DIADS: Addressing the “My-Problem-or-Yours” Syndrome with Integrated SAN and Database Diagnosis

Nedyalko Borisov
Duke University

Shivnath Babu, Duke
Sandeep Uttamchandani, IBM
Ramani Routray, IBM
Aameek Singh, IBM
Current State

➢ Databases (DBMSs) and SANs have separate admin teams
  ➢ Each team has limited visibility into full system

➢ Database admin (DBA) opens problem ticket
➢ SAN admin responds
➢ To and fro may continue

- Business Intelligence (BI) Queries
- 30% slowdown compared to 2 weeks ago
- 40% IO increase, but response time is within normal bounds
What is the Natural Solution?

➢ Separate admin teams do not have holistic view of query execution

➢ Easy if we have low-level tracing
  ➢ May be infeasible
  ➢ May have high overhead
Our Solution: DIADS

➢ DBMS level and SAN level monitoring tools - e.g., Hyperic HQ, TPC
➢ Need to integrate these separate pieces of data to create a holistic view of query execution
➢ **DIADS: DIAgnosis for Databases and SANs**
  ➢ Inputs
    ▶ Poorly performing query
    ▶ Monitoring data from DBMS
    ▶ Monitoring data from SAN
Our Solution: DIAADS

➢ DBMS level and SAN level monitoring tools - e.g., Hyperic HQ, TPC
➢ Need to integrate these separate pieces of data to create a holistic view of query execution
➢ **DIAADS**: DIAgnosis for Databases and SANs
  ➢ Outputs
    ▶ Root cause of query's poor performance (ideal)
    ▶ Localization of problem
<table>
<thead>
<tr>
<th>Feature</th>
<th>Novelty</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Annotated Plan Graph (APG) across DBMS and SAN</td>
<td>- Holistic view of query execution</td>
</tr>
<tr>
<td>- Diagnosis workflow</td>
<td>- Generated from commonly-available monitoring data</td>
</tr>
<tr>
<td></td>
<td>- Careful combination of machine-learning (ML) techniques and expert knowledge (EK)</td>
</tr>
<tr>
<td></td>
<td>- Deals with flood of monitoring data (ML)</td>
</tr>
<tr>
<td></td>
<td>- Deals with noisy monitoring data in real systems (ML + EK)</td>
</tr>
<tr>
<td></td>
<td>- Deals with fault propagation (EK)</td>
</tr>
<tr>
<td></td>
<td>- Incorporates checks and balances</td>
</tr>
</tbody>
</table>
Roadmap

➢ Motivation
➢ Running Example
➢ Workflow
➢ Evaluation
➢ Conclusions & Future work
Running Example

➢ Report-generation query (TPC-H Query 2) is running periodically

```sql
SELECT s_acctbal, s_name, n_name, p_partkey, p_mfgr, 
s_address, s_phone, s_comment 
FROM part, supplier, partsupp, nation, region 
WHERE p_partkey = ps_partkey 
  AND s_suppkey = ps_suppkey AND p_size = 28 
  AND p_type like '%COPPER' AND s_nationkey = n_nationkey 
  AND n_regionkey = r_regionkey AND r_name = 'AMERICA'
  AND ps_supplycost = ( 
    SELECT min(ps_supplycost) 
    FROM partsupp, supplier, nation, region 
    WHERE p_partkey = ps_partkey 
      AND s_suppkey = ps_suppkey 
      AND s_nationkey = n_nationkey 
      AND n_regionkey = r_regionkey AND r_name = 'AMERICA' )
ORDER BY s_acctbal desc, n_name, s_name, p_partkey;
```
SAN (Mis)configuration Issue

- Server
- HBA
- Switch Sw1
- Switch Sw2
- Switch Sw3
- Storage Subsystem
- Pool P1
- Pool P2
- Volume V1
- Volume V2
- Volume V3
- Disks
- Disks
- Workload
- Table: Partsupp
- Tables: Region, Nation, Supplier, Part
Observations

15.2 minutes
15.1 minutes
14.9 minutes
15.2 minutes
33.1 minutes
31.3 minutes

Diagnose the cause for the slowdown
Query Plan Execution

1. Hash
2. Merge Join
3. Sort
4. Materialize
5. Hash
6. Sort
7. Hash Join
8. Partsupp
9. Hash
10. Hash Join
11. Supplier
12. Hash
13. Hash Join
14. Nation
15. Hash
16. Region
17. Aggregate
18. Nested Loop
19. Region
20. Hash Join
21. Nested Loop
22. Partsupp
23. Supplier
24. Hash
25. Supplier

DBMS SAN
Running Example of APG

DBMS
SAN

Storage Subsystem

Server

Switch Sw1

Switch Sw2

Switch Sw3

Pool P1

Pool P2

Volume V1

Volume V2

Disks

Disks

Part

Region

Supplier

Supplier

Nation

Partsupp

O8

O22

O4

O19

O25

O23

O11

O16

Volume V1

Volume V2
DBMS: Tables
  - Tables
  - Tablespaces

SAN: File System
  - File System
  - Volumes
  - Disks & Pools
  - Storage Subsystem
  - Ports
  - FC Switches
  - HBA
  - Server
➢ Monitoring data
  ➢ DBMS
    ➢ Plan-level data (e.g., running time of operator, # of records)
    ➢ DBMS-level data (e.g., hits in the buffer pool, event logs)
APG Annotations

- Monitoring data
  - SAN
    - Component-level data (e.g., for volumes - #reads, #writes, latency, bytes transferred)
    - Event logs
Workflow

Admin identifies run instances when query Q ran fine and when it did not

Correlate with change in operator costs
Module Correlated Operators

➢ Which operators have a change in running time that explains change in running time of the entire plan?

➢ Anomaly Score computed with Kernel Density Estimation (KDE)

<table>
<thead>
<tr>
<th>O16</th>
<th>O14</th>
<th>O11</th>
<th>O8</th>
<th>O4</th>
<th>O25</th>
<th>O23</th>
<th>O22</th>
<th>O19</th>
<th>Plan</th>
</tr>
</thead>
<tbody>
<tr>
<td>APG #1</td>
<td>1</td>
<td>2</td>
<td>43</td>
<td>377</td>
<td>277</td>
<td>1</td>
<td>44</td>
<td>24</td>
<td>1</td>
</tr>
<tr>
<td>APG #2</td>
<td>1</td>
<td>1</td>
<td>44</td>
<td>382</td>
<td>281</td>
<td>1</td>
<td>39</td>
<td>22</td>
<td>2</td>
</tr>
<tr>
<td>APG #3</td>
<td>2</td>
<td>2</td>
<td>43</td>
<td>380</td>
<td>272</td>
<td>1</td>
<td>38</td>
<td>26</td>
<td>1</td>
</tr>
<tr>
<td>APG #4</td>
<td>2</td>
<td>1</td>
<td>43</td>
<td>628</td>
<td>401</td>
<td>1</td>
<td>51</td>
<td>45</td>
<td>1</td>
</tr>
<tr>
<td>APG #5</td>
<td>1</td>
<td>1</td>
<td>45</td>
<td>596</td>
<td>390</td>
<td>1</td>
<td>40</td>
<td>51</td>
<td>2</td>
</tr>
</tbody>
</table>

Running times (seconds)

<table>
<thead>
<tr>
<th>Anomaly Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>O8</td>
</tr>
<tr>
<td>O4</td>
</tr>
<tr>
<td>O22</td>
</tr>
</tbody>
</table>

KDE picture borrowed from Internet
Admin identifies run instances when query Q ran fine and when it did not

Correlate with change in operator costs

Dependency path analysis

Correlate with change in data flow
Module Dependency Analysis

- Correlation analysis of annotations in each dependency path
- Uses KDE

<table>
<thead>
<tr>
<th></th>
<th>Anomaly Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1, writeIO</td>
<td>0.894</td>
</tr>
<tr>
<td>V1, writeTime</td>
<td>0.823</td>
</tr>
<tr>
<td>V2, writeIO</td>
<td>0.063</td>
</tr>
<tr>
<td>V2, writeTime</td>
<td>0.479</td>
</tr>
</tbody>
</table>
Workflow

Admin identifies run instances when query Q ran fine and when it did not

Correlate with change in operator costs

Dependency path analysis

Correlate with change in data flow

Lookup symptoms database
Module Symptom Database

➢ Mapping from symptoms to root causes
   ▶ Handling event (fault) propagation

➢ Machine learning is not enough. Need to incorporate expert knowledge about DBMS and SAN systems

➢ Many implementation choices
   ▶ Codebook (ex: EMC)
   ▶ Rules (ex: Oracle)
   ▶ Bayesian networks
Our Impl. of Symptom Database

Challenges

➢ How are symptoms expressed?

➢ How is database populated and maintained?

➢ How to prevent database bloat?

➢ What about missing/extra symptoms due to noise?

Our Solution

➢ Language for expressing complex symptoms
  ➢ Intuitive built-in patterns
  ➢ Temporal patterns

➢ Currently, by administrators; Working on partial automation

➢ Parameterized symptoms and root causes

➢ Support for partial matching with confidence score
Workflow

- APGs for Q
- Correlate with change in operator costs
- Dependency path analysis
- Correlate with change in data flow
- Lookup symptoms database
- Impact analysis

Admin identifies run instances when query Q ran fine and when it did not.

Plans

Operators

Components

Data

Symptoms

Root cause
Module Impact Analysis

- What fraction of the slowdown does this root cause explain?
  - Impact score (0-100%)

- Uses
  - Separating high-impact causes from others
  - Safeguard against false positives
  - Identifying presence of false negatives

- Suite of techniques to compute impact score
  - Reverse dependency analysis: Bottom-up traversal of the correlated dependency paths
  - Use of models (DBMS cost models, SAN device models)
Reverse Dependency Analysis

➢ SAN misconfiguration cause – High Impact score
Roadmap

- Motivation
- Running Example
- Workflow
- Evaluation
- Conclusions & Future work
Evaluation Methodology

➢ Testbed
  ▶ TPC-H Queries
  ▶ PostgreSQL
  ▶ IBM DS6000 storage manager
  ▶ On production system

DBMS

Affects only DBMS

DIADS:
Concurrent problems
Fault propagation
Spurious symptoms

Affects only SAN

SAN
Recap of Running Example (Scenario 1)

- **Problem**
  - SAN misconfiguration
- **Correlated Operators**
  - O4, O8, O22
- **Anomaly Scores**

<table>
<thead>
<tr>
<th>Operator</th>
<th>Anomaly Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>O8</td>
<td>1.0</td>
</tr>
<tr>
<td>O4</td>
<td>0.965</td>
</tr>
<tr>
<td>O22</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Diagram:
- Server
- HBA
- Switch Sw1
- Switch Sw2
- Switch Sw3
- Storage Subsystem
- Pool P1
- Volume V1
- Pool P2
- Volume V2
- Partsupp O8
- Partsupp O22
Recap of Running Example (Scenario 1)

- Dependency Analysis
- Anomaly Scores

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Anomaly Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1, writeIO</td>
<td>0.894</td>
</tr>
<tr>
<td>V1, writeTime</td>
<td>0.823</td>
</tr>
</tbody>
</table>

- Symptom Database
  - SAN misconfiguration
Recap of Running Example (Scenario 1)

➢ Impact analysis
  ➢ High score
Scenario 2

➢ Problem
  ➢ Concurrent IO
  ➢ In bursty manner
  ➢ Query is not affected

➢ SAN-only tool will fail to distinguish between the two causes
Scenario 2

- Correlated Operators
- Symptom Database
  - V1 misconfiguration – High confidence score
  - V2 workload – low confidence score
Other Scenarios

- Change in data properties
- With or without concurrent SAN problems
- Spurious/missing symptoms
- More details in the paper
Related work

➢ DBMS level diagnosis
  ▶ For example: Dageville et al. [VLDB'04]

➢ SAN level diagnosis
  ▶ For example: Genesis [ICDCS'06]

➢ Machine learning techniques for diagnosis
  ▶ For example: PeerPresure [OSDI'04]

➢ Incorporating expert knowledge in diagnosis
  ▶ For example: Yemini et al. [IEEE Comm. Magazine '96]
Conclusions & Future work

➢ DIADS
  ▶ APG: Provides holistic view across DBMS and SAN
  ▶ Diagnosis workflow: Careful integration of machine learning and expert knowledge
  ▶ Can succeed where DBMS-only and SAN-only tools fail

➢ Future directions
  ▶ Alternative techniques for each module
  ▶ Automated fix recommendation
  ▶ Other applications of DIADS, e.g., what-if for SAN changes