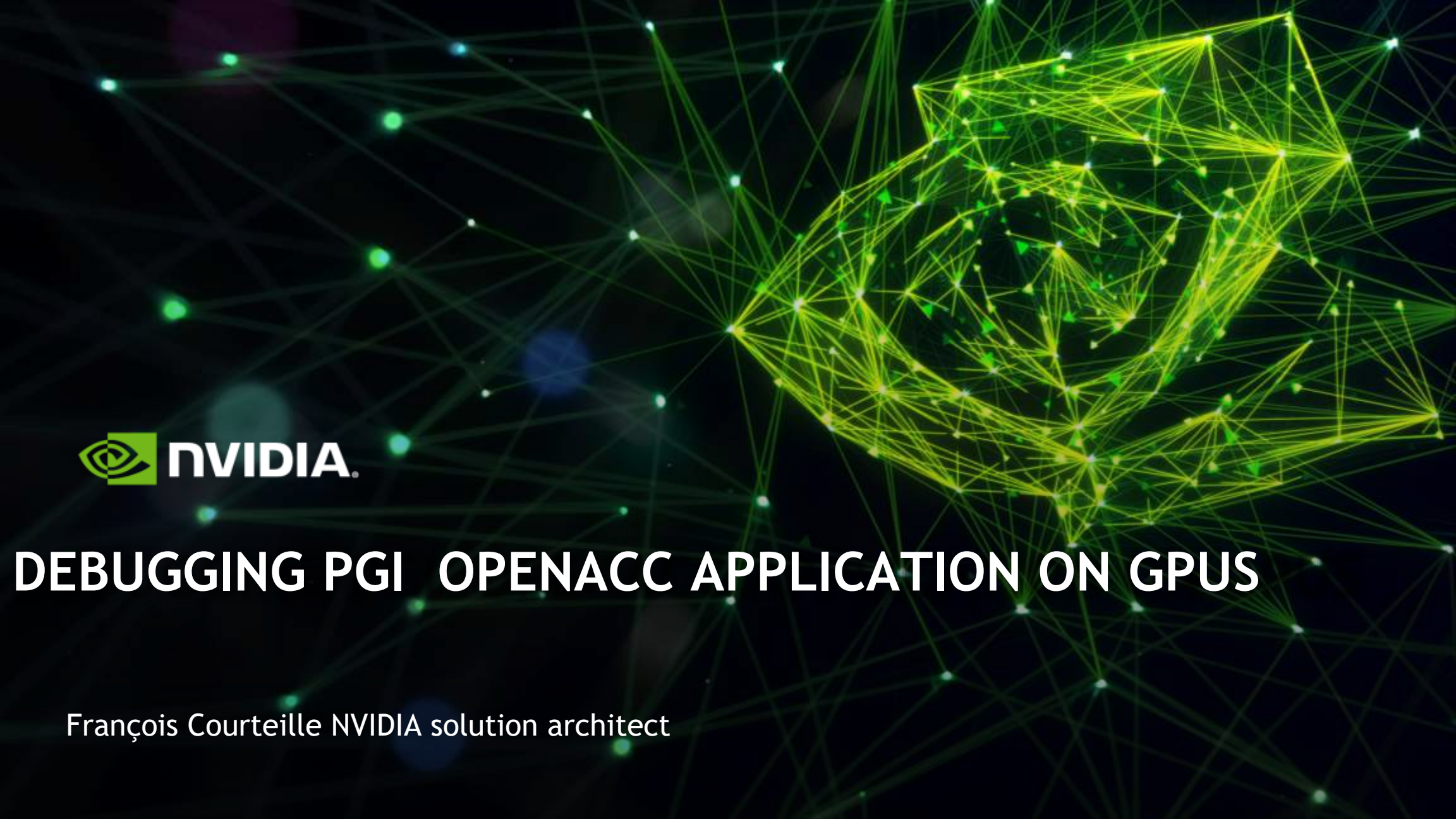


DEBUGGING PGI OPENACC APPLICATION ON GPUS

François Courteille NVIDIA solution architect

COMMON OPENACC ERRORS

acc parallel or loop independent errors (not a parallel loop)

data bounds errors (not enough data moved to the device)

stale data on device or host (missing update)

present error (missing data clause somewhere)

roundoff error (differences in float arithmetic host vs device)

roundoff error for summation (parallel accumulation)

async errors (missing wait)

compiler error (ask for help)

other runtime error (need debugger or other help)

COMMON OPENACC ERRORS - TALK HIGHLIGHTS

roundoff error (differences in float arithmetic host vs device)

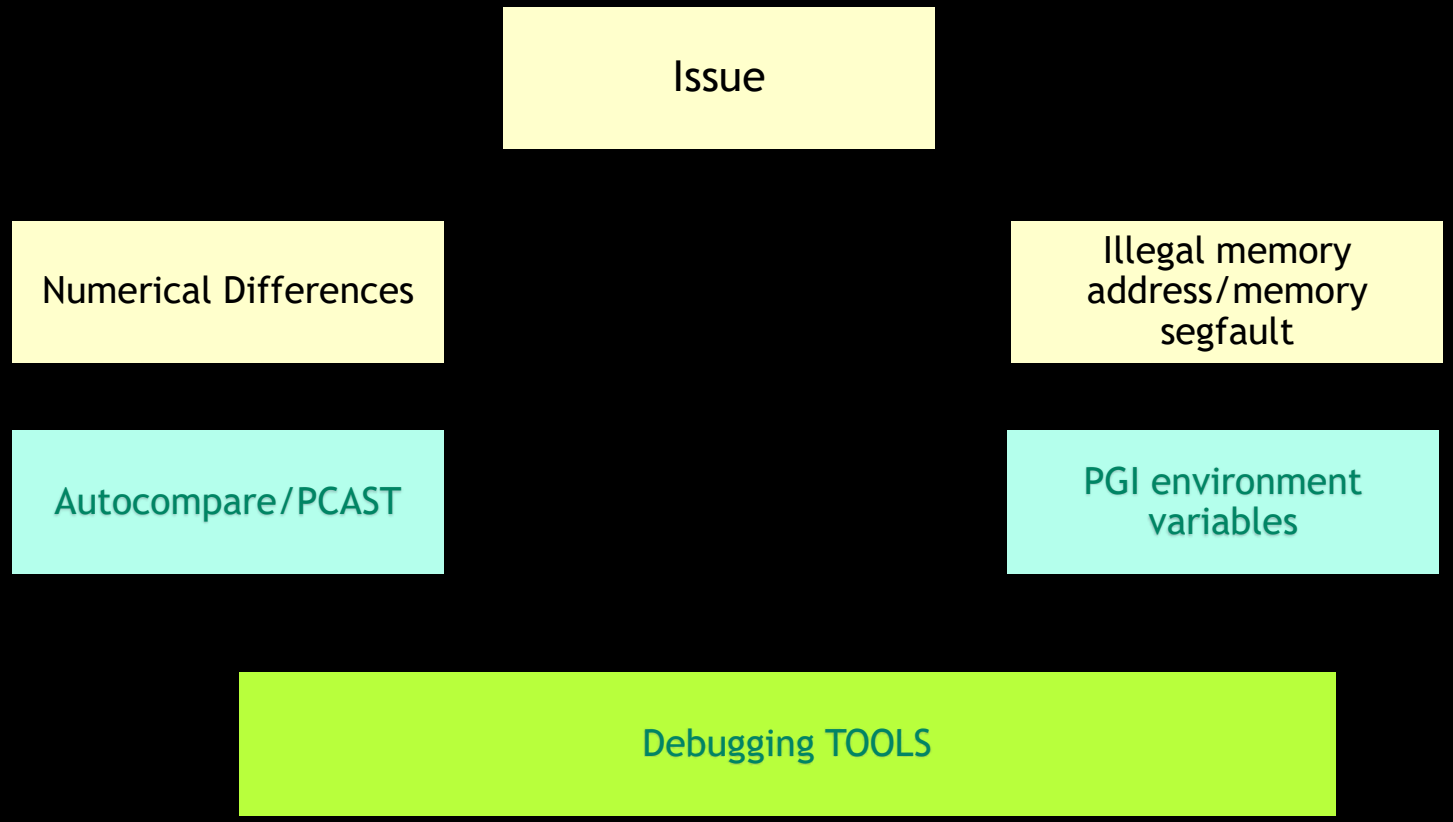
roundoff error for summation (parallel accumulation)

other runtime error (need debugger or other help)

call to cuStreamSynchronize returned error 700: Illegal address during kernel execution

DEBUGGING APPROACH

DEBUGGING A GPU APPLICATION



SUPPORT FOR PRINTF() IN OPENACC

Debugging / tracing GPU-accelerated code regions

```
#pragma acc kernels copy(x[0:256])
{
  for (int i = 0; i < 256; i++) {
    if (x[i] < small) {
      printf("x at loc %03d is small: %d\n",i,x[i]);
      x[i] = small;
    } else if (x[i] > large) {
      printf("x at loc %03d is large: %d\n",i,x[i]);
      x[i] = large;
    }
 }
```

Support for formatted output using printf() statements in OpenACC compute regions

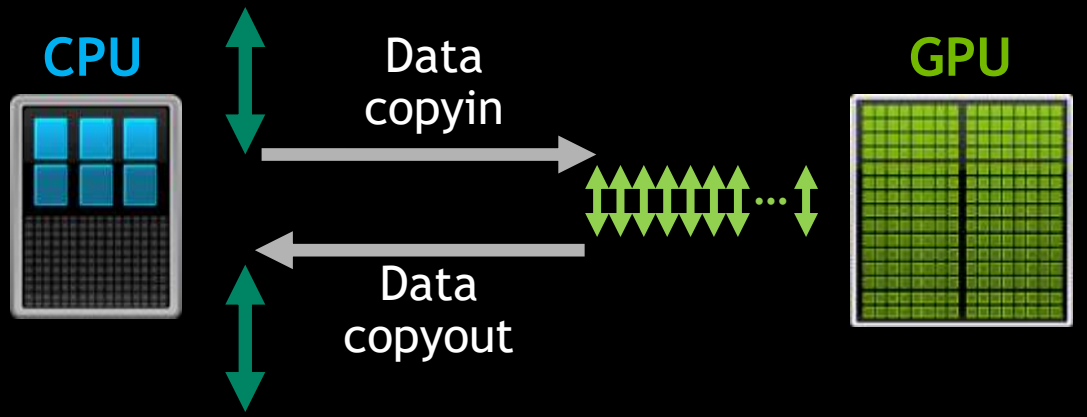
Debug and trace OpenACC programs during development and tuning

Works with both CPU and GPU

INVESTIGATING NUMERICAL DIFFERENCES : OPENACC PCAST

OPENACC AUTO-COMPARE

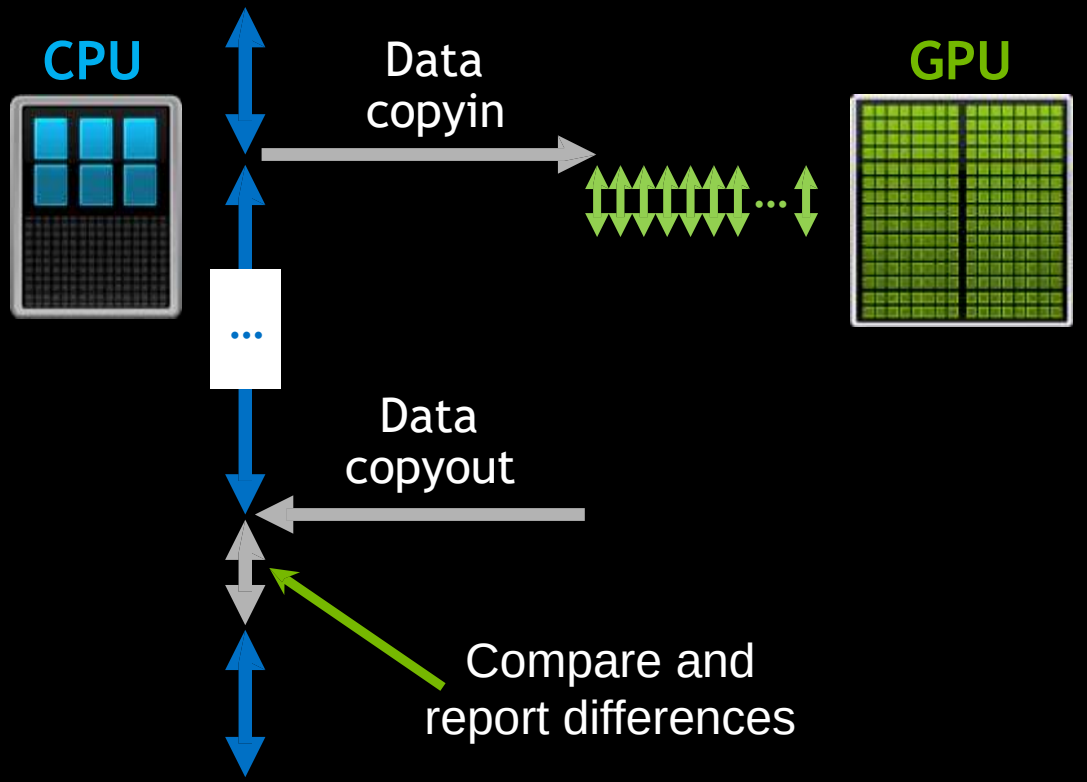
Find where CPU and GPU numerical results diverge



Normal OpenACC execution mode, no auto-compare

OPENACC AUTO-COMPARE

Find where CPU and GPU numerical results diverge



- **-ta=tesla:autocompare**
- Compute regions run redundantly on CPU and GPU
- Results compared when data copied from GPU to CPU
- **pgicompilers.com/pcast**

OPENACC AUTO-COMPARE OUTPUT

Directives / environment variables to control fidelity

```
$ pgcc -ta=tesla:autocompare -o a.out example.c

$ PGI_COMPARE=summary,compare,abs=1 ./a.out
PCAST a1 comparison-label:0 Float
        idx: 0 FAIL ABS  act: 8.40187728e-01
exp: 1.00000000e+00 tol: 1.00000001e-01
        idx: 1 FAIL ABS  act: 3.94382924e-01
exp: 1.00000000e+00 tol: 1.00000001e-01
        idx: 2 FAIL ABS  act: 7.83099234e-01
exp: 1.00000000e+00 tol: 1.00000001e-01
        idx: 3 FAIL ABS  act: 7.98440039e-01
exp: 1.00000000e+00 tol: 1.00000001e-01
```

Test for program correctness, and determine points of divergence

Detect diverging results between CPU & GPU or multiple CPU code versions

Flexible calling and instrumentation options

PCAST: PGI COMPILER ASSISTED SOFTWARE TESTING

PCAST and OpenACC Auto-compare Summary

- **OpenACC, GPU to CPU comparison**
  - `-ta=tesla,autocompare`, compare at `copyout` or `update host`
  - `-ta=tesla,redundant`, redundant execution
  - `#pragma acc compare(a[0:n])`
    - dynamically compare device-to-host data
- **CPU to CPU comparison**
  - `#pragma pgi compare(b[0:n])`
    - save data to file, then to compare to saved data

https://www.pgroup.com/resources/pcast.htm

https://www.pgroup.com/blogs/posts/pcast.htm

USING PGI ENVIRONMENT VARIABLES FOR A FIRST DEBUGGING APPROACH

PGI ENVIRONMENT VARIABLES

Valuable PGI ENVs

```
PGI_ACC_DEBUG
PGI_ACC_NOTIFY
PGI_ACC_TIME
PGI_ACC_PROFILE
PGI_ACC_FILL
PGI_ACC_SYNCHRONOUS
```

PGI RUNTIME ANALYSIS
FOR QUICK SANITY CHECKS

Applications compiled with PGI compiler: Analyze via environment variables

PGI_ACC_TIME Lightweight profiler for time of data movement and kernels

PGI_ACC_NOTIFY Print information for GPU-related events.

=1 … kernel launches only
=2 … data transfers only
=3 … kernel launches and data transfers
=4 … region entry/exits only
=5 … region entry/exits and kernel launches
=8 … wait operations, synchronizations
=16 … (de)allocation of device memory

PGI RUNTIME EXAMPLE

```
$> PGI_ACC_NOTIFY=3  ./spmv.bin

upload   CUDA data    file=.../spmv.c
function=main line=36  device=0  threadid=1 variable=row_ptr  bytes=15705192
upload   CUDA data  file=.../spmv.c
function=main line=36  device=0  threadid=1 variable=row_ptr  bytes=16777216 [...]
launch CUDA  kernel    file=.../spmv.c
function=main line=42  device=0  threadid=1 num_gangs=65535  [...]
block=128 shared memory=1024
launch CUDA  kernel    file=.../spmv.c
function=main line=42  device=0  threadid=1 num_gangs=65535  [...]
block=128 shared  memory=1024 [...]
download   CUDA data  file=.../spmv.c
function=main line=59  device=0  threadid=1 variable=y  bytes=14633352
download   CUDA data  file=.../spmv.c
function=main line=59  device=0  threadid=1 variable=y  bytes=16777152 [...]
Runtime 0.172498  s.
```

PGI RUNTIME ANALYSIS
FIRST DEBUGGING INFORMATION

Applications compiled with PGI compiler: Analyze via environment variables

PGI_ACC_DEBUG=1 … To get detailed debug information

DEBUGGING SUMMARY

- PGI_ACC_NOTIFY={bit mask}
  1 - kernels launch, 2 - data transfers, 4 - sync operations,
  8 - region entry/exit, 16 - data allocation/free
- PGI_ACC_DEBUG=1
- PGI_ACC_FILL=<value>
- PGI_ACC_SYNCHRONOUS=1
- Use “if” clause and “update” directives