Making Fast Databases

FASTER

@andy_pavlo
If a system is already fast, can we make it faster without giving up ACID?
H-Store
Main Memory • Parallel • Shared-Nothing Transaction Processing
H-Store

Client Application

Transaction Result

Database Node
Core
Execution Engine
Partition Data
Main Memory
Partition Data

Database Node
Core
Execution Engine
Partition Data
Main Memory
Partition Data

Database Node
Core
Execution Engine
Partition Data
Main Memory
Partition Data

Database Node
Core
Execution Engine
Partition Data
Main Memory
Partition Data

Database Node
Core
Execution Engine
Partition Data
Main Memory
Partition Data

Database Node
Core
Execution Engine
Partition Data
Main Memory
Partition Data

Database Node
Core
Execution Engine
Partition Data
Main Memory
Partition Data

Database Node
Core
Execution Engine
Partition Data
Main Memory
Partition Data

Database Node
Core
Execution Engine
Partition Data
Main Memory
Partition Data

Database Node
Core
Execution Engine
Partition Data
Main Memory
Partition Data

Database Node
Core
Execution Engine
Partition Data
Main Memory
Partition Data

Database Node
Core
Execution Engine
Partition Data
Main Memory
Partition Data

Database Node
Core
Execution Engine
Partition Data
Main Memory
Partition Data
TPC-C NewOrder

H-Store

txn/s

Partitions

No Distributed Txns
20% Distributed Txns

4 8 12 16 20 24 28 32 36 40 44 48 52 56 60 64

0 40,000 80,000 120,000 160,000 200,000 240,000 280,000 320,000 360,000 400,000

Optimization #1:
Partition database to reduce the number of distributed txns.
### CUSTOMER

<table>
<thead>
<tr>
<th>c_id</th>
<th>c_w_id</th>
<th>c_last</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1001</td>
<td>5</td>
<td>RZA</td>
<td>-</td>
</tr>
<tr>
<td>1002</td>
<td>3</td>
<td>GZA</td>
<td>-</td>
</tr>
<tr>
<td>1003</td>
<td>12</td>
<td>Raekwon</td>
<td>-</td>
</tr>
<tr>
<td>1004</td>
<td>5</td>
<td>Deck</td>
<td>-</td>
</tr>
<tr>
<td>1005</td>
<td>6</td>
<td>Killah</td>
<td>-</td>
</tr>
<tr>
<td>1006</td>
<td>7</td>
<td>ODB</td>
<td>-</td>
</tr>
</tbody>
</table>

### ORDERS

<table>
<thead>
<tr>
<th>o_id</th>
<th>o_c_id</th>
<th>o_w_id</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>78703</td>
<td>1004</td>
<td>5</td>
<td>-</td>
</tr>
<tr>
<td>78704</td>
<td>1002</td>
<td>3</td>
<td>-</td>
</tr>
<tr>
<td>78705</td>
<td>1006</td>
<td>7</td>
<td>-</td>
</tr>
<tr>
<td>78706</td>
<td>1005</td>
<td>6</td>
<td>-</td>
</tr>
<tr>
<td>78707</td>
<td>1005</td>
<td>6</td>
<td>-</td>
</tr>
<tr>
<td>78708</td>
<td>1003</td>
<td>12</td>
<td>-</td>
</tr>
<tr>
<td>i_id</td>
<td>i_name</td>
<td>i_price</td>
<td>...</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
<td>---------</td>
<td>-----</td>
</tr>
<tr>
<td>603514</td>
<td>XXX</td>
<td>23.99</td>
<td>-</td>
</tr>
<tr>
<td>267923</td>
<td>XXX</td>
<td>19.99</td>
<td>-</td>
</tr>
<tr>
<td>475386</td>
<td>XXX</td>
<td>14.99</td>
<td>-</td>
</tr>
<tr>
<td>578945</td>
<td>XXX</td>
<td>9.98</td>
<td>-</td>
</tr>
<tr>
<td>476348</td>
<td>XXX</td>
<td>103.49</td>
<td>-</td>
</tr>
<tr>
<td>784285</td>
<td>XXX</td>
<td>69.99</td>
<td>-</td>
</tr>
<tr>
<td>c_id</td>
<td>c_w_id</td>
<td>c_last</td>
<td>...</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
<td>----------</td>
<td>-----</td>
</tr>
<tr>
<td>1001</td>
<td>5</td>
<td>RZA</td>
<td></td>
</tr>
<tr>
<td>1002</td>
<td>3</td>
<td>GZA</td>
<td></td>
</tr>
<tr>
<td>1003</td>
<td>2</td>
<td>Raekwon</td>
<td></td>
</tr>
<tr>
<td>1004</td>
<td>7</td>
<td>ODB</td>
<td></td>
</tr>
<tr>
<td>1005</td>
<td>8</td>
<td>Killah</td>
<td></td>
</tr>
<tr>
<td>1006</td>
<td>7</td>
<td>ODB</td>
<td></td>
</tr>
</tbody>
</table>
DDL

- CUSTOMER
- ITEM
- ORDERS

NewOrder

```
SELECT * FROM WAREHOUSE
WHERE W_ID = 10;

SELECT * FROM DISTRICT
WHERE D_W_ID = 10 AND D_ID = 9;

INSERT INTO ORDERS
(O_W_ID, O_D_ID, O_C_ID, ...)
VALUES
(10, 9, 12345, ...);
```
Large-Neighborhood Search Algorithm
The graphs illustrate the performance of Horticulture and State of the Art in TATP and TPC-C benchmarks. The x-axis represents the number of partitions, ranging from 4 to 64, while the y-axis shows the transaction per second (txn/s) ranging from 0 to 18,000.

In both TATP and TPC-C, Horticulture consistently outperforms State of the Art across all partition sizes. The performance improvement is evident as the number of partitions increases, with Horticulture achieving higher txn/s values compared to State of the Art in all depicted scenarios.
Optimization #2: Predict what txns will do before they execute.

On Predictive Modeling for Optimizing Transaction Execution in Parallel OLTP Systems
in PVLDB, vol 5. issue 2, October 2011
### Houdini Performance Comparison

- **TATP**: +57%
- **TPC-C**: +126%
- **AuctionMark**: +117%

**Graphs**

- **Axes**:
  - **TXN/s**
  - **Number of Partitions** (4, 8, 16, 32, 64)

- **Legend**:
  - Houdini
  - Assume Single-Partitioned

Graphs illustrate performance improvements across different workloads and partition sizes.
Conclusion: Achieving fast performance is more than just using only RAM.

Future Work: Reduce distributed txn overhead through creative scheduling.
h-store
hstore.cs.cs.brown.edu
github.com/apavlo/h-store
Help is Available

+1- 617-258-6643

Graduate Student Abuse Hotline
Available 24/7
Collect Calls Accepted