original.*
- Fail to detect logic bugs in spatial-related features (*e.g.,* #968 can not be detected)

Approach

AEI: We propose Affine Equivalent Inputs to provides the expected results for SDBMSs.

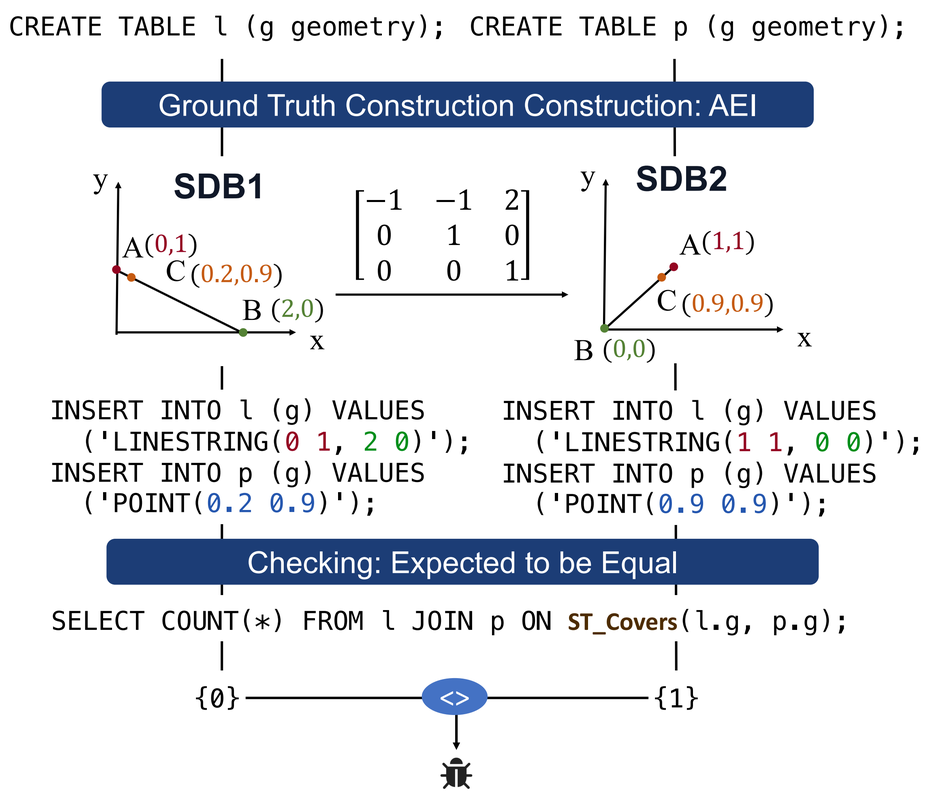
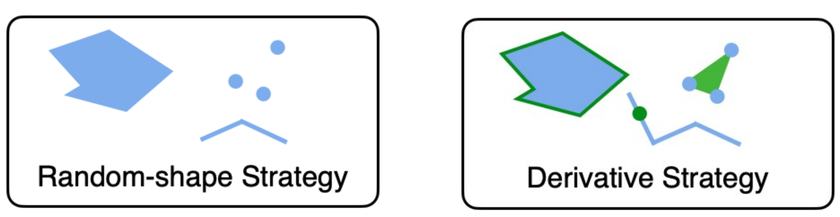


Fig 2. Affine Equivalent Inputs (AEI)

Key Insight: If two geometries affine transform (e.g., rotate, scale, and translate) in the same way, topological relationships (e.g., intersects, covers, or disjoint) are preserved.

How to generate Spatial DB to to exercise the SDBMSs?

One naive method is to randomly generate syntactically valid geometries --- our ***random-shape strategy***.
However, the random-shape strategy makes it unlikely to observe a variety of topological relationships, making it difficult to exercise the SDBMSs.
To improve the efficiency of bug finding, we propose the derivative strategy that derives existing geometries by applying spatial functions --- our ***derivative strategy***.



Tool

Spatter (Spatial DBMSs Tester): An automated testing tool that combines AEI with a geometry-aware generator specific for SDBMSs.

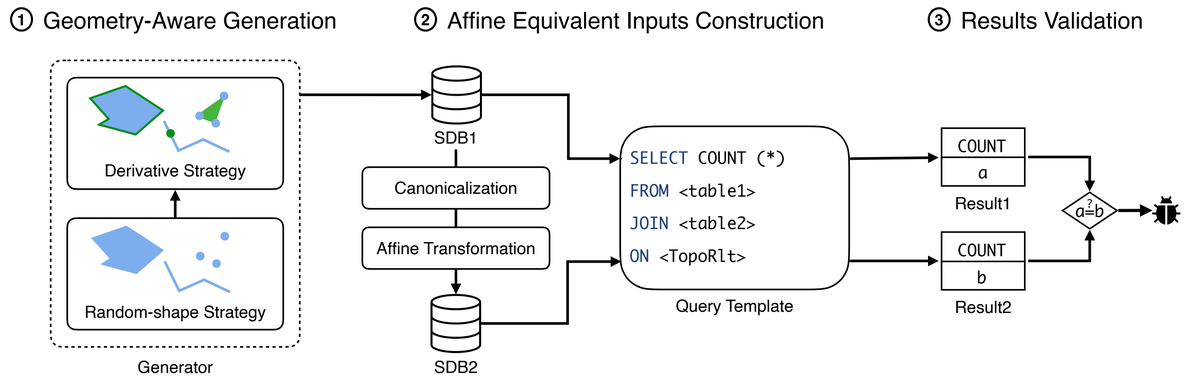


Fig 3. Overview of Spatter

Evaluation

🛠️ **New Bugs**

We consider 34 of them as previously unknown, unique bugs, 30 of which have been confirmed or fixed by the developers.

Table 1. Status of the reported bugs in SDBMSs.

SDBMS	Fixed	Confirmed	Unconfirmed	Duplicate	Sum
GEOS	4	8	0	0	12
PostGIS	8	1	1	1	11
DuckDB Spatial	5	0	1	0	6
MySQL	1	3	0	0	4
SQL Server	0	0	2	0	2
Sum	18	12	4	1	35

Table 2. A Classification of the Confirmed Bugs.

	Logic Bugs		Crash Bugs		
SDBMS	Fixed	Confirmed	Fixed	Confirmed	Sum
GEOS	1	8	3	0	12
PostGIS	6	1	2	0	9
MySQL	1	3	0	0	4
DuckDB Spatial	0	0	5	0	5
Sum	8	12	10	0	30

Note: GEOS is a third-party library used by PostGIS and DuckDB Spatial.

🛠️ **Comparison to the State of the Art**

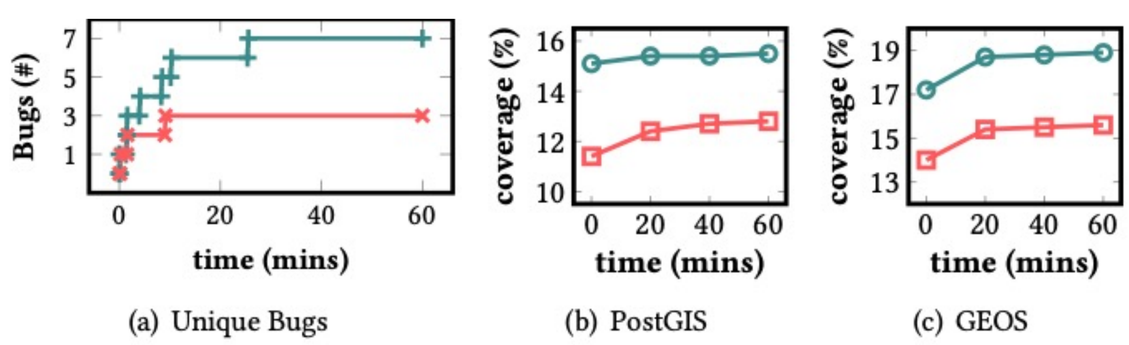
📌 4 logic bugs could be detected by comparing PostGIS and MySQL; however, such differential testing suffers from false alarms.
📌 All logic bugs were missed when comparing PostGIS and DuckDB.
📌 Two index-related bugs could be found. However, applying the *Index* method heavily depends on the test case design.
📌 TLP detected one index-related bug, since the lack awareness of spatial relationships.

Table 3. Logic bugs detection comparison.

	Differential Testing			-	
SDBMS	*AEI*	PostGIS vs. MySQL	PostGIS vs. DuckDB	*Index*	*TLP*
GEOS	9	3	0	0	0
PostGIS	7	0	0	1	1
MySQL	4	1	0	1	0
Sum	20	4	0	2	1

🛠️ **Efficiency of the Geometry-Aware Generator**

Self-constructed baseline: generator based solely on the random-shape strategy
Target SDBMS: PostGIS
📌 Triggering cases:
Geometry-Aware Generator +○ : 9,913; Random-shape Generator ×□ : 2,366
📌 As (a), after deduplication, the geometry-aware generator with our proposed strategy, ***the derivative strategy***, significantly outperformed the baseline.
📌 As (b)(c), the geometry-aware generator achieved higher coverage, given that the derivative strategy exploits spatial functions inherent in PostGIS.



(a) Unique Bugs (b) PostGIS (c) GEOS

Fig. 4. Ablation study of the Geometry-Aware Generator

Conclusion

🛠️ **Key Contributions**

Geometry-Aware SQL Generator --- Produces high-quality spatial queries
Affine Equivalent Inputs (AEI) --- Novel validation method to detect incorrect results
Spatter Tool --- Automated testing framework for SDBMSs

🔬 **Experimental results**

📌 34 unique bugs found, 30 confirmed by developers, and 18 already fixed.
📌 AEI can identify 14 logic bugs that were overlooked by previous approaches.
📌 The geometry-aware generator significantly outperforms the random-shape generator in detecting unique bugs.