Increasing Performance in Byzantine Fault-Tolerant Systems with On-Demand Replica Consistency

Tobias Distler and Rüdiger Kapitza

System Software Group
Friedrich-Alexander University Erlangen-Nuremberg

EuroSys
April 11, 2011
Byzantine Fault Tolerance (BFT)

- Agreement-based Byzantine fault tolerance
  - $3f + 1$ replicas to tolerate $f$ faults
  - BFT agreement protocol
  - Client-side voting

- Drawbacks
  - **High resource usage**
  - Performance overhead for agreement

**REFIT Project Research Goal**

**Resource-efficient BFT systems**

[Castro et al., Practical Byzantine fault tolerance, OSDI '99]
Making BFT Systems More Resource-Efficient

REFIT Project Research Goal

Resource-efficient BFT systems

Optimizing Resource Usage
- Reduced number of replicas
- Same performance
- Recent examples
  - SPARE [Distler et al., NDSS ’11]
  - ZZ [Wood et al., EuroSys ’11]

Optimizing Performance
- Default number of replicas
- Increased performance

⇒ ODRC

Tobias Distler (distler@cs.fau.de)
Where to Optimize?

- **Agreement stage**
  - BFT agreement protocol
  - Sequence of agreed requests

- **Execution stage**
  - Service application
  - Request processing

**Observations**
- Past optimizations have **significantly reduced agreement overhead**
- Non-trivial services: response times are dominated by execution stage

**ODRC Approach**
Reducing the load on the execution stage

[Tobias Distler (distler@cs.fau.de)]
Increasing Performance in BFT Systems with ODRC
Basic Approach

- **Traditional BFT systems**
  - All $3f + 1$ replicas process all requests
  - Client waits for $f + 1$ identical replies

**Insight**

In the absence of faults, a client only needs $f + 1$ replies to make progress

- **ODRC**
  - Each request is processed by only $f + 1$ replicas
  - Load distribution across replicas
  - Additional replicas process the request in case of faults
ODRC

- Selective Request Execution
- On-Demand Replica Consistency
- Evaluation
- Conclusion
ODRC

- Selective Request Execution
- On-Demand Replica Consistency
- Evaluation
- Conclusion
Selector

- Selects requests for execution
- Stores requests that have not been selected
Application State

- Set of objects
  - Examples: files, directories, ...
  - Assumption: requests carry information about object access

![Diagram showing the flow of agreement, selection, and execution states with colored nodes representing state objects]
Application State Distribution

Object distribution scheme
- Each object is **maintained** on \( f + 1 \) replicas, **unmaintained** on others
- State of unmaintained objects may be outdated

![Diagram of object distribution]

Client

- Agreement
- Selector
- Execution

Maintained Objects

Tobias Distler (distler@cs.fau.de)  Increasing Performance in BFT Systems with ODRC
Selective Request Execution in Action

- Normal-case operation

- Garbage collection of stored requests
  - Periodic **object checkpoints** of maintained objects
  - Stable checkpoint: \( f + 1 \) identical checkpoints
Selective Request Execution in Action

- Normal-case operation

- Garbage collection of stored requests
  - Periodic **object checkpoints** of maintained objects
  - Stable checkpoint: $f + 1$ identical checkpoints

Tobias Distler (distler@cs.fau.de) Increasing Performance in BFT Systems with ODRC
Selective Request Execution in Action

- Normal-case operation

- Garbage collection of stored requests
  - Periodic **object checkpoints** of maintained objects
  - Stable checkpoint: $f + 1$ identical checkpoints

Client

Graph showing the process of request execution with nodes for Agreement, Selector, and Execution.
Selective Request Execution in Action

- Normal-case operation

- Garbage collection of stored requests
  - Periodic **object checkpoints** of maintained objects
  - Stable checkpoint: $f + 1$ identical checkpoints
Selective Request Execution in Action

- Normal-case operation

- Garbage collection of stored requests
  - Periodic **object checkpoints** of maintained objects
  - Stable checkpoint: \( f + 1 \) identical checkpoints
Selective Request Execution in Action

- Normal-case operation

  ![Diagram showing agreement and execution]  

  - Agreement
  - Selector
  - Execution

- Garbage collection of stored requests
  - Periodic **object checkpoints** of maintained objects
  - Stable checkpoint: \( f + 1 \) identical checkpoints

Tobias Distler (distler@cs.fau.de) Increasing Performance in BFT Systems with ODRC
Selective Request Execution in Action

**Normal-case operation**

- Agreement
- Selector
- Execution

**Garbage collection of stored requests**
- Periodic **object checkpoints** of maintained objects
- Stable checkpoint: $f + 1$ identical checkpoints

Tobias Distler (distler@cs.fau.de) Increasing Performance in BFT Systems with ODRC
Talk Outline

ODRC

- Selective Request Execution

- On-Demand Replica Consistency

- Evaluation

- Conclusion
Multi-Object Operations

- Access of multiple objects
  - Only unmaintained objects $\Rightarrow$ store request
  - At least one maintained object
    - Update unmaintained objects
    - Process request

![Diagram showing multi-object operations with agreements and executions for different objects.](image-url)
Multi-Object Operations

Access of multiple objects
- Only unmaintained objects ⇒ store request
- At least one maintained object
  - Update unmaintained objects
  - Process request

On-Demand Replica Consistency
- Only if a request demands it
- Only to the extent demanded by a request
Multi-Object Operations

- Access of multiple objects
  - Only unmaintained objects ⇒ store request
  - At least one maintained object
    - Update unmaintained objects
    - Process request

Client

Agreement$_0$ → $\text{inc()}$ → Agreement$_1$ → $\text{inc()}$ → Agreement$_2$ → $\text{inc()}$ → Agreement$_3$ → $\text{inc()}$

Execution$_0$ → $\text{inc()}$ → Execution$_1$ → $\text{inc()}$ → Execution$_2$ → $\text{inc()}$ → Execution$_3$
Multi-Object Operations

- Access of multiple objects
  - Only unmaintained objects ⇒ store request
  - At least one maintained object
    - Update unmaintained objects
    - Process request

Client

Agreement$_0$ $\rightarrow$ inc()$_0$ $\rightarrow$ Execution$_0$

Agreement$_1$ $\rightarrow$ inc()$_1$ $\rightarrow$ Sinc()$_1$ $\rightarrow$ Execution$_1$

Agreement$_2$ $\rightarrow$ inc()$_2$ $\rightarrow$ Sinc()$_2$ $\rightarrow$ Execution$_2$

Agreement$_3$ $\rightarrow$ Sinc()$_3$ $\rightarrow$ inc() $\rightarrow$ Execution$_3$
Multi-Object Operations

Access of multiple objects
- Only unmaintained objects ⇒ store request
- At least one maintained object
  - Update unmaintained objects
  - Process request

On-Demand Replica Consistency

only if
a request demands it
only to the extent
demanded
by a request

Tobias Distler (distler@cs.fau.de) Increasing Performance in BFT Systems with ODRC
Multi-Object Operations

Access of multiple objects
- Only unmaintained objects ⇒ store request
- At least one maintained object
  - Update unmaintained objects
  - Process request

Cross-border request
Multi-Object Operations

- Access of multiple objects
  - Only unmaintained objects ⇒ store request
  - At least one maintained object
    - Update unmaintained objects
    - Process request

---

Tobias Distler (distler@cs.fau.de)
Increasing Performance in BFT Systems with ODRC 13
Multi-Object Operations

- Access of multiple objects
  - Only unmaintained objects ⇒ store request
  - At least one maintained object
    - Update unmaintained objects
    - Process request

On-Demand Replica Consistency

only if a request demands it
only to the extent demanded by a request

Tobias Distler (distler@cs.fau.de) Increasing Performance in BFT Systems with ODRC
Multi-Object Operations

- Access of multiple objects
  - Only unmaintained objects ⇝ store request
  - At least one maintained object
    - Update unmaintained objects
    - Process request
**Multi-Object Operations**

- **Access of multiple objects**
  - Only unmaintained objects $\Rightarrow$ store request
  - At least one maintained object
    - Update unmaintained objects
    - Process request

---

**On-Demand Replica Consistency**

- Only if a request demands it
- Only to the extent demanded by a request

---

**Tobias Distler** (distler@cs.fau.de)  
**Increasing Performance in BFT Systems with ODRC**
Multi-Object Operations

- Access of multiple objects
  - Only unmaintained objects ⇒ store request
  - At least one maintained object
    - Update unmaintained objects
    - Process request

On-Demand Replica Consistency
- only if a request demands it
- only to the extent demanded by a request

Tobias Distler (distler@cs.fau.de) Increasing Performance in BFT Systems with ODRC
Multi-Object Operations

- Access of multiple objects
  - Only unmaintained objects ⇒ store request
  - At least one maintained object
    - Update unmaintained objects
    - Process request

---

On-Demand Replica Consistency

only if a request demands it

only to the extent demanded by a request
Multi-Object Operations

- Access of multiple objects
  - Only unmaintained objects $\Rightarrow$ store request
  - At least one maintained object
    - Update unmaintained objects
    - Process request

On-Demand Replica Consistency

- Only if a request demands it
- Only to the extent demanded by a request
Cross-Border Requests

- Additional consistency overhead
  - Processed by more than \( f + 1 \) replicas
  - Goal: minimize number of cross-border requests

- Optimized object distribution
  - Application-centric strategies
  - Consider object dependencies

- Example: Network File System (NFS)
  - Assign files and their parent directories to the same replicas
  - Subdirectories may be assigned to different replicas
Handling Faults

- Providing additional replies on demand
  - Standard BFT clients
  - Request retransmission after timeout
  - Additional replicas process the request

![Diagram of BFT System with ODRC](image-url)

Client replies are processed by different Agreement and Selector stages, leading to Execution stages with replicated outputs.
Handling Faults

- Providing additional replies on demand
  - Standard BFT clients
  - Request retransmission after timeout
  - Additional replicas process the request

![Flowchart diagram showing agreement and execution processes with retransmit option.]

Tobias Distler (distler@cs.fau.de)  Increasing Performance in BFT Systems with ODRC 15
Handling Faults

- Providing additional replies on demand
  - Standard BFT clients
  - Request retransmission after timeout
  - Additional replicas process the request
Handling Faults

- Providing additional replies on demand
  - Standard BFT clients
  - Request retransmission after timeout
  - Additional replicas process the request

![Diagram showing the process of handling faults in BFT systems.](image)
Handling Faults

- Providing additional replies on demand
  - Standard BFT clients
  - Request retransmission after timeout
  - Additional replicas process the request

Tobias Distler (distler@cs.fau.de)  Increasing Performance in BFT Systems with ODRC
Talk Outline

ODRC

- Selective Request Execution
- On-Demand Replica Consistency
- Evaluation
- Conclusion
NFS Evaluation: Postmark Benchmark

4 replicas (f=1), write-mostly workload

- **BFT**
- **Unreplicated NFS**
- **ODRC**

61% throughput increase over BFT
NFS Evaluation: Postmark Benchmark

4 replicas (f=1), write-mostly workload

- **BFT**
- **Unreplicated NFS**
- **ODRC**
- **ODRC (optimized distribution)**

90% throughput increase over BFT

Tobias Distler (distler@cs.fau.de) Increasing Performance in BFT Systems with ODRC
NFS Evaluation: Append-Only Micro-Benchmark

- Replica fault
- 4 ODRC replicas
- 20 clients

Average response time [ms]

Time [s]

Tobias Distler (distler@cs.fau.de) Increasing Performance in BFT Systems with ODRC 18
Talk Outline

ODRC

- Selective Request Execution
- On-Demand Replica Consistency
- Evaluation
- Conclusion
Conclusion

- Execution matters!

- Traditional BFT systems
  - All replicas process all requests
  - Consistency overhead

- ODRC
  - Selective request execution based on object access
  - On-demand replica consistency
  - Additional replies in case of faults

Thank you very much.

Questions?