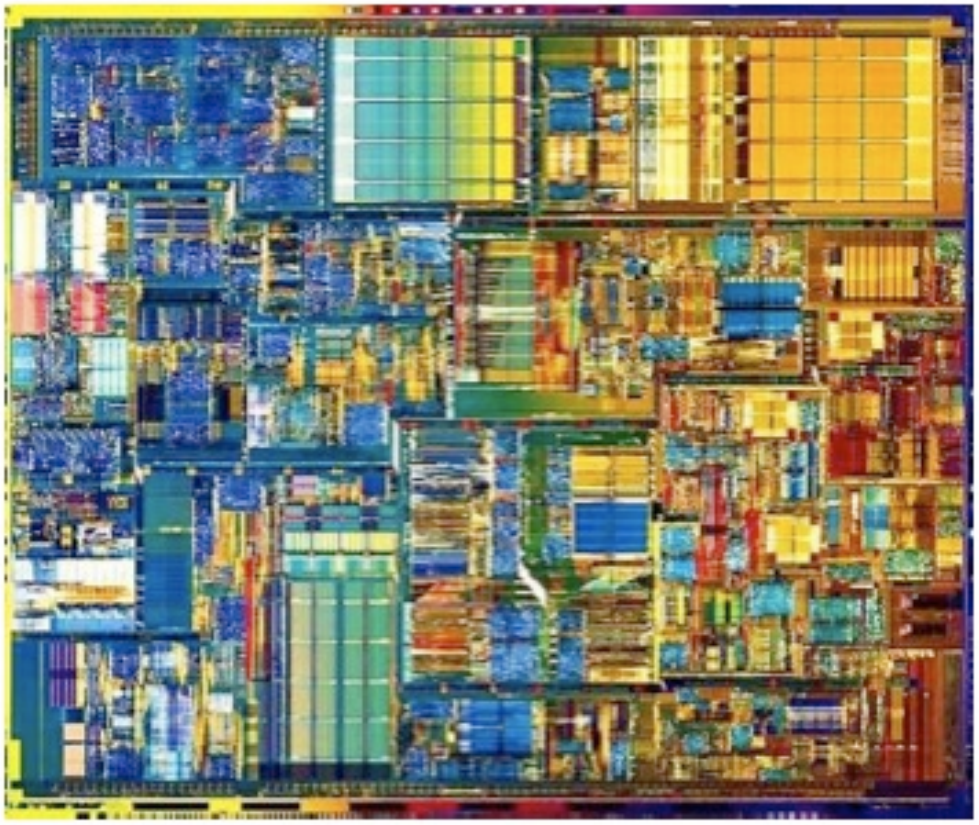
Instruction Scheduling for a Tiled Dataflow Architecture

Martha Mercaldi
Steven Swanson, Andrew Petersen, Andrew Putnam, Andrew Schwerin
Mark Oskin and Susan Eggers

University of Washington

Tiled Architectures



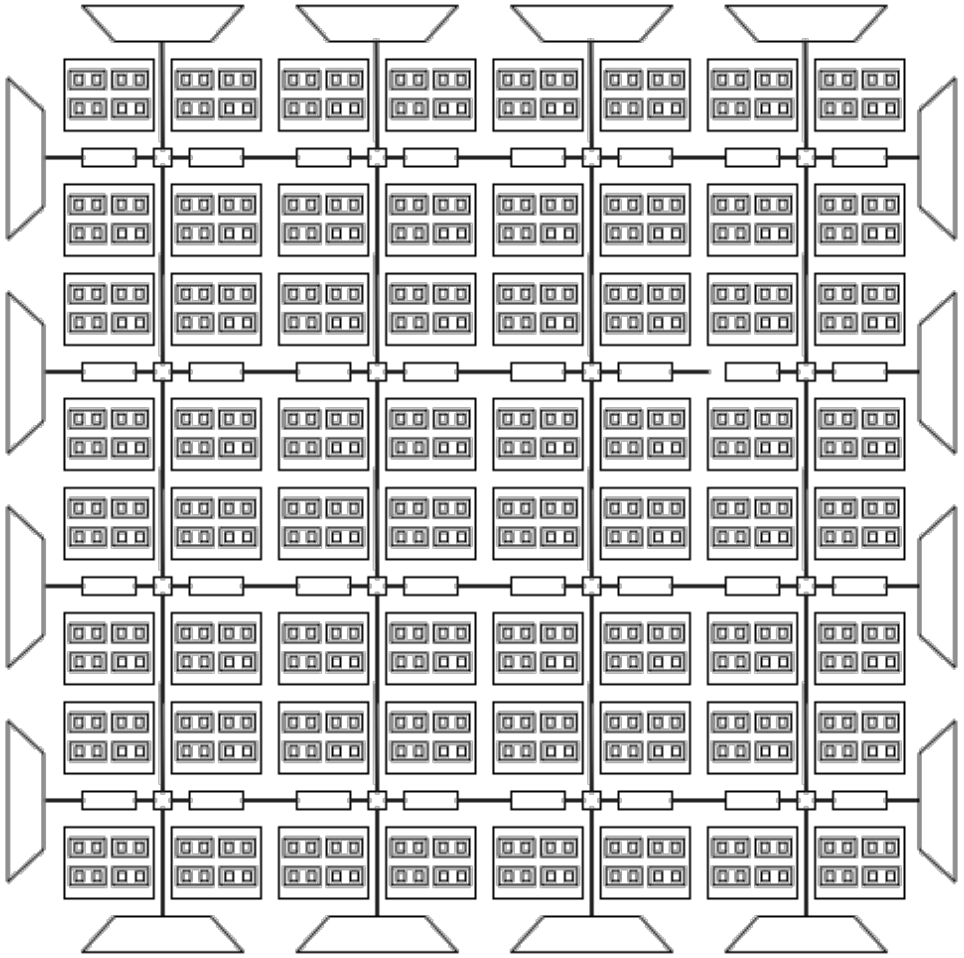
- Scalability
  - Short wires
- Complexity
  - Simple, replicated unit
- Power
  - Turn off unneeded tiles

What should execute where?

Talk Outline

- WaveScalar Instruction Placement
- Hierarchical Placement
- Summary of Preliminary Algorithm Survey
- DAWG Placement Algorithm
- Conclusions

WaveScalar Processor

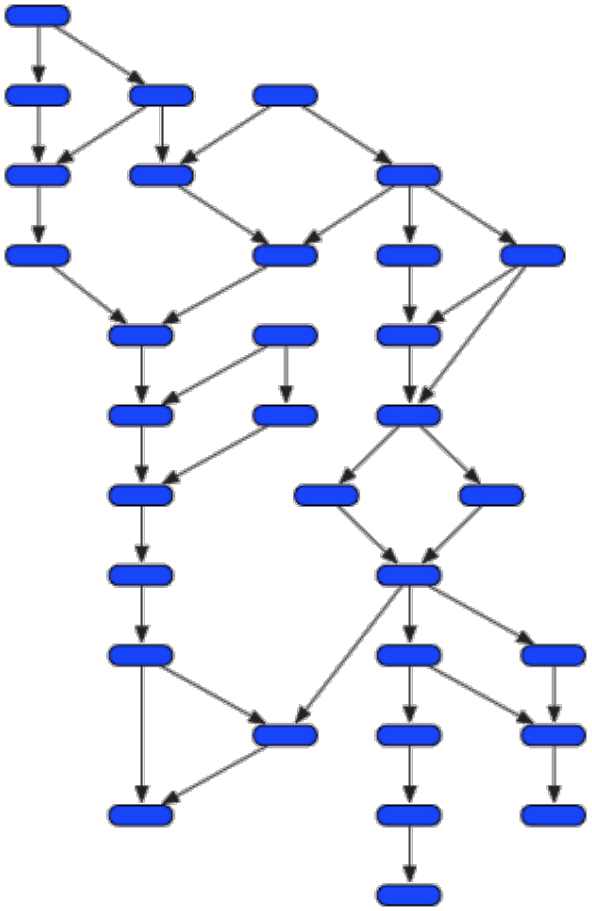


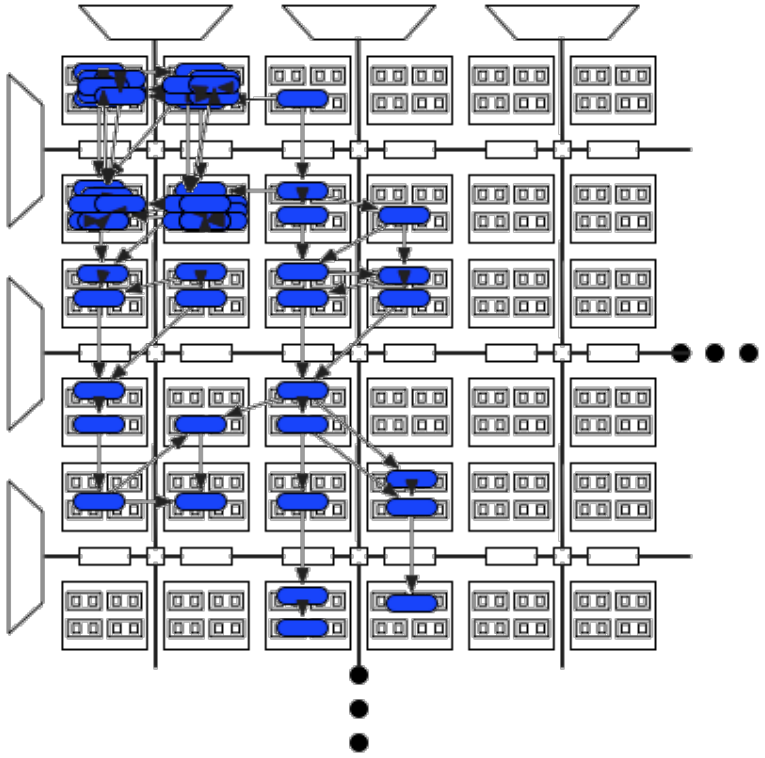
Dataflow execution model

Regular, hierarchical, microarchitecture

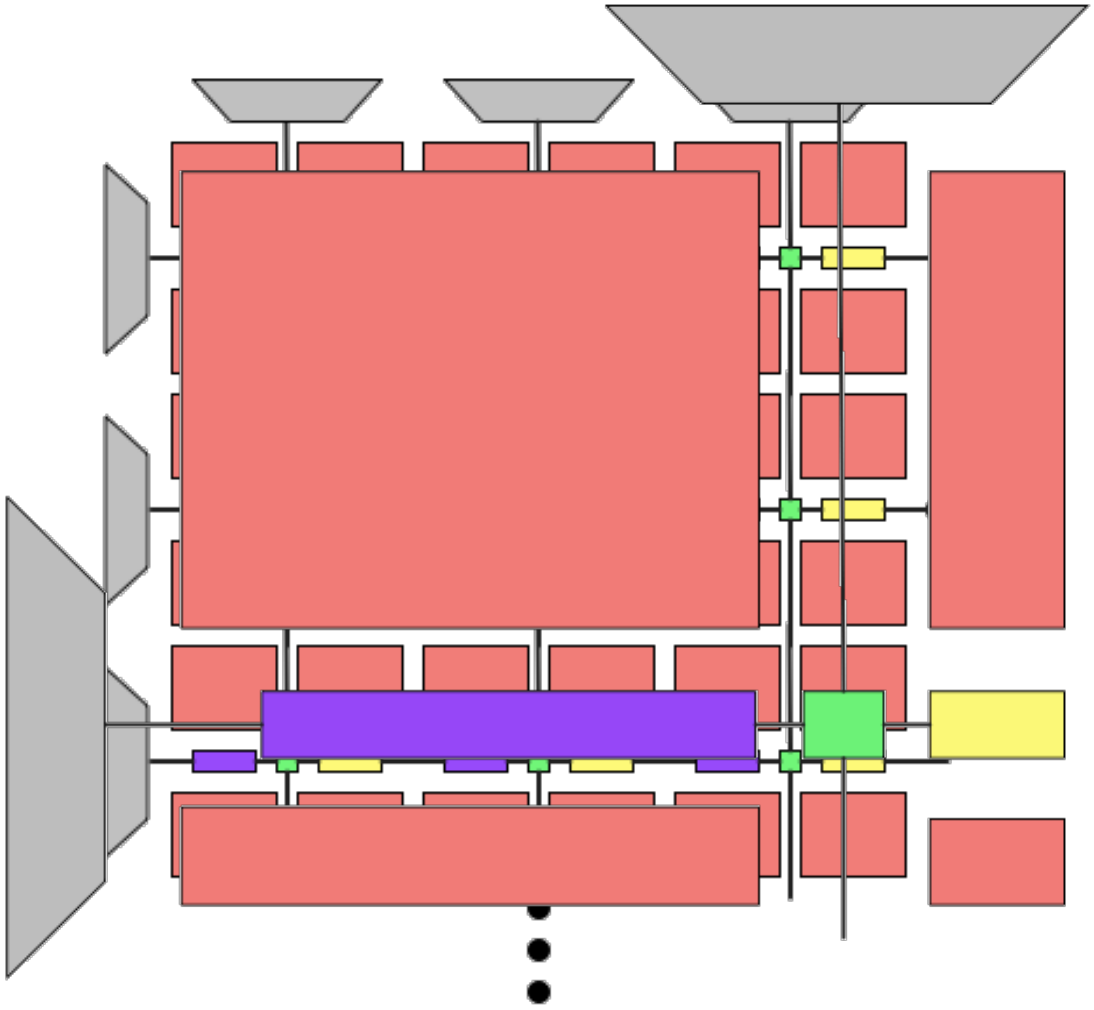
[ISCA 2006]

WaveScalar Application Execution





WaveScalar: Processor



Domains

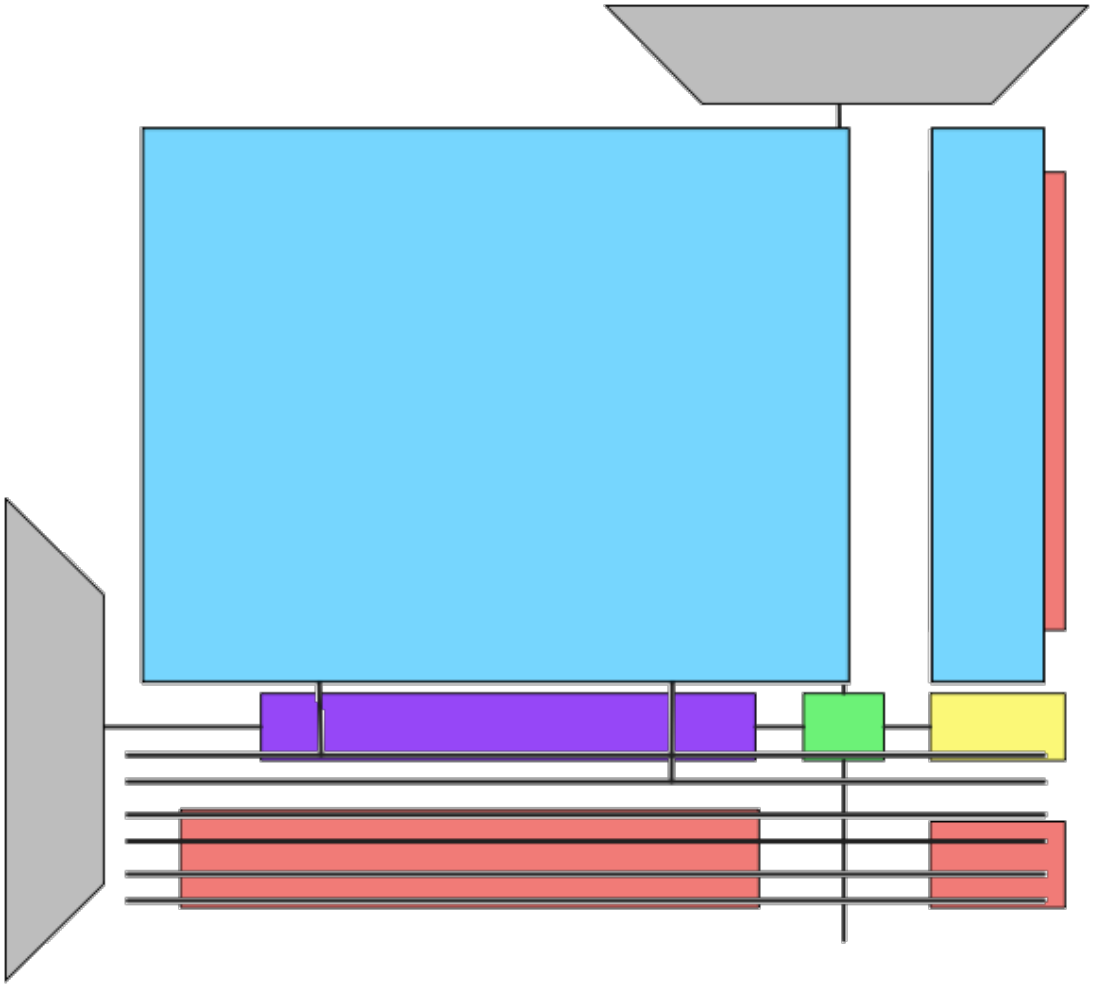
Network Switches
- packet switched
- min 7 cycle latency

Store Buffers

L1 Data Caches

L2 Data Cache

WaveScalar: Domain

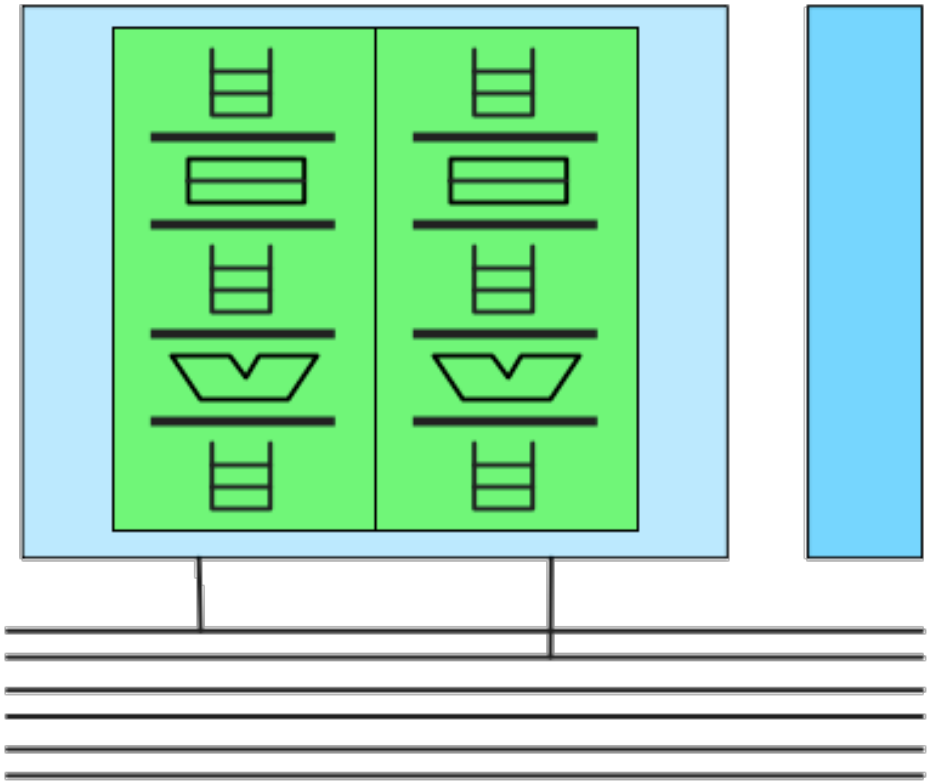


4 Pods

Crossbar Interconnect

Fixed, 4-cycle latency

WaveScalar: Pod



2 Processing Elements (PEs)

1-cycle operand latency

Talk Outline

WaveScalar Instruction Placement

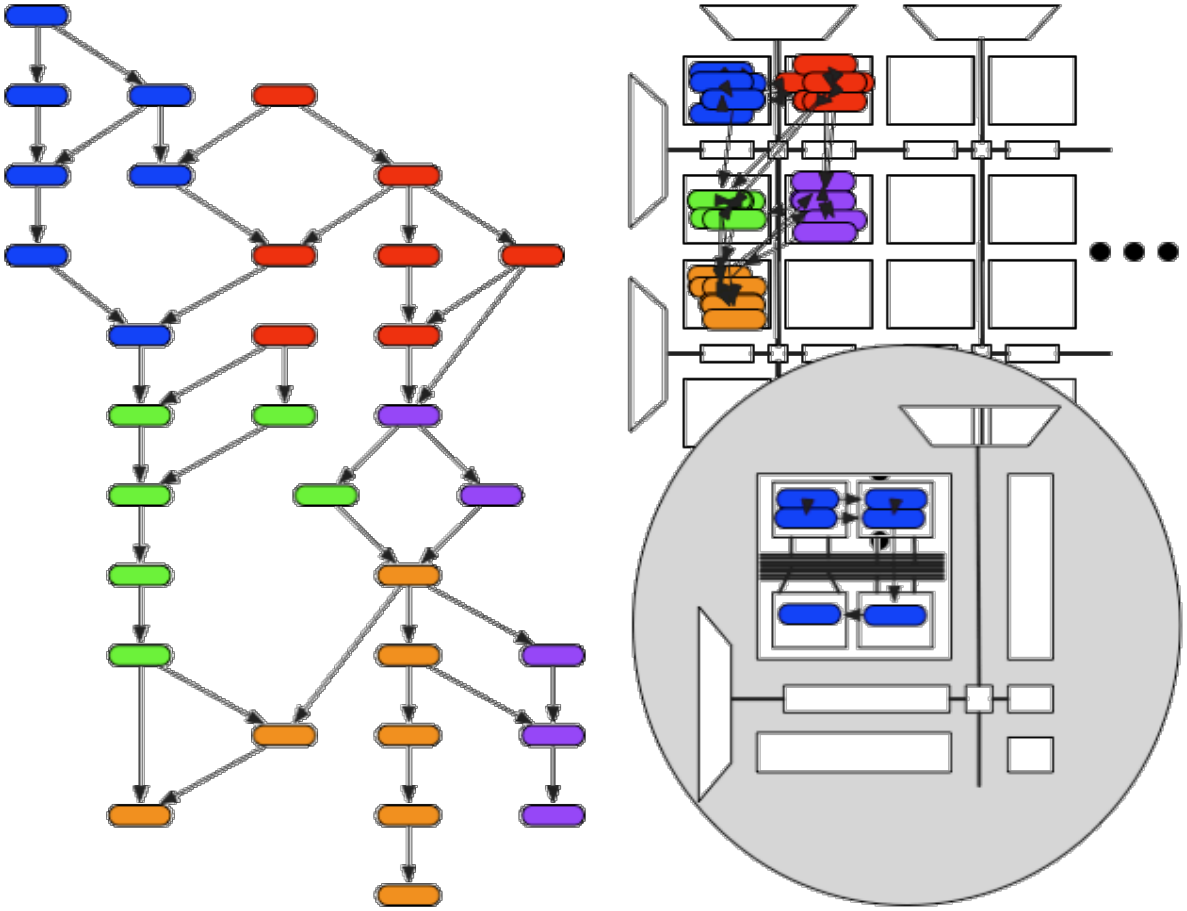
Hierarchical Placement

Summary of Preliminary Algorithm Survey

DAWG Placement Algorithm

Conclusions

Hierarchical Placement



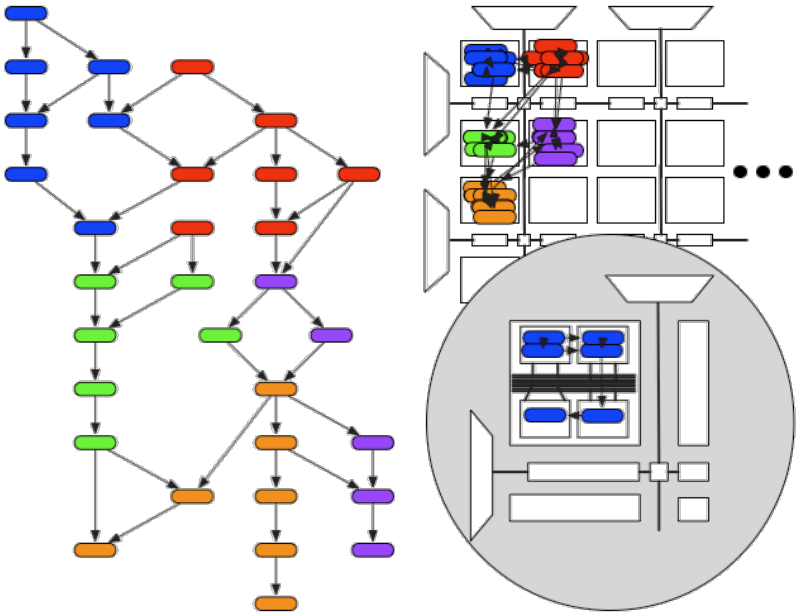
Coarse

Placement

Fine

Placement

Why Hierarchical?



Processor is hierarchical

- *Different network designs inside and outside domains*
- *Consider coarse and fine placement effects separately*

Manage complexity

- *Two subproblems smaller than total problem*

Talk Outline

WaveScalar Instruction Placement

Hierarchical Placement

Summary of Preliminary Algorithm Survey

DAWG Placement Algorithm

Conclusions

Preliminary Algorithm Study

Coarse Placement

- By Function
- By Topology
- By Execution Order

Min Operand Latency ⇒ Best Placements

Preliminary Algorithm Study: BUG

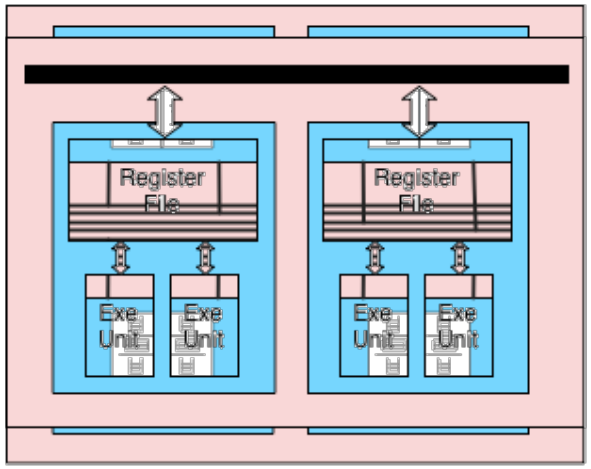
Fine Placement

- Bulldog VLIW compiler
  [J.R. Ellis Thesis, ‘85]
- Later, Multiflow

  – Bottom-Up Greedy

  – Unified Assign and Schedule

  – By Execution Order



Preliminary Algorithm Study: UAS

Fine Placement

- Bottom-Up Greedy
- Unified Assign and Schedule
- By Execution Order

- Also for clustered microarchitectures

  [J Ozer, MICRO '98]
- Determine WHERE and WHEN an instruction will execute

Preliminary Algorithm Study: By Exe. Order

Fine Placement

- Bottom-Up Greedy
- Unified Assign and Schedule
- By Execution Order

- Profile-based algorithm

Preliminary Algorithm Study: Results

Fine Placement

- Bottom-Up Greedy
- Unified Assign and Schedule

+ Operand Latency &
-- Exe. Resource Conflicts ⇒
Better Placement

- By Execution Order

Min Operand Latency ⇒
Worst Placement

Min Operand Latency &
Most Exe. Resource Conflicts ⇒
Worst Placement

Talk Outline

WaveScalar Instruction Placement

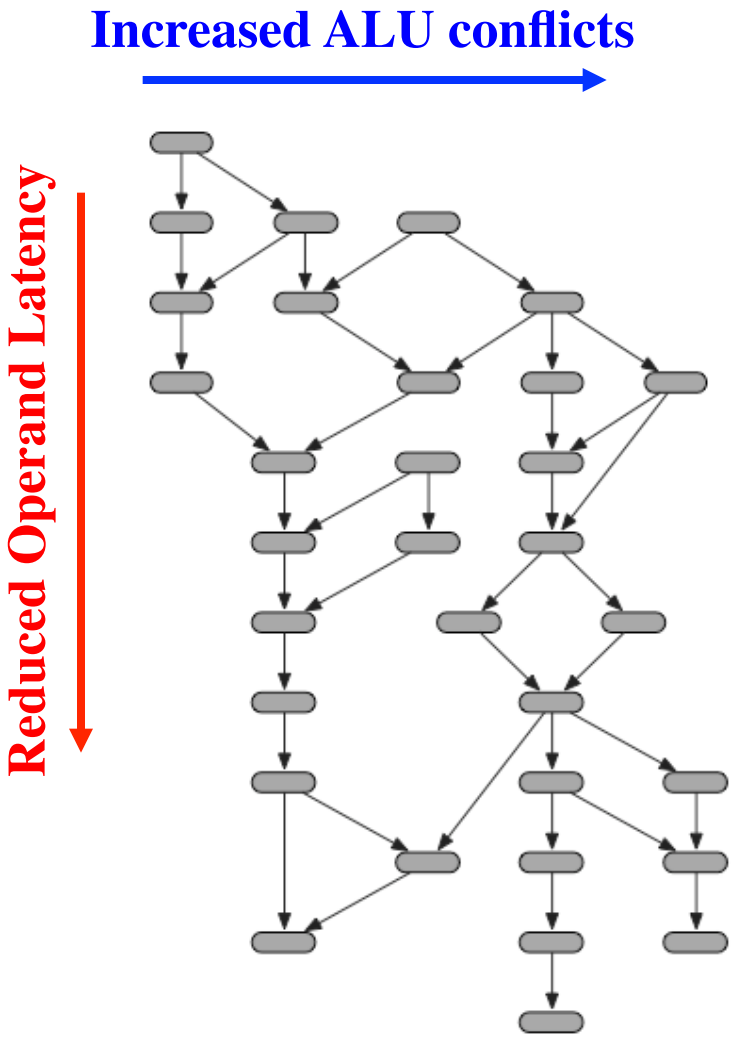
Hierarchical Placement

Summary of Existing Algorithm Survey

DAWG Placement Algorithm

Conclusions

Exploring Tradeoff



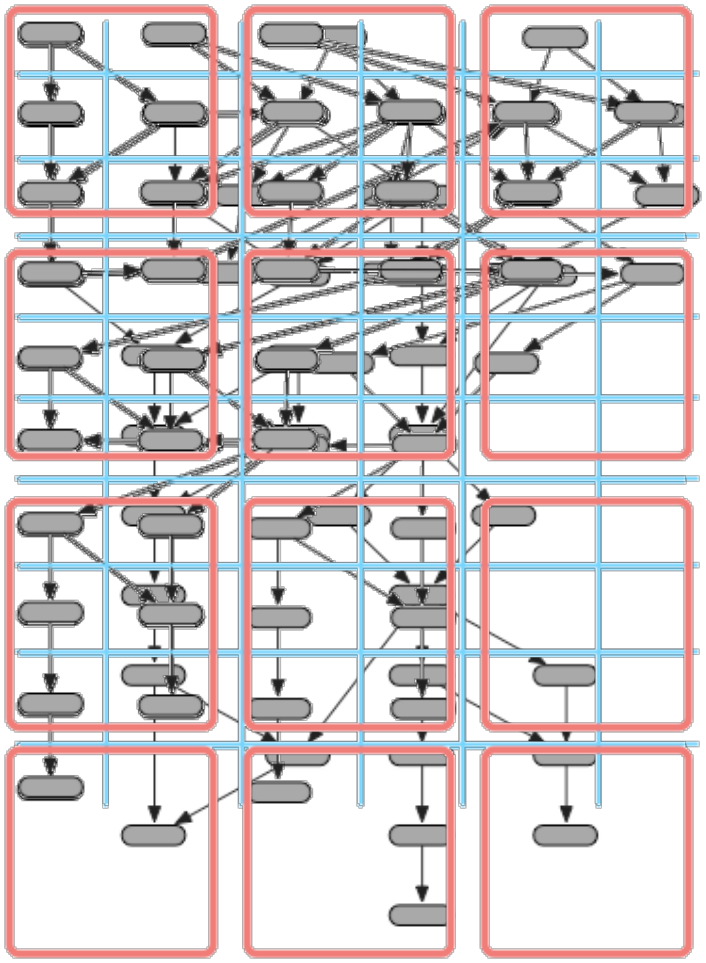
DAWG Placement

Depth And Width Graph Placement

1. `create_subgraphs(max_depth,max_breadth)`
2. `place_subgraphs(dep_degree)`

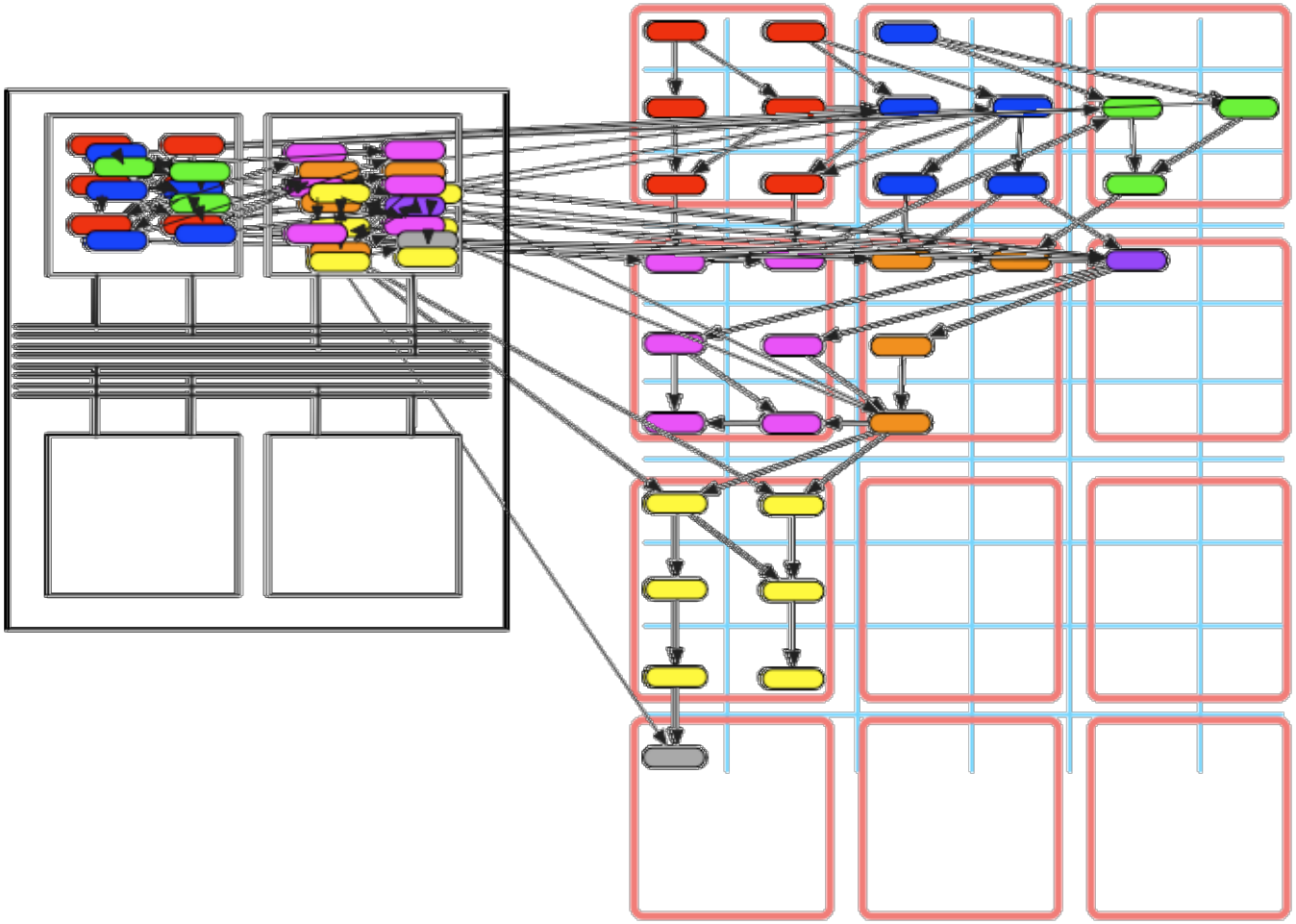
DAWG Placement:

1. `create_subgraphs(max_depth,max_breadth)`



DAWG Placement:

`2. place_subgraphs(dep_degree)`



DAWG Placement as a Vehicle

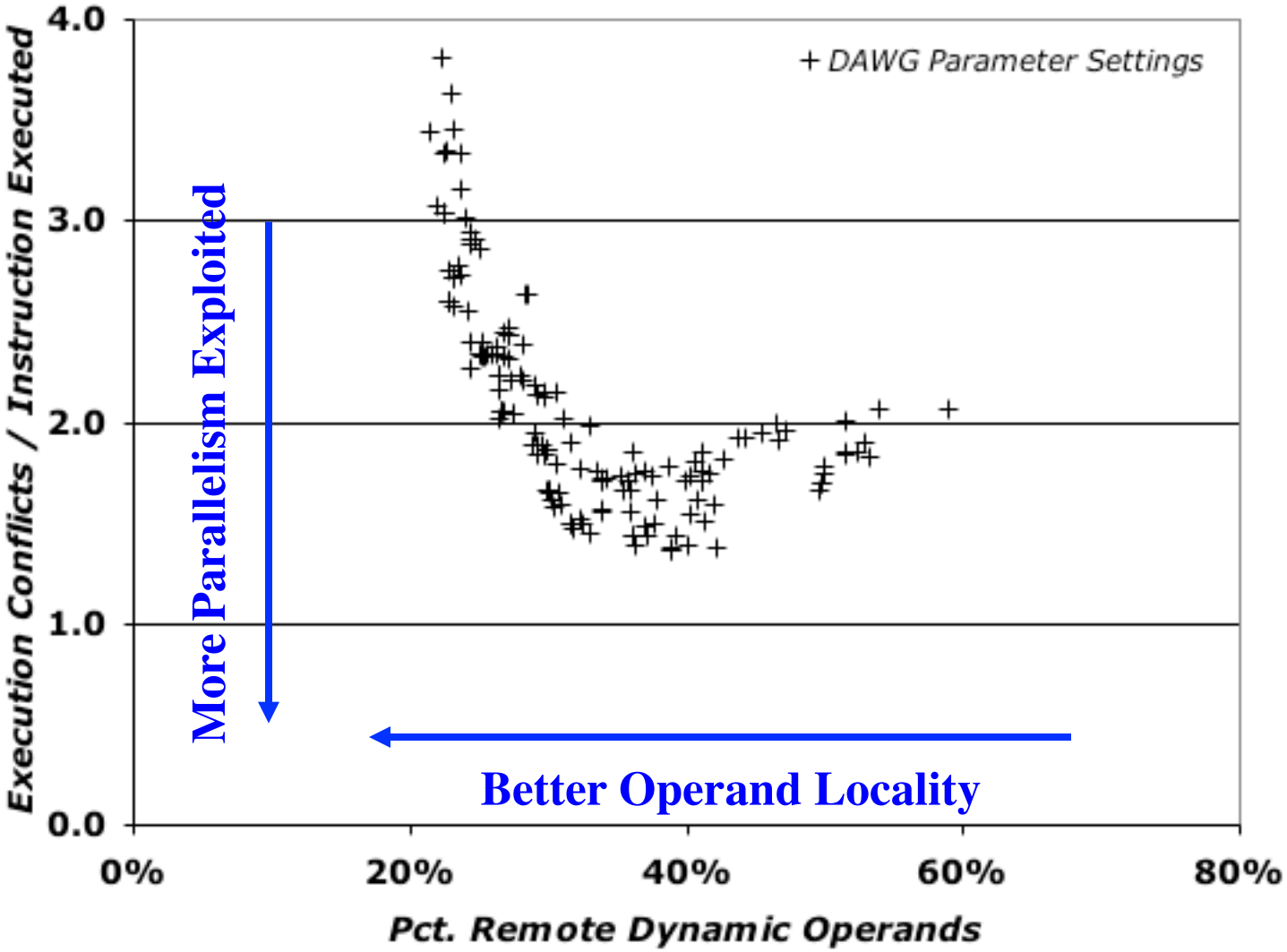
1. **`create_subgraphs(max_depth,max_breadth)`**
2. **`place_subgraphs(dep_degree)`**

Explore parameter space ⇒

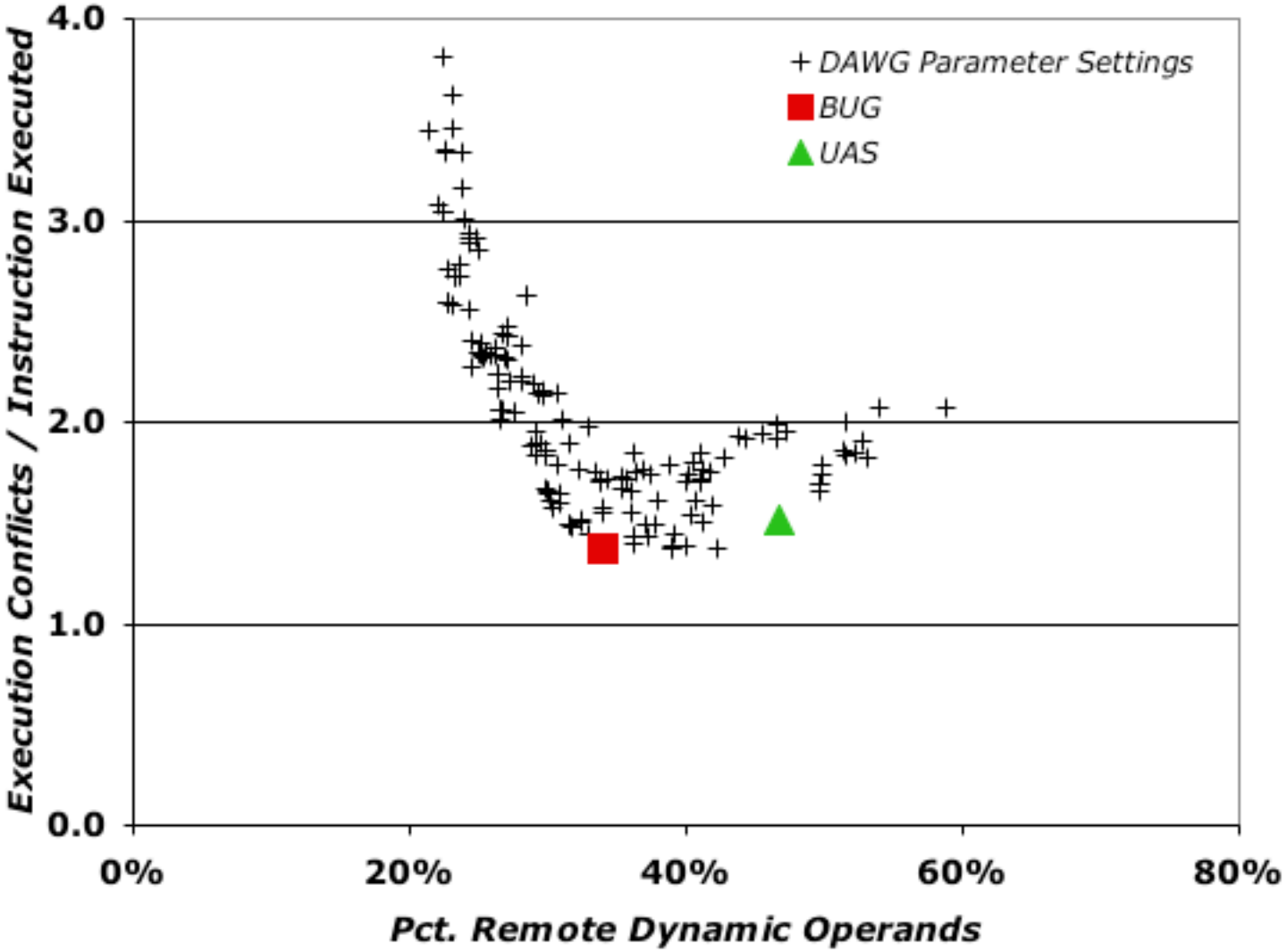
explore latency/conflict tradeoff

```
max_depth = {2,4,8,12,16,32,50,64,128}
max_breadth = {1,2,3,4,6,10}
dep_degree = {.1,.5,.9}
```

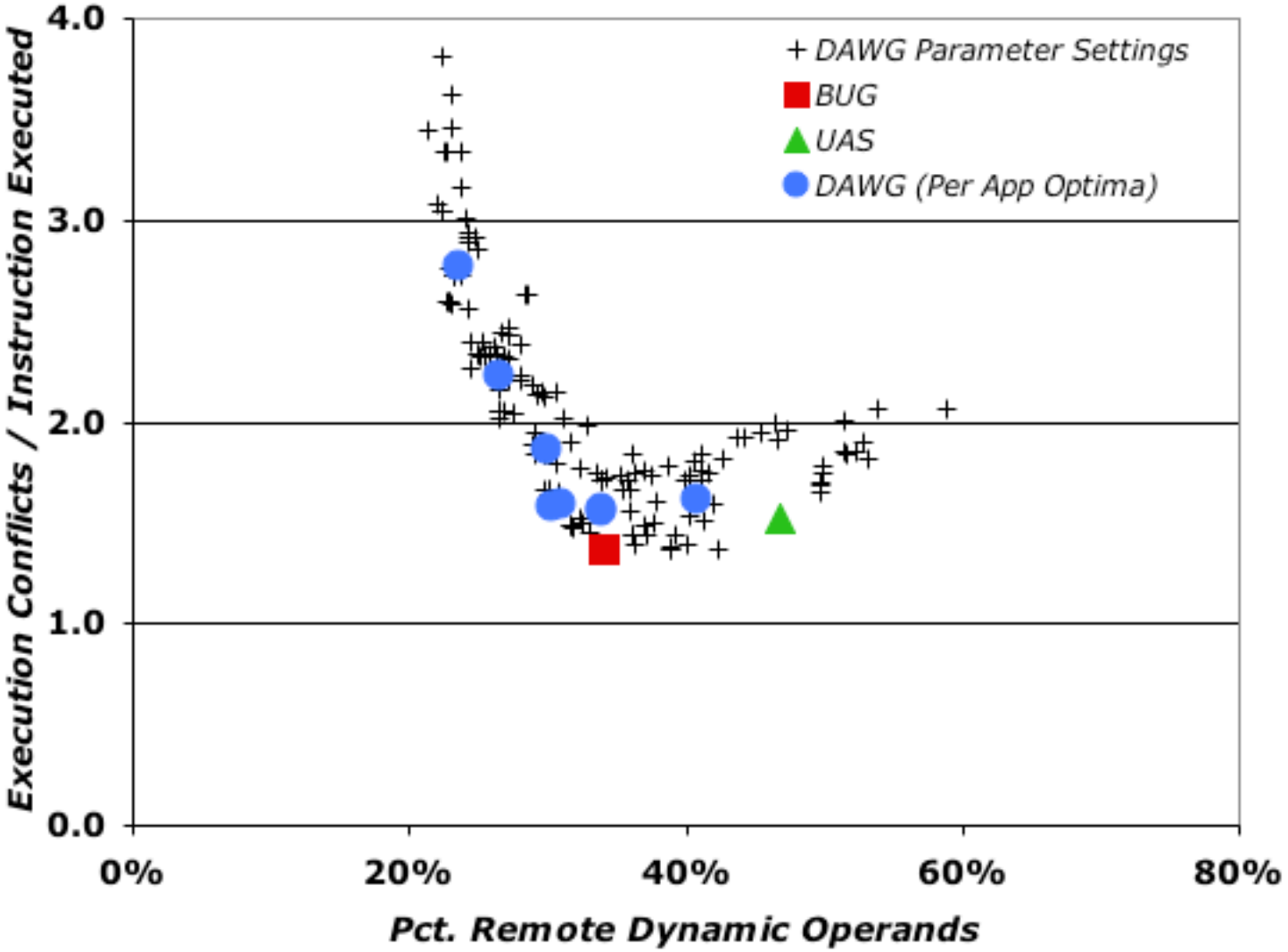
DAWG Placement: Design Space



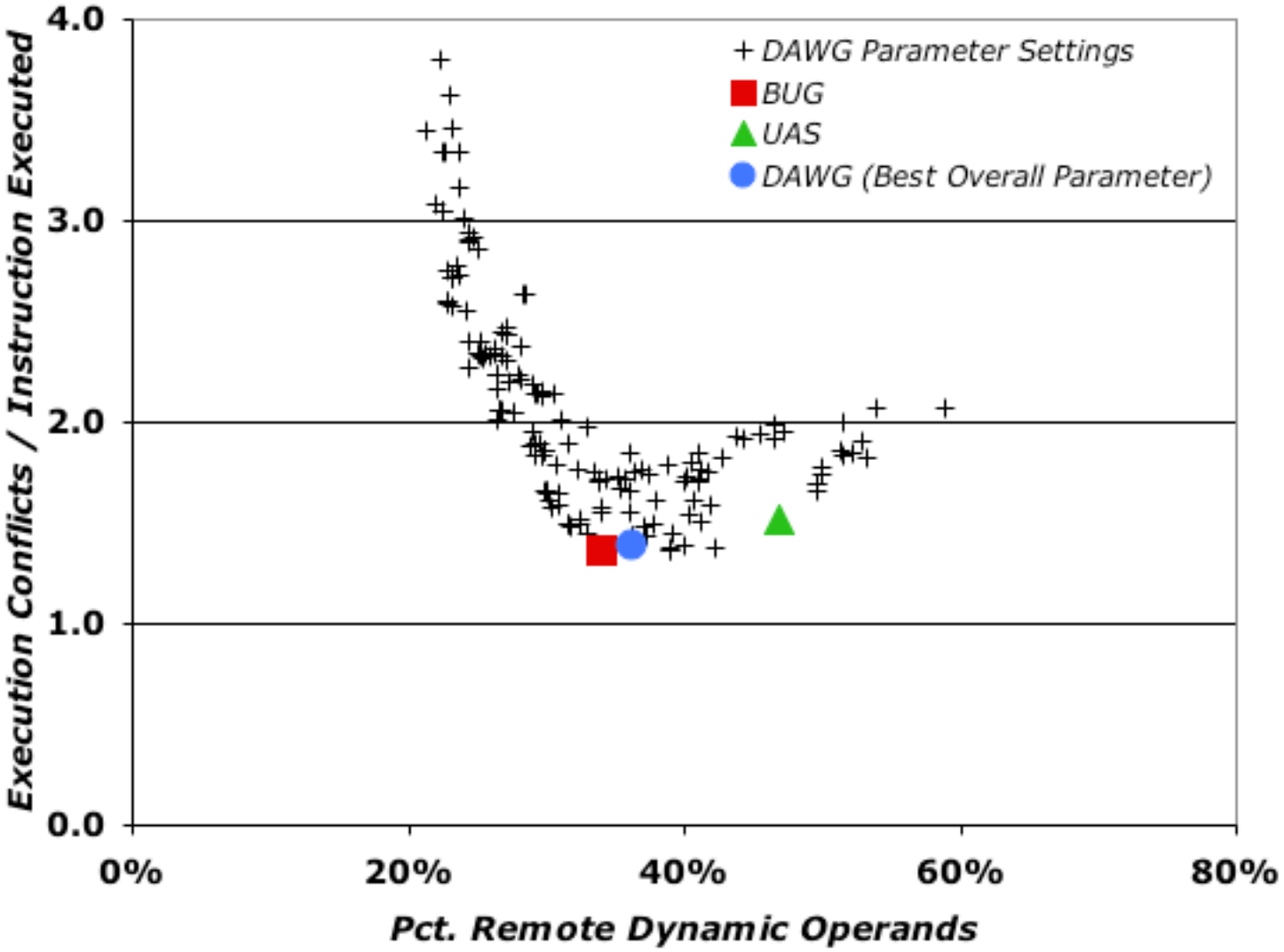
DAWG Placement: Design Space



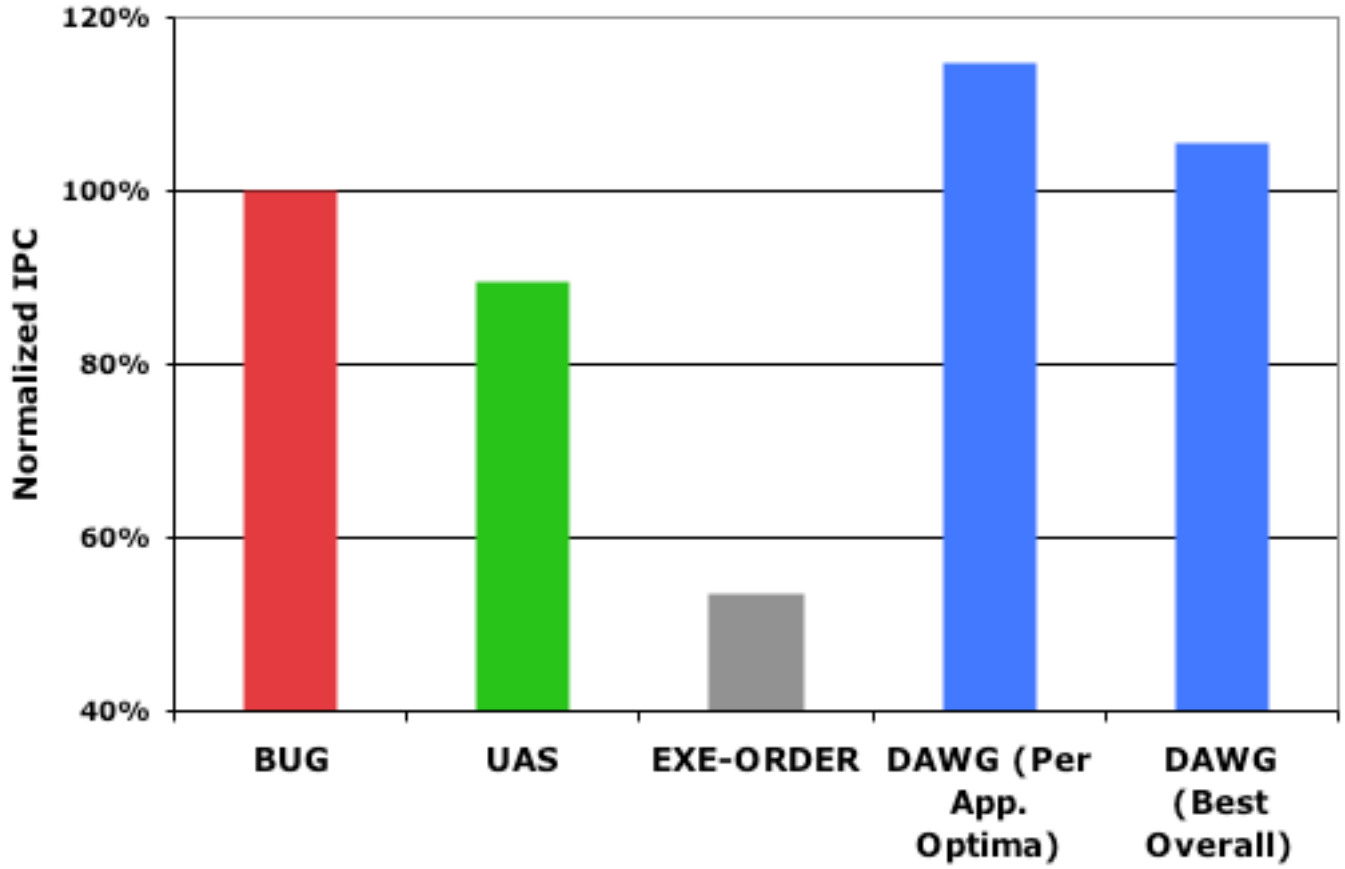
DAWG Placement: Design Space



DAWG Placement: Design Space



DAWG Placement: Performance



Conclusions

Hierarchical placement well-suited to WaveScalar

Correct balance between parallelism and operand communication latency essential

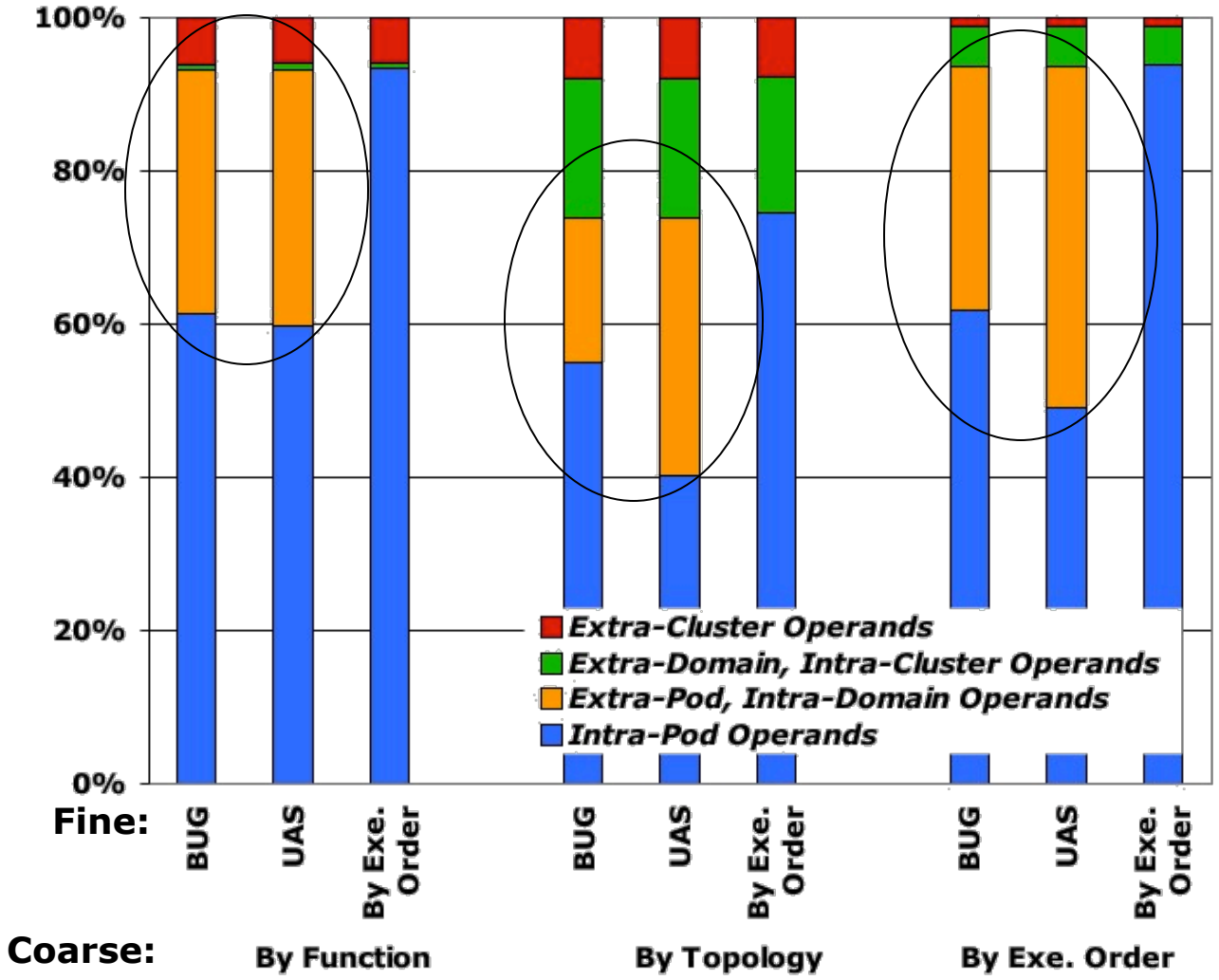
DAWG Placement is tunable to match balance to architecture and application

For more information:

http://wavescalar.cs.washington.edu

WaveScalar Team

Operand Traffic Distribution



Execution Conflicts