Fault Tolerance for OpenSHMEM

Pengfei Hao
Pavel Shamis, Manjunath Gorentla
Swaroop Pophale, Aaron Welch
Stephen Poole, Barbara Chapman

Previous Related Work

N.Ali, S.Krishnamoorthy, N.Govind, and B.Palmer. **A redundant communication approach to scalable fault tolerance in PGAS programming models.** In *Parallel Distributed and Network-Based Processing(PDP) , 2011 19th Euromicro International Conference*.
Limitation:

- Algorithm specific solution. "Maintaining and continuously updating shadow data structures."
- Manually effort to manage backup and checkpoint restore

OpenSHMEM Memory Model

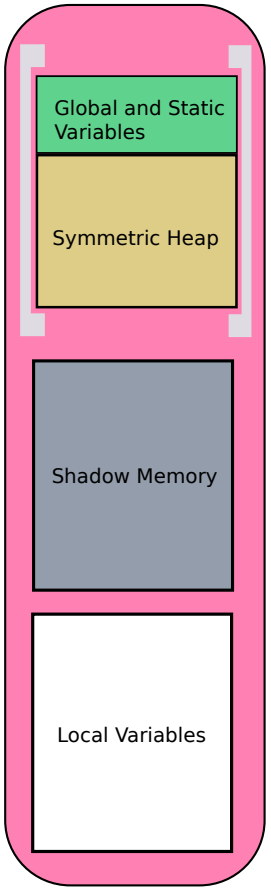
Symmetric Regions:

- Fortran data object in common blocks or with the SAVE attribute
- Global and Static C and C++ variables
- Fortran mem allocated by shpalloc()
- C and C++ mem allocated by shmalloc()

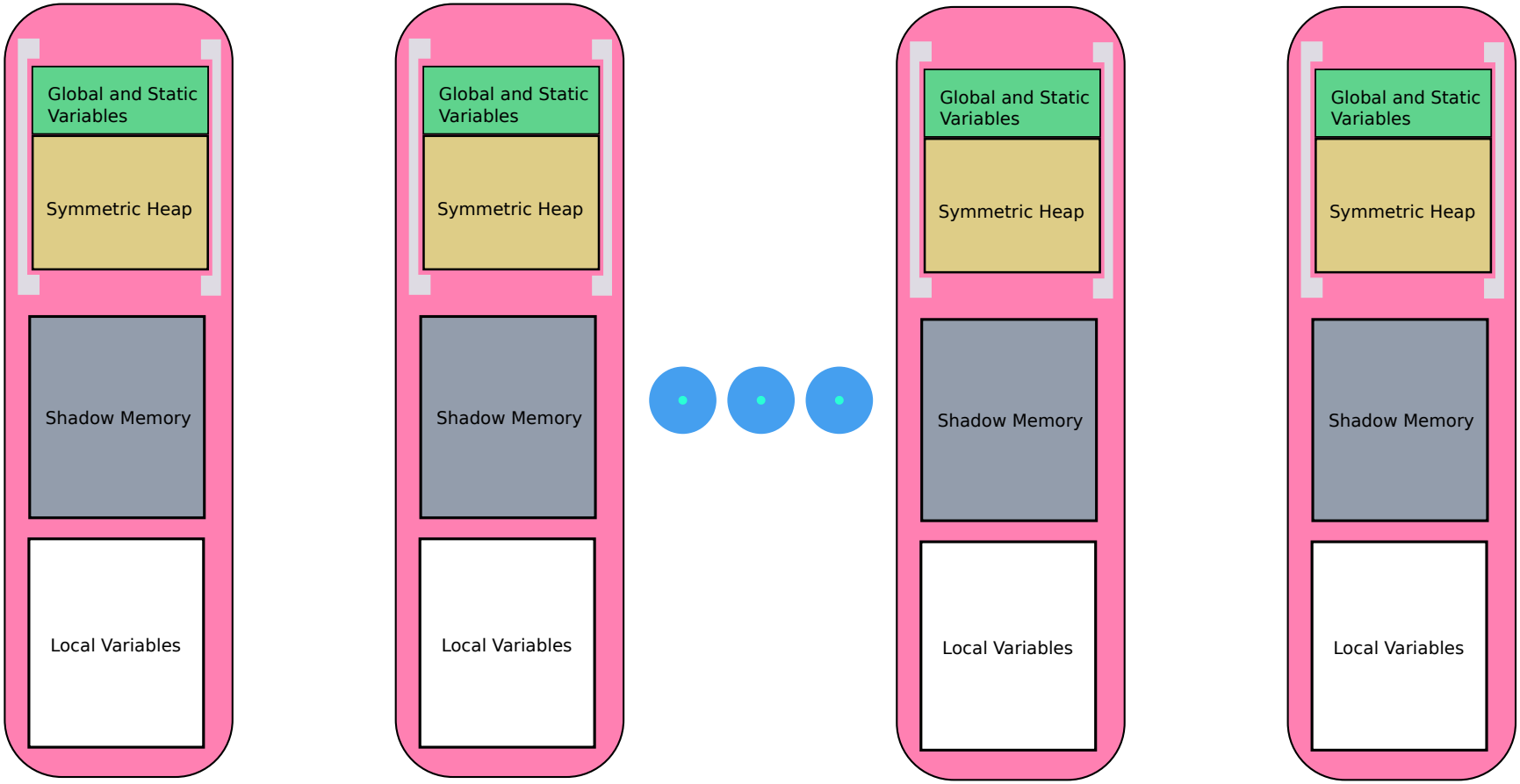
Fault tolerance Design

- Backup all symmetric regions
- Checkpoint mechanism
- Partner PE and shadow memory

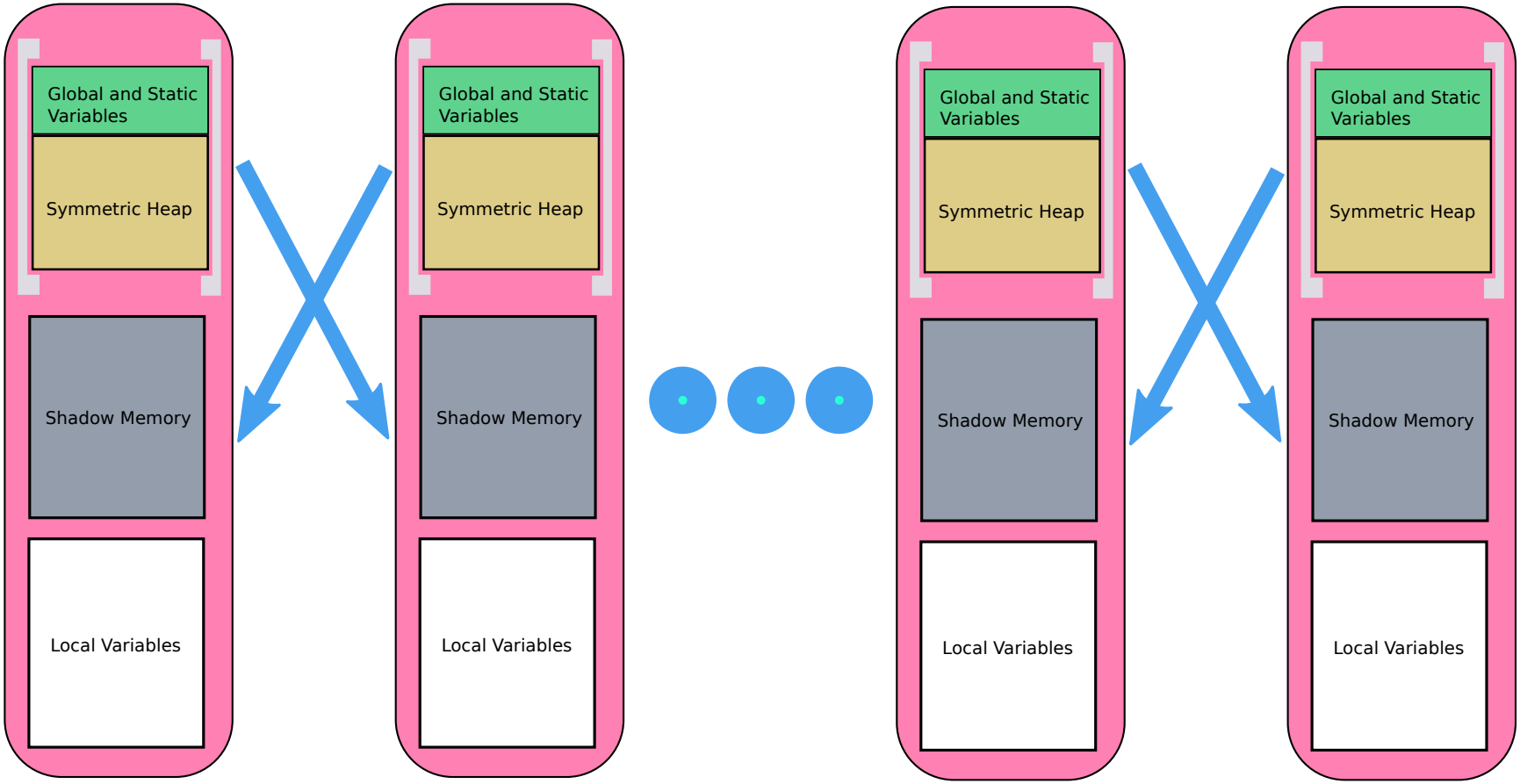
Fault tolerance Design - Memory Layout



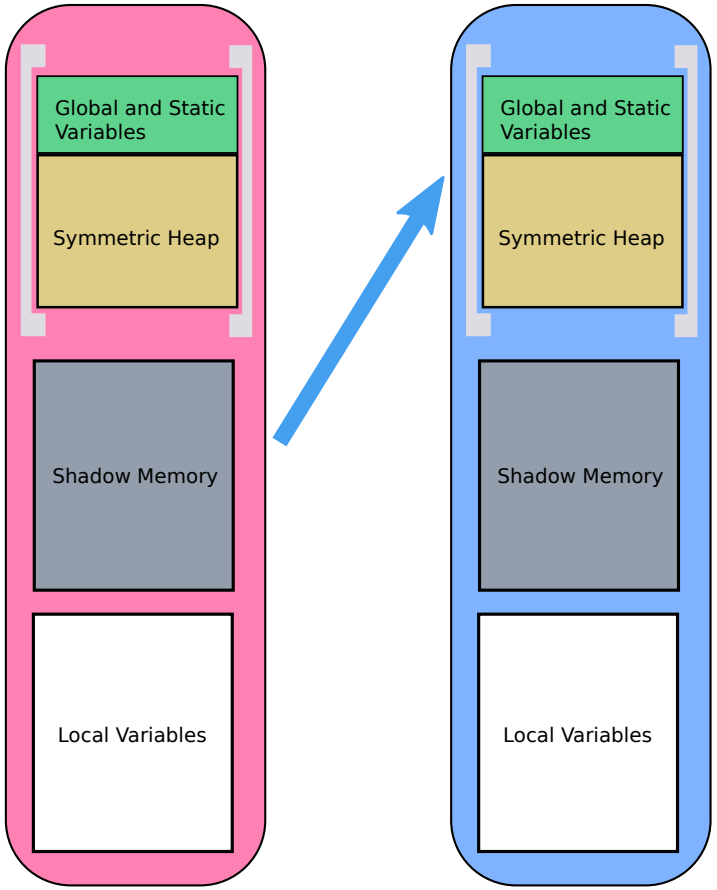
Fault tolerance Design - Backup



Fault tolerance Design - Backup



Fault tolerance Design - Restore



API

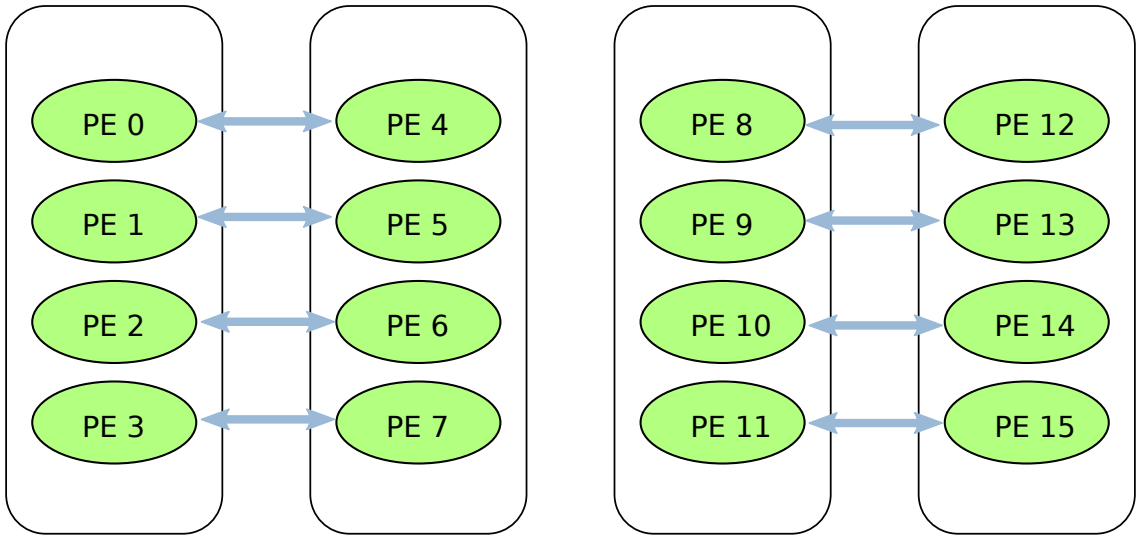
- int shmem_checkpoint_all(void);
  Fault detection by return value.
- int shmem_query_fault(int **pes, int **pes_status, size_t *numpes);
  Check failed PE list.
- int shmem_restart_pes(int *pes, size_t num_pes);
  Collective reconstuction of PE to process mapping.
- int shmem_restore_all(void);
  Restore of symmetric memory.

User participation model

```
do{
  if(shmem_checkpoint_all()){
    /* Failure event was reported */
    int *pes,*status; size_t num_pes;
    /* Query indexes of failed PEs */
    shmem_query_fault(&pes, &status, &num_pes);
    /* Restart failed PEs */
    shmem_restart_pes(pes, num_pes);
    /* Restore memory */
    shmem_restore_all();
    /* Reset the algorithm one step back */
    step--;
  }
  execute_computation(step);
  step++;
}while(step<MAX_STEPS);
```

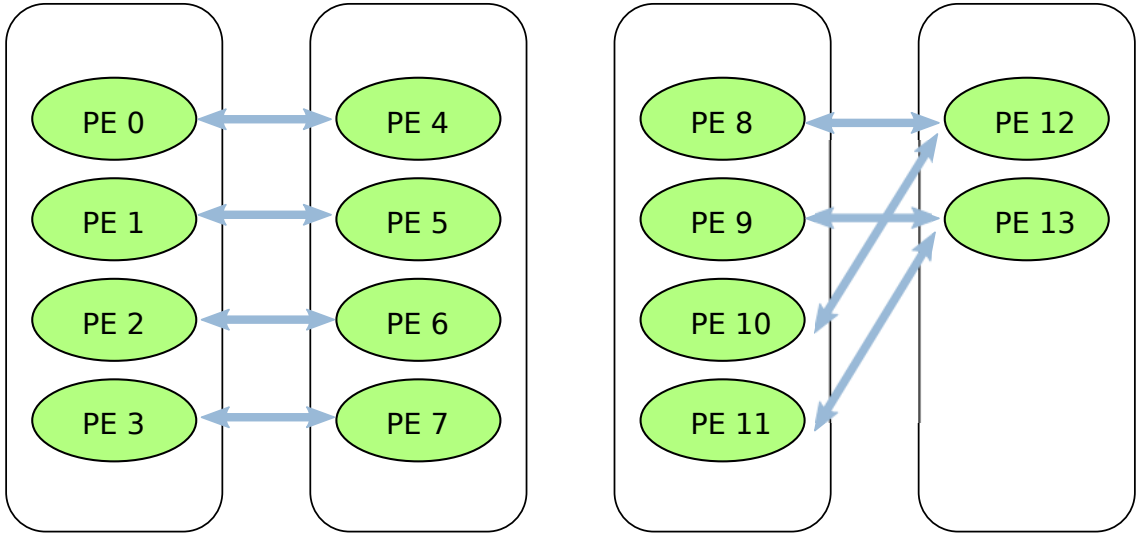
PE layout awareness

- Each PE have knowledge of distribution of all PEs.
- Consistence algorithm to find Partner PE.
- Backup in remote node



Asymmetric backup

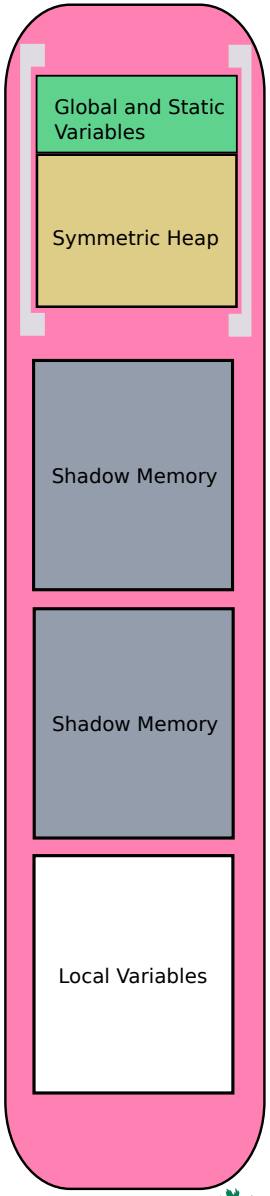
- Uneven Distribution
- Require Asymmetric Memory



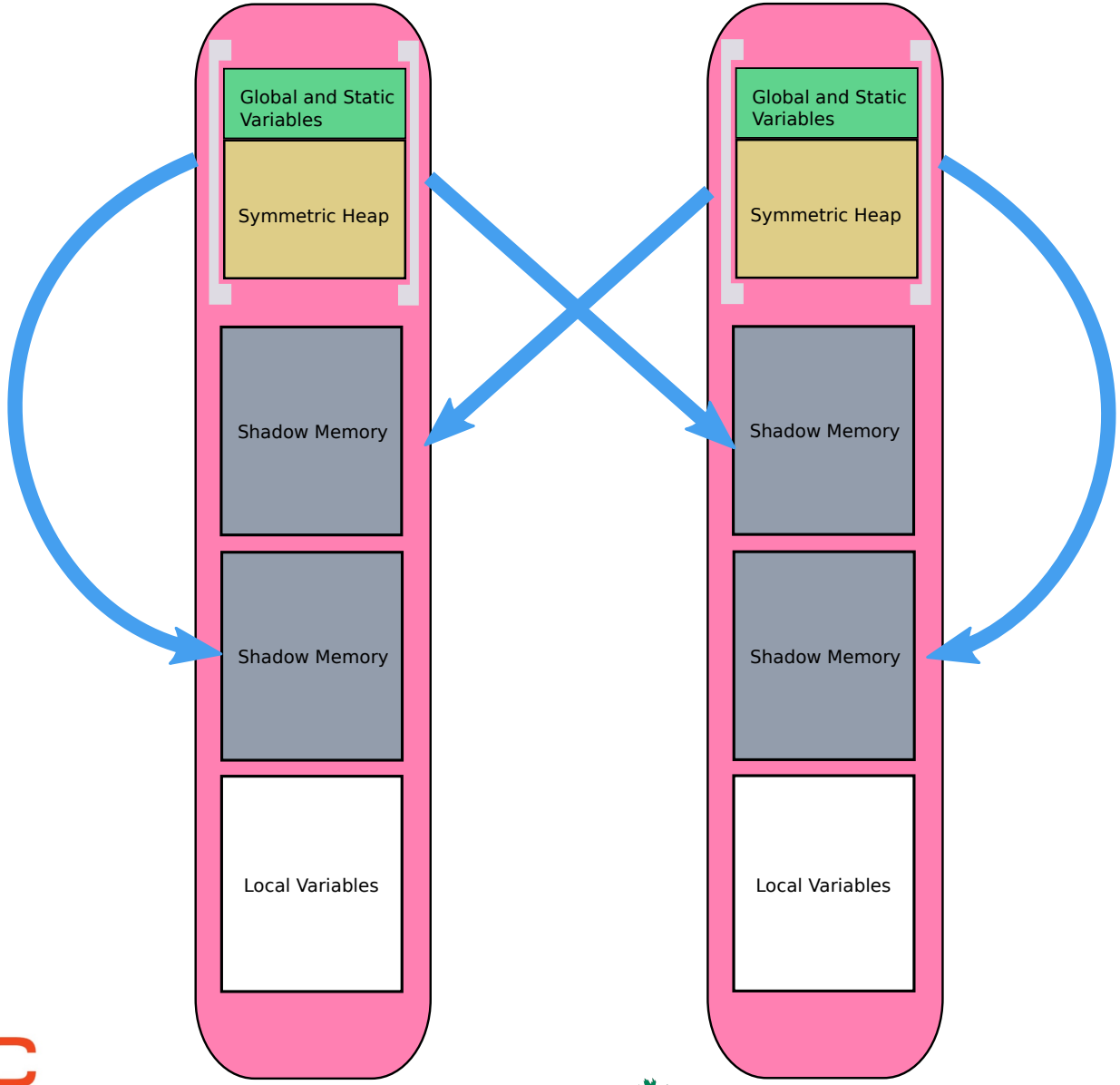
Restore selection

- PE fails – Checkpoint of partner lose
- Single backup VS. Dual backup

Dual backup


Global and Static Variables
Symmetric Heap
Shadow Memory
Shadow Memory
Local Variables

Dual backup


Global and Static Variables
Symmetric Heap
Shadow Memory
Shadow Memory
Local Variables
Global and Static Variables
Symmetric Heap
Shadow Memory
Shadow Memory
Local Variables

Single Backup VS. Dual backup

- Dual backup
  - Local restore benefit
  - Fully functional restore
- Single backup
  - Better memory usage
  - Fit for some algorithms

Summary

- Fault Tolerance for OpenSHMEM
- Shadow memory Model
- APIs
- Backup/Restore Strategy

Summary

Still in progress . . .

Acknowledgements

This work was supported by the United States Department of Defense & used resources of the Extreme Scale Systems Center at Oak Ridge National Laboratory.