PUG : A Symbolic Verifier of GPU Programs

Gudong Li[1] Ganesh Gopalakrishnan[1] Robert M Kirby[1] Dan Quinlan[2]

[1] University of Utah

[2]Lawrence Livermore National Laboratory

Introduction

PUG is a automated verifier for GPU programs written in C/CUDA

PUG verifies GPU kernels for

- Data Races (e.g. unexpected conflicts on shared variables)
- Barrier mismatches (e.g. deadlocks)
- Totally wrong results (eg : off by one)
- Weak memory model related bugs

SMT-based correctness checking methods for these errors are often more *scalable*, *general* and *modular*

Overview of PUG

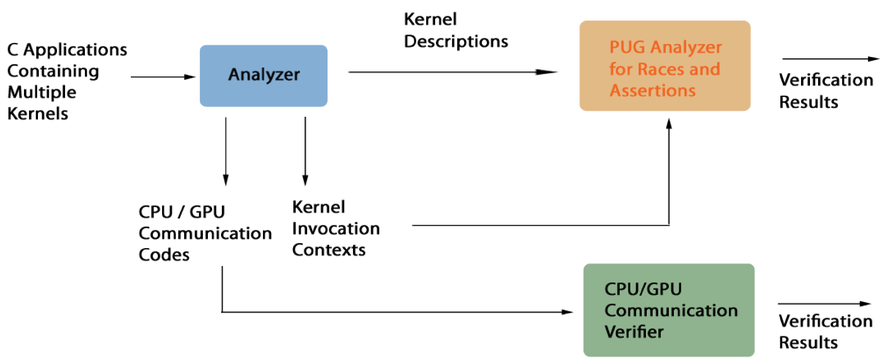


Fig 1 : PUG Architecture

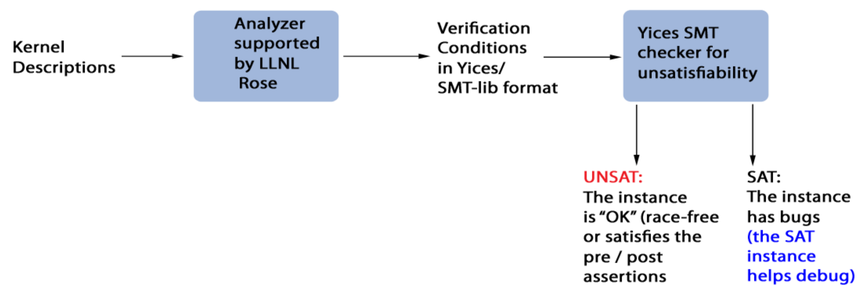
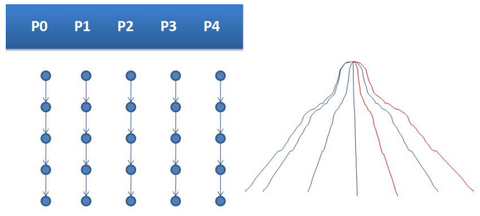


Fig 2 : Workflow of Kernel Verification

Traditional Testing : Problems

Testing is ineffective

- It usually gets stuck in some of the interleavings
- It has to be lucky to choose the right initial values



TOTAL > 10 billion interleavings

Symbolic Methods

- Cover all possible input values in one go
- Cover all interleavings in one go
- Can smartly avoid un-necessary interleavings
- Made possible by CONSTRAINT SOLVING ENGINES (Satsifialbility Modulo Theory solvers or SMT solvers) which work far better than exhaustive search

Example CUDA Kernel

```
void __global__ kernel
(int *a, int b, int N) {
 int idx = blockIdx.x * blockDim.x +
 threadIdx.x;
 if (idx < N) {
 a[idx] = a[idx] + b;
 /* Data Race Access */
 // a[idx] = a[idx-1] + b;
 }
}
```

Race Caught by PUG

- Constraint Formula encoded by PUG is *satisfiable*

```
(= idx1@t1 0b0000000001)
(= idx1@t2 0b0000000000)
```

- Witness shows data race when idx@t1 is 1 and idx@t2 is 0

PUG Encodes the Kernel as Constraint Formula

```
(∧ (∧ (≠ t1.x t2.x)
(< t1.x bdim.x)
(< t2.x bdim.x))
(∧ (= idx1@t1(+ (* bid.x bdim.x) t1.x))
(ite (< idx1@t1 N0) true (= a1 a0)))
(∧ (= idx1@t2(+ (* bid.x bdim.x) t2.x))
(ite (< idx1@t2 N0) true (= a1 a0)))
true
true
(∨ (∧ (< idx1@t1 N0)
(< idx1@t2 N0)
(= idx1@t1 idx1@t2))
(∧ (< idx1@t1 N0)
(< idx1@t2 N0)
(= idx1@t1 idx1@t2))))
unsatisfiable
```

References

[1] Gudong Li, Ganesh Gopalakrishnan, Robert M Kirby, Dan Quinlan, " PUG : A Symbolic Verifier for CUDA Programs", *Workshop on Language, Compiler and Architecture Support for GPGPU, PPOPP'10* (2010)

Results

Program	Time (Pass)	Time (Bug)
Matrix Transpose	1.1	< 0.1
Matrix Multiply	< 0.5	< 0.1
Scan Large	2.7	1.2
Nbody	5.7	2.4

Table: Race Checking Results

Program	Corr	Bug	Corr (+POR)	Bug (+POR)
Reduction(n=4)	T.O	240	2.84	0.16
Bitonic Sort(n=4)	T.O	T.O	3.7	0.45
Scalar Product(n=4)	T.O	T.O	6.9	1.85
Matrix Transpose(n=4)	T.O	325	3.85	0.24

Table: Assertion Checking Results

Supported by

- "Formal Specification, Verification and Test Generation for Multi-core CPUs"
- "CPA-DA: Formal Methods for Multicore Shared Memory protocol Design"
- Microsoft