PHX: Memory Speed HPC I/O with NVM

Pradeep Fernando

Sudarsun Kannan, Ada Gavrilovska, Karsten Schwan

Node Local Persistent I/O ?

- Node local checkpoint/ restart
  - Recover from transient failures (node restart)
  - Transient/ soft failures – not permanent failures
  - More soft failures in future machines

- Locally stored analytics output
  - Co-running analytics

NVRAM is Ideal for Persistent I/O

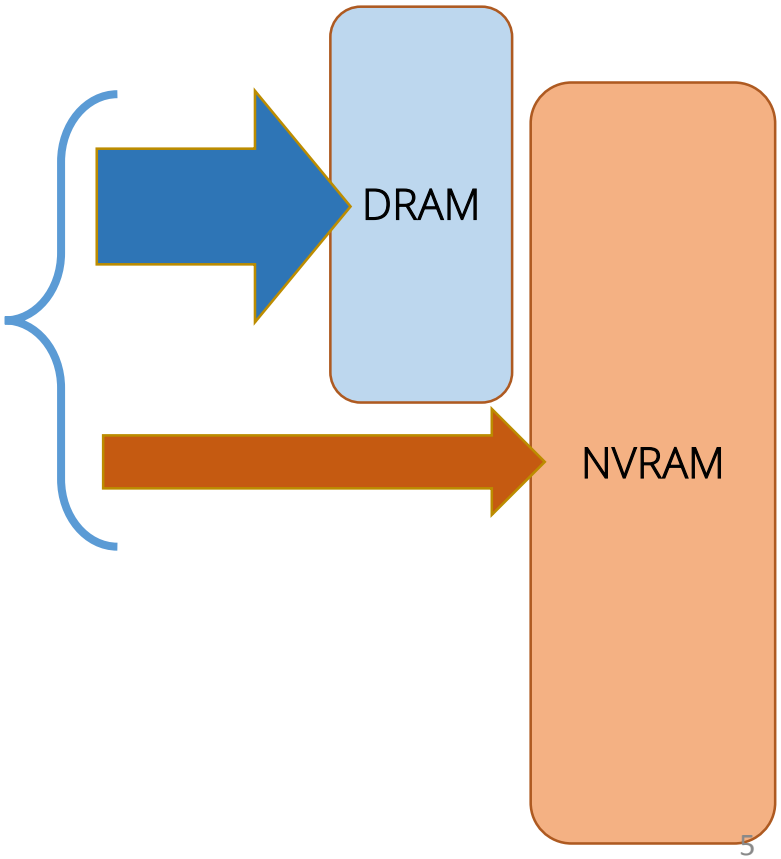
- Read/ write latencies comparable to DRAM
  - 100x faster than SSD
- Persistent data writes
  - Data retention time is >10 years for Memristor
- Denser memory
  - Low chip area per bit -> more capacity
- Memory device -> load/ store operations

NVRAM Bandwidth?

- HPC applications move data in bulk
  - I/O after implicit/explicit synchronization
- More I/O in future Exascale simulations
  - More cores per node -> more data
- Poor bandwidth scaling of NVRAM
  - Device physics
  - RAID like structures -> more energy

Key Idea: Aggregate Bandwidth Checkpoints

- NVRAM provides denser persistent memory, but has limited bandwidth
- DRAM has superior bandwidth compared to NVRAMs (4x-8x)
- Accelerate critical path data movement with bandwidth aggregation



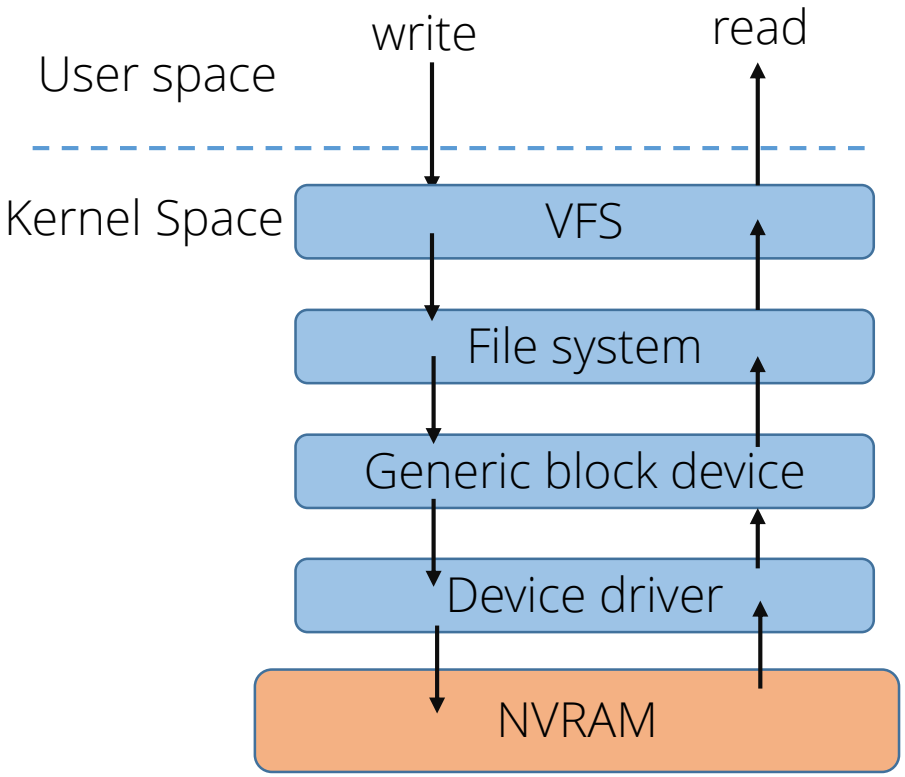
Phoenix (PHX) Design

Problem : Software Stack Overheads

- Using NVRAM as a block device under file system lengthens the I/O path.

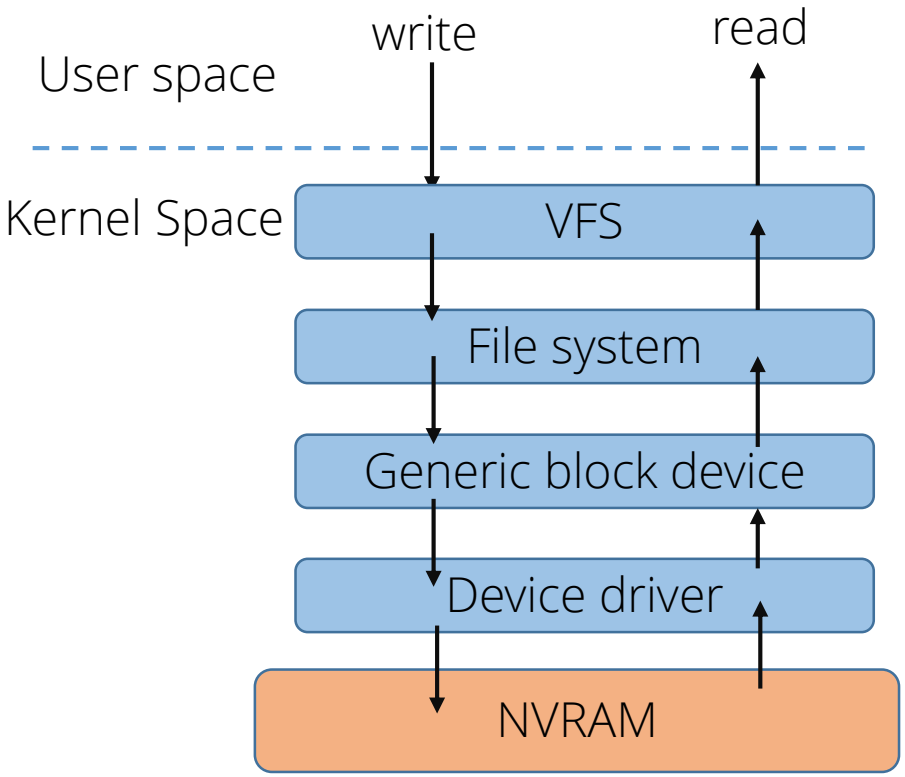
Problem : Software Stack Overheads

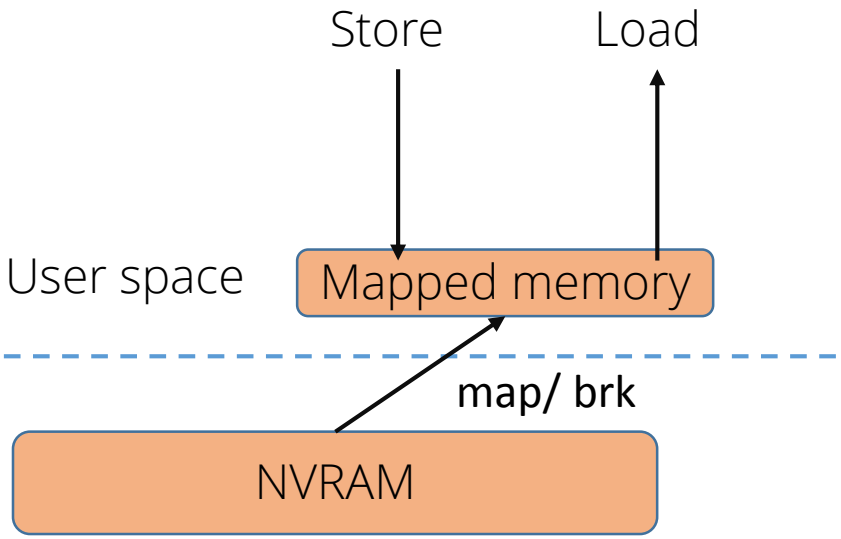
- Using NVRAM as a block device under file system lengthens the I/O path.



Problem : Software Stack Overheads

- Using NVRAM as a block device under file system lengthens the I/O path.



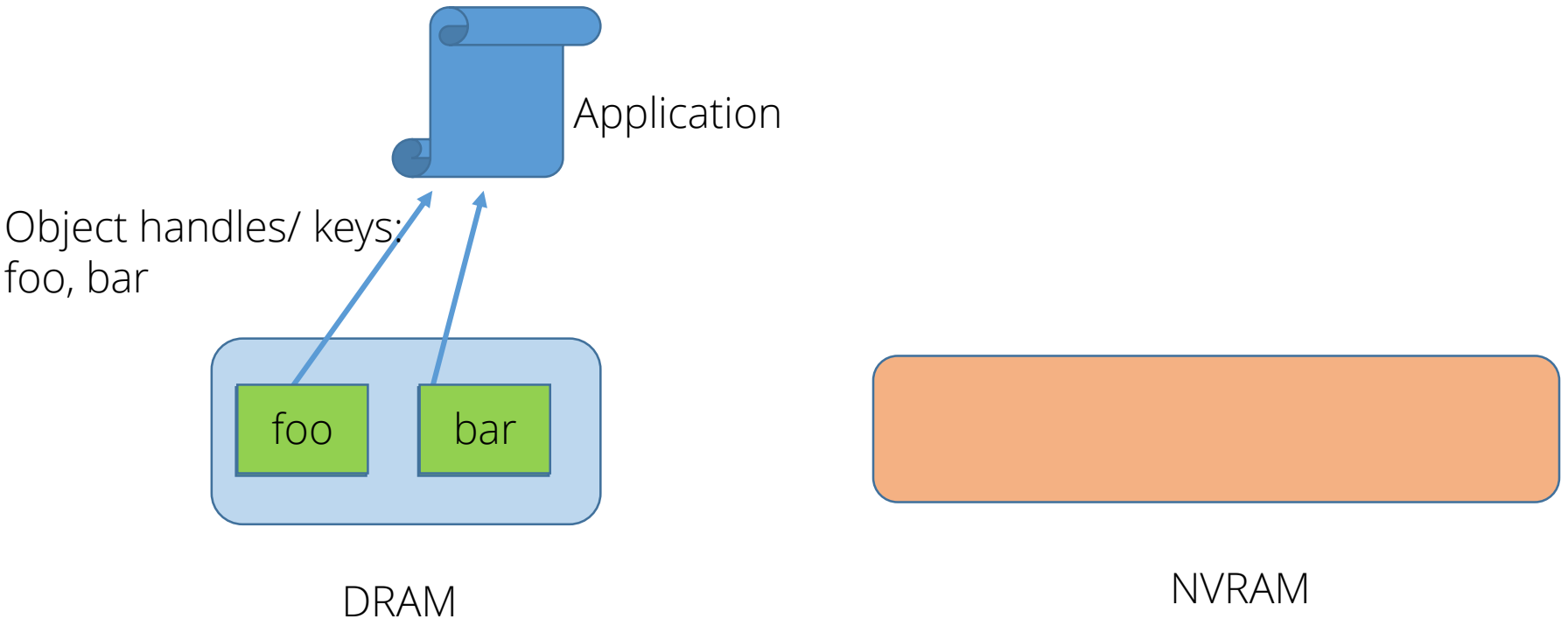


Problem : NVRAM Access Latencies

- NVRAM writes/ reads are expensive
  - Writes takes 4x more time than DRAM
  - Application works with a DRAM allocated object

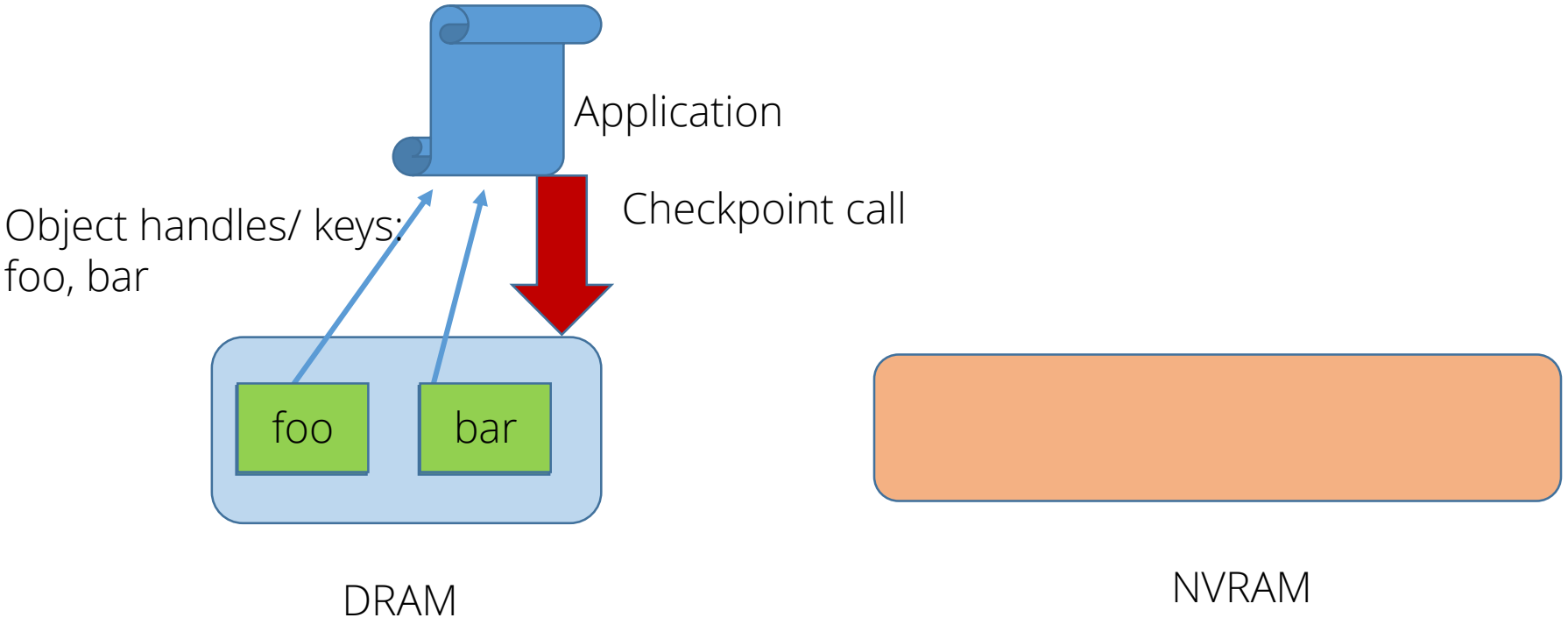
Problem : NVRAM Access Latencies

- NVRAM writes/ reads are expensive
  - Writes takes 4x more time than DRAM
  - Application works with a DRAM allocated object



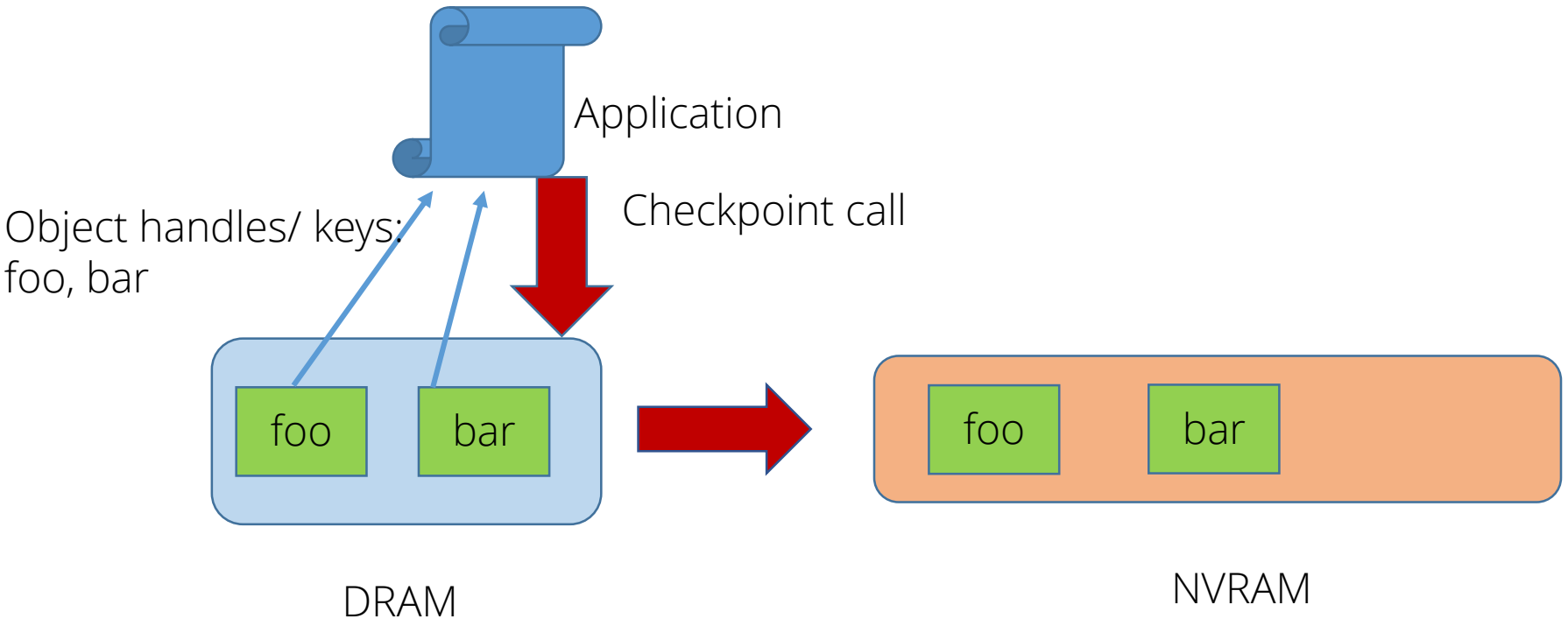
Problem : NVRAM Access Latencies

- NVRAM writes/ reads are expensive
  - Writes takes 4x more time than DRAM
  - Application works with a DRAM allocated object



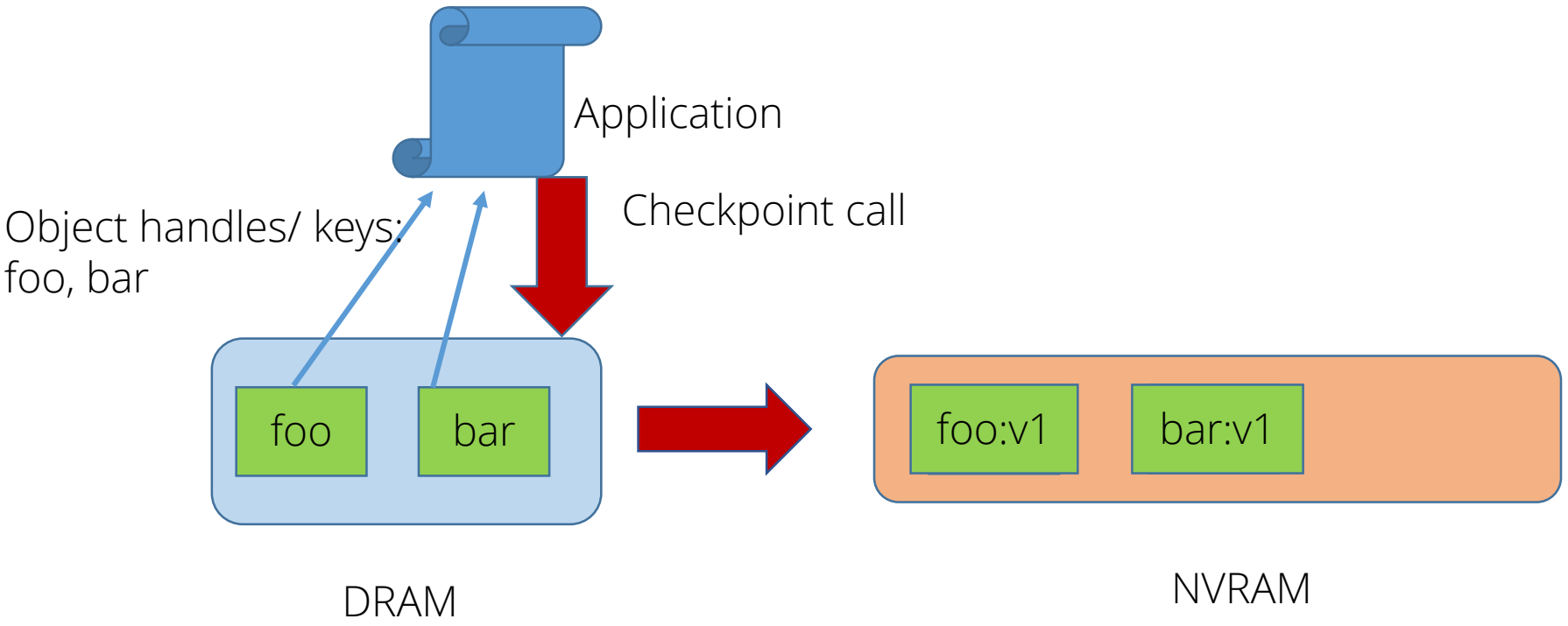
Problem : NVRAM Access Latencies

- NVRAM writes/ reads are expensive
  - Writes takes 4x more time than DRAM
  - Application works with a DRAM allocated object



Problem : NVRAM Access Latencies

- NVRAM writes/ reads are expensive
  - Writes takes 4x more time than DRAM
  - Application works with a DRAM allocated object



Resulting PHX C/R API

```
1 char key[]="foo";
2 /*create checkpointable object instance*/
3 struct properties prop = {.checkpoint = true, ...};
4 void *ptr=create_obj(key,SIZE,prop);
5 .
6 /*do computation using data_ptr*/
7 .
8 checkpoint_commit();
```

Resulting PHX C/R API

```
1  char key[]="foo";
2  /*create checkpointable object instance*/
3  struct properties prop = {.checkpoint = true, ...};
4  void *ptr=create_obj(key,SIZE,prop);
5  .
6  /*do computation using data_ptr*/
7  .
8  checkpoint_commit();
```

Allocate checkpoint object

Resulting PHX C/R API

```
1  char key[]="foo";
2  /*create checkpointable object instance*/
3  struct properties prop = {.checkpoint = true, ...};
4  void *ptr=create_obj(key,SIZE,prop);
5  .
6  /*do computation using data_ptr*/
7  .
8  checkpoint_commit();
```

Allocate checkpoint object

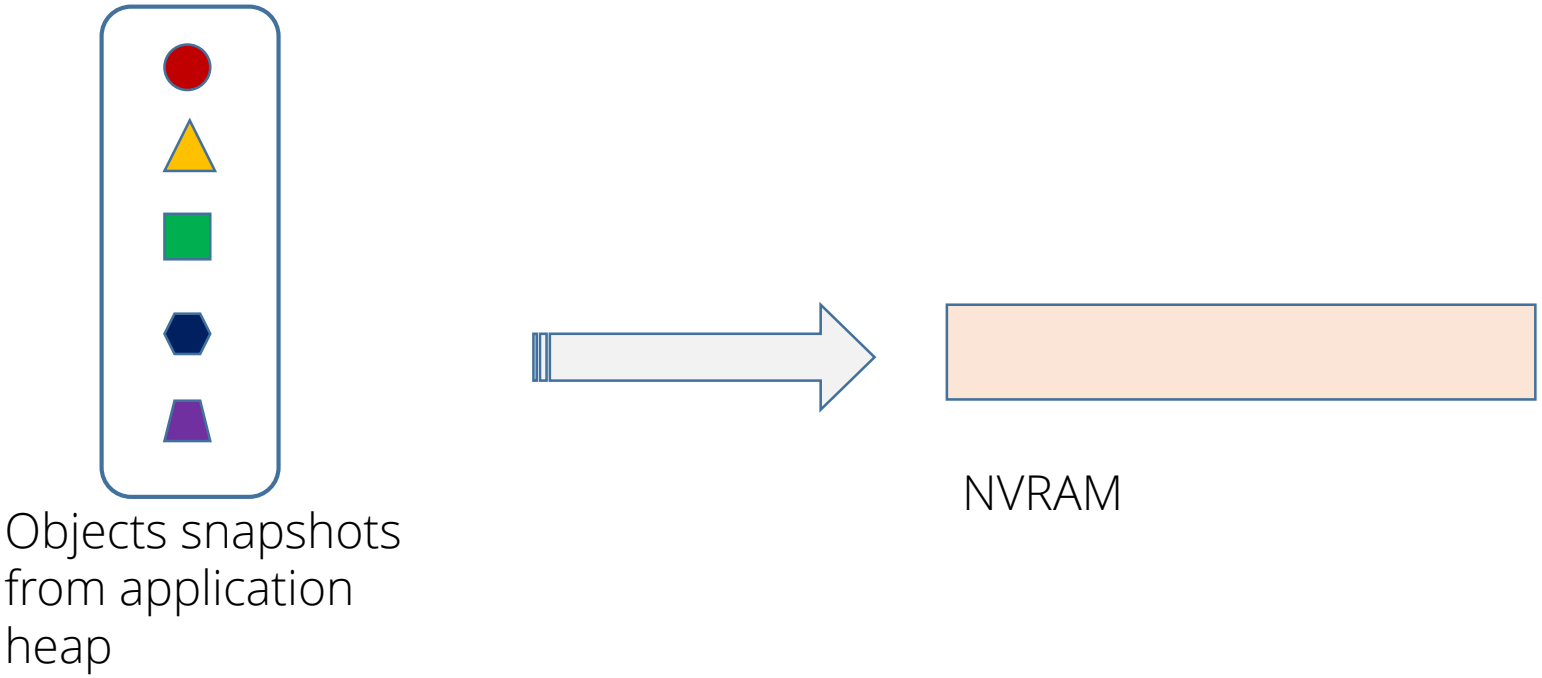
Create a versioned copy and move it to NVRAM

Problem: Limited NVRAM Bandwidth

- Simultaneous bandwidth usage of both NVRAM and DRAM

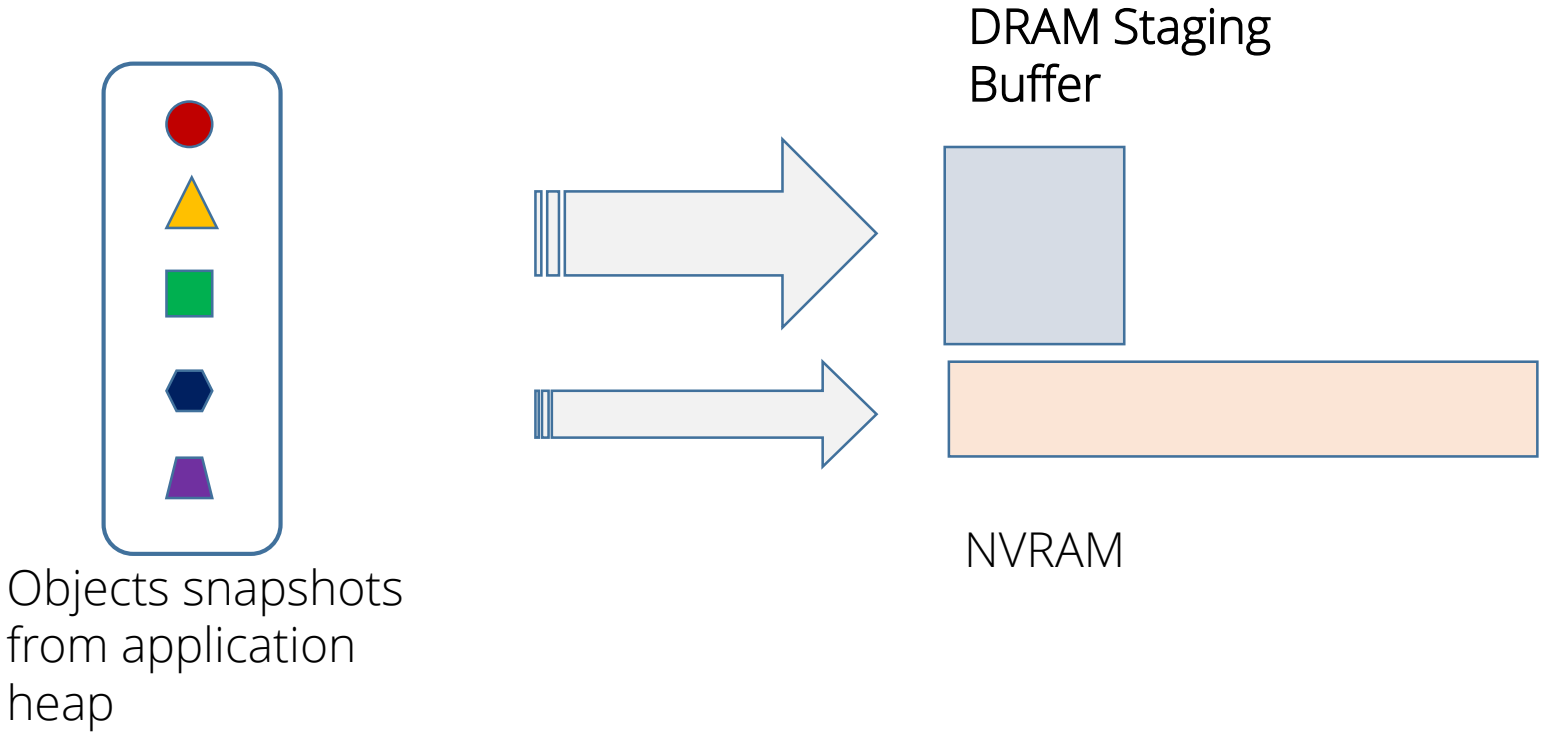
Problem: Limited NVRAM Bandwidth

- Simultaneous bandwidth usage of both NVRAM and DRAM



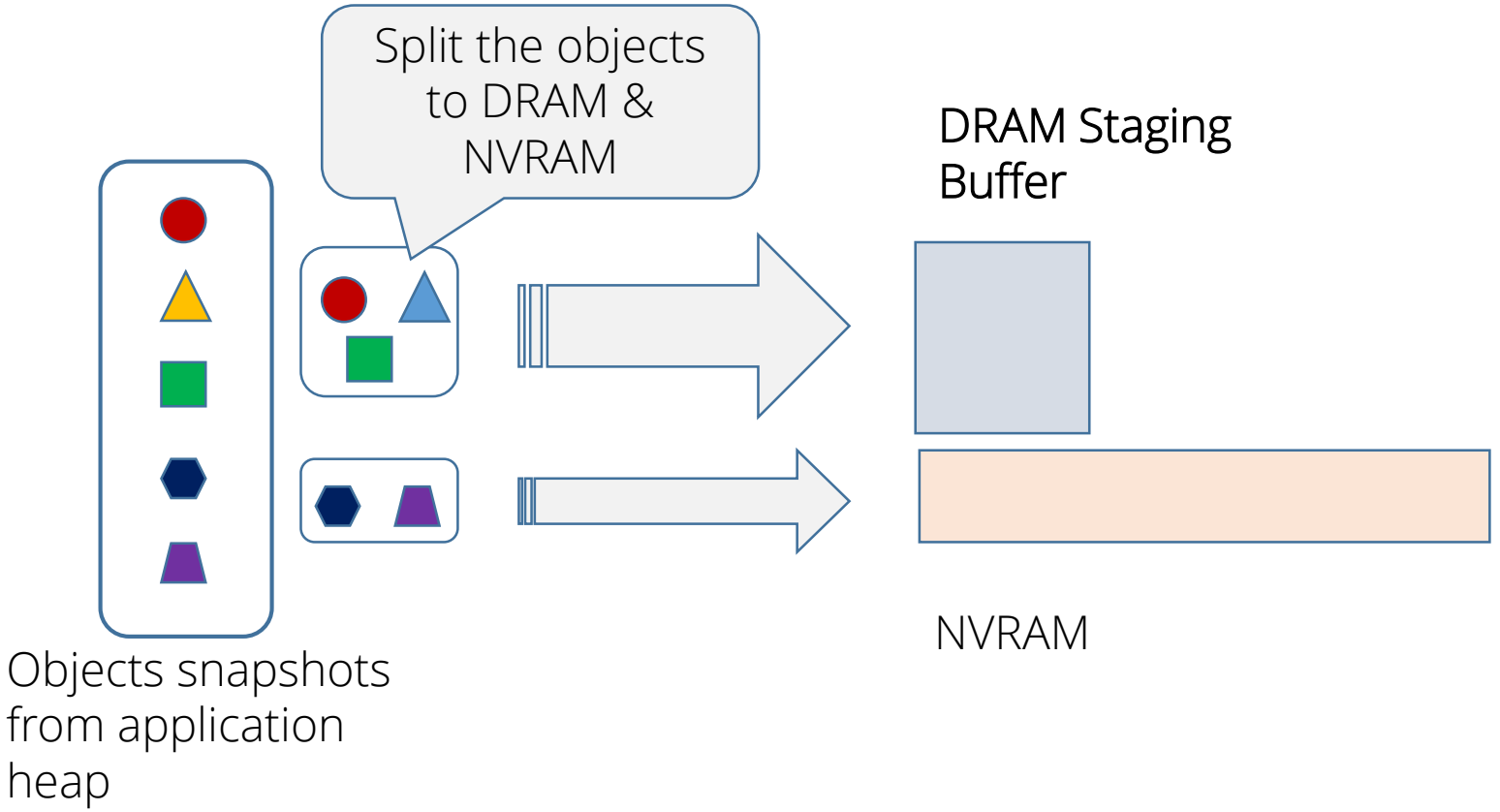
Problem: Limited NVRAM Bandwidth

- Simultaneous bandwidth usage of both NVRAM and DRAM



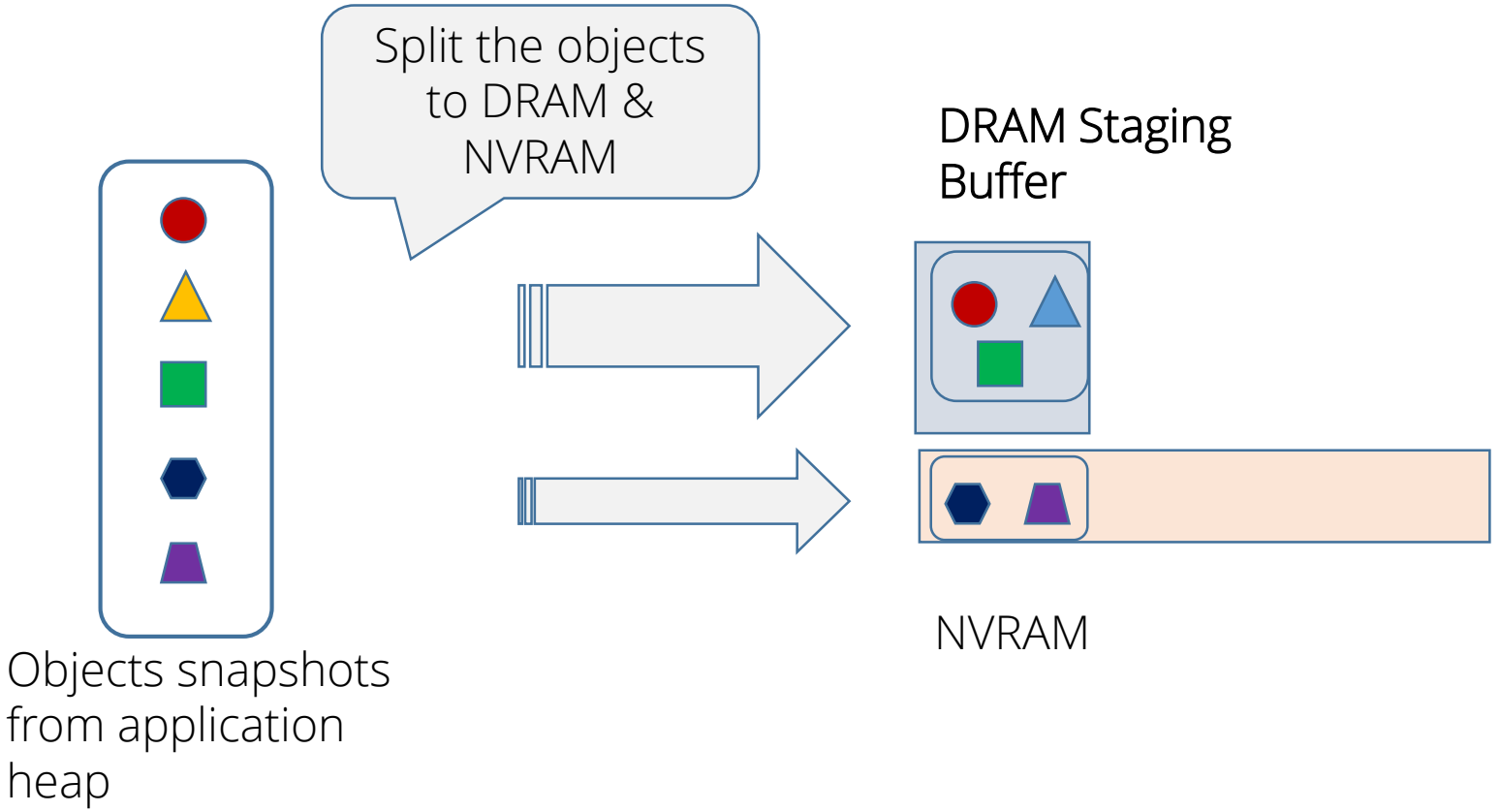
Problem: Limited NVRAM Bandwidth

- Simultaneous bandwidth usage of both NVRAM and DRAM



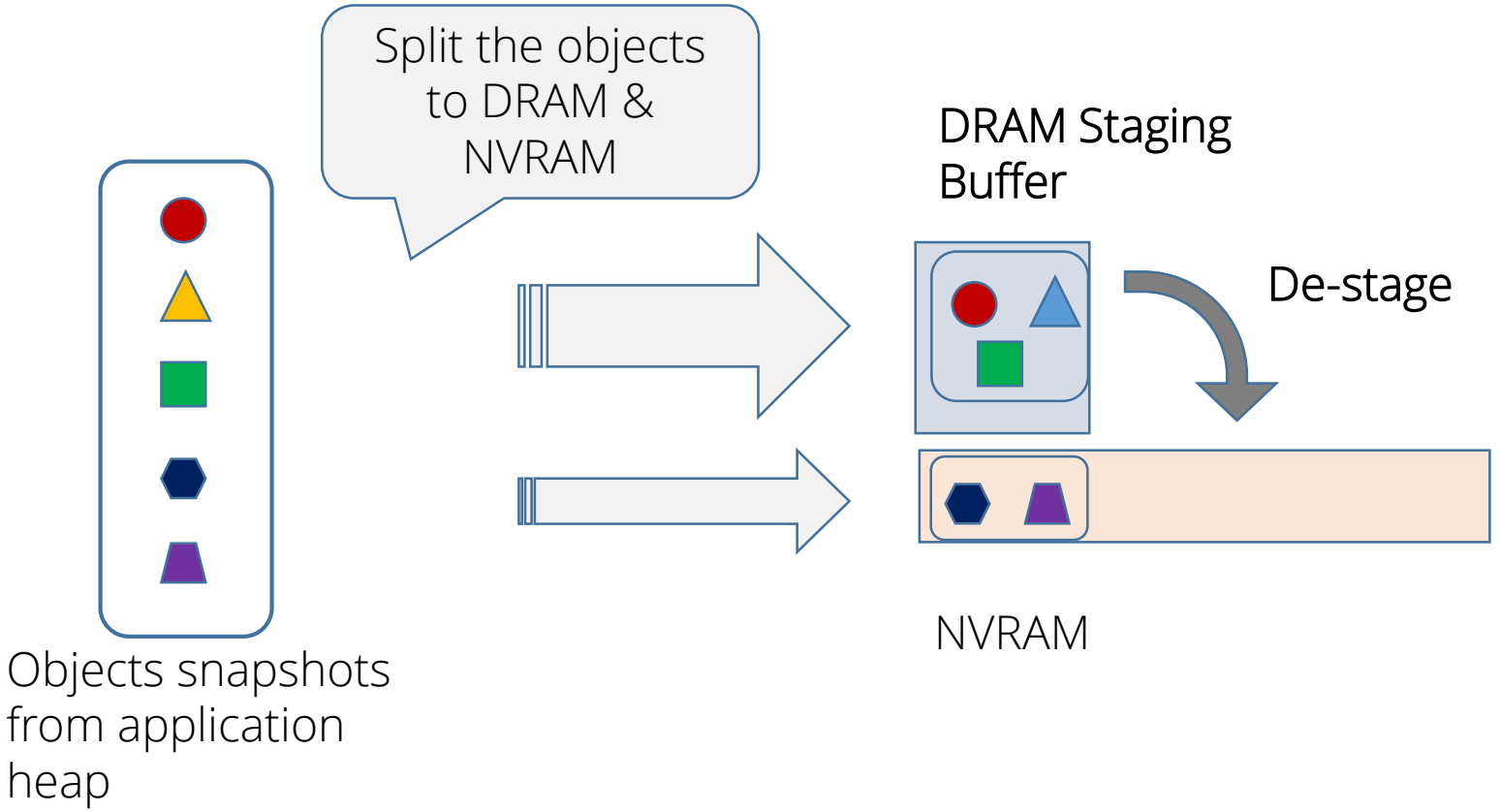
Problem: Limited NVRAM Bandwidth

- Simultaneous bandwidth usage of both NVRAM and DRAM



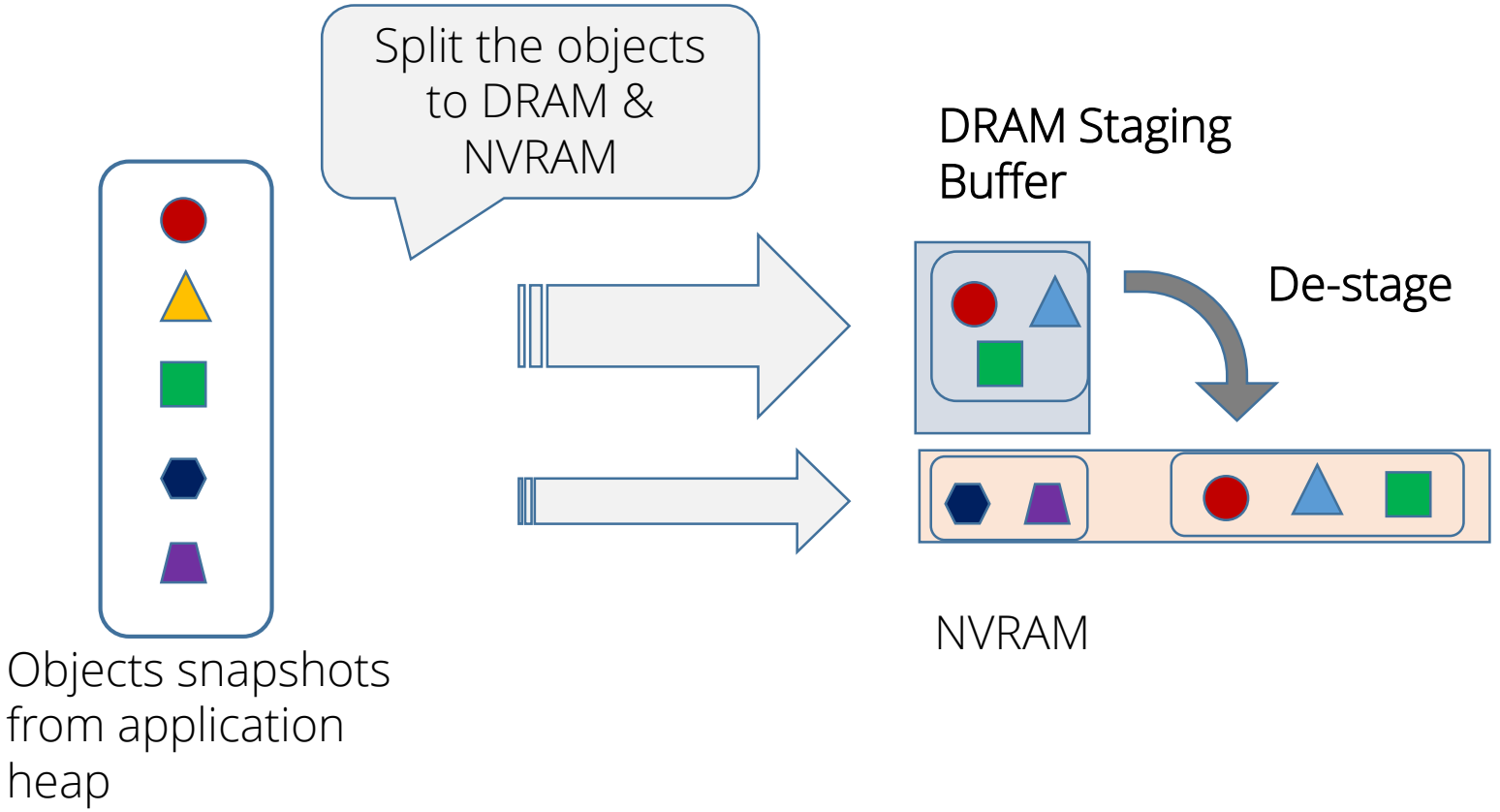
Problem: Limited NVRAM Bandwidth

- Simultaneous bandwidth usage of both NVRAM and DRAM



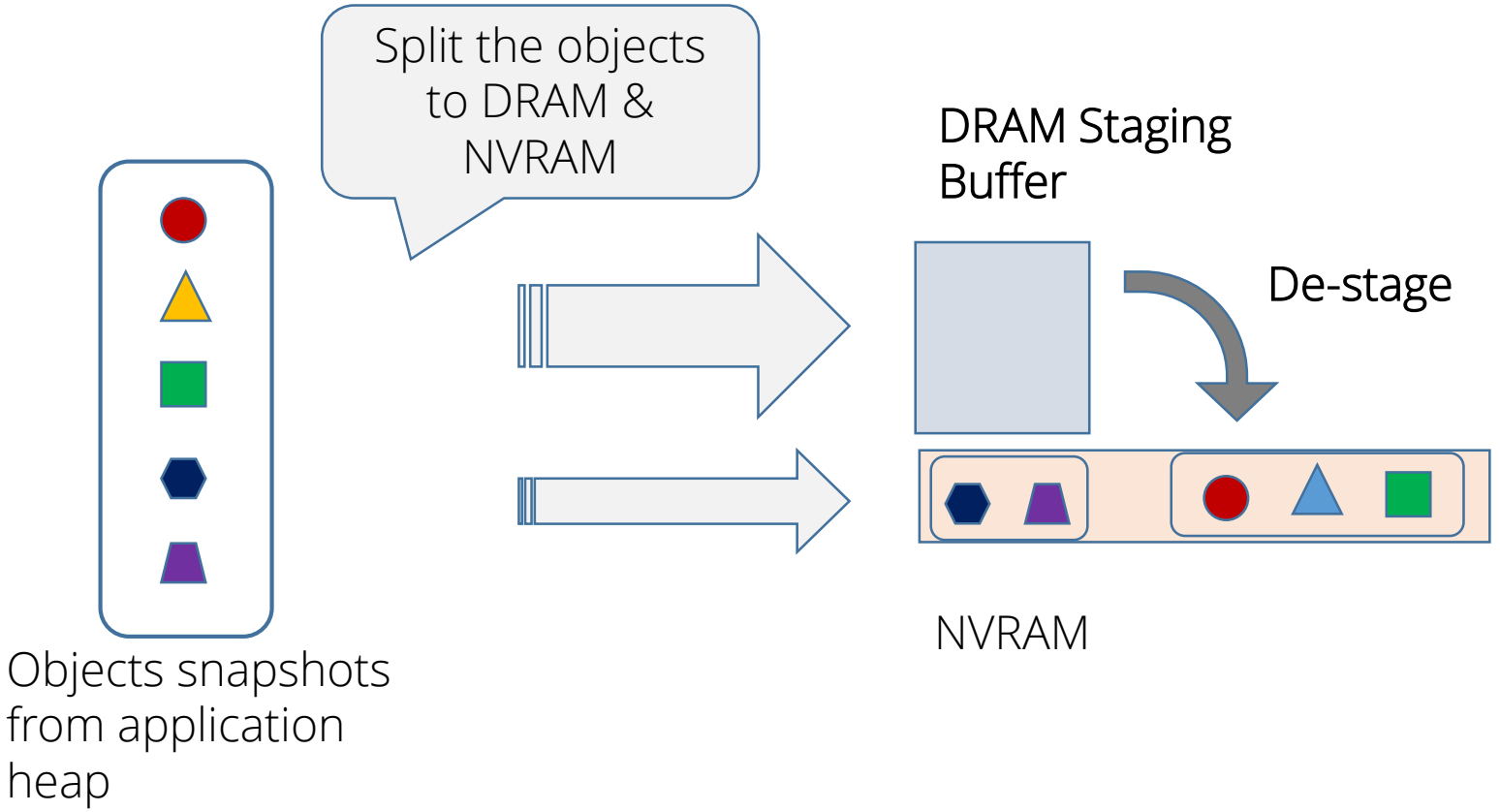
Problem: Limited NVRAM Bandwidth

- Simultaneous bandwidth usage of both NVRAM and DRAM

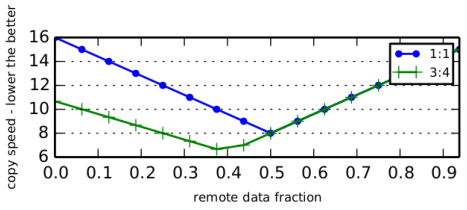


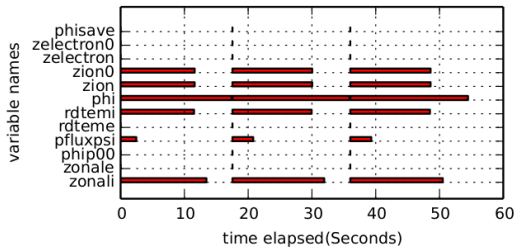
Problem: Limited NVRAM Bandwidth

- Simultaneous bandwidth usage of both NVRAM and DRAM



Checkpoint Data Split





1. Optimal data split ratio is same as the device bandwidth ration
2. Prioritize staging for late access variables – immune to optimizations (e.g. Pre-copy)
3. Memory budget allocated for staging.

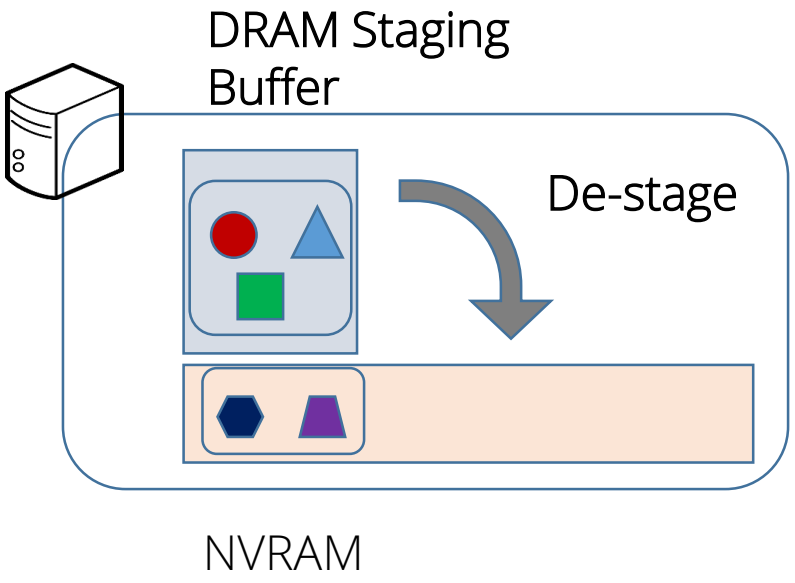
Fault Tolerance of Staged Data

Staging Buffer Data Loss

- Staged data is vulnerable to transient node failures.

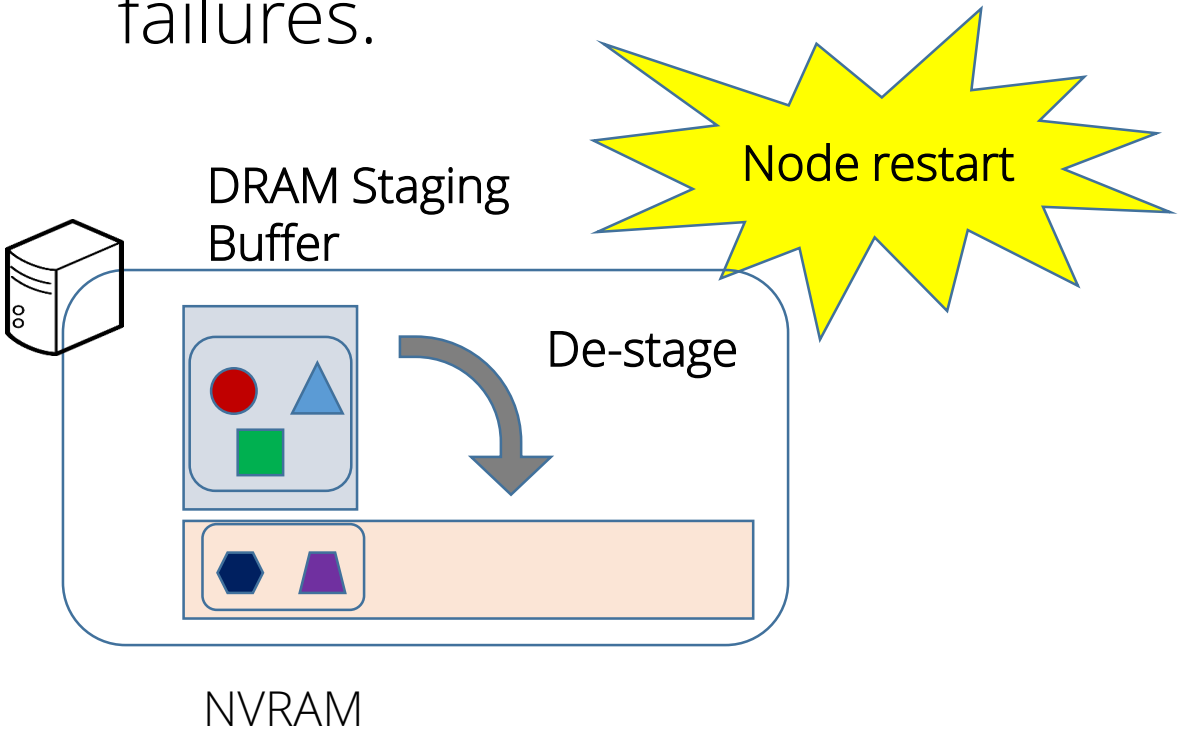
Staging Buffer Data Loss

- Staged data is vulnerable to transient node failures.



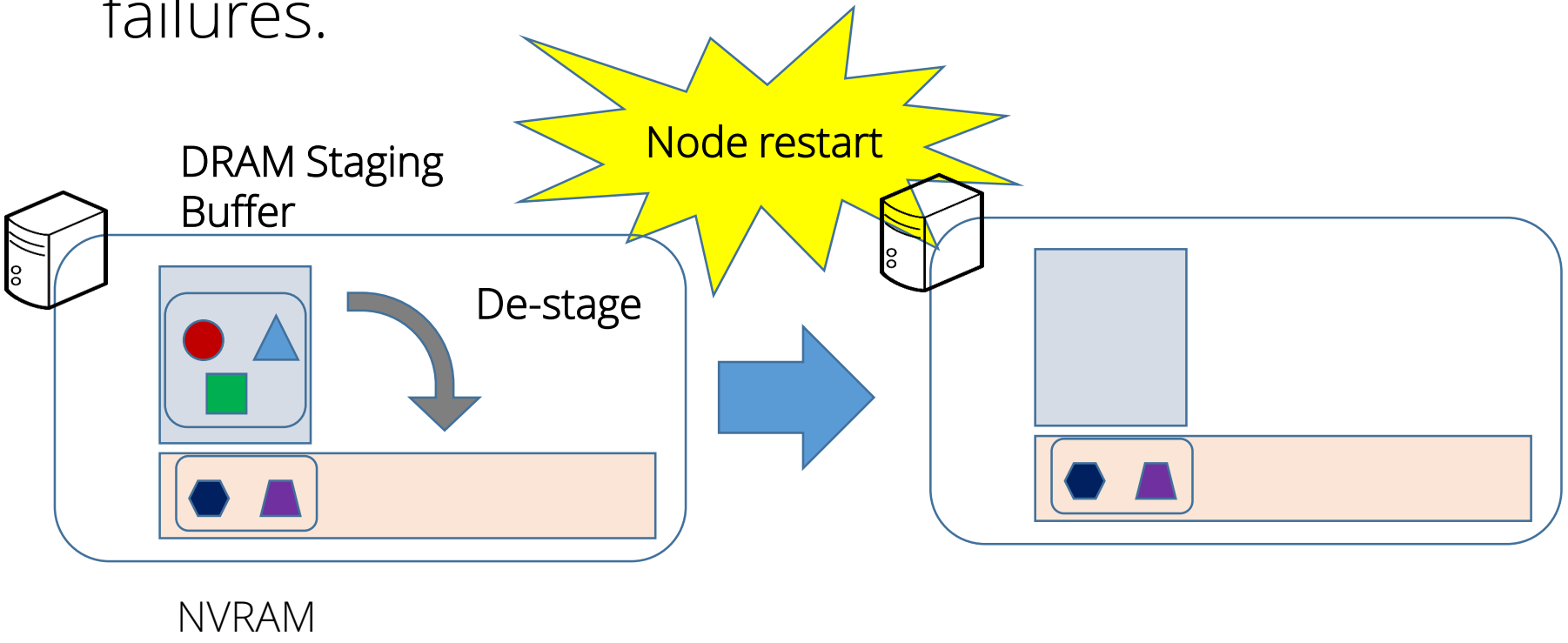
Staging Buffer Data Loss

- Staged data is vulnerable to transient node failures.



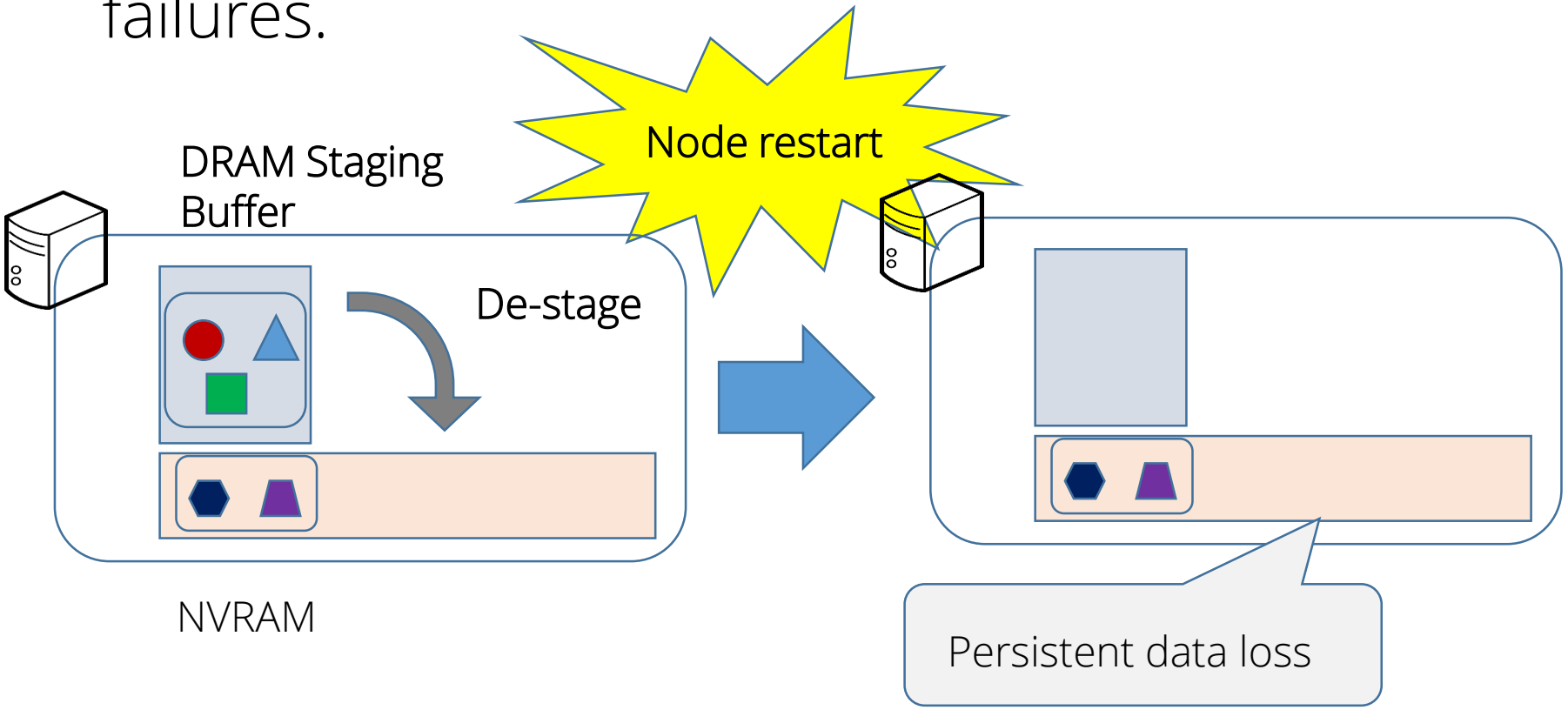
Staging Buffer Data Loss

- Staged data is vulnerable to transient node failures.

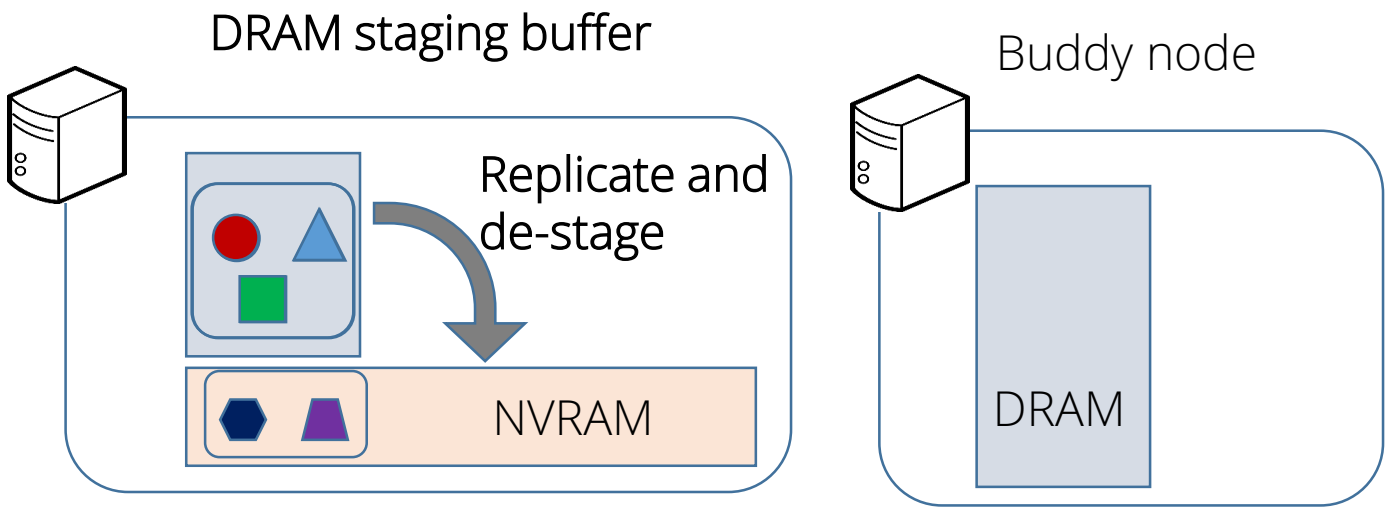


Staging Buffer Data Loss

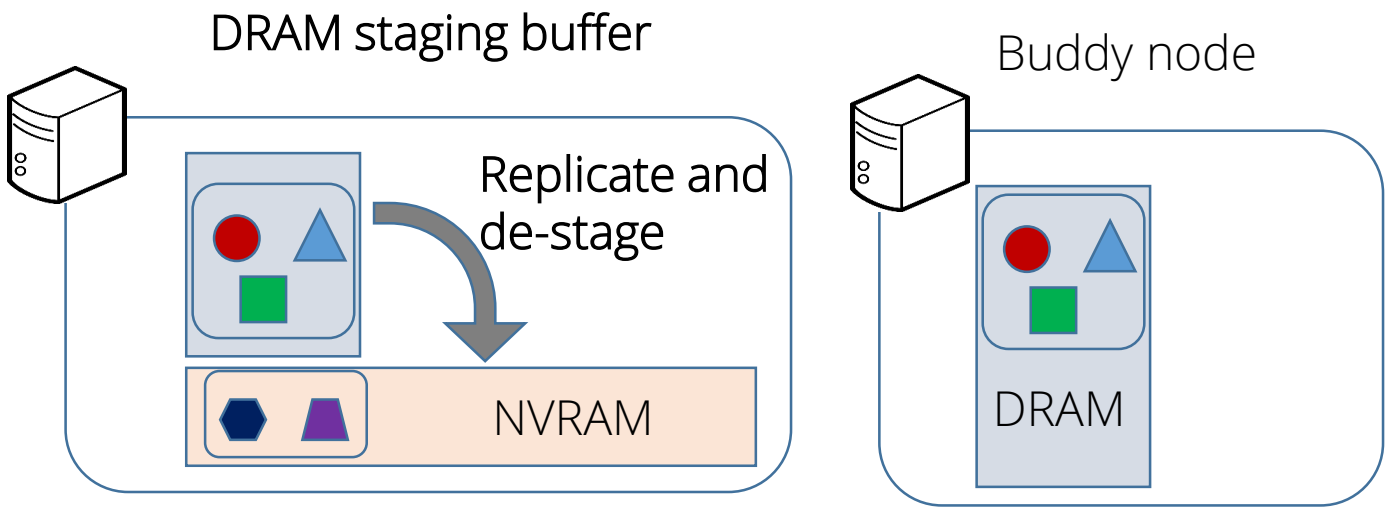
- Staged data is vulnerable to transient node failures.



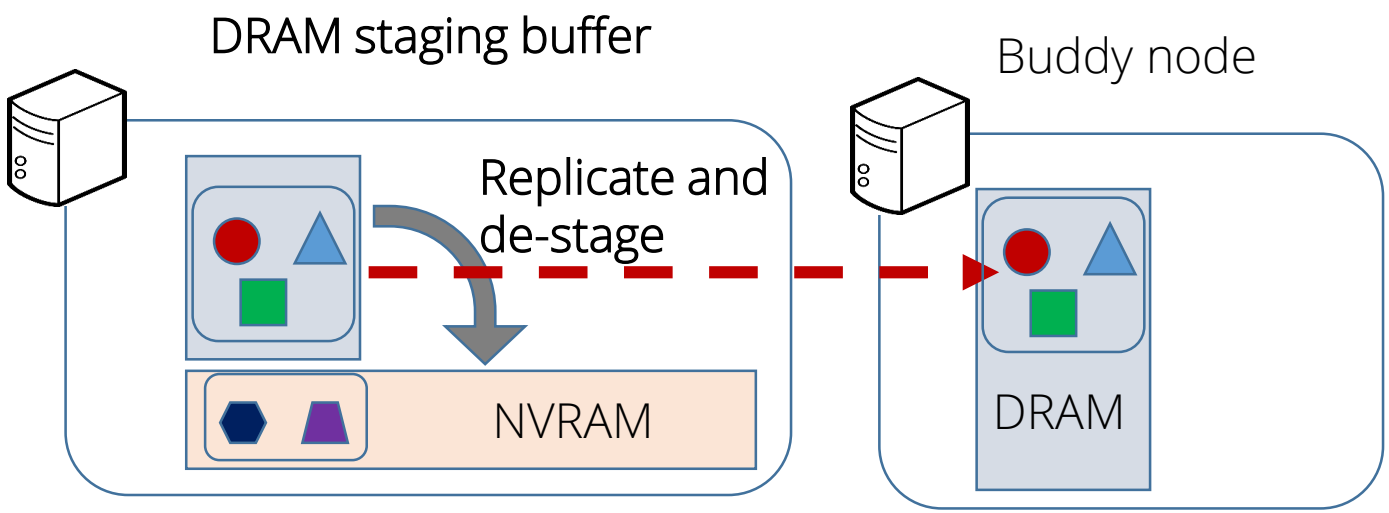
Buddy Node Replication (N=2)



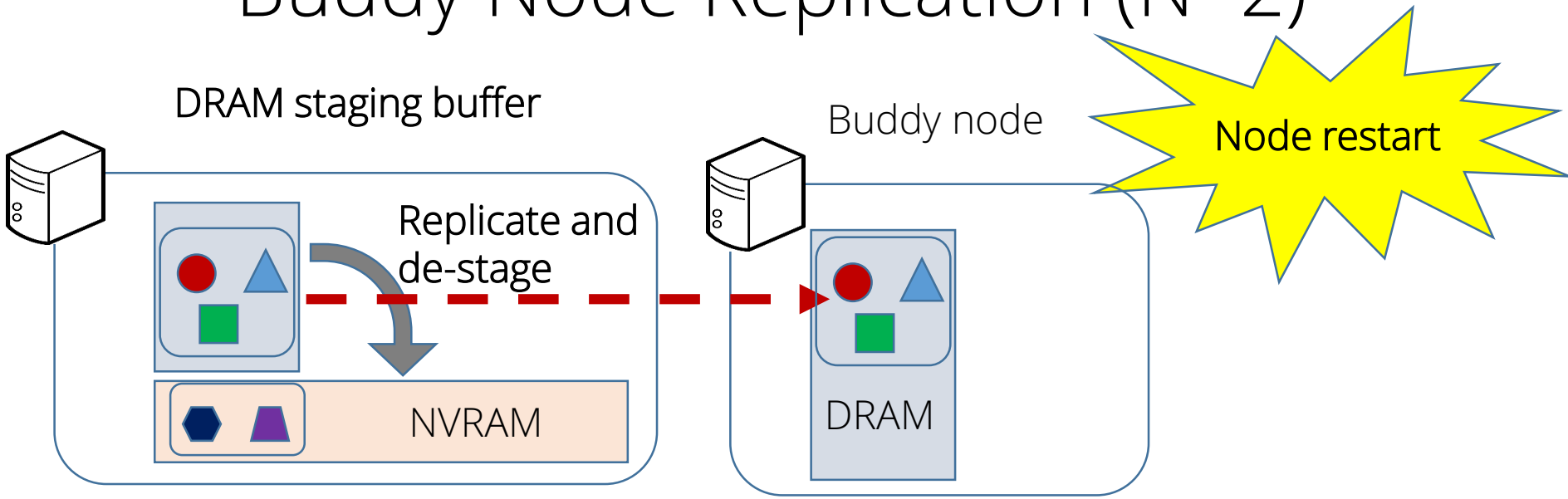
Buddy Node Replication (N=2)



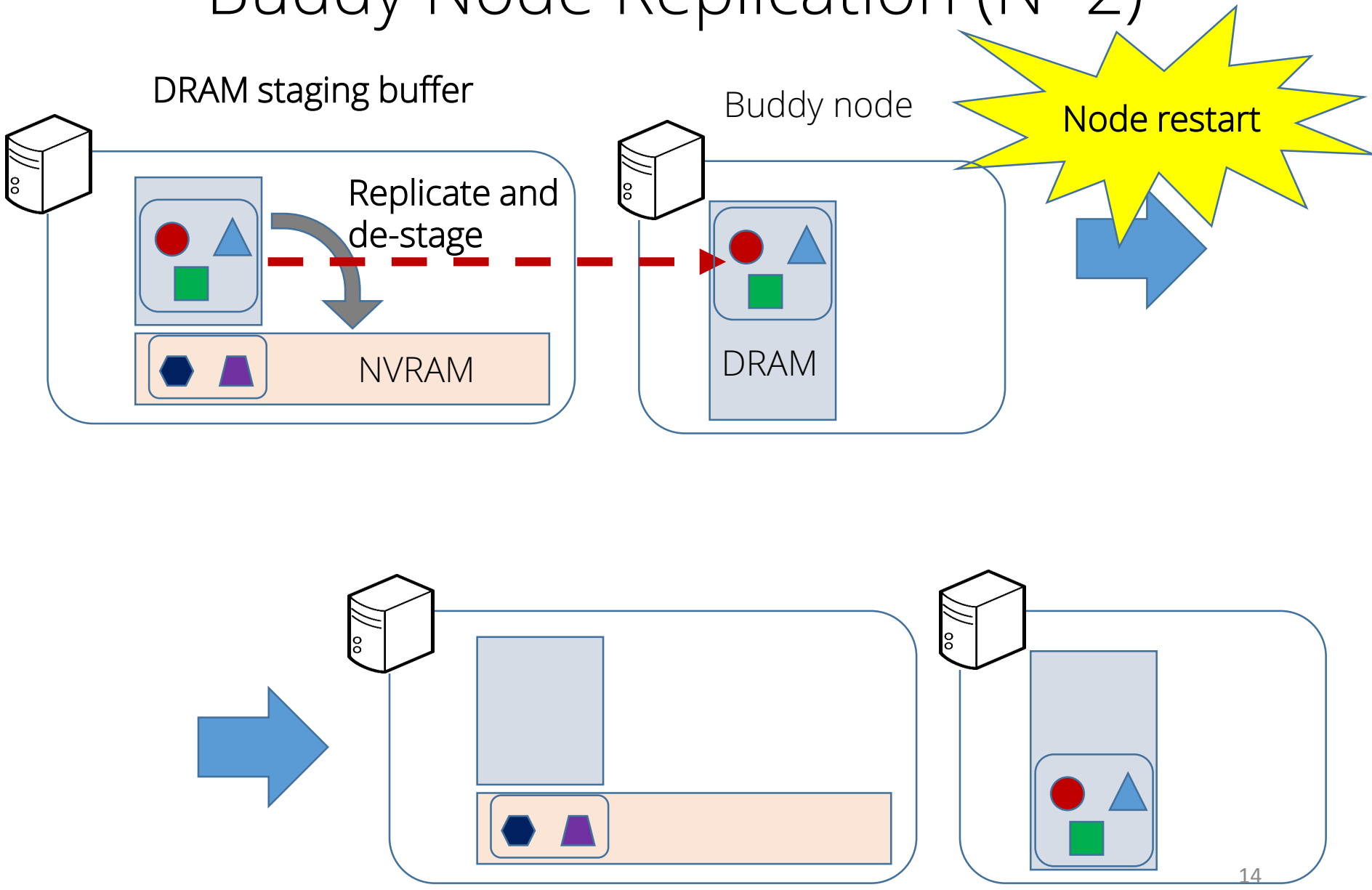
Buddy Node Replication (N=2)



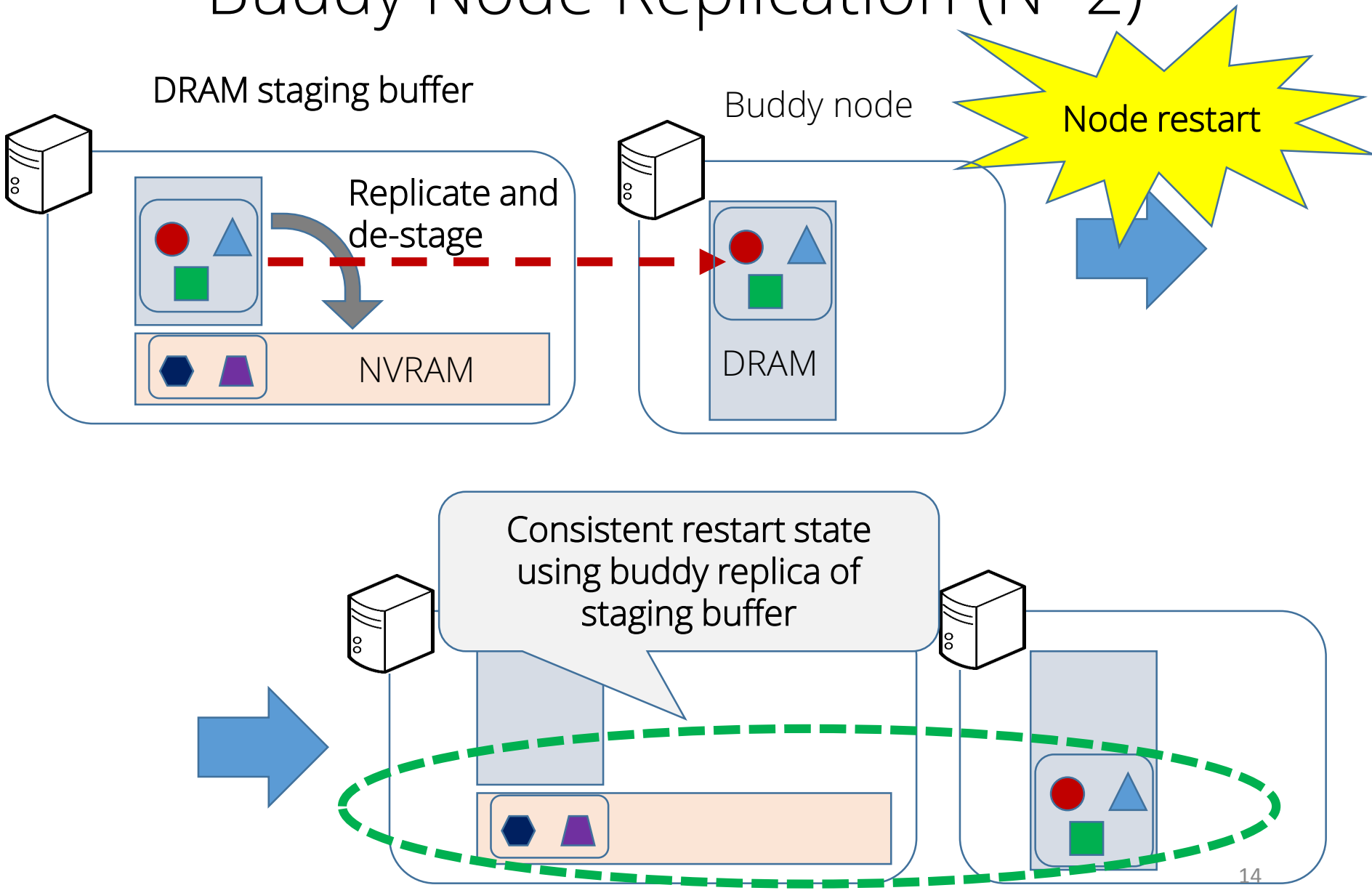
Buddy Node Replication (N=2)



Buddy Node Replication (N=2)



Buddy Node Replication (N=2)



Failure Probability of Staged Data

Scheme	Execution time	Checkpoint interval	Max de-stage interval	Failure probability
Free run	24 hrs	-	-	0.99999887
PHX, N=1	48 hrs	60 s	10 s	0.98960270
PHX, N=2	48 hrs	60 s	10 s	0.00000003

Failure Probability of Staged Data

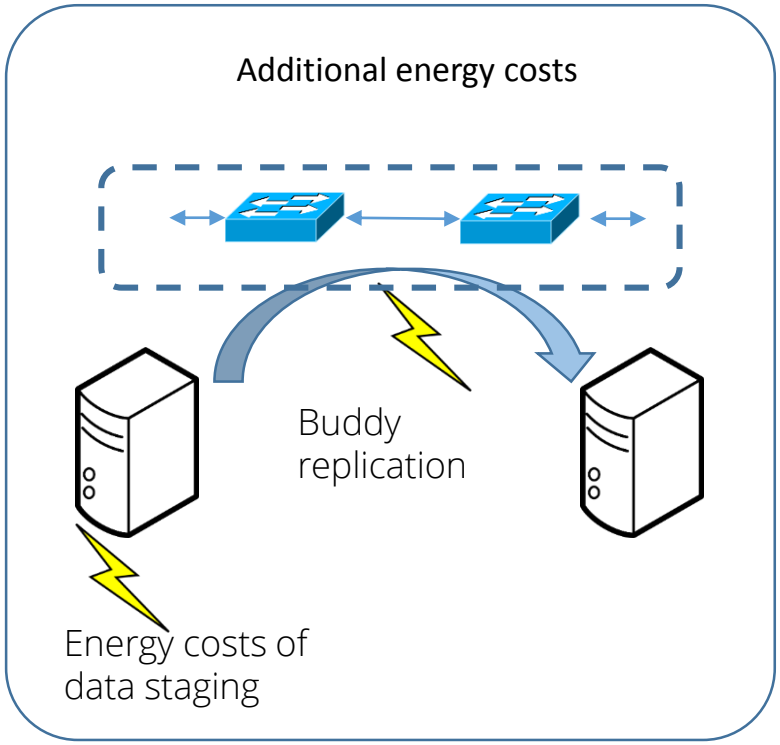
Scheme	Execution time	Checkpoint interval	Max de-stage interval	Failure probability
Free run	24 hrs	-	-	0.99999887
PHX, N=1	48 hrs	60 s	10 s	0.98960270
PHX, N=2	48 hrs	60 s	10 s	0.00000003

- N=2, buddy replication scheme brings down the staged data loss probability to negligible level.

Energy Cost Analysis

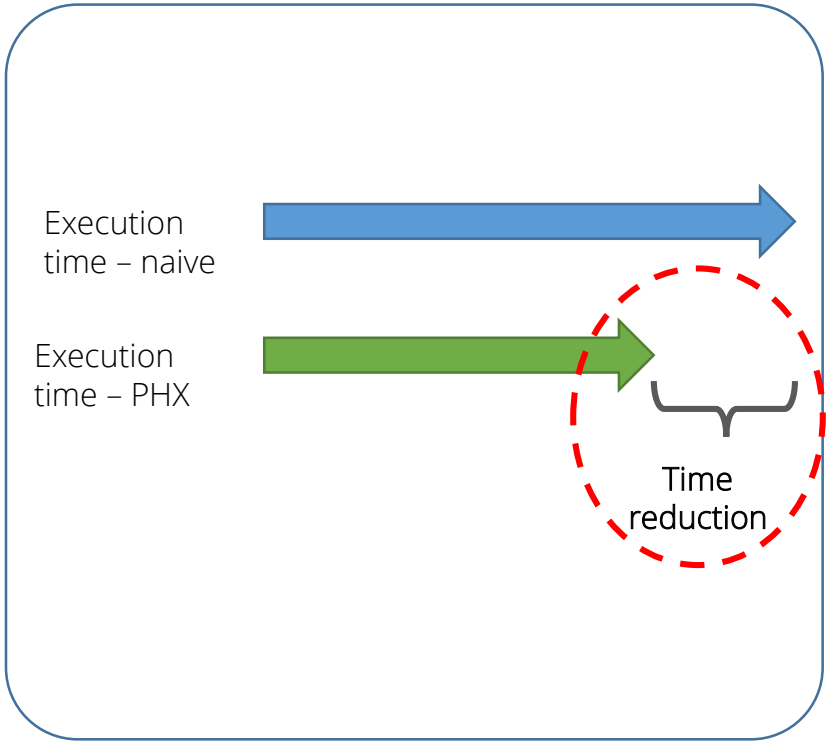
Energy Overheads of PHX

- PHX energy overheads
  - Checkpoint staging
  - Stage buffer replication
- Depends on
  - Interconnect technology
  - Distance to buddy
  - DRAM write costs

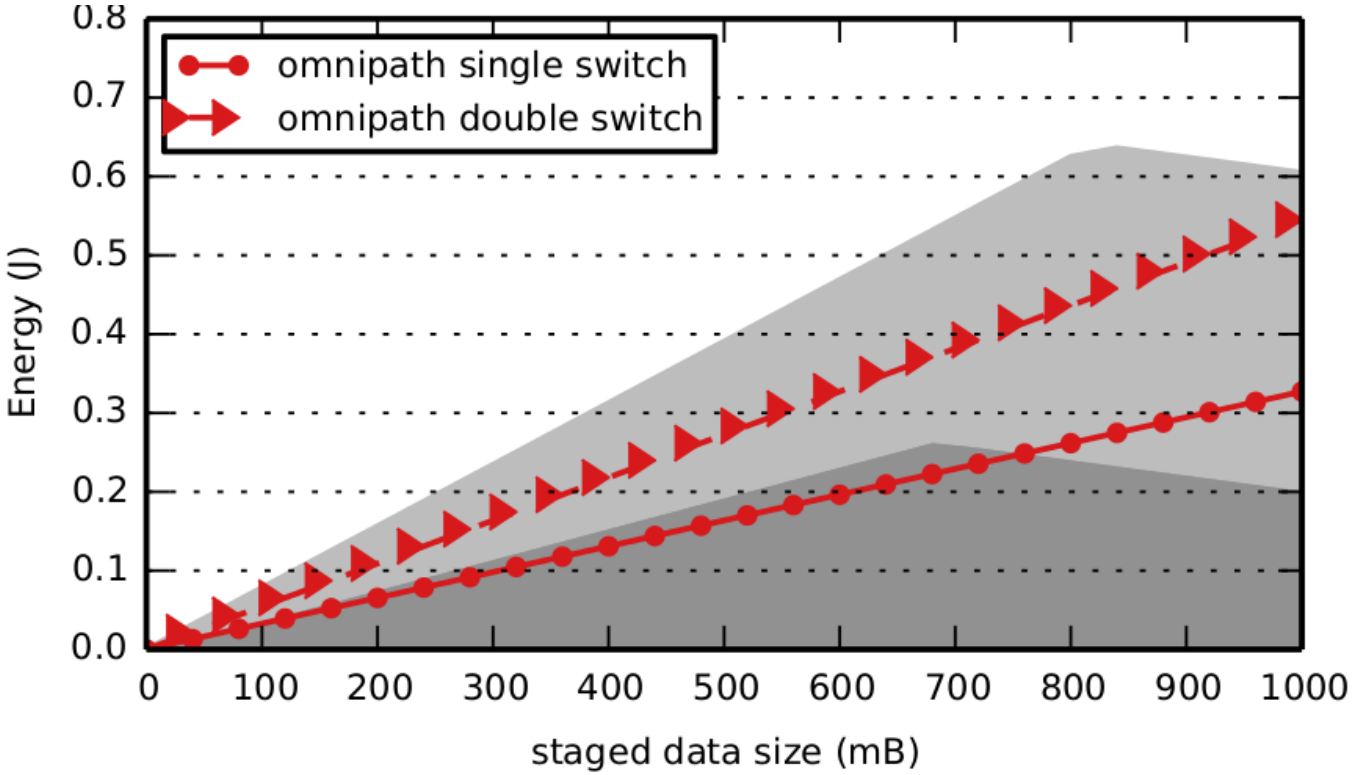


Energy Savings of PHX

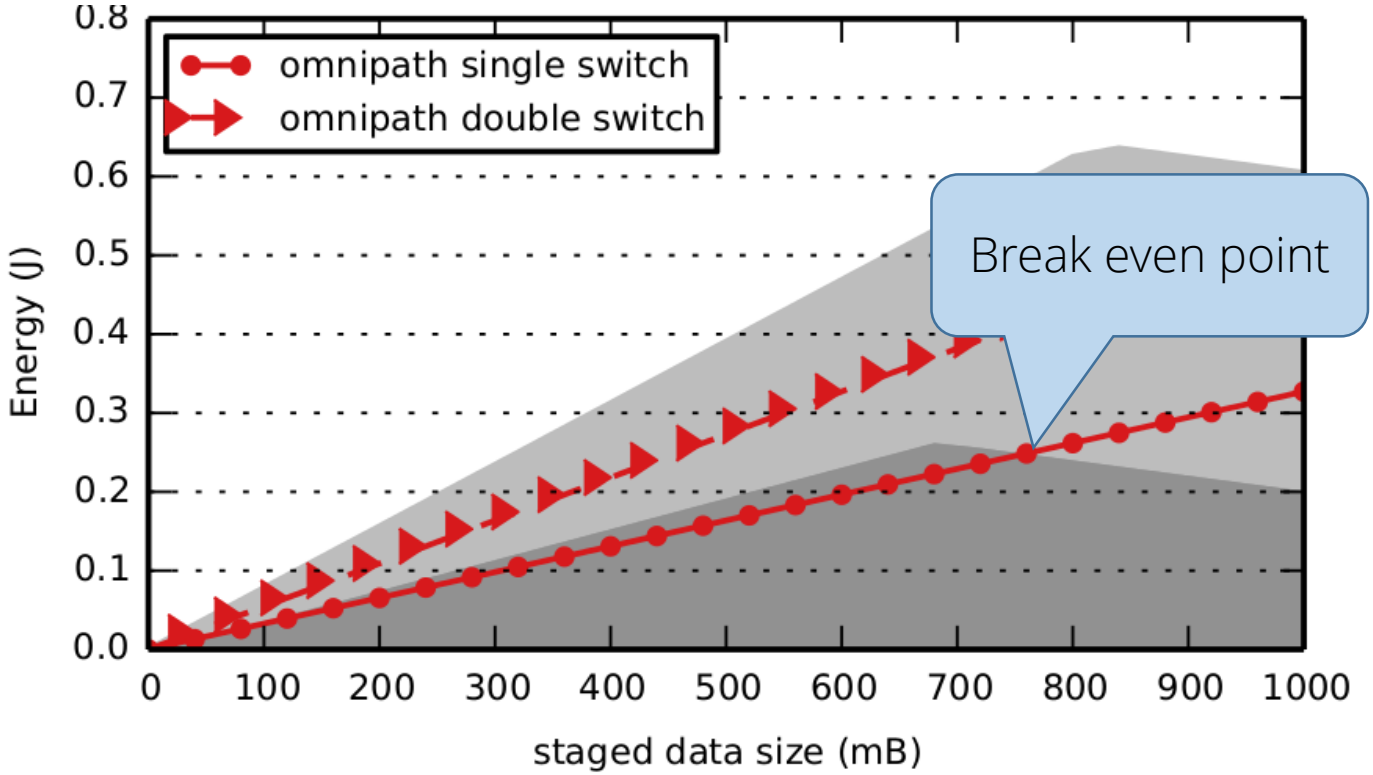
- Fast PHX checkpoints cut down the total simulation time.
- Energy savings!



PHX Saves Energy

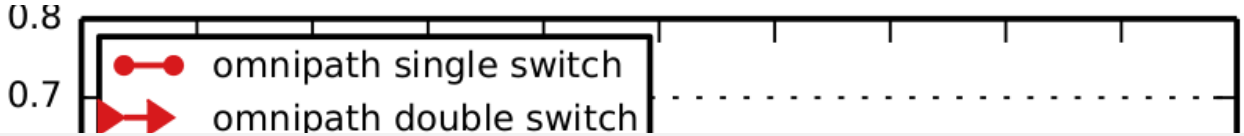


PHX Saves Energy



- For DRAM:NVRAM bandwidth ratio of 2:1 PHX can stage up to ~750 MB (total checkpoint size is 1GB) when the buddy node is one switch away

PHX Saves Energy



Careful selection of PHX configuration parameters will lead to both fast checkpoints and energy savings.

- For DRAM:NVRAM bandwidth ratio of 2:1 PHX can stage up to ~750 MB (total checkpoint size is 1GB) when the buddy node is one switch away

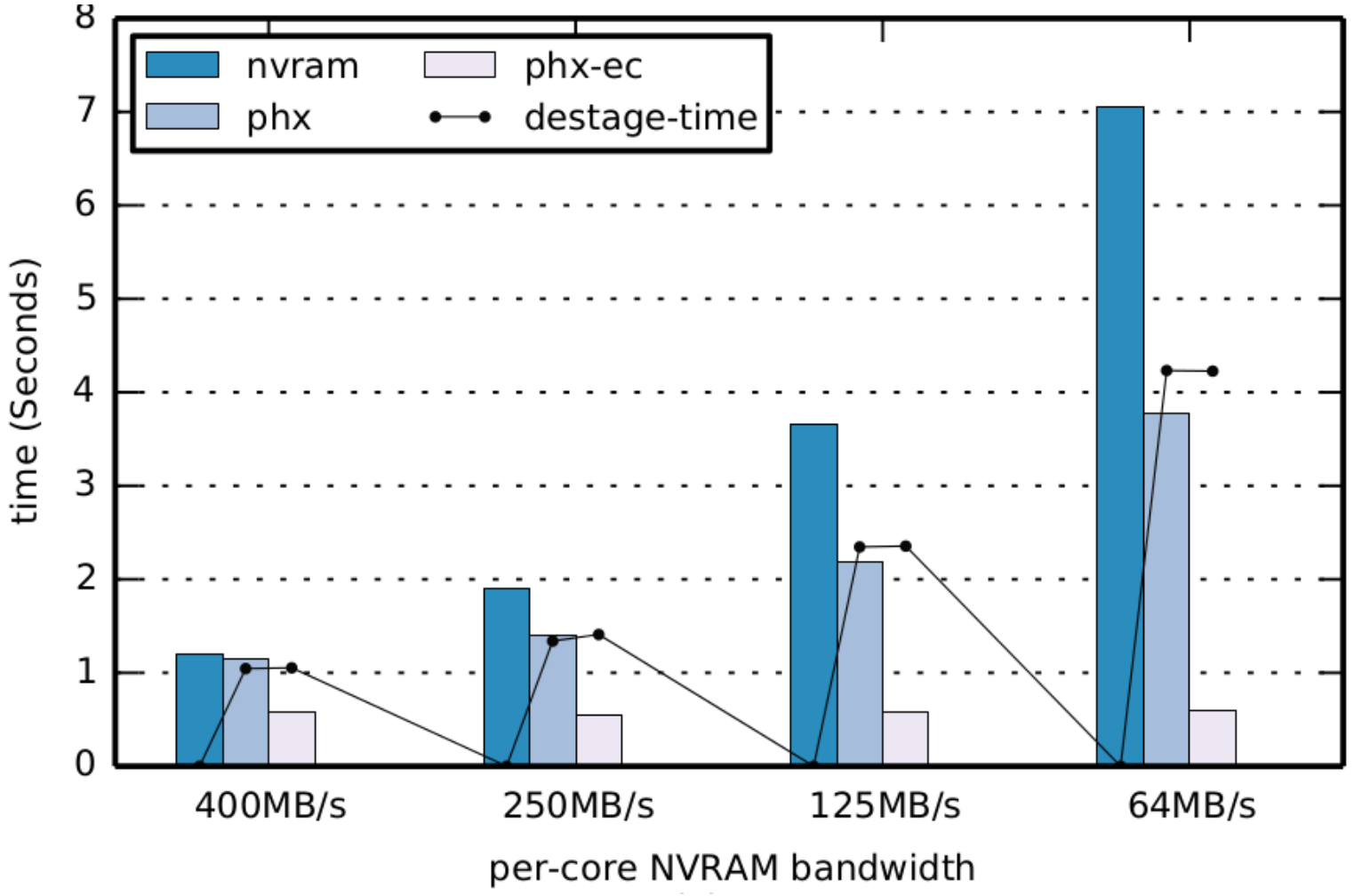
Evaluation

- Four node cluster run in the stampede supercomputer
- Emulated NVRAM using software delays
- Three application benchmarks
  - GTC
  - CM1
  - S3D

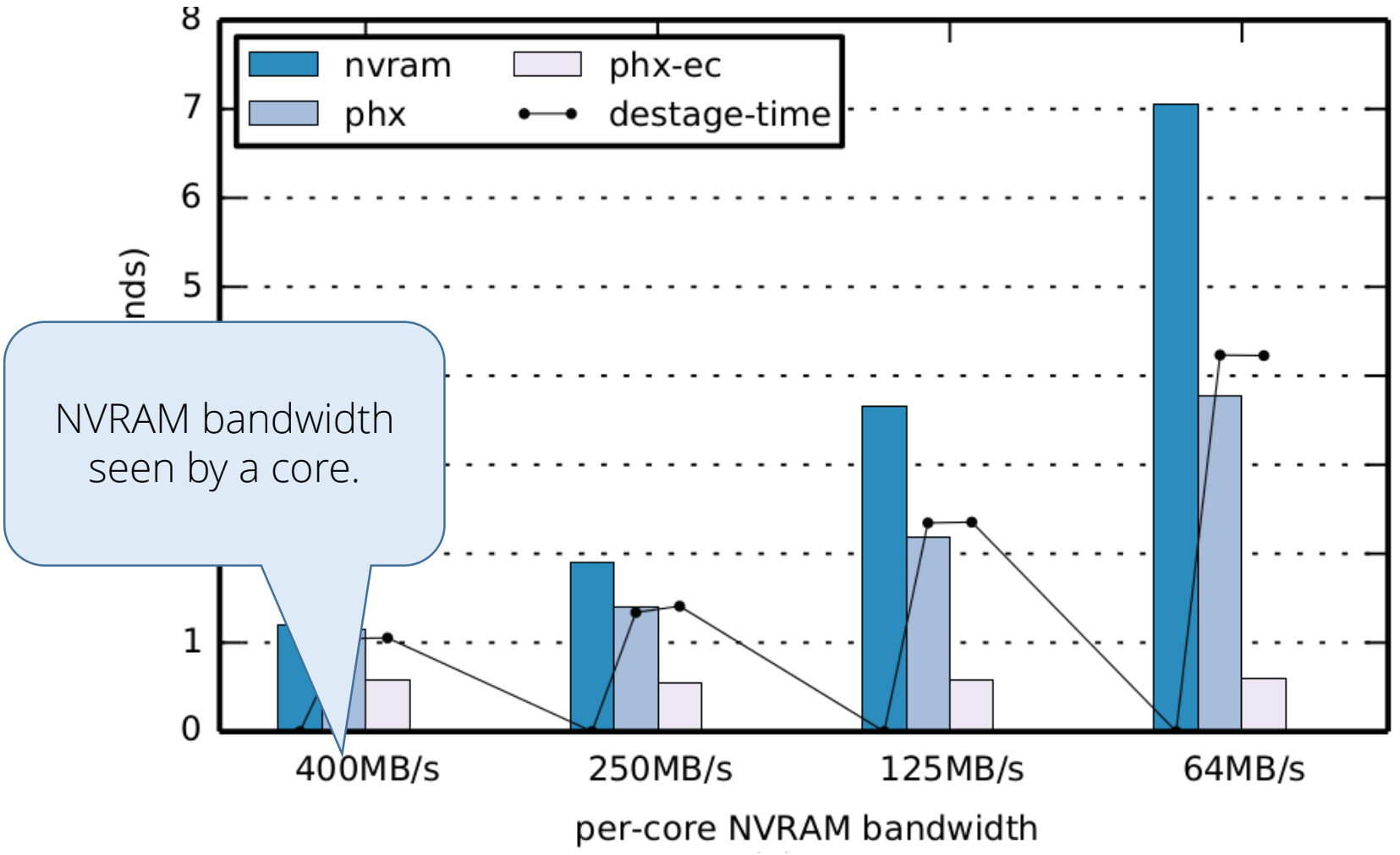
GTC Benchmark

- Checkpoint data-intensive application
- Staging buffer size -- 50% of checkpoint data per interval
- Lot of early access variables

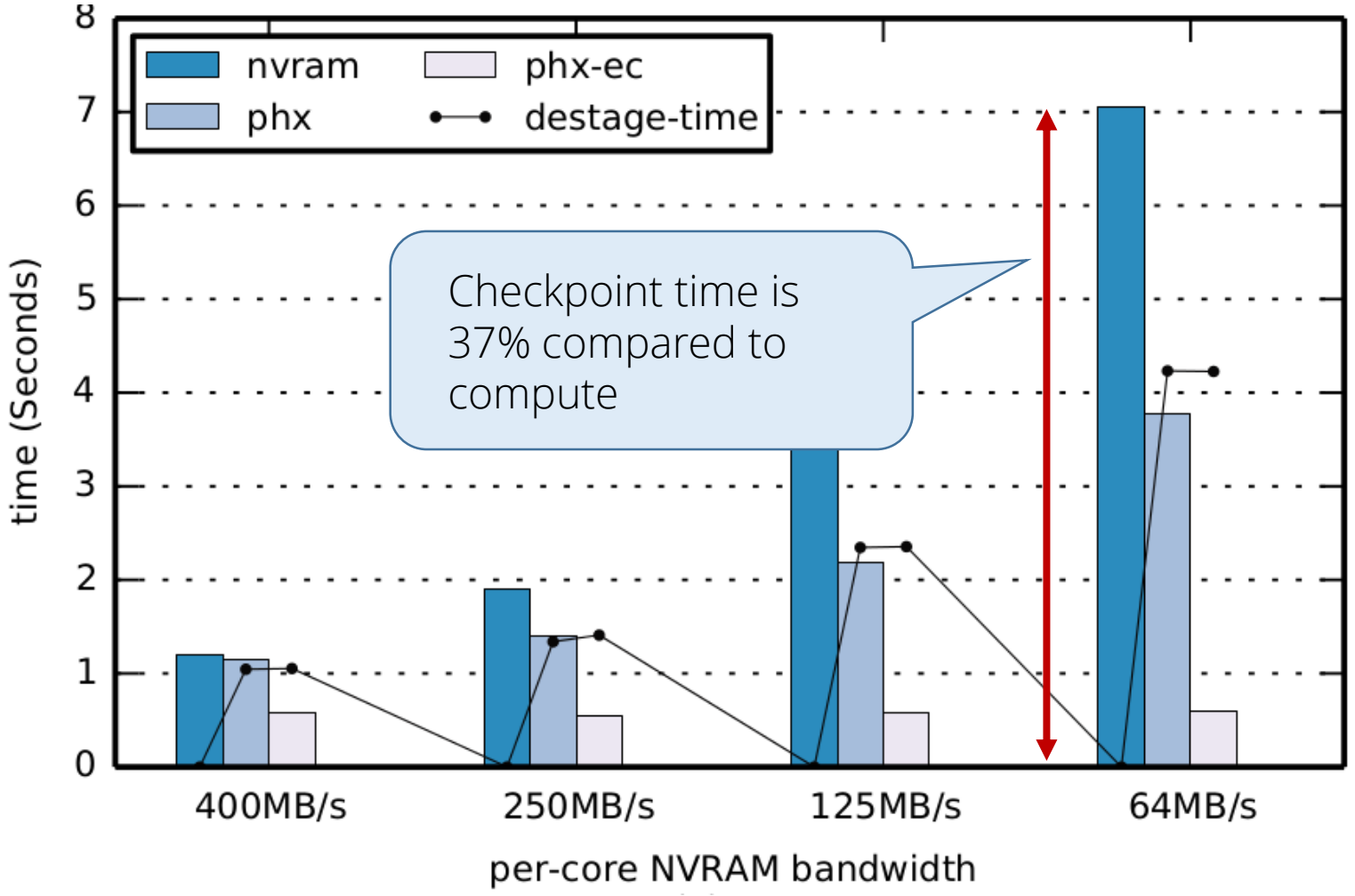
GTC Benchmark



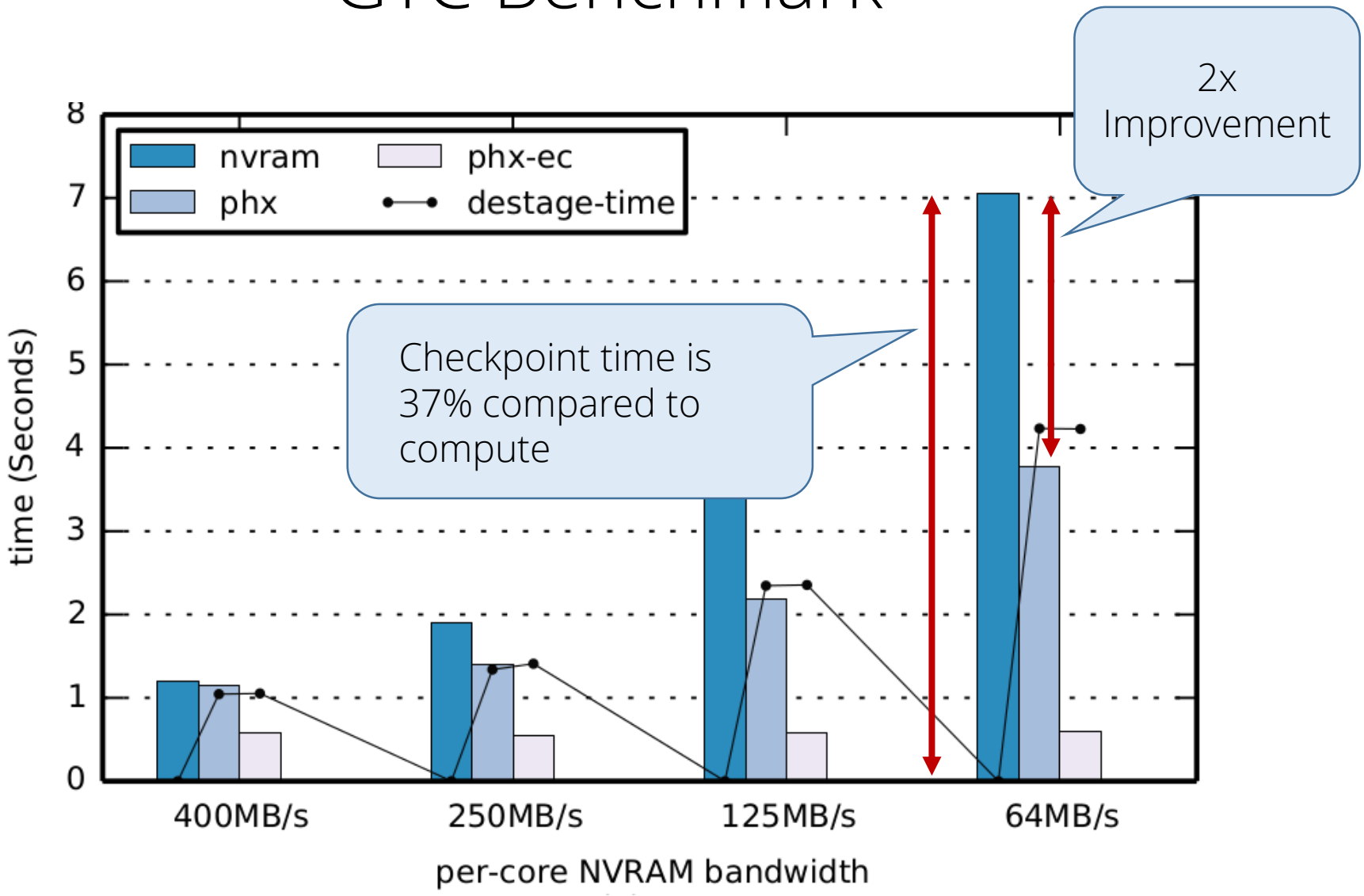
GTC Benchmark



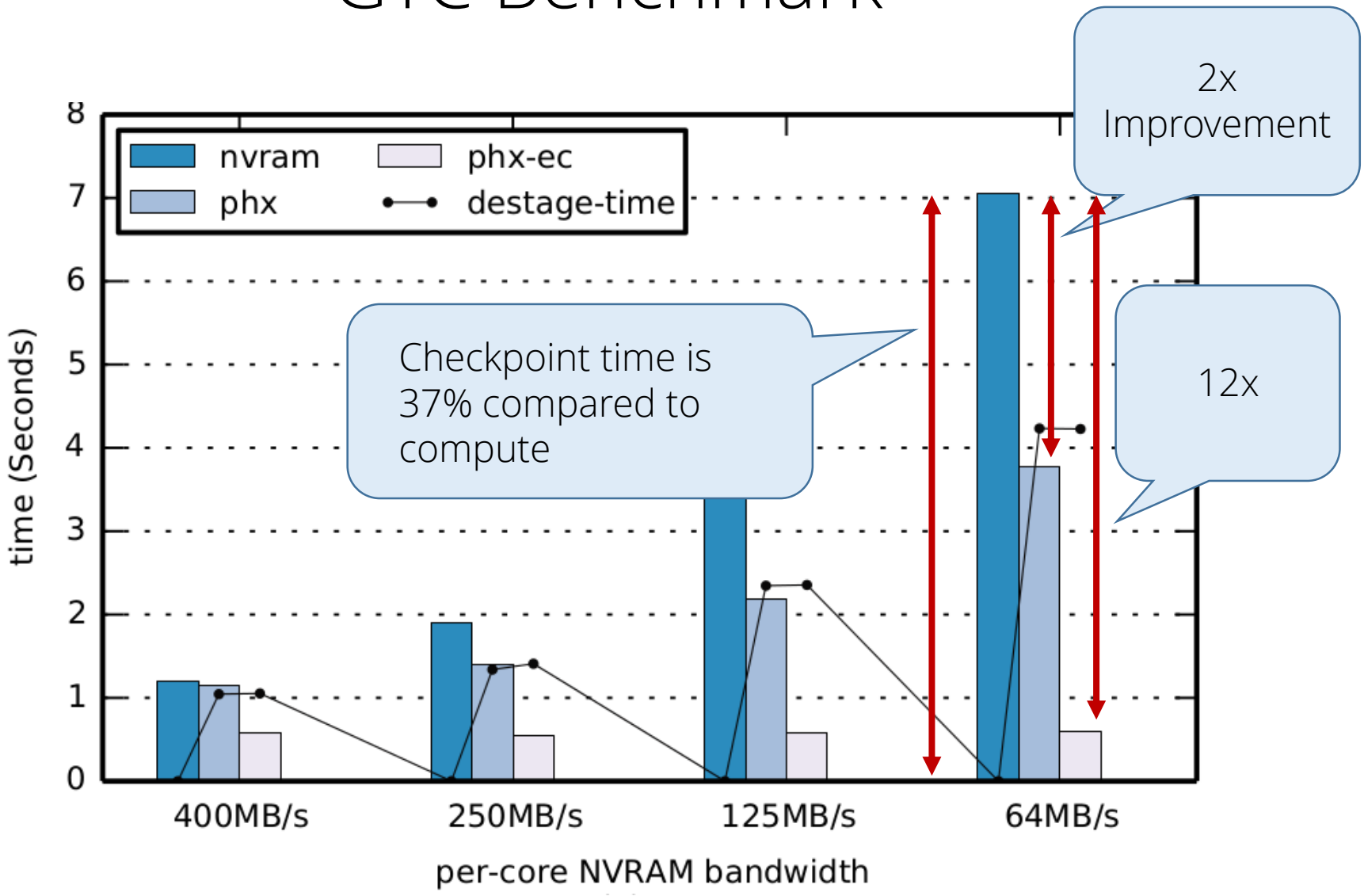
GTC Benchmark



GTC Benchmark



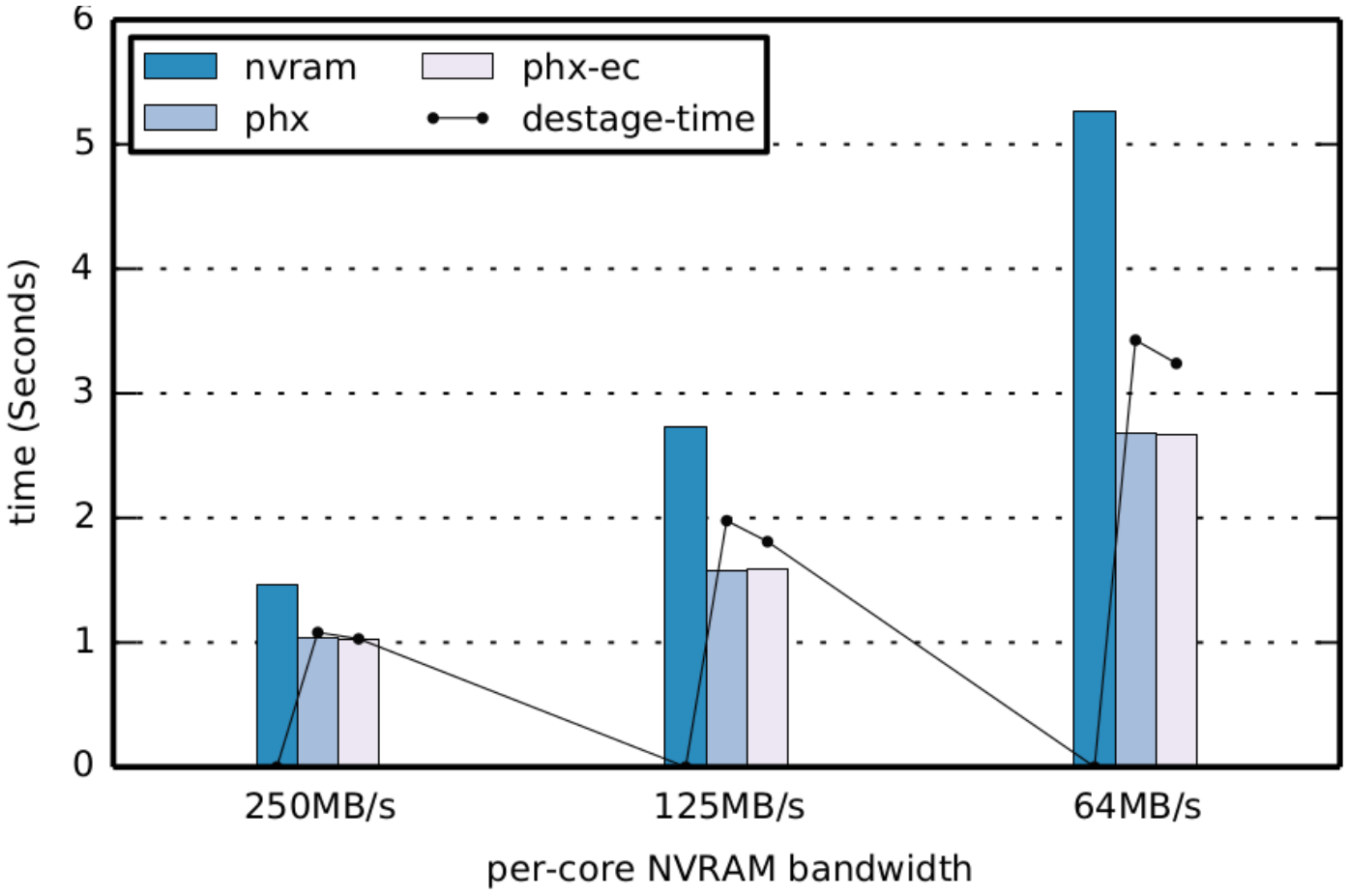
GTC Benchmark



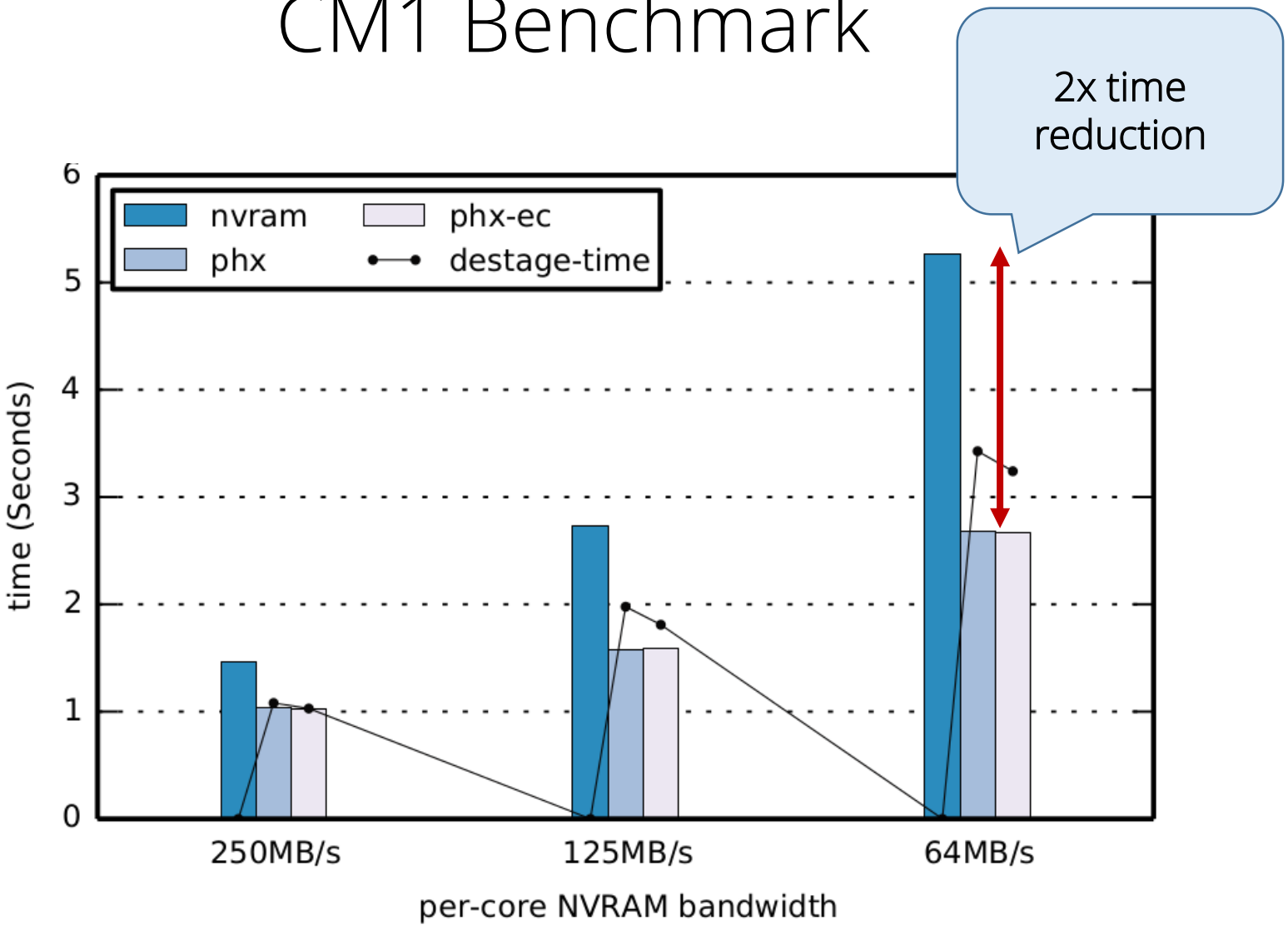
CM1 Benchmark

- Compute heavy application
- Staging buffer = 50% checkpoint data per interval
- No early access variables

CM1 Benchmark

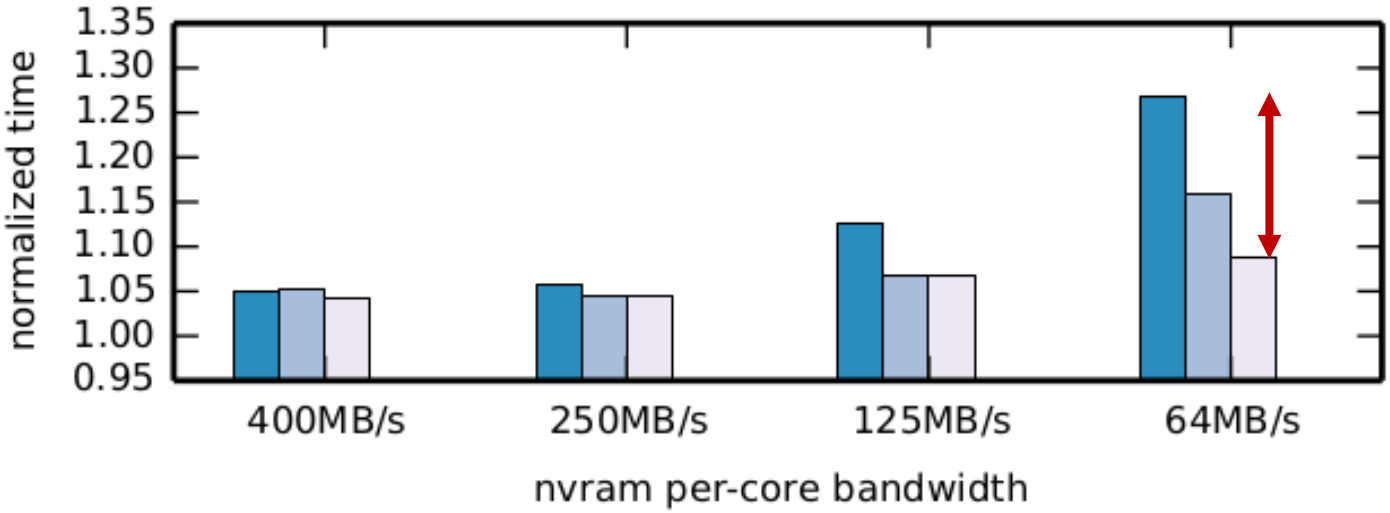


CM1 Benchmark



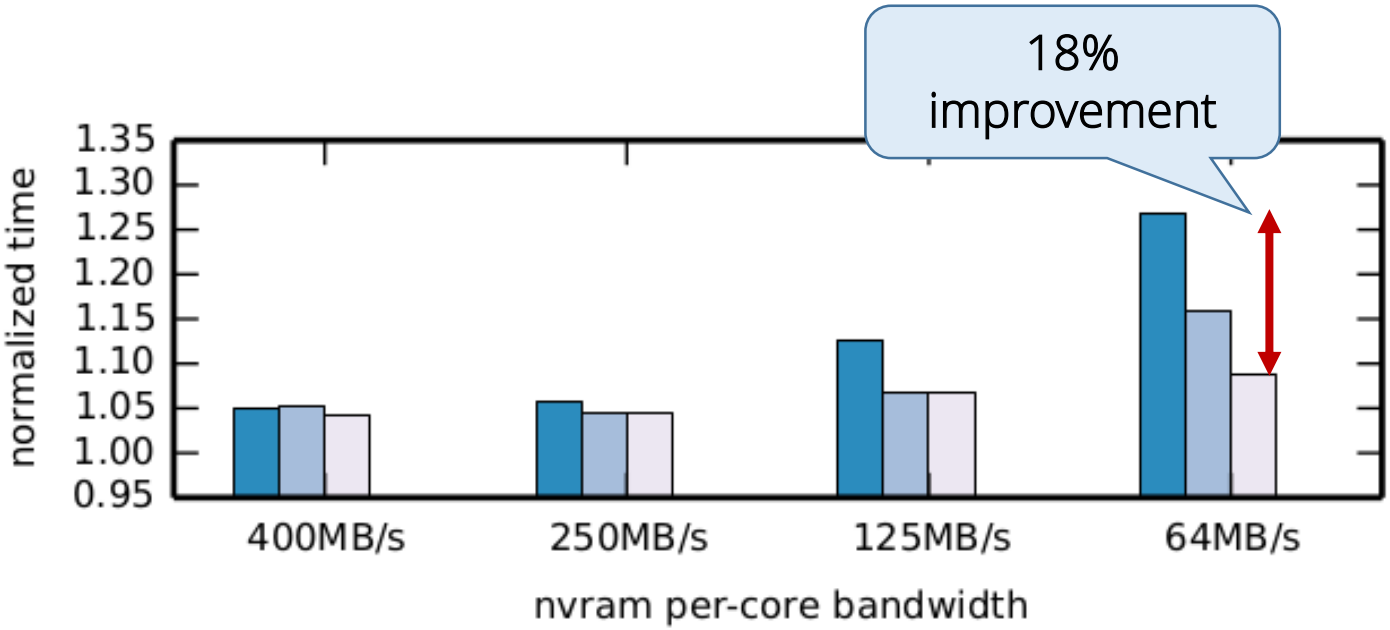
Total Simulation Time

- Application – GTC
- Total time over 10 compute iterations



Total Simulation Time

- Application – GTC
- Total time over 10 compute iterations



Summary

- PHX – a bandwidth aware checkpoint/ restart scheme for NVM
  - Reduce C/R time , accelerate simulation time
  - Reduce energy requirements
- The technique shows promising results in the evaluated scale
  - C/R costs will only increase with scale, so solutions like PHX will only gain importance
- Future work
  - Evaluate against analytics workloads
  - Deployment and evaluation at scale
  - Will be carried out as part of the UNITY SSIO/ SICM ECP projects, funder by US DoE

Thanks!

Pradeep Fernando

pradeepfn@gatech.edu

Sudarsun Kannan, Ada Gavrilovska, Karsten Schwan

We thank IEEE TCPP and the National Science Foundation for providing travel support to attend IEEE HiPC 2016

Q&A