On the Cost of Database Clusters Reconfiguration

R. Vilaça, J. Pereira, R. Oliveira
Universidade do Minho
Portugal

J.E. Armendáriz-Iñigo, J.R. González de Mendívil
Universidad Pública de Navarra
Spain
Online Database Recovery

- Reconfiguration of a database cluster required to face load spikes or to restore the resilience of the system

- Update new replicas to the most current database state online

- Minimize time required to finish recovery

- Minimize impact on resource usage and performance of the cluster as a whole
Prominent practical problem

Often addressed before (eg., [Kemme et al., 2001], [Jiménez-Peris et al., 2002], [Armendáriz-Íñigo et al., 2007])

Applied to consistent database replication

Existing refined techniques to improve performance

Existing work

Assumes simplified models without taking into account system limitations such as I/O and CPU

Does not provide a detailed evaluation under representative workload scenarios
Introduction

Goals

- Combine the proposed techniques in a single protocol
- Systematically benchmark different reconfiguration scenarios
- Assess each technique’s performance impact and overhead on the clustered database service
- Determine fundamental limits to cluster reconfiguration
- Discuss the relative merits of each approach
Based on the algorithm presented in [Kemme et al., 2001]

Parallel Recovery [Jiménez-Peris et al., 2002]

Convergence Phases [Armendáriz-Íñigo et al., 2007]

Includes several obvious optimizations, e.g:

- purging of redundant data changes
- data compression
- parallel and batch applier for received recovery data at the recovering replica.

Configurable: number of donors for Parallel Recovery and number of Convergence Phases
Recovery Protocol

- Full Transfer
- Missed Updates or Delta Transfer
- Parallel Recovery
- Convergence Phases
Full Transfer

r1

r2

r3

t
Full Transfer

View Change

r1

r2

r3

© 2009 Ricardo Vilaça  SRDS 09, Niagara Falls, NY

terça-feira, 2 de Fevereiro de 2010
Full Transfer

r1

View Change

r2

r3

t
Full Transfer

View Change

r1

View Change

r2

View Change

r3
Full Transfer

View Change

r1

r2

r3

t
Full Transfer

View Change

Send Full Database

r1

r2

r3

t
Full Transfer

r1

r2

r3

Send Full Database

Received Full Database

View Change

t
Full Transfer

Send Full Database

Received Full Database

terça-feira, 2 de Fevereiro de 2010
Full Transfer

Send Full Database

Received Full Database

Applied Full Database

View Change

r1

r2

r3
Full Transfer

Send Full Database

Received Full Database

Applied Full Database

Catch-Up

Normal Processing

t

r1

r2

r3

View Change

View Change

terça-feira, 2 de Fevereiro de 2010
Missed Updates

View Change

r1

View Change

r2

View Change

r3

t

terça-feira, 2 de Fevereiro de 2010
Missed Updates

[r1, r2, r3]

View Change

View Change
Missed Updates

View Change

Send Missed Updates

Received Missed Updates

r1

r2

r3

terça-feira, 2 de Fevereiro de 2010
Missed Updates

Send Missed Updates

Received Missed Updates

View Change

View Change
Missed Updates

r1

View Change

r2

View Change

r3

Send Missed Updates

Received Missed Updates

Applied Missed Updates
Missed Updates

Send Missed Updates

Received Missed Updates

Applied Missed Updates

Catch-Up

r1

r2

r3

View Change

View Change

t
Missed Updates

Send Missed Updates

View Change

Received Missed Updates

Applied Missed Updates

Catch-Up

Normal Processing

t
Missed Updates

Minimal set of update, insert or deletes statements
Missed Updates

View Change

Send Missed Updates

View Change

Received Missed Updates

Applied Missed Updates

Catch-Up

Normal delegate transaction processing

Minimal set of update, insert or deletes statements

terça-feira, 2 de Fevereiro de 2010
Parallel Recovery

View Change

Send Missed Updates

Send Missed Updates

t
Parallel Recovery

r1

r2

r3

View Change

Send Missed Updates

Received Missed Updates

View Change

t

Send Missed Updates
Parallel Recovery

View Change

Send Missed Updates

Received Missed Updates

terça-feira, 2 de Fevereiro de 2010
Parallel Recovery

- View Change
- Send Missed Updates
- Received Missed Updates
- Applied Missed Updates
- Catch-Up
Parallel Recovery

View Change

Send Missed Updates

Received Missed Updates

Applied Missed Updates

Catch-Up

Normal Processing

Send Missed Updates

Vanced Change

r1

r2

r3

t
Convergence Phases
Convergence Phases

View Change

r1

r2

r3

t
Convergence Phases

View Change

Send Missed Updates

r1

r2

r3

t
Convergence Phases

Send Missed Updates

View Change

r1

r2

r3
Convergence Phases

Phase 1

Send Missed Updates

View Change

r1
r2
r3

View Change

t

© 2009 Ricardo Vilaça
SRDS 09, Niagara Falls, NY

terça-feira, 2 de Fevereiro de 2010
Convergence Phases

View Change

Send Missed Updates

Phase 1

r1

r2

r3

terça-feira, 2 de Fevereiro de 2010
Convergence Phases

Phase 1

View Change

Send Missed Updates

r1

r2

r3

terça-feira, 2 de Fevereiro de 2010
Convergence Phases

View Change

Send Missed Updates

Phase 1

Phase 2

r1

r2

r3

terça-feira, 2 de Fevereiro de 2010
Convergence Phases

View Change

Send Missed Updates

Phase 1

Phase 2

r1

r2

r3

terça-feira, 2 de Fevereiro de 2010
Convergence Phases

Phase 1

Phase 2

Send Missed Updates

View Change

r1

r2

r3

t
Convergence Phases

Send Missed Updates

View Change

Phase 1

Phase 2

Phase n

r1

r2

r3

terça-feira, 2 de Fevereiro de 2010
Convergence Phases

View Change

Send Missed Updates

Phase 1
Phase 2
Phase n
Catch-Up

Normal Processing
Experimental Setting

- 4 commodity servers on par with the systems used in recent related work
  - Intel Core 2 Duo at 2.13GHz, 1GB RAM and dedicated SATA HD.
- Replicas ran an instance of PostgreSQL 8.1 and a Java Virtual Machine (1.5.0) for the Replication service.
- No failures during the recovery process
- At most one replica was recovering during evaluation
- Flow Control on the incoming rate of update transactions during recovery
- Average of three independent samples
Two standard benchmarks

- **TPC-C**: write intensive workload, IO stressing (15 warehouses, 150 clients, 2.2GB database)
- **TPC-W**: read intensive workload, CPU stressing (Shopping Mix, 400 clients, 10000 items, 2.4GB database)
Experiments’ Workload

Two standard benchmarks

- **TPC-C**: write intensive workload, IO stressing (15 warehouses, 150 clients, 2.2GB database)
- **TPC-W**: read intensive workload, CPU stressing (Shopping Mix, 400 clients, 10000 items, 2.4GB database)

Evaluation of recovery with all replicas close to their **nominal capacity**, maximum load that does not saturate machines.
Recovery Time: Full Transfer

TPC-C 150 clients
2.2GB database
throughput 110 tpm

Specific database dump and restore tools

TPC-W 400 clients
2.4GB database
throughput 3000 tpm
Recovery Time: Convergence Phases

TPC-C 150 clients
2.2GB database
throughput 110 tpm

1 phase
2 phases
5 phases

TPC-W 400 clients
2.4GB database
throughput 3000 tpm
**TPC-C** 150 clients
- 2.2GB database
- throughput 110 tpm

**TPC-W** 400 clients
- 2.4GB database
- throughput 3000 tpm
### Recovery Time: Flow Control

#### TPC-C 150 clients
- 2.2GB database
- Throughput: 110 tpm

#### TPC-W 400 clients
- 2.4GB database
- Throughput: 3000 tpm

**Graphs:**
- **Top Graph:** Recovery time vs. outdatedness for 2 phases, 3 donors.
- **Bottom Graph:** Recovery time vs. outdatedness for 2 phases, 3 donors.

**Legend:**
- Red: 0% FlowControl
- Orange: 5% FlowControl
- Brown: 10% FlowControl
- Purple: 20% FlowControl
- Blue: 50% FlowControl
- Green: 100% FlowControl
TPC-W, 400 clients, 2.4GB database

Before and After recovery

1 -> 1 state donor, 2 convergence phases and no flow control
2 -> 1 state donor, 5 convergence phases and no flow control
3 -> 2 state donor, 2 convergence phases and no flow control
4 -> 3 state donor, 2 convergence phases and no flow control
5 -> 3 state donor, 2 convergence phases and 5% flow control
6 -> 3 state donor, 2 convergence phases and 10% flow control

TPC-C, 150 clients, 2.2GB database
Donation vs. Recovery Time

Recovery is about 7x longer than donation

For a TPC-C workload, a 86MB recovery log and a single donor:

- Donor replica
- Recovering replica

<table>
<thead>
<tr>
<th>Time (s)</th>
<th>300</th>
<th>600</th>
<th>900</th>
<th>1200</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recovery</td>
<td>168</td>
<td>168</td>
<td>1145</td>
<td></td>
</tr>
<tr>
<td>Donation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Database online recovery protocol combining several previously proposed optimization techniques
Database online recovery protocol combining several previously proposed optimization techniques

The results of our tests do not reveal any relevant effect of the optimizations on the recovery time or on the overall cluster performance either.
Database online recovery protocol combining several previously proposed optimization techniques.

The results of our tests do not reveal any relevant effect of the optimizations on the recovery time or on the overall cluster performance either.

The capacity of the recovering replica to apply the received state turns out to be the salient limiting factor.
Database online recovery protocol combining several previously proposed optimization techniques

The results of our tests do not reveal any relevant effect of the optimizations on the recovery time or on the overall cluster performance either.

The capacity of the recovering replica to apply the received state turns out to be the salient limiting factor.

Most research has been targeted at optimizing the operations that are not, by a large margin, limiting factors in overall performance.