

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

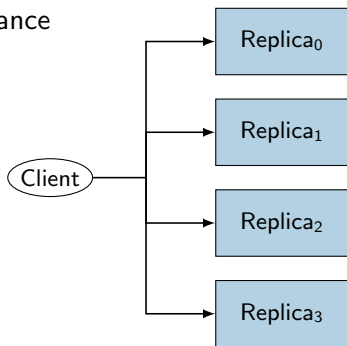
Tobias Distler and Rüdiger Kapitza

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



[Castro et al., Practical Byzantine fault tolerance, OSDI '99]

REFIT Project Research Goal
Resource-efficient BFT systems



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

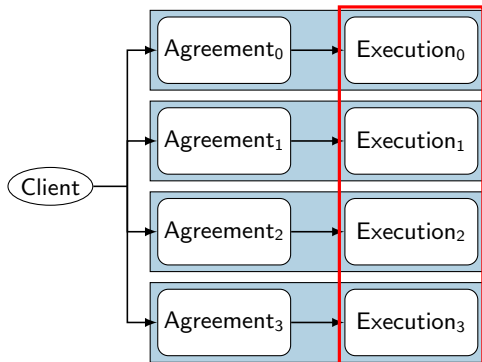
⇒ **ODRC**



Where to Optimize?

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

- Execution stage
 - Service application
 - Request processing



[Yin et al., Separating agreement from execution for Byzantine fault tolerant services, SOSP '03]

- 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

- 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
- **O**n-**D**emand **R**eplica **C**onsistency
- Evaluation
- Conclusion

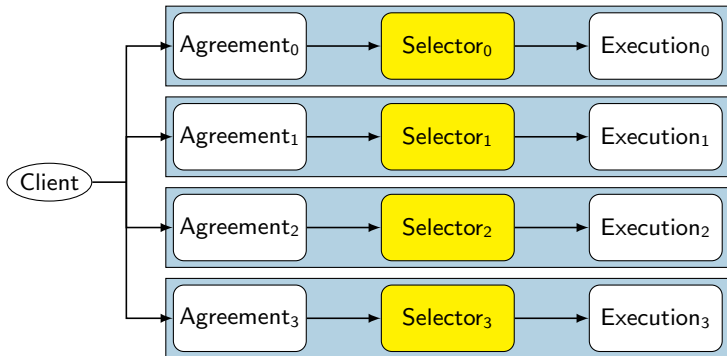


ODRC

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

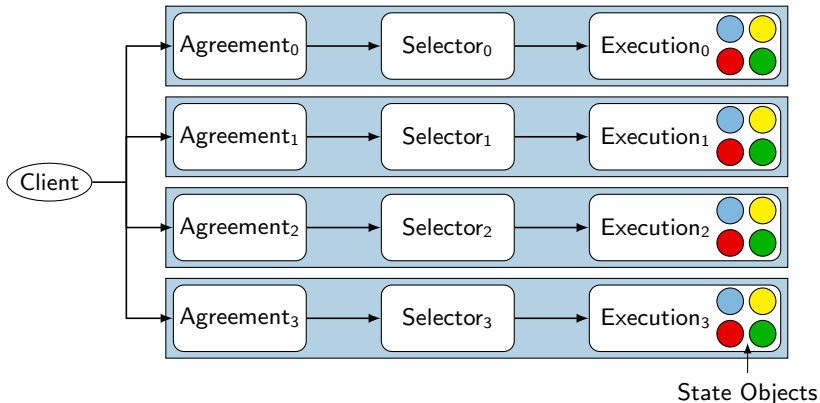


- Selector
 - Selects requests for execution
 - Stores requests that have not been selected



Application State

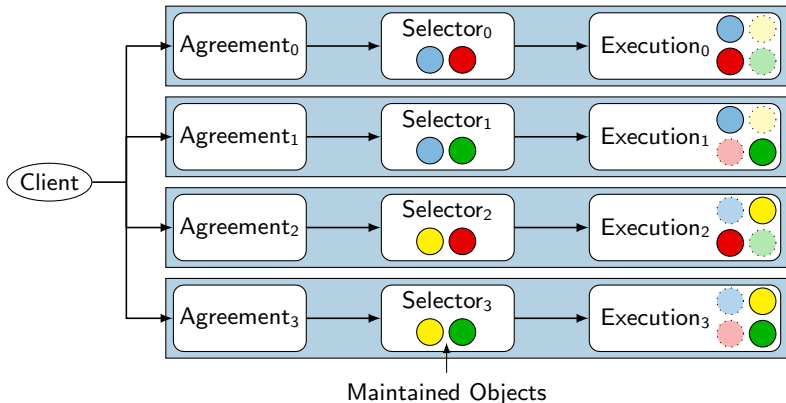
- Set of objects
 - Examples: files, directories, ...
 - Assumption: requests carry information about object access



Application State Distribution

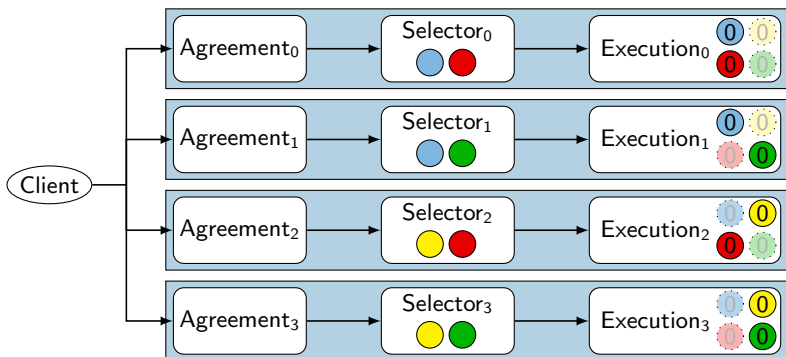
- Object distribution scheme

- Each object is **maintained** on $f + 1$ replicas, **unmaintained** on others
- State of unmaintained objects may be outdated



Selective Request Execution in Action

■ Normal-case operation



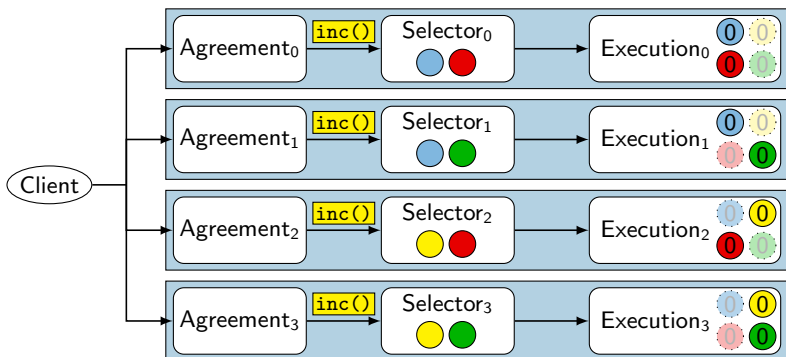
■ Garbage collection of stored requests

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



Selective Request Execution in Action

■ Normal-case operation



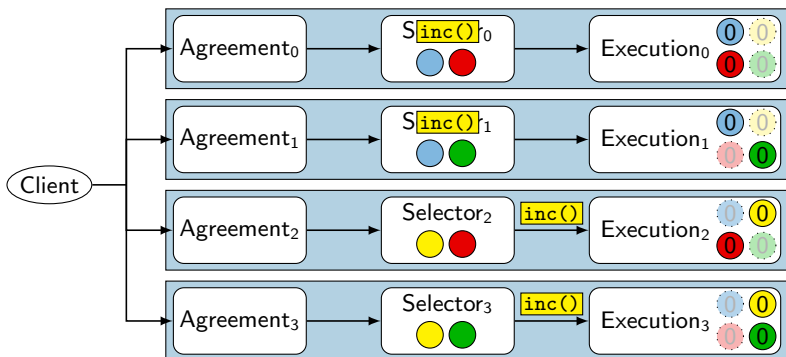
■ Garbage collection of stored requests

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



Selective Request Execution in Action

■ Normal-case operation



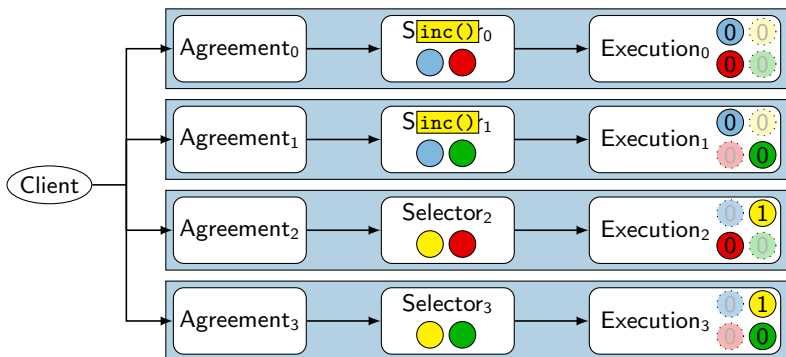
■ Garbage collection of stored requests

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



Selective Request Execution in Action

■ Normal-case operation



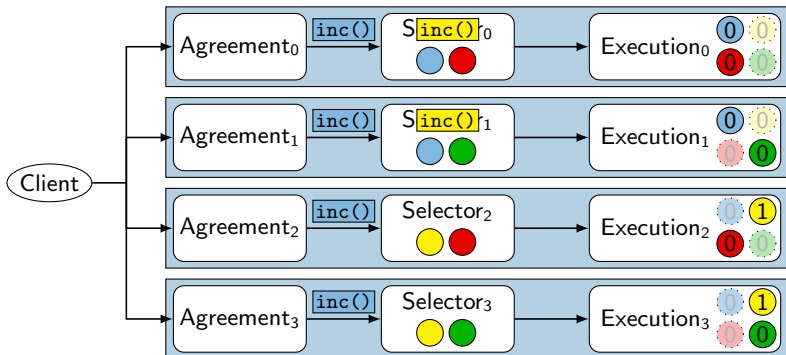
■ Garbage collection of stored requests

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



Selective Request Execution in Action

Normal-case operation



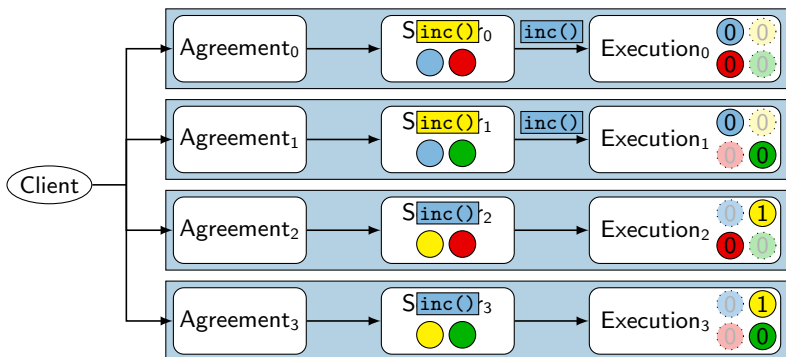
Garbage collection of stored requests

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



Selective Request Execution in Action

■ Normal-case operation



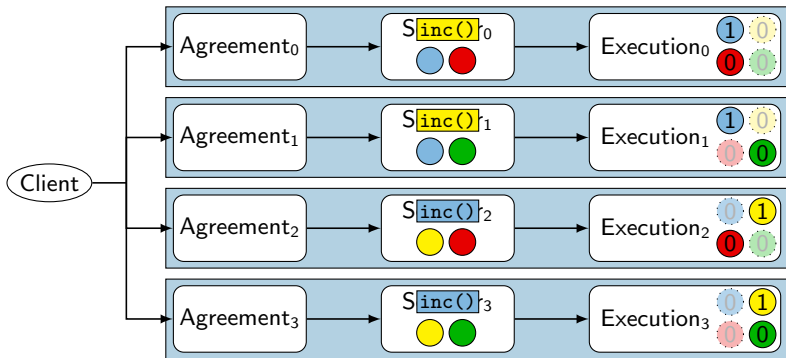
■ Garbage collection of stored requests

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



Selective Request Execution in Action

■ Normal-case operation



■ Garbage collection of stored requests

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



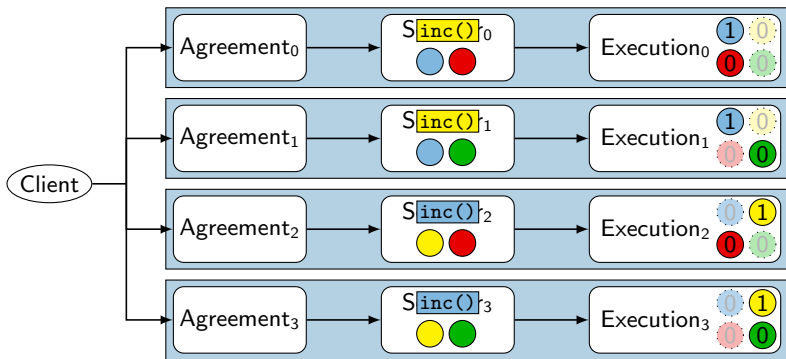
ODRC

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



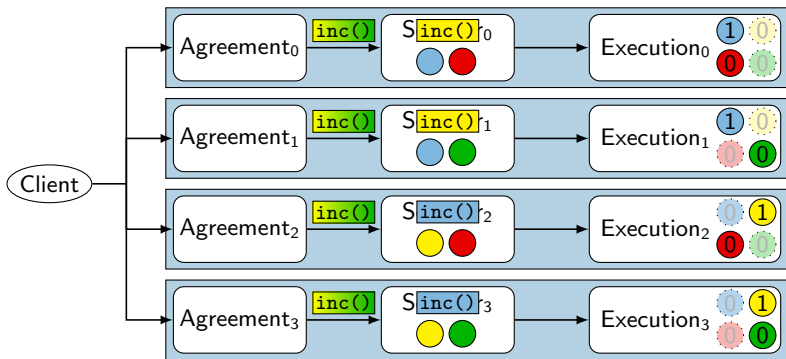
Multi-Object Operations

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



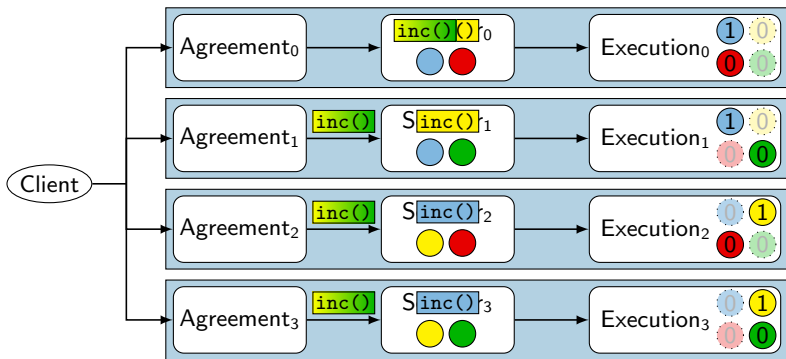
Multi-Object Operations

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



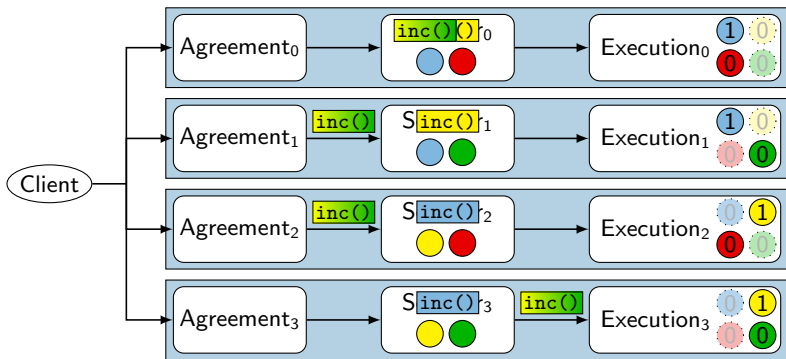
Multi-Object Operations

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



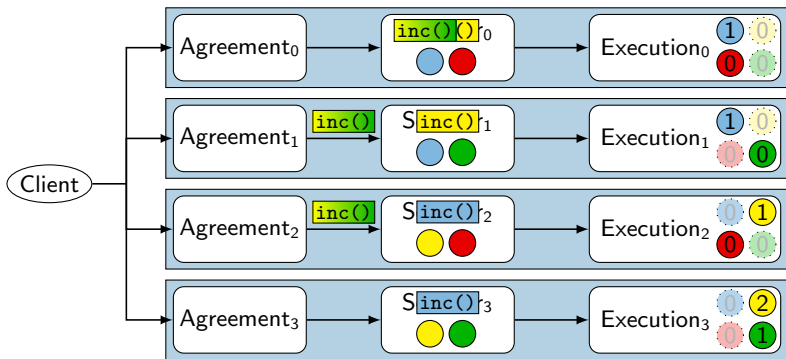
Multi-Object Operations

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



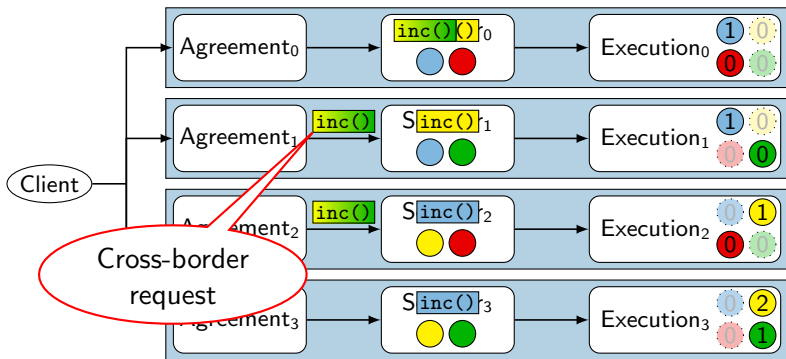
Multi-Object Operations

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



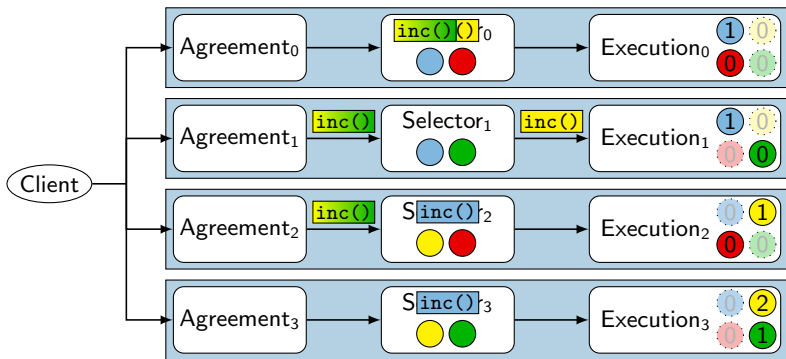
Multi-Object Operations

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



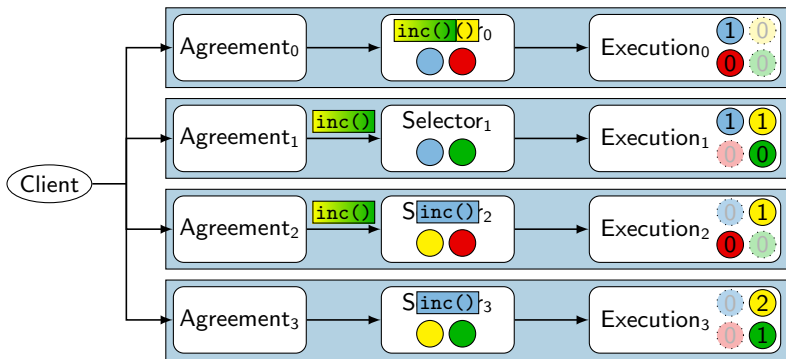
Multi-Object Operations

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



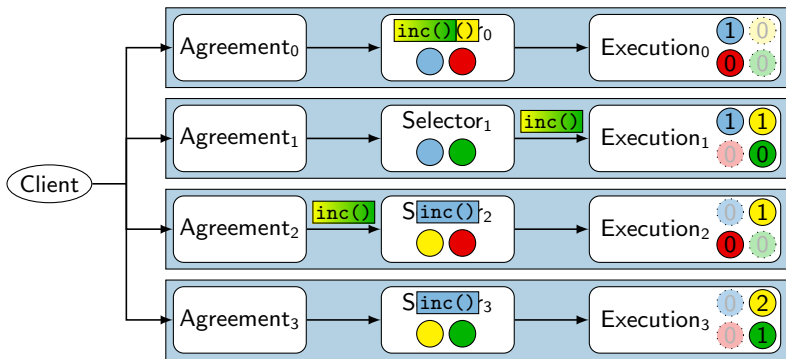
Multi-Object Operations

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



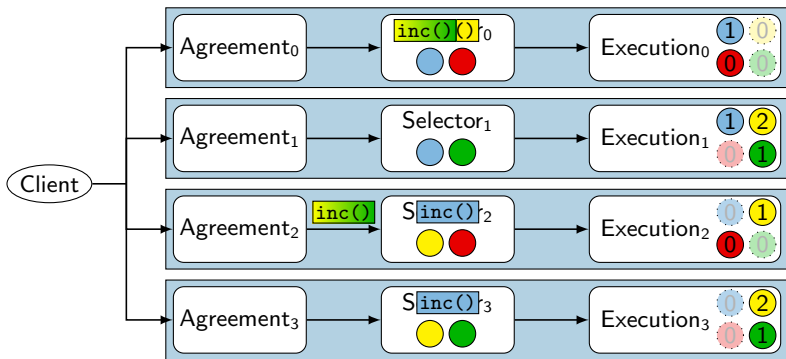
Multi-Object Operations

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



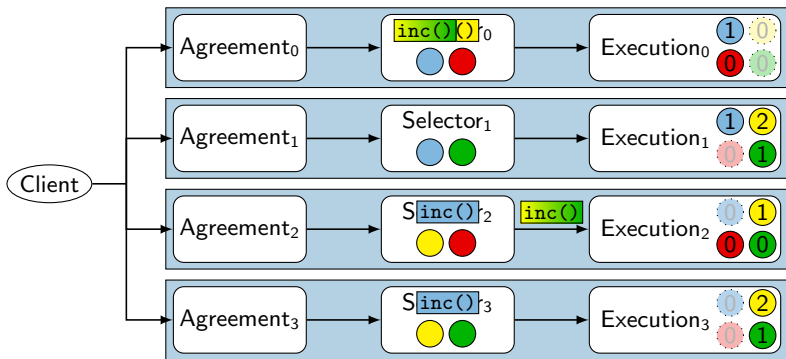
Multi-Object Operations

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



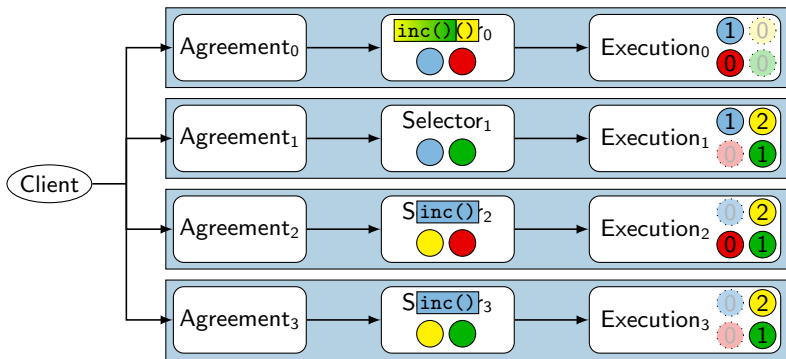
Multi-Object Operations

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



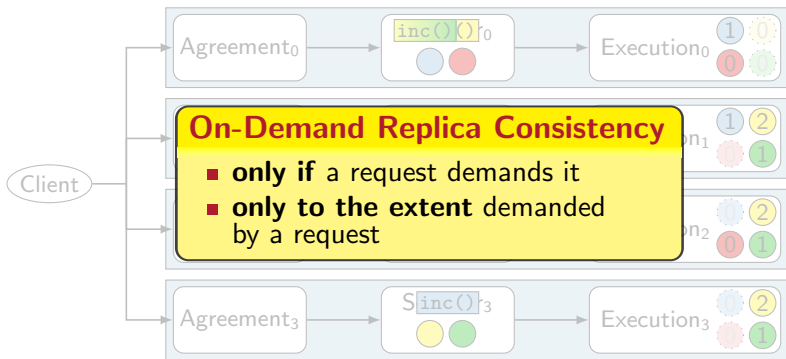
Multi-Object Operations

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



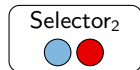
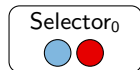
Multi-Object Operations

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



Cross-Border Requests

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

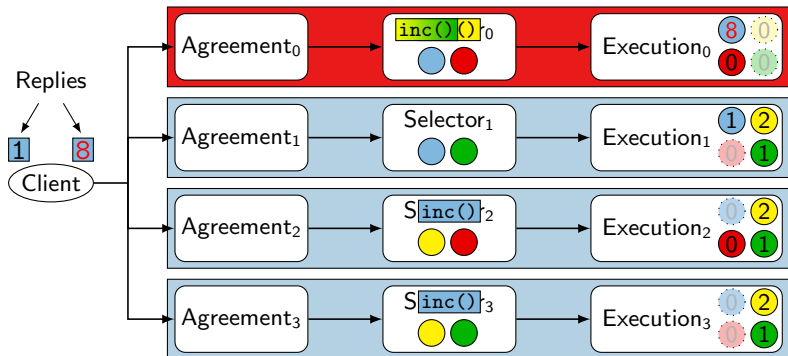


- Optimized object distribution
 - Application-centric strategies
 - Consider object dependencies
- Example: Network File System (NFS)
 - Assign files and their parent directories to the same replicas
 - Subdirectories may be assigned to different replicas



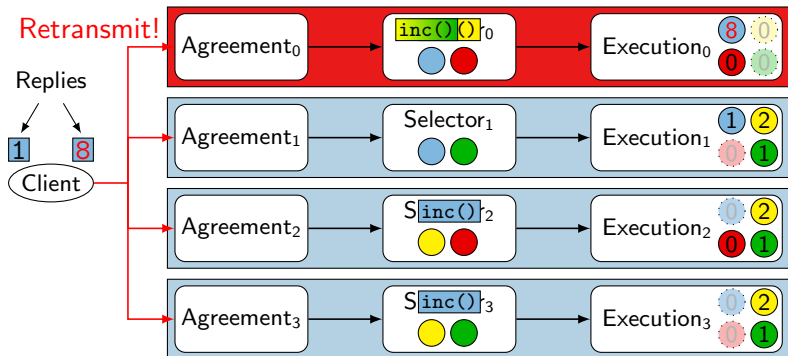
Handling Faults

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



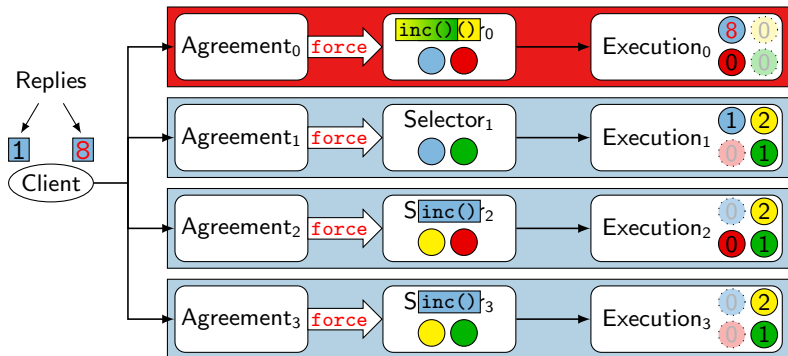
Handling Faults

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



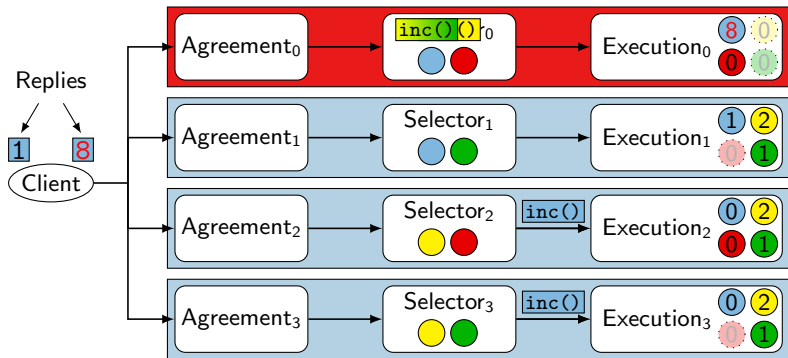
Handling Faults

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



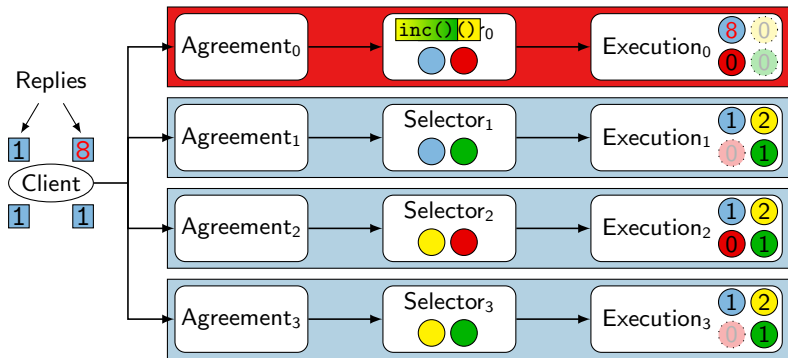
Handling Faults

- 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

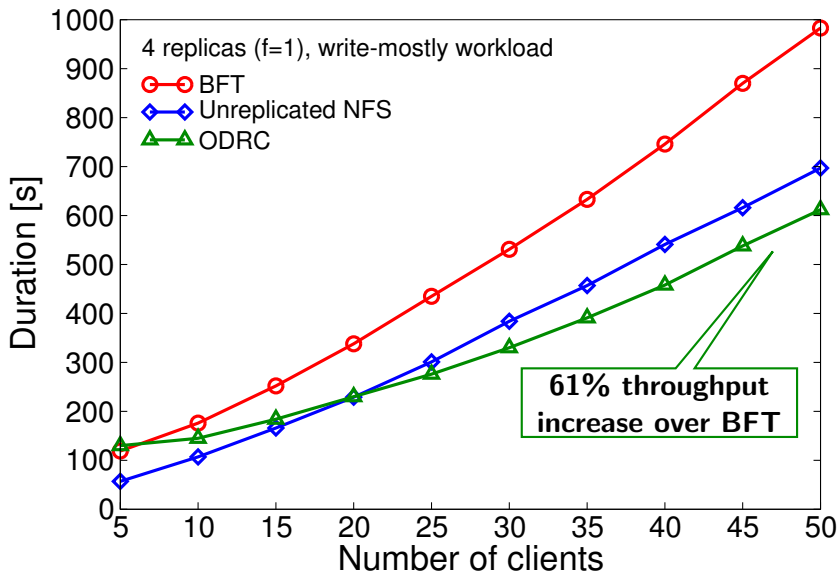


ODRC

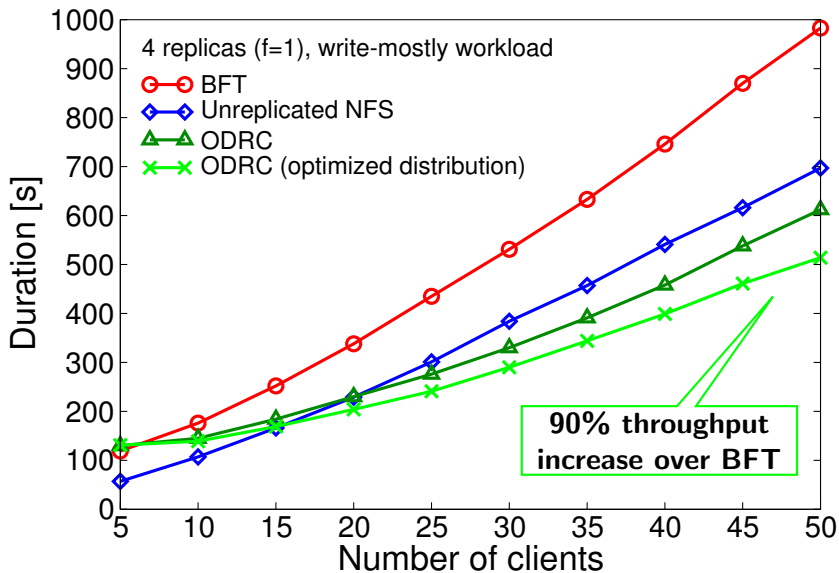
- Selective Request Execution
- On-Demand Replica Consistency
- Evaluation
- Conclusion



NFS Evaluation: Postmark Benchmark



NFS Evaluation: Postmark Benchmark



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?

