Active replication at (almost) no cost

André Martin¹, Andrey Brito², Christof Fetzer¹

¹ Technische Universität Dresden, Germany
² Universidade Federal de Campina Grande, Brazil

SRDS'11 – October, 5th 2011
Outline

- Observations & Motivation
- StreamMine Architecture (Stages, Operators and Slices)
- Active Replication Approach
- Evaluation
- Conclusion
- Q&A
MOTIVATION & OBSERVATIONS
IN CLOUD COMPUTING
Observations & Motivation

- Recent trend: Moving distributed applications to public clouds such as Amazon EC2
- Most cloud providers charge their customers on an hourly basis rather than CPU cycles consumed
- Launching of new instances in a cloud may take several minutes in order to adjust to new workloads
- To accommodate sudden load spikes, stream processing application need to limit their CPU utilization to conservative levels (often as low as 50%)

- What can those spare (& paid) CPU cycles used for?
Observations & Motivation (cont’d)

- Thinking about fault tolerance – how about **Active Replication**?

- Advantages & disadvantages of **active replication**:
  - almost no recovery time (+)
  - requires twice as much resources (-)

- **Passive replication** – trad. based on logging & checkpointing:
  - long recovery times (reading the chkpnt & replaying the log) (-)
  - no additional computational resources needed (+)

⇒ **Paying for the CPU cycles anyway: active replication is the way to go!**
Observations & Motivation (cont’d)

**Environmental aspects:**

- Energy consumption does not increase linearly with CPU utilization

  ⇒ No extra costs for cloud providers if customers fully utilize their dedicated resources

- It’s “green computing”
StreamMine Architecture

- **Stages** – host **operators**
  - custom written or standard CEP ones such as join, selection etc.
- Events are represented as key-value pairs such as in MapReduce
- **Stateful** operators - state is partitioned to scale out
- Unit of partitioning is a **slice**
ACTIVE REPLICATION
OUR APPROACH
Active Replication Approach

**Key ideas**
- Active replication based on deterministic execution
- Fine grained partitioning of data through slices
- Dynamically switch between active and passive replication:
  - Under moderate load, process primary as well as secondary
  - Under high load, solely process primary – en-queue events for secondary
  - If a spike lasts too long, synchronize state of secondary with its primary counterpart and prune the secondary’s queue
- Distribute primary as well secondary slices using interleaved partitioning to minimize load increase in case of node failures
Active Replication Approach

subscribe(s')

state transfer

primary

secondary

node

slice

Active replication at (almost) no cost
Active replication – under moderate load

Source node

Node #1
- Queue
- State
- Primary
- Queue
- State
- Secondary

Node #2
- Queue
- State
- Primary
- Queue
- State
- Secondary
Active replication – under high load /spike
EVALUATION
Experiments setup

- **50-nodes cluster** w/ 2x Intel Xeon E5405 (quad core) CPUs and **8GB** of RAM
- Gigabit Ethernet (1000BaseT full duplex) LAN

- Application:
Scalability - with number of nodes

![Graph showing scalability with number of nodes.](image)
Scalability – within a single processing node

The diagram illustrates the throughput (in KEvents/s) as a function of the number of threads. It compares different execution modes:
- Active replication
- Deterministic execution
- No order

The throughput increases with the number of threads, showing that active replication is significantly more efficient than deterministic execution and no order.
Event throughput and queue length behavior of secondary replica queues

![Graph showing event throughput and queue length behavior](image)
Event throughput and queue length behavior of secondary replica queues (cont’d)

- Throughput (kEvents/s)
- Secondary queue length (kItems)
- Peaks in load and queue length
- State transfer during load spikes

High load / more frequent & longer spikes
Impact on throughput while node failure

- Moderate load

SRDS'11, October, 5th 2011
Spike lengths & state synchronization update interval length

![Graph showing the relationship between peak interval length and average state sync interval length. The graph compares passive replication and active replication, indicating that active replication reduces the interval length significantly compared to passive replication.]

Passive replication
Active replication

SRDS'11, October, 5th 2011
Active replication at (almost) no cost
Slide 21 of 23
Conclusion & Contribution

- Hybrid approach of active & passive replication
  - passive replication until the spike is gone or the load balancer kicks in
- Usage of spare CPU cycles ⇒ cost effective for cloud consumers
- Energy efficient ⇒ good for the environment
- Recovery after a node failure introduces only a minor load increase to surviving nodes due to fine grained distribution of slices
THANK YOU
FOR YOUR UNDIVIDED ATTENTION

andre.martin@se.inf.tu-dresden.de