Generating Configurable Hardware from Parallel Patterns

Raghu Prabhakar, David Koeplinger, Kevin J. Brown, HyoukJoong Lee, Chris De Sa, Christos Kozyrakis, Kunle Olukotun

Motivation
- Increasing interest to use FPGAs as accelerators
  - Key advantage: Performance/Watt
- Key domains:
  - Big data analytics, image processing, financial analytics, scientific computing, search

Problem: Programmability
- Verilog and VHDL too low level for software developers
- High level synthesis (HLS) tools need user pragmas to help discover parallelism
- Use Higher-level Abstractions
  - Parallel Patterns
  - Productivity: Developer focuses on application
  - Performance:
    - Capture Locality by tiling application programs
    - Exploit Parallelism at multiple nesting levels
  - Smart compiler generates efficient hardware

New Parallel Pattern: multiFold
- multiFold describes tiled computation
- Polychdral analysis limits data access patterns to affine functions of loop indices

Pattern Transformation 1: Strip Mining
- Transform parallel pattern → nested patterns
- Strip mined patterns enable computation reordering
- Insert copies to enhance locality
- Copies guide creation of on-chip buffers

Parallel Patterns
\[ \text{map}(d)[i] \Rightarrow 2^x(k(i)) \]

Strip Mined Patterns
\[ \text{multiFold}(d/b)[i] \Rightarrow \text{xTl}(i) \]
\[ xTl = \text{x.copy}(b0+i, b1+j) \]
\[ \{(i,j), \text{map}(b0,b1)[i,j] \} \]

Pattern Transformation 2: Pattern Interchange
- Reorder nested patterns
- Move ‘copy’ operations out toward outer pattern(s) to improve locality

Strip Mined Patterns
\[ \text{multiFold}(m/b0,n/b1)[i,j] \Rightarrow \text{xTl} = \text{x.copy}(b0+i, b1+j) \]
\[ \{(i,j), \text{multiFold}(p/b2)[k] \} \]

Interchanged Patterns
\[ \text{multiFold}(m/b0,n/b1)[i,j] \Rightarrow \text{xTl} = \text{x.copy}(b0+i, b1+j) \]
\[ \{(i,j), \text{multiFold}(p/b2)[k] \} \]

Experimental Results: Performance and Resource Utilization
- Speedup with tiling alone: up to 15.5x
- Speedup with metapipelining: up to 39.4x
- Tiled designs have fewer off-chip data loaders and storers
- Minimal (often positive!) impact on resource usage

Contact Information
E-mail Addresses (Student)
raghu@cs.stanford.edu
dkoeplin@stanford.edu