Data race detection for large OpenMP applications

Ignacio Laguna, Harshitha Menon
Lawrence Livermore National Laboratory

Michael Bentley, Ian Briggs, Pavel Panchekha, Ganesh Gopalakrishnan
University of Utah

Hui Guo, Cindy Rubio González
University of California at Davis

Michael O. Lam
James Madison University

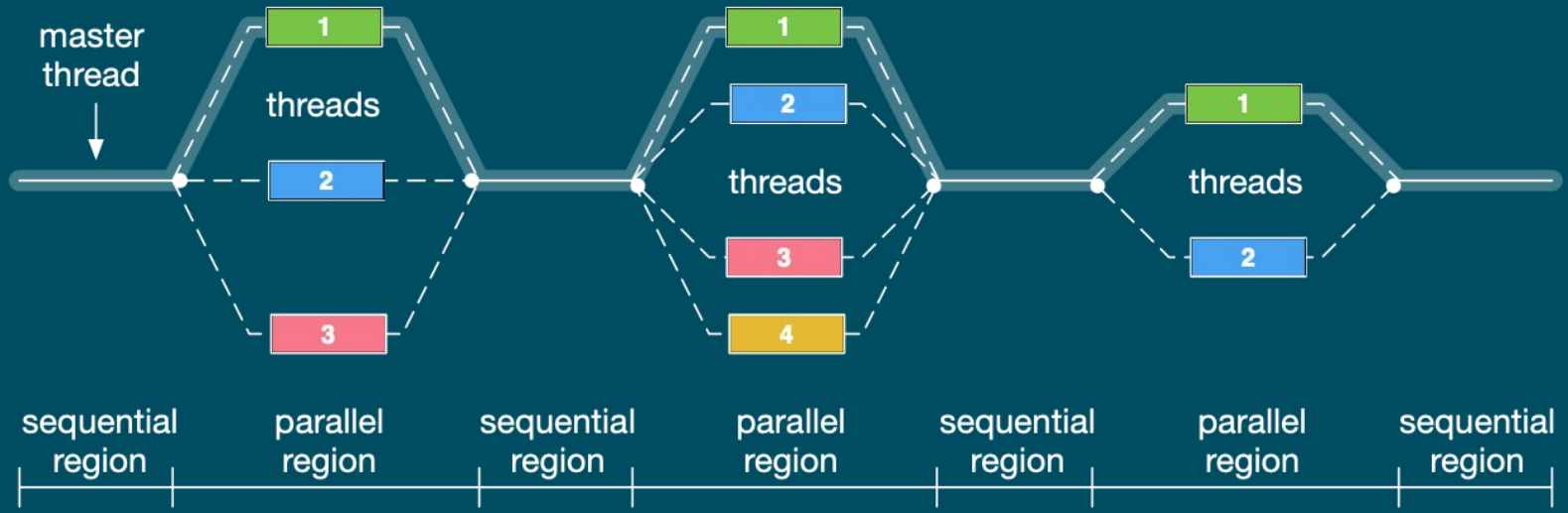
http://fpanalysistools.org/

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 (LLNL-PRES-780623).

OpenMP

Enabling HPC since 1997

- Shared memory parallel programming
- Standard for on node parallelism in HPC

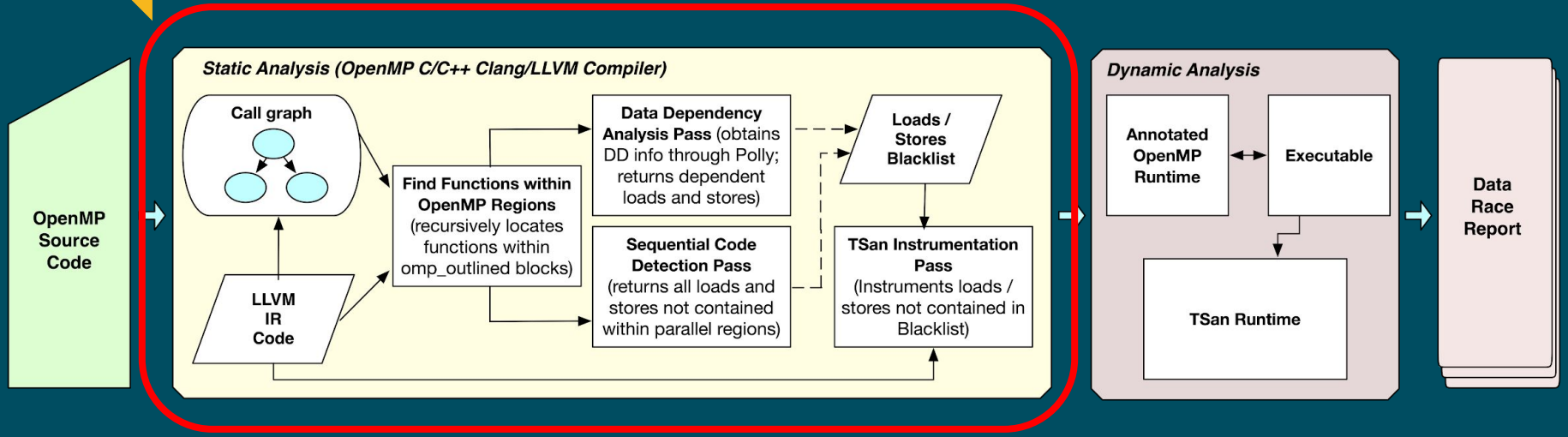


Data Race

Thread 1	Thread 2	Value of X
		0
local = X		0
local++		0
	Local = X	0
X = local		1
	local++	1
	X = local	1

- When shared data is accessed by two or more threads with no synchronization and at least one access is a write
- Leads to non deterministic behavior, crashes, and incorrect results
- Races are undefined behavior in C/C++
- Even “benign” races can be transformed by compilers into harmful races

Archer Tool Flow



Static Analysis

- Used to not instrument race free code
- Polly, an llvm polyhedral analysis
  - Used to exclude code with no data dependencies
- A custom llvm pass
  - Used to exclude serial code
  - Excludes loads/stores not reachable from OpenMP

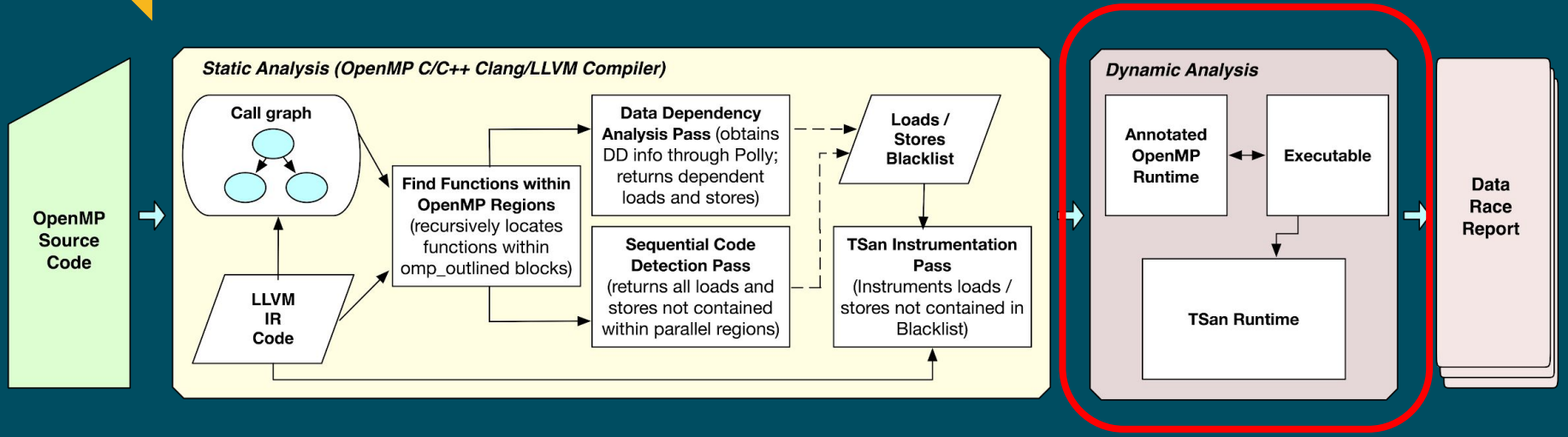
```
#pragma omp parallel for
  for (int i=0; i<N; i++) {
    a[i] = a[i] + 1;
  }
```

No data dependency code blacklisted

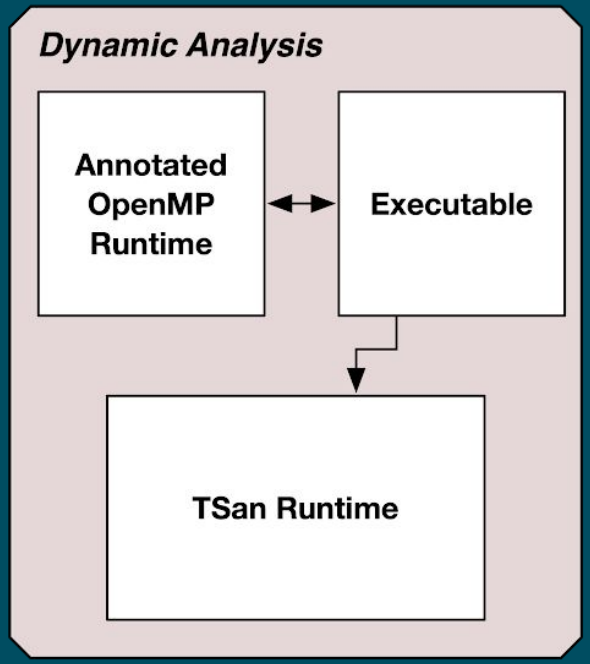
```
#pragma omp parallel for
  for (int i=0; i<N-1; i++) {
    a[i] = a[i+1];
  }
```

Potentially racy code instrumented

Archer Tool Flow



Dynamic Analysis



- ThreadSanitizer is used for dynamic analysis
  - Works well for Pthreads, but breaks on OpenMP constructs
- Annotated OpenMP runtime to highlight synchronization

Results

- AMG2013, algebraic multigrid solver for linear systems
  - Three unknown races found
- HYDRA, a simulator used at the National Ignition Facility
  - Found race which caused crashes when porting to Sequoia
  - First solution was to **disable** OpenMP
- Z-Pinch, a high-density physics code
  - Data race in macro extremely hard to pinpoint
  - Same value written in the same shared variable
  - IBM XL compiler transformations make the program crash

Examples

Exercise 1: Using Archer on an example from DataRaceBench

- OpenMP data race benchmark suite developed at LLNL
- Available at https://github.com/LLNL/dataracebench

Exercise 1

minusminus-orig-yes.c

```
57 const int len = 100;
58 int x[len];
59 int numNodes = len, numNodes2 = 0;
60
61 for (int i=0; i<len; i++) {
62   if (i%2 == 0) {
63     x[i] = 5;
64   } else {
65     x[i] = -5;
66   }
67 }
68
69 #pragma omp parallel for
70 for (int i=numNodes-1; i > -1; --i) {
71   if (x[i] <= 0) {
72     numNodes2-- ;
73   }
74 }
```

```
75
76 if (numNodes2 != -numNodes/2) {
77   printf ("numNodes2 = %d\n", numNodes2);
78   //printf ("A race occured\n");
79 } else {
80   //printf ("No race found\n");
81 }
```

Just because no race was found doesn't mean that there is no race! It just means that we were "lucky". Also these prints are only for illustration (not recommended) — **Sutter's "Pink Elephant" values!!**

Exercise 1 - ./step-01.sh

```
exercise-1 $ clang -g -fopenmp minusminus-orig-yes.c \
                  -o minusminus-without-archer
exercise-1 $ ./minusminus-without-archer
exercise-1 $ ./minusminus-without-archer
exercise-1 $ ./minusminus-without-archer
exercise-1 $ ./minusminus-without-archer
exercise-1 $ ./minusminus-without-archer
exercise-1 $ ./minusminus-without-archer
exercise-1 $ ./minusminus-without-archer
[...]
```

Exercise 1 - ./step-02.sh

```
 exercise-1 $ clang-archer -g minusminus-orig-yes.c -o minusminus-with-archer
 exercise-1 $ ./minusminus-with-archer
==================
WARNING: ThreadSanitizer: data race (pid=29573)
  Write of size 4 at 0x7ffdab010858 by thread T1:
    #0 .omp_outlined._debug__
~/Module-Archer/excercise-1/minusminus-orig-yes.c:72:16
(minusminus-with-archer+0x4a7cb1)
    #1 .omp_outlined. ~/Module-Archer/excercise-1/minusminus-orig-yes.c:70:3
(minusminus-with-archer+0x4a7e02)
    #2 __kmp_invoke_microtask <null> (libomp.so+0x89cc2)

  Previous write of size 4 at 0x7ffdab010858 by main thread:
    #0 .omp_outlined._debug__
~/Module-Archer/excercise-1/minusminus-orig-yes.c:72:16
(minusminus-with-archer+0x4a7cb1)
    #1 .omp_outlined. ~/Module-Archer/excercise-1/minusminus-orig-yes.c:70:3
(minusminus-with-archer+0x4a7e02)
[...]
```

Exercise 1

minusminus-orig-yes.c

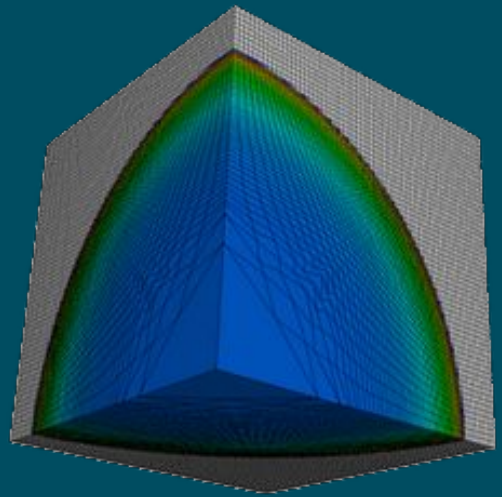
```
57 const int len = 100;
58 int x[len];
59 int numNodes = len, numNodes2 = 0;
60
61 for (int i=0; i<len; i++) {
62   if (i%2 == 0) {
63     x[i] = 5;
64   } else {
65     x[i] = -5;
66   }
67 }
68
69 #pragma omp parallel for
70 for (int i=numNodes-1; i > -1; --i) {
71   if (x[i] <= 0) {
72     numNodes2-- ;
73   }
74 }
```

```
75
76 if (numNodes2 != -numNodes/2) {
77   printf ("numNodes2 = %d\n", numNodes2);
78   //printf ("A race occured\n");
79 } else {
80   //printf ("No race found\n");
81 }

// Again the prints are for illustration.
// Racy code has “catch-fire” semantics.
// So don’t rely upon printing after a race!
// Pink Elephant values!
```

Exercise 2 Application: LULESH

- Proxy application developed at LLNL
- Models a shock hydrodynamics problem



Exercise 2 - ./step-03.sh

```
exercise-2 $ mkdir build
exercise-2 $ cd build
exercise-2 $ cmake ../LULESH -DCMAKE_BUILD_TYPE=Release \
                             -DCMAKE_CXX_COMPILER=`which clang-archer++` \
                             -DWITH_MPI=Off \
                             -DWITH_OPENMP=On
-- The CXX compiler identification is Clang 8.0.1
-- Check for working CXX compiler:
/home/ibriggs/ARCHER/archer_install/bin/clang-archer++
-- Check for working CXX compiler:
/home/ibriggs/ARCHER/archer_install/bin/clang-archer++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Found OpenMP_CXX: -fopenmp=libomp (found version "3.1")
-- Found OpenMP: TRUE (found version "3.1")
[...]
```

Exercise 2 - ./step-03.sh

```
exercise-2 $ make
Scanning dependencies of target lulesh2.0
[ 16%] Building CXX object CMakeFiles/lulesh2.0.dir/lulesh-comm.cc.o
[ 33%] Building CXX object CMakeFiles/lulesh2.0.dir/lulesh-init.cc.o
[ 50%] Building CXX object CMakeFiles/lulesh2.0.dir/lulesh-util.cc.o
[ 66%] Building CXX object CMakeFiles/lulesh2.0.dir/lulesh-viz.cc.o
[ 83%] Building CXX object CMakeFiles/lulesh2.0.dir/lulesh.cc.o
[100%] Linking CXX executable lulesh2.0
[100%] Built target lulesh2.0
```

Exercise 2 - ./step-03.sh

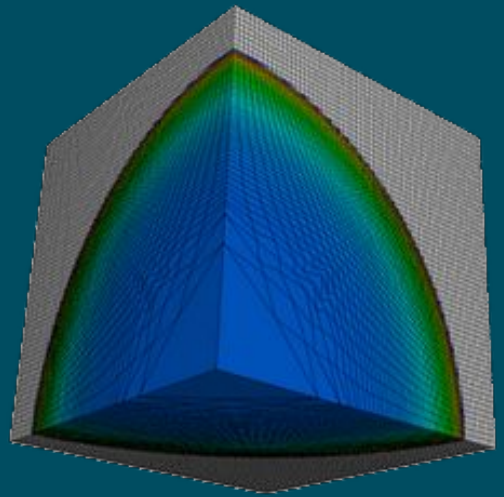
```
exercise-2 $ ./lulesh2.0
Running problem size 8^3 per domain until completion
Num processors: 1
Num threads: 3
Total number of elements: 512
[...]
Run completed:
   Problem size        =  8
   MPI tasks           =  1
   Iteration count     =  163
   Final Origin Energy =  1.788182e+04
   Testing Plane 0 of Energy Array on rank 0:
        MaxAbsDiff   = 1.136868e-12
        TotalAbsDiff = 1.120390e-11
        MaxRelDiff   = 5.521293e-14

Elapsed time         =        1.5 (s)
Grind time (us/z/c)  =  17.881075 (per dom)  (  1.492283 overall)
FOM                  =  55.925049 (z/s)
```

Exercise 3 Application: LULESH

- Proxy application developed at LLNL
- Models a shock hydrodynamics problem

Goal: find a race condition in a modified Lulesh



Exercise 3 - ./step-04.sh

```
exercise-3 $ mkdir build
exercise-3 $ cd build
exercise-3 $ cmake ../LULESH -DCMAKE_BUILD_TYPE=Release \
                             -DCMAKE_CXX_COMPILER=`which clang-archer++` \
                             -DWITH_MPI=Off \
                             -DWITH_OPENMP=On
-- The CXX compiler identification is Clang 8.0.1
-- Check for working CXX compiler:
/home/ibriggs/ARCHER/archer_install/bin/clang-archer++
-- Check for working CXX compiler:
/home/ibriggs/ARCHER/archer_install/bin/clang-archer++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Found OpenMP_CXX: -fopenmp=libomp (found version "3.1")
-- Found OpenMP: TRUE (found version "3.1")
[...]
```

Exercise 2 - ./step-04.sh

```
 exercise-3 $ make
Scanning dependencies of target lulesh2.0
[ 16%] Building CXX object CMakeFiles/lulesh2.0.dir/lulesh-comm.cc.o
[ 33%] Building CXX object CMakeFiles/lulesh2.0.dir/lulesh-init.cc.o
[ 50%] Building CXX object CMakeFiles/lulesh2.0.dir/lulesh-util.cc.o
[ 66%] Building CXX object CMakeFiles/lulesh2.0.dir/lulesh-viz.cc.o
[ 83%] Building CXX object CMakeFiles/lulesh2.0.dir/lulesh.cc.o
[100%] Linking CXX executable lulesh2.0
[100%] Built target lulesh2.0
```

Exercise 3 - ./step-04.sh

```
 exercise-3 $ ./lulesh2.0
Running problem size 8^3 per domain until completion
[...]
==================
WARNING: ThreadSanitizer: data race (pid=28978)
  Write of size 8 at 0x7b880000e730 by thread T1:
    #0 .omp_outlined._debug__.33
~/Module-Archer/excercise-3/LULESH/lulesh.cc:983:37 (lulesh2.0+0x4c9eea)
    #1 .omp_outlined..34 ~/Module-Archer/excercise-3/LULESH/lulesh.cc:970
(lulesh2.0+0x4c9eea)
    #2 __kmp_invoke_microtask <null> (libomp.so+0x89cc2)

  Previous write of size 8 at 0x7b880000e730 by thread T2:
    #0 .omp_outlined._debug__.33
~/Module-Archer/excercise-3/LULESH/lulesh.cc:983:37 (lulesh2.0+0x4c9eea)
    #1 .omp_outlined..34 ~/Module-Archer/excercise-3/LULESH/lulesh.cc:970
(lulesh2.0+0x4c9eea)
    #2 __kmp_invoke_microtask <null> (libomp.so+0x89cc2)

[...]
```

Exercise 3

lulesh.c

```c
969 #pragma omp parallel for firstprivate(numNode)
970     for( Index_t gnode=0 ; gnode<numNode ; ++gnode )
971     {
972        Index_t count = domain.nodeElemCount(gnode) ;
973        Index_t *cornerList = domain.nodeElemCornerList(gnode) ;
974        Real_t fx_tmp = Real_t(0.0) ;
975        Real_t fy_tmp = Real_t(0.0) ;
976        Real_t fz_tmp = Real_t(0.0) ;
977        for (Index_t i=0 ; i < count ; ++i) {
978           Index_t ielem = cornerList[i] ;
979           fx_tmp += fx_elem[ielem] ;
980           fy_tmp += fy_elem[ielem] ;
981           fz_tmp += fz_elem[ielem] ;
982        }
983        domain.fx(gnode + gnode%2) += fx_tmp ;
984        domain.fy(gnode) += fy_tmp ;
985        domain.fz(gnode) += fz_tmp ;
986     }
```

Thank You!

Questions?

pruners.github.io/archer