Extending Polyhedral Model for Analysis and Transformations of OpenMP Programs

Prasanth Chatarasi and Vivek Sarkar

Department of Computer Science, Rice University

http://habanero.rice.edu

Contact: prasanth@rice.edu

(1) Introduction & Motivation

- **Moving towards Exa-scale and Extreme-scale machines**
  Enabling applications to fully exploit them is not easy
- **Software stack**
  - Explicitly parallel programming models
  - Optimizing compilers
  - Efficient runtime systems
- **Our past work (PoPP) [PACT'15, IMPACT'15]**
  - Automatically optimize OpenMP programs
    - Task-parallel, Loop-parallel
  - Polyhedral compilation techniques
  - Fine-grained synchronization in run-time
- **Limitations of PoPP:**
  - Applicable only to set of parallel constructs that satisfy **serial-elision property**
- **Motivation:**
  - **Analyze and Transform** more generic OpenMP constructs such as parallel regions, work-sharing, barriers, sections etc. — **PolyOMP**

(2) Extending Polyhedral Representation

- **Polyhedral representation of a statement instance**
  - Domain + **Schedule** + Access relations
- **Domain** - Set of statement instances
- **Schedule** - Ordering among statement instances
  - Extended with Phase, Space dimensions
  - **Phase - Computation phase id**
  - **Space - Thread id that executes it**
- **Access relations** - Array subscripts referenced

3) Analysis - Data race Detection

- **Step-1: Conditions for Data-race b/w statement instances S&T**
  - S & T should touch the same memory location and one being a write
  - S & T should be in same phase of computation
  - S & T should be executed by different threads
- **Step-2: Use Z3 solver for the existence of the solutions**
- **Implementation details**
  - Frameworks:
    - PET (Polyhedral extraction tool), Z3 (SMT solver)
  - Implementation is **in-progress**
- **Preliminary Experimental Evaluation**
  - Manual evaluation on OMP SRC benchmarks
  - Data races category
    - Accessing overlapped memories
    - Inserting barrier statements
    - Conditional executions among threads.
  - All races were detected and Z3 solver took less time (1-2 seconds) for the existence of the solutions

	Pathg (LCTES'12)	OAT (ICPP'13)	ompVerify (IWOMP'11)	PolyX10 (PPoPP'13)	PolyOMP (Ours)
Target parallelism	OpenMP Work sharing (loop), Barriers, Atomic	OpenMP Work sharing (loop, sections), locks, barriers, master, single, critical, atomic	OpenMP Work sharing (loop)	X10 Async/ finish parallelism	OpenMP SPMD regions, work sharing (loop, sections), master, single, barrier, doacross (OpenMP 4.1)
Approach	Extended Thread Automata	Symbolic execution	Polyhedral model	Polyhedral model	Extended Polyhedral model
Guarantees	Per- no.of threads, chunk size	Per- no. of threads	Per- Program	Per- Program	Per- Program
Dependent on scheduling techniques	Yes	Yes	No	No	Yes
Nature of analysis	Precise[1]	False +Ve and False -Ve	Precise[1]	Precise[1]	Precise[1]

[1] In case of affine array subscripts and static affine control flow

4) Transformations

- **Data dependence relations from S to T**
  - S & T should touch same memory location and one of them is write
  - S should happens-before T (Computed based on phase and space)
- **Transformations**
  - Fusion of work-sharing directives
  - Removal of redundant barriers
  - Fusion of SPMD regions
- **Example**

```
#pragma omp parallel {
  #pragma omp for schedule(static, c) nowait
  for(int i = 0; i < N; i++)
    A[i] = B[i]; // S1
  #pragma omp barrier      - Can be removed
  #pragma omp for schedule(static, c)
  for(int j = 0; j < N; j++)
    C[j] = A[j]; // S2
}
```

	PoPP (PACT'15)	PIR-OPT (TOPLAS'13)	OMPD (PPoPP'12)	PolyOMP (Ours)
Target parallelism	OpenMP task-level and loop-level parallelism	X10 async-finish parallelism	Extended OpenMP	SPMD regions, work sharing (loop, sections), master, single, barrier, doacross (OpenMP 4.1)
Ordering relations in data dependences	Intersection of conservative dependences with HB relations	HB relations	Dataflow analysis	HB relations
Optimizations	Loop-level transformations	Finish-elimination, forall-coarsening, loop-chunking	Communication reduction	SPMD fusion, Barrier removal, Worksharing fusion

- **Acknowledgements** Jun Shirako and Habanero Extreme Scale Software Research Project