Motivation

- Maintaining optimized programs for different devices is costly
- Programs written once should run on different devices with performance, which is known performance portability

Limitations of Current Practice

- OpenCL is not performance portable

- Composition-based languages highly rely on high-performance base-rule implementations

TANGRAM Platform

- TANGRAM adopts codelet programming model
  - A codelet is defined as a code snippet reusable for one or many kernels
  - Users write interchangeable alternative codelets, and corresponding composition and partition rules for a computation pattern (called spectrum)
  - We do Not ask users to write multiple versions of kernels
- TANGRAM supports recursive composition to adapt to different hierarchies of devices and cooperative codelets for SIMD architectures
- TANGRAM also provides performance tuning annotation to enable parameterization

Performance Portability

- TANGRAM’s device specification model is highly extensible to support CPU SIMD unit, GPU Warp, ILP, and GPU Dynamic Parallelism

TANGRAM Workflow

Device Specification:

- Specified composition rules: $S$, regroup($S$), distribut(e), $S$, regroup($S$), distribut(e), $S$, regrou(p), distribut(e), $S$, regroup($S$)
- Compute($S$), distribut(e), $S$, regroup($S$)
- Distribut(e), $S$, regroup($S$)
- Reorder($S$), distribut(e), $S$, regroup($S$)

Experimental Results

- TANGRAM delivers 70% or higher performance compared to highly-optimized libraries, such as Intel MKL, NVIDIA CUBLAS, CUSPARSE, or Thrust, or experts’ optimized benchmarks in Rodinia

Conclusion

- We propose TANGRAM, a programming system for performance portability across devices
- Our results show TANGRAM can achieve promising performance across devices