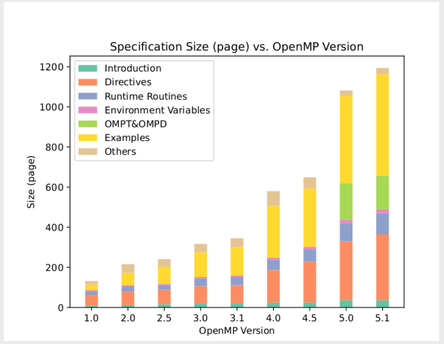
Dynamic Analysis of Data Inconsistencies and Data Races in OpenMP Programs

Lechen Yu[1], Feiyang Jin[1], Joachim Protze[2], Vivek Sarkar[1]

Georgia Institute of Technology, RWTH University

Georgia Tech College of Computing
Center for Research into Novel Computing Hierarchies

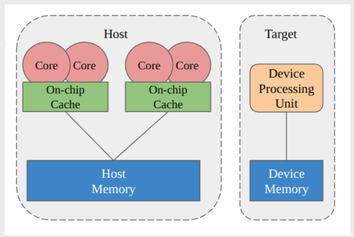
OpenMP's Complexity



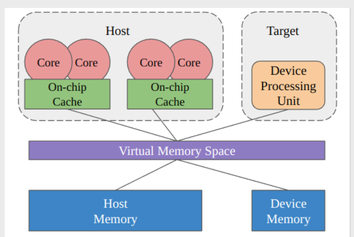
- When introducing a new parallel paradigm, the size of the specification increases significantly
- Optimal, or even correct, usage of OpenMP constructs can be non-trivial and error-prone

OpenMP's Support for Accelerator

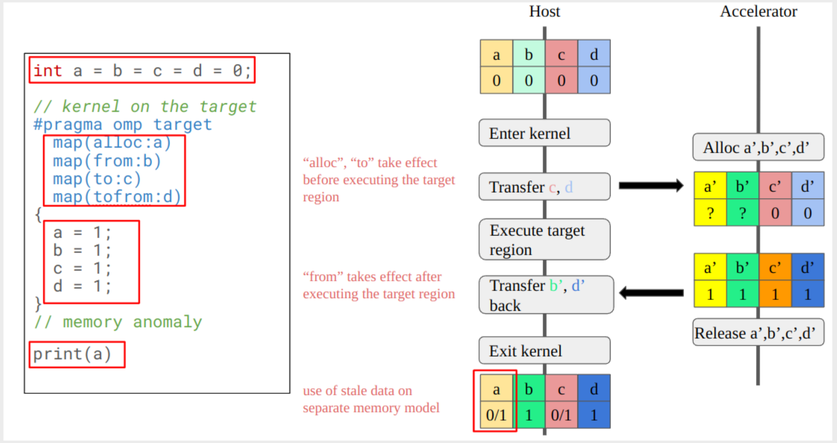
- Separate Memory Model



- Unified Memory Model



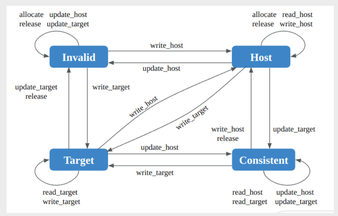
Dynamic Data Inconsistency Detection for OpenMP



- OpenMP programs may encounter incorrect usage of the target construct and map clause
- We refer to these programming errors as "data inconsistency"
- Make a single variable have inconsistent values between the host and accelerator
- Manually detecting and debugging data mapping issues can be challenging

Dynamic Detector "ARBALEST"

- Use host as a virtual device to simulate device offloading
- Simulate data transfer using *malloc* and *memcpy*
- All kernels execute on the host memory
- Use a state machine to track the validity of each mapped variable

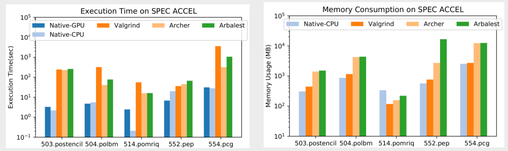


ARBALEST: https://github.com/lechenyu/Arbalest

- Precision Evaluation on DRACC

Benchmark ID	Effect	Effectiveness				
		Arbalest	Valgrind	Archer	ASan	MSan
22, 24, 49, 50, 51	UUM	✓	-	-	-	✓
23, 25, 28, 29, 30, 31	BO	✓	✓	-	✓	-
26, 27, 32, 33, 34	USD	✓	-	-	-	-
Overall		16/16	6/16	0/16	6/16	5/16

- Performance Evaluation on SPEC-ACCEL

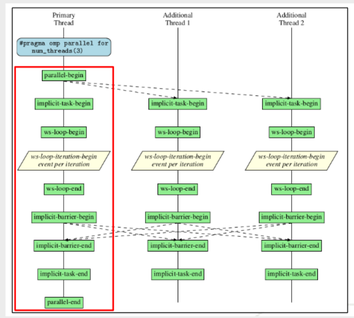


DRACC: https://github.com/RWTH-HPC/dracc

SPEC-ACCEL: https://www.spec.org/accel/

Dynamic Data Race Detection for OpenMP Tasking

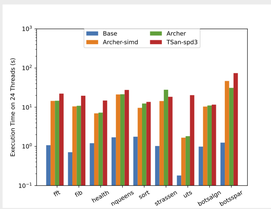
- OpenMP introduces a task-centric execution model
- Supported parallel paradigms
  - single program multiple data
  - task parallelism
  - heterogeneous parallelism



Dynamic Data Race Detector "TSan-SPD3"

- Explored the feasibility of extend to OpenMP programs
  - SPD3 stands for Scalable Precise Dynamic Data race Detection (PLDI 12)
- Why did we choose SPD3?
  - no prior work applies SPD3 to OpenMP programs
  - works for parallel execution
  - requires fixed-size of shadow memory for each memory location
- Implemented SPD3 on top of ThreadSanitizer (TSan), reusing its infrastructure

Bench	H	Wait Node	Finish Node	Async Node	Step Node
fft	22	19,604,368	2	28,808,579	77,221,530
fib	37	14,930,351	1	29,860,723	74,651,800
health	7	17,515,985	1	17,515,626	52,547,240
nqueens	16	4,601,178	1	59,815,326	124,231,831
sort	20	1,157,801	1	2,490,380	6,138,563
strassen	10	19,608	1	137,279	294,169
uts	1,576	4,112,897	1	4,112,920	12,338,741
botsalgn	4	0	1	2,312	4,626
botsspar	4	300	1	292,547	585,398



TSan-SPD3: https://github.com/lechenyu/tsan-spd3