IOWA STATE UNIVERSITY
Department of Computer Science

Computer Science Graduate Student Organization
February 14, 2019

Nathan Weeks, Glenn R. Luecke[1], Pieter Maris[2], James P. Vary[2]

1. Department of Mathematics
2. Department of Physics & Astronomy

Recovery from Fail-Stop Failures in Parallel Fortran Applications

Summary

The Fortran 2018 standard defines syntax and semantics to allow a parallel application to recover from *failed images* (processes) during execution. This poster presents work to extend the GFortran compiler front end and OpenCoarrays library to support fault tolerant *teams* of images, enabling use of collective routines after an image failure.

Problem

- The largest supercomputers today have tens of thousands of compute nodes—and a low Mean Time Between Failure (several hours)
  - Probability of failure unacceptably high for applications that use most/all of the system
  - Checkpoint/restart can be inefficient
- Fortran 2018 standardizes an API that allows an application to detect and recover from image failures, but only a partial implementation exists

Try It

Docker image with complete software environment (MPICH + modified GFortran & OpenCoarrays):

```
$ alias d='docker run -it -v $PWD:/mnt \
-w /mnt nathanweeks/opencoarrays:cs-gso-2019'
$ curl -L https://goo.gl/NpD9Ac \
> cs-gso-2019.F90
$ d caf cs-gso-2019.F90
$ d cafrun -np 16 ./a.out
 pi (est.):    3.1410373333333337
```

Example

Fault-Tolerant Parallel Monte Carlo Pi Calculation

```
…
do sample = 1, NSAMPLES
  call random_number(x); call random_number(y)
  if (hypot(x, y) <= 1) n = n + 1
end do

if (this_image() == 1) n_copy = n

do
  form team (1, team, stat=status)
  ! simulate image failure
  fail = size(failed_images()) < NFAIL &
         .and. this_image() == num_images()
  change team (team, stat=status)
    if (fail) fail image
    ! result undefined if image failure during
    ! co_sum(); use copy of n on image 1
    if (this_image() == 1) n = n_copy
    call co_sum(n, result_image=1, stat=status)
  end team (stat=status)
  if (status /= STAT_FAILED_IMAGE) exit
end do

if (this_image() == 1) write(*,*) 'pi (est.):',&
 (4.0d0*n/NSAMPLES)/
 (num_images()-size(failed_images()))
```

FORM TEAM creates a new team of images that excluded failed images. NEW_INDEX= specifier (not shown) enables deterministic ordering of images in team, facilitating non-shrinking recovery

Image indexes are preserved in the event of image failure, allowing one-sided gets/puts with coarrays (eventually: once teams supported with coarrays)

FORM/CHANGE/END/SYNC TEAM and **CRITICAL** now accept STAT= and ERRMSG= arguments, and handle failed images only if STAT= is present

CHANGE TEAM/END TEAM synchronize only on active images of new team (previously all images of current team)

FAILED_IMAGES() now works for teams other than the initial team, and supports an optional team argument, returning a list of failed images in that team

Future Work

- Support teams with failed images containing coarrays (for one-sided gets/puts between images)
- Prototype fault-tolerant sparse matrix eigensolver from application MFDn

Acknowledgements This research used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.
This material is based upon work supported by the U.S. Department of Energy, Office of Science, under Award Numbers DE-FG02-87ER40371 and DE-SC0018223 (SciDAC-4/NUCLEI).