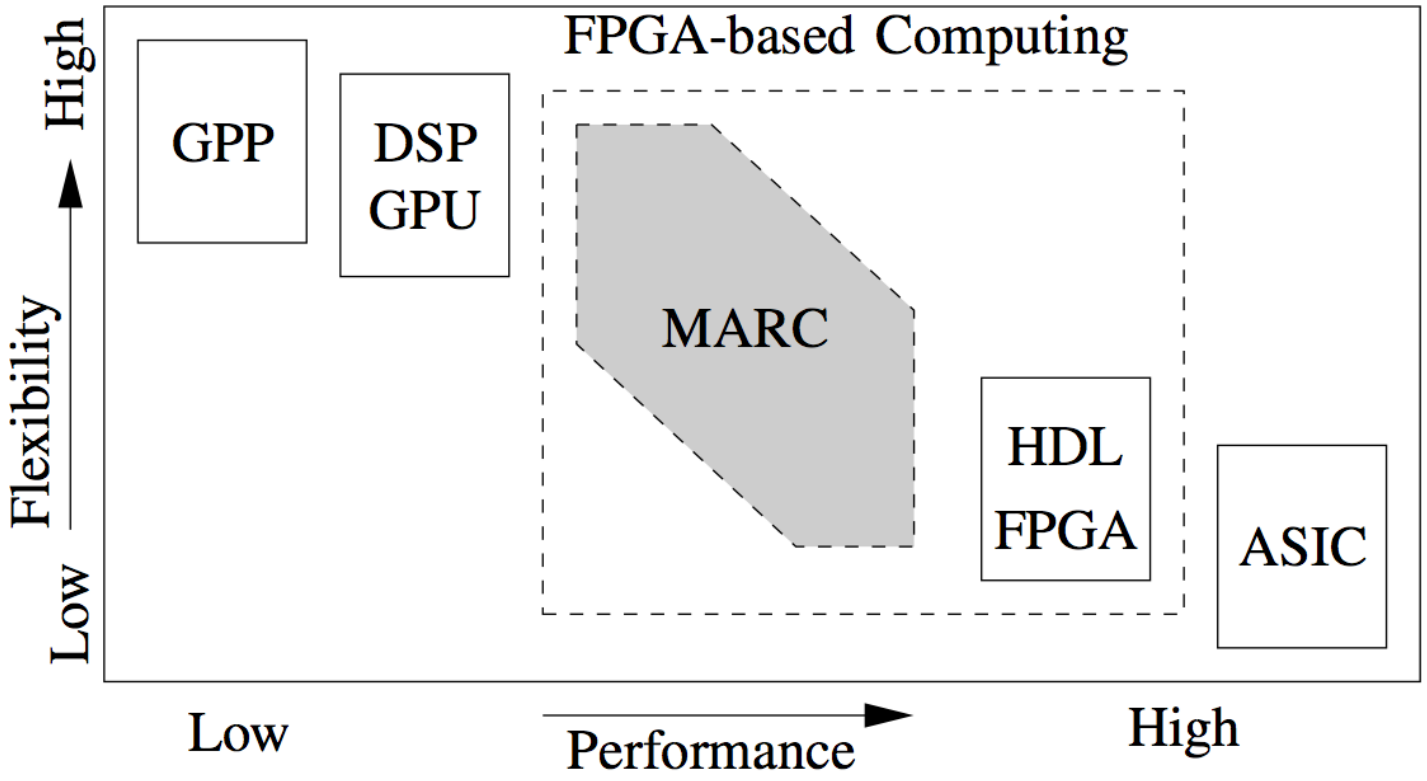
MARC: A Many-Core Approach to Reconfigurable Computing

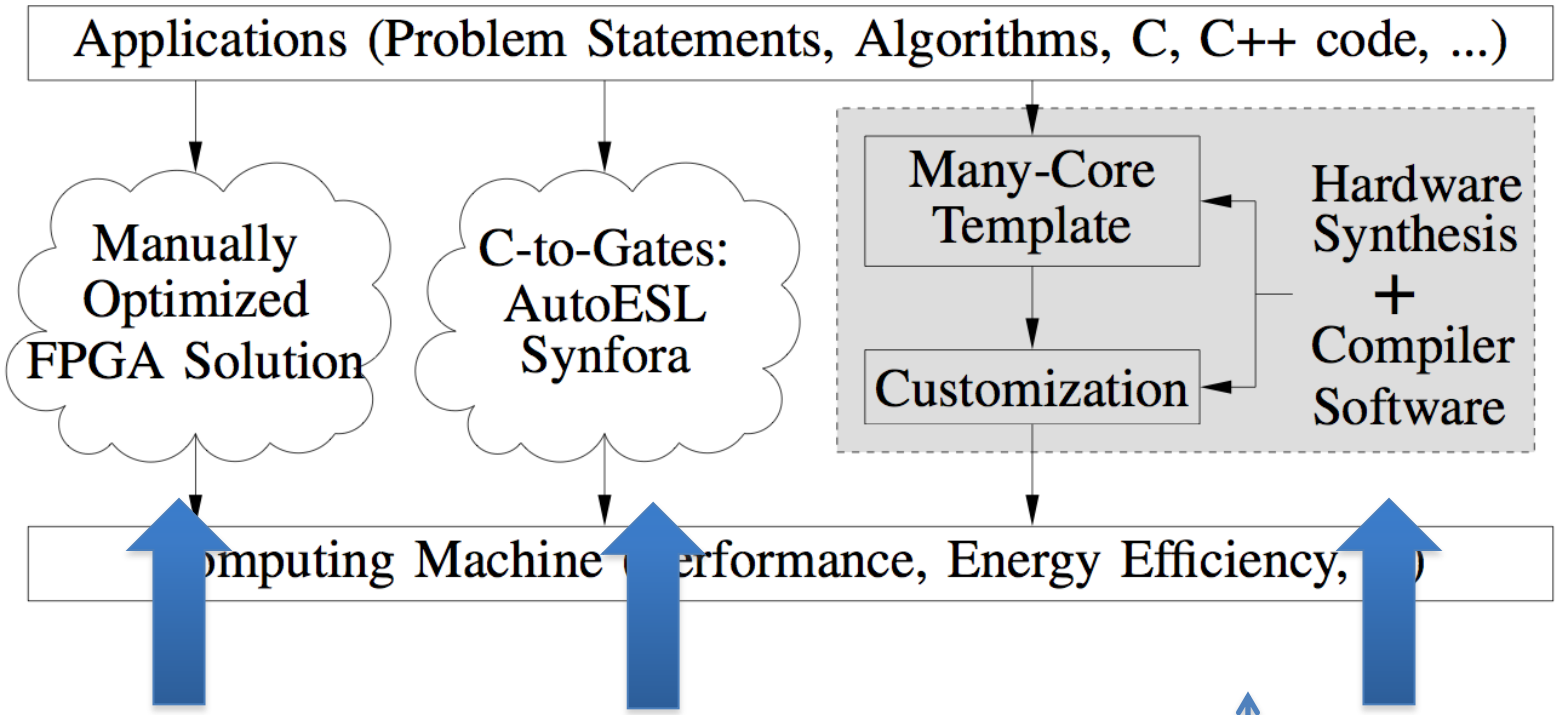
Ilia Lebedev, Shaoyi Cheng, Austin Doupnik,
James Martin, Christopher Fletcher, Daniel Burke,
Mingjie Lin, and John Wawrzynek

Landscape of Computing



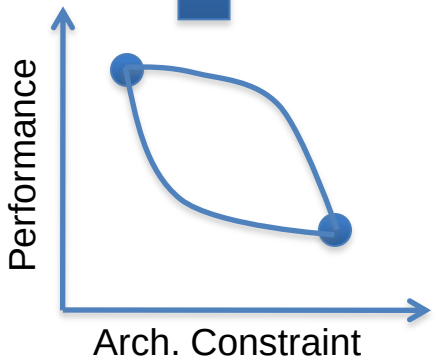
MARC = Many-core Approach to Reconfigurable Computing

FPGA Computing



1. Tons of research
2. Not very portable
3. HDL programming
4. Domain knowledge

1. C-style HLP
2. Huge potential
3. Code fudging
4. Complementary to MARC

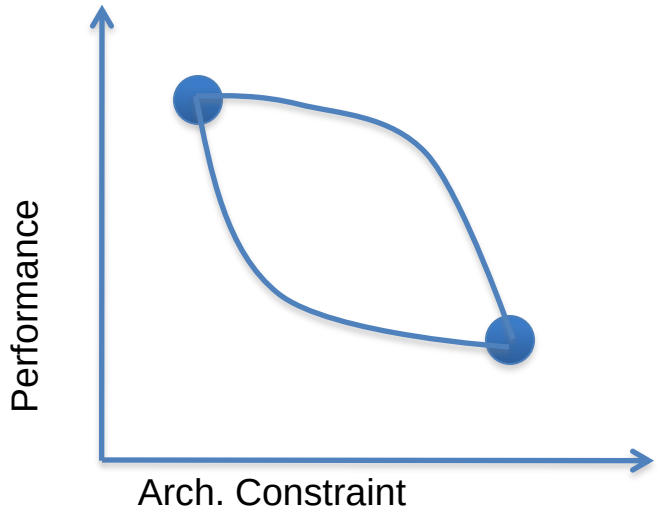


Motivations

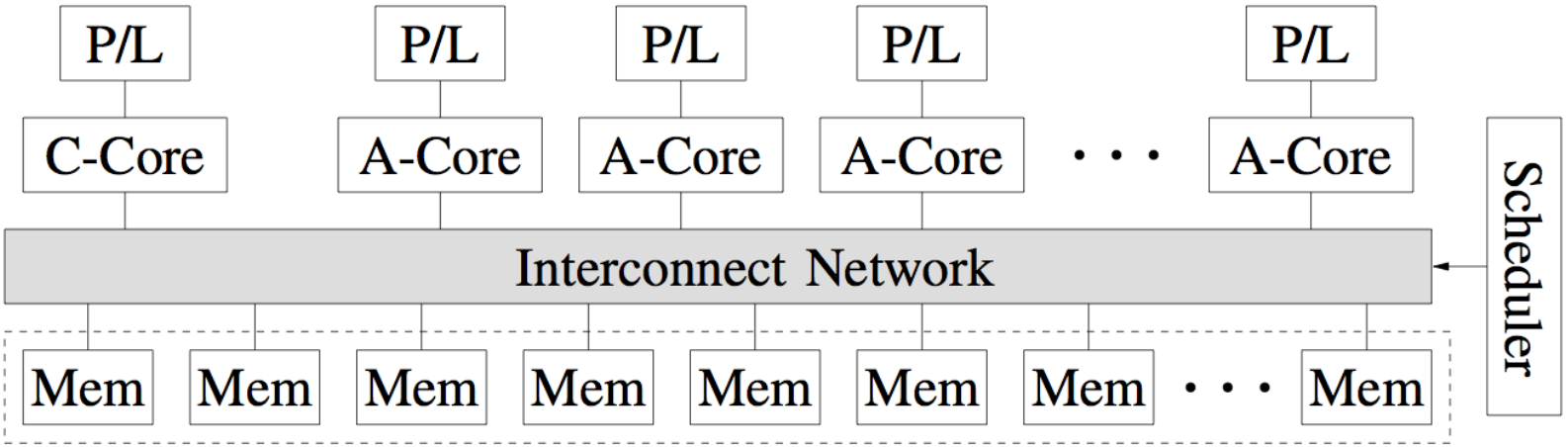
- HLS heavily studied
  - Most studies focused on application-specific customization
  - Benchmarks tend to be small, not system-scale
- This study
  - Tradeoff between customization and flexibility
  - Realistic, relatively large-scale applications
    - ParaLearn bio-inference problem
    - Bayesian learning

We Ask:

- For a given class of computation-intensive applications, is it possible to build a reconfigurable computing machine *constrained to resemble a many-core computer*, program it using a high-level *imperative language* such as C/C++, and yet still achieve orders of magnitude in *performance gain* relative to conventional computing means?

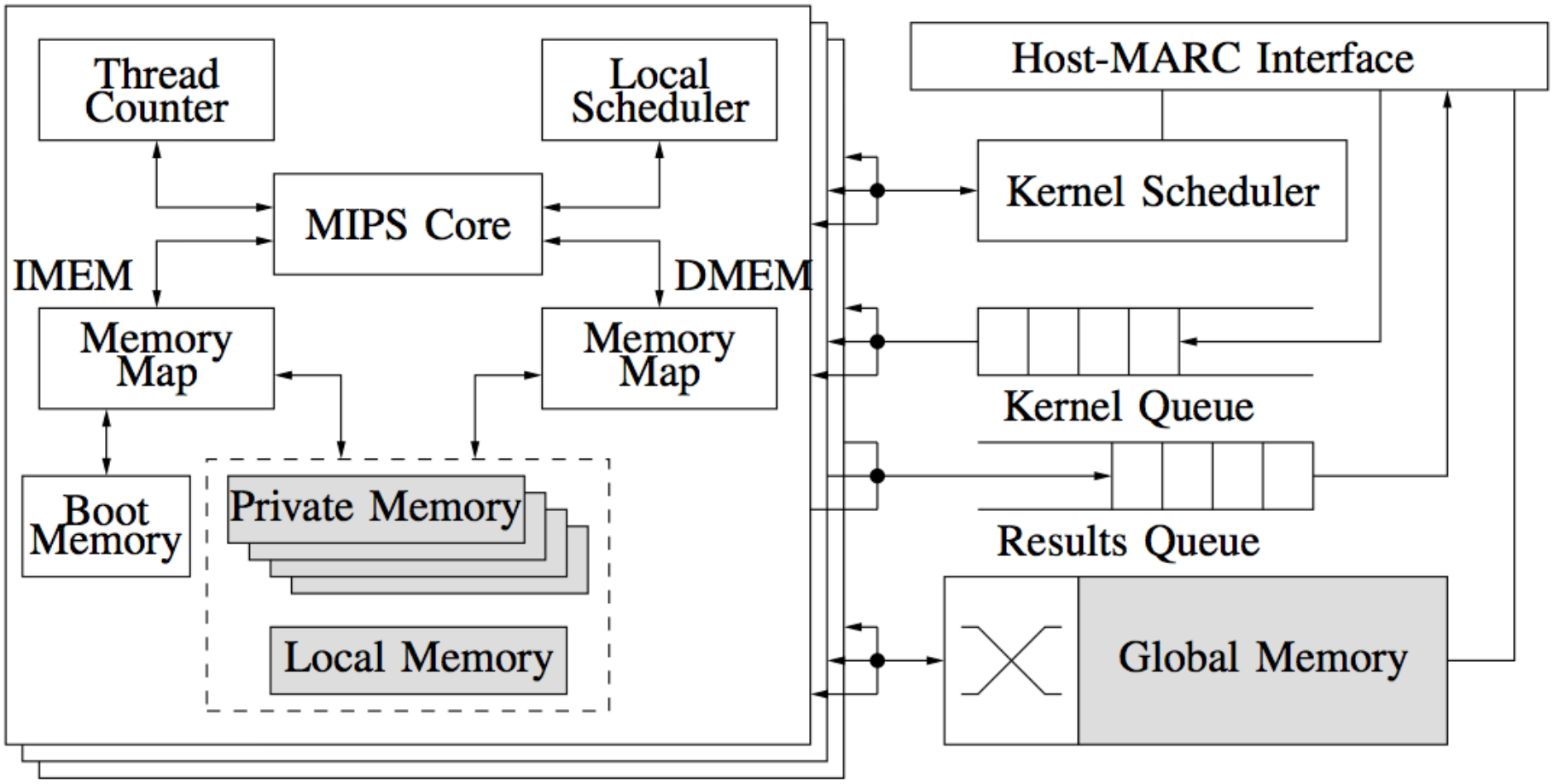


MARC: Overview

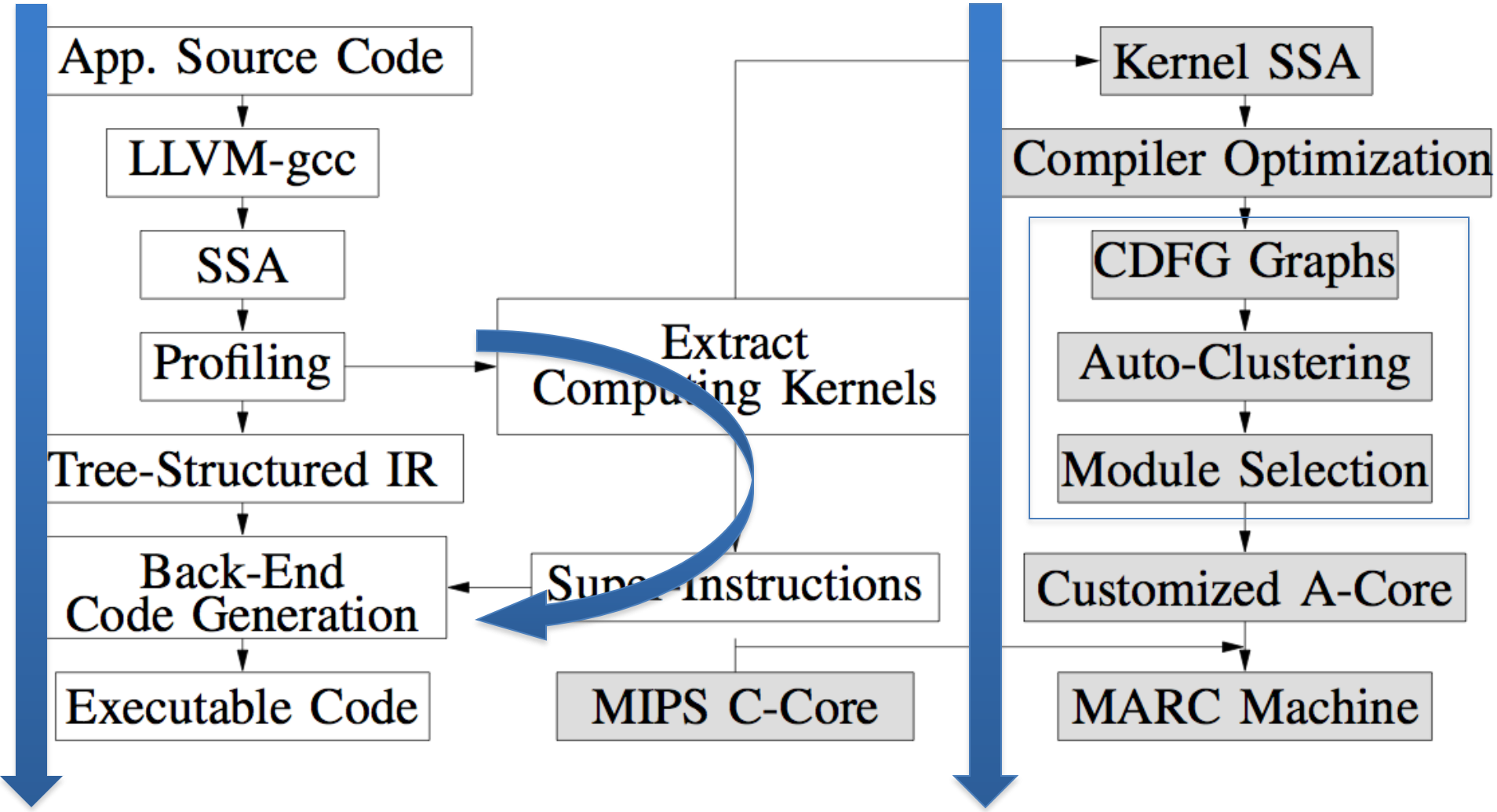


- C-Core (Control Processing Core) ➔ MIPS
- A-Cores (Arithmetic Processing Cores) ➔ data-path
- Parameterizable (bit-width, multi-threading, …)
- Objectives:
  - Reducing hardware inefficiency between ASIC- and FPGA-based computing platforms
  - Effective RC but with low design effort

Schematic: MARC machine's implementation



MARC: CAD Flow

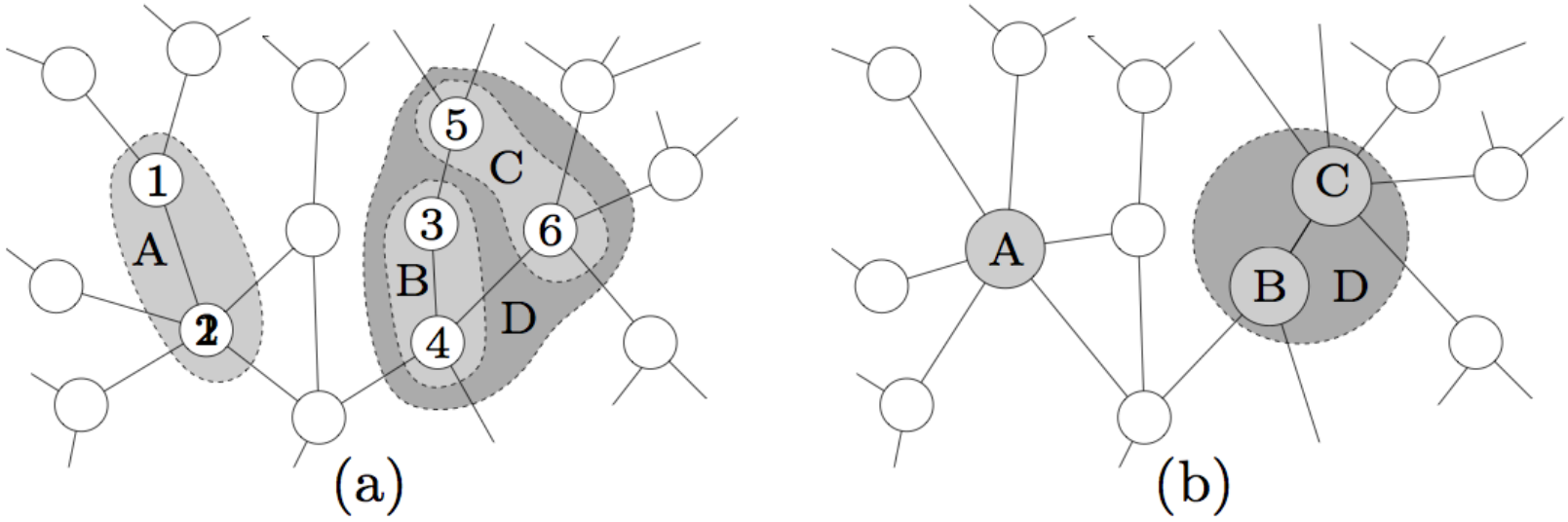


Design Aspects of MARC System

- Many-Core Template
- Execution Model and Software Infrastructure
- Application-Specific Processing Core
- Host-MARC Interface
- Memory Organization
- Kernel Scheduler
- System Interconnect

A-Core Synthesis: CDFG

- IR ➔ mixed control/data flow graph

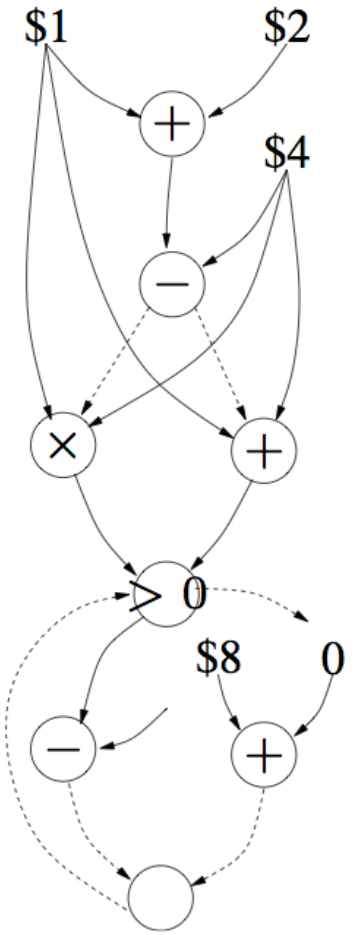


A-Core Synthesis: CDFG

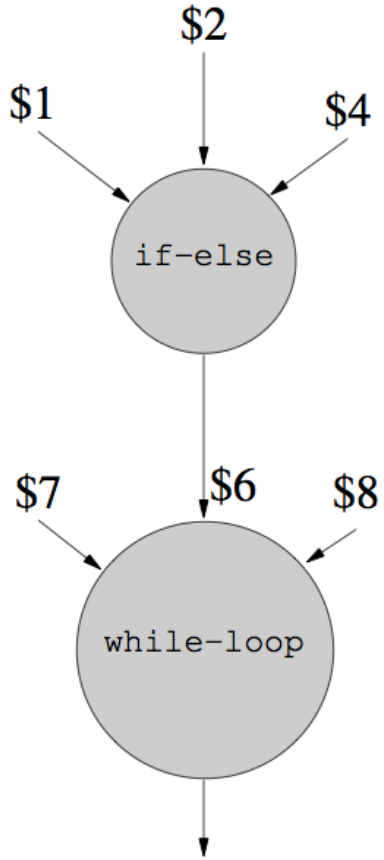
- IR ➔ mixed control/data flow graph

```
      x=a+b;
      if (x>c) {
        y=a+c;
      } else {
        y=a*c;
      }
      sum = 0;
      read z;
      while (y>0) {
        sum=sum+z;
        y=y-1;
      }
------------------
    load $1, a
    load $2, b
     add $3, $1, $2
    load $4, c
     sub $5, $3, $4
  branch $5, l1
    mult $6, $1, $4
    jump l2
l1:  add $6, $1, $4
l2: load $7, z
    load $8, 0
  branch $6, l3
    jump l4
l3: jump l2
l4:
```

(a)



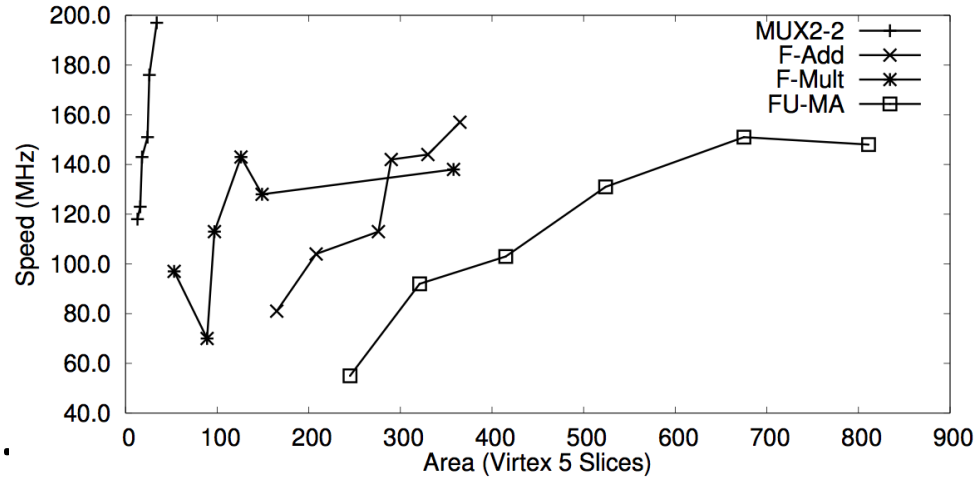
(b)



(c)

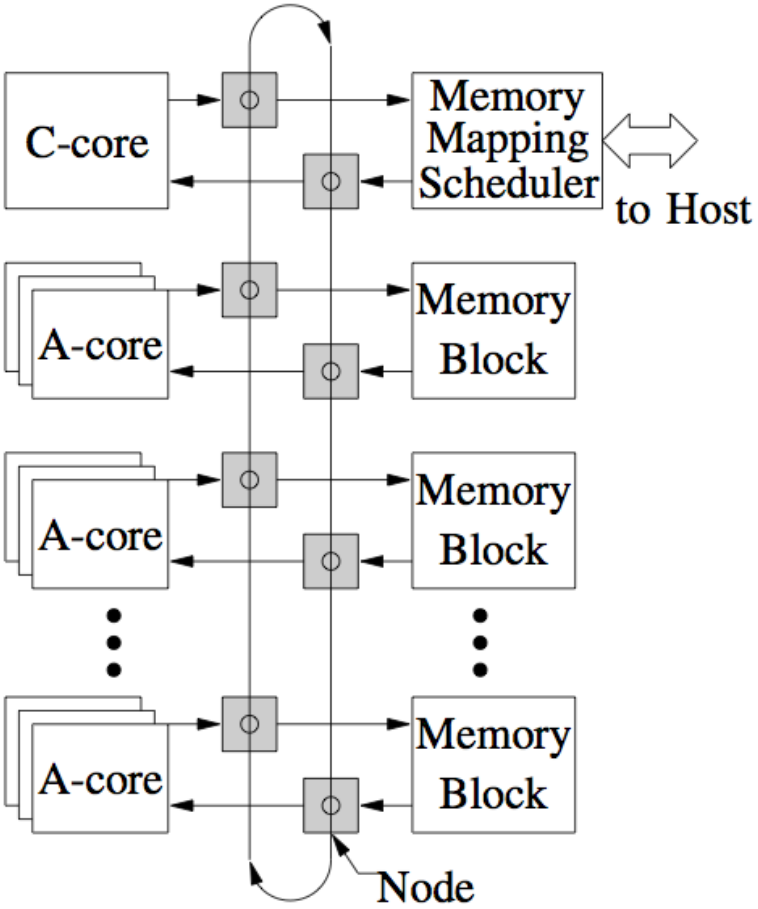
A-Core Synthesis: *Auto-Clustering and Module Selection*

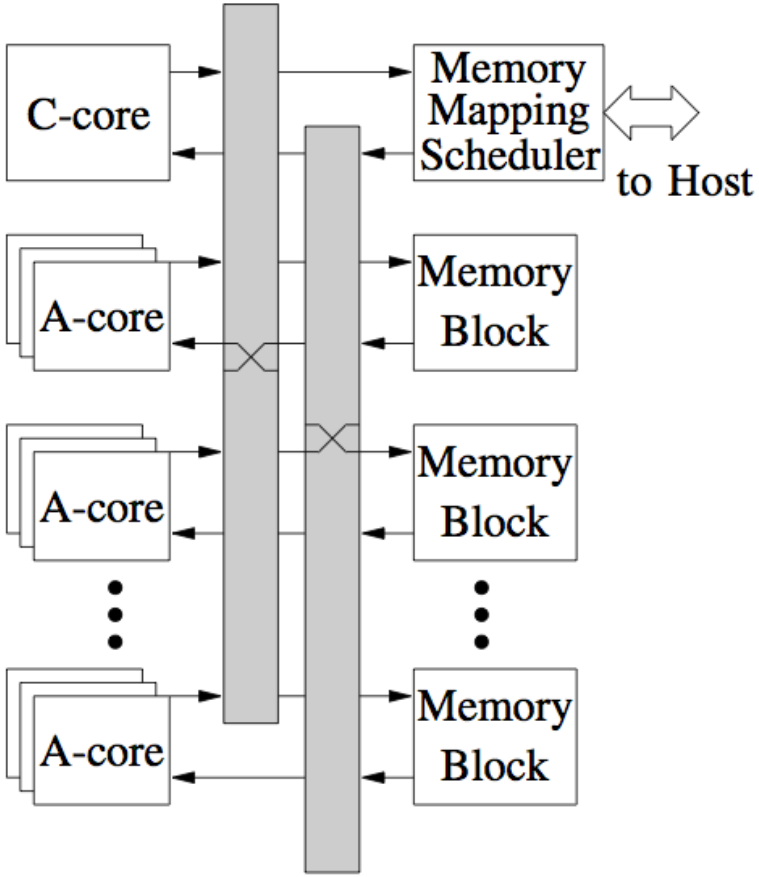
- Inputs: CDFG network + module library
- MARC synthesis = graph matching
  - Resembles technology mapping in FPGA
- NP-complete
- Simulated annealing
  - Moves
    - Module selection
    - Auto-clustering
  - Cost function
    - Aera, perf., timing,.



System Interconnect

- Exploit application-specific communication patterns in the hardware system



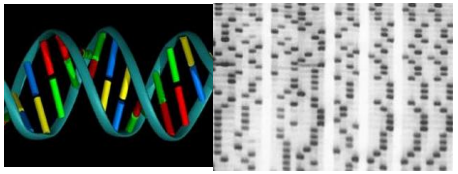


Benchmarking

- General-purpose Bayesian computing
- Specific application for large-scale Bayesian inference → ParaLearn

Bayesian Network Computing

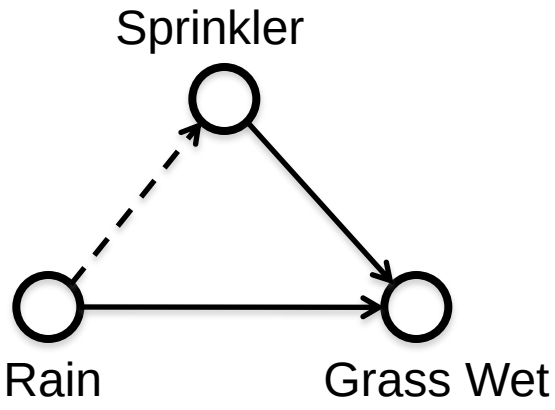
- Artificial intelligence and signal processing
  - Forward/backward algo., Viterbi algo., iterative "turbo" decoding algo., Pearl's belief propagation algo., Kalman filter, certain FFT algo., …
  - Real-world applications: Early vision: stereo and image restoration, DNA pyrosequencing: sequencing--synthesis
- Representative
- Known baseline

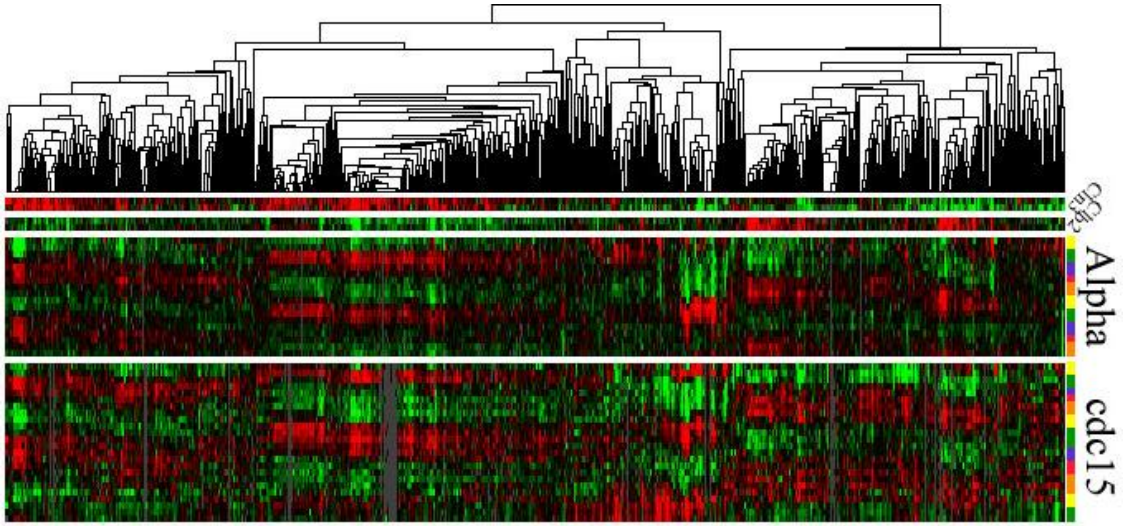


Felzenszwalb'06

What is Bayesian Computing

- Bayesian network/belief propagation network
  - Nodes ⇔ RVs, edges ⇔ dependency
  - Joint probability,
  - Point probability,





Gene Expression Data, *Stanford Bio-X center*

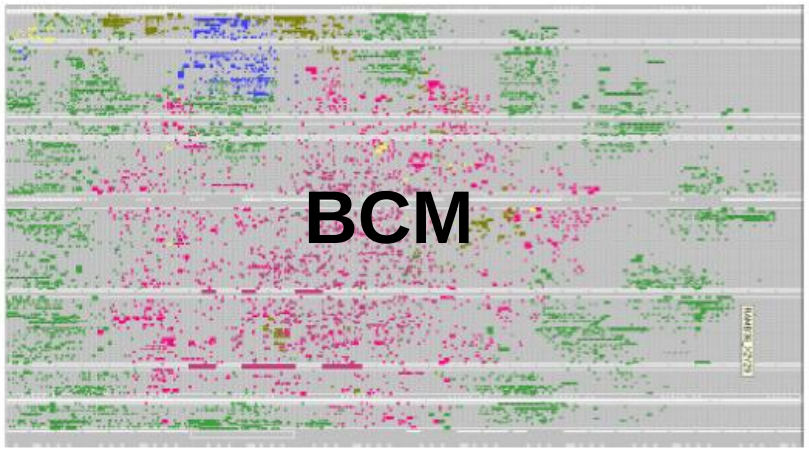
Case 1: Bayesian Computing Machine

Goal: quickly evaluating

- Challenges
  - # of terms exponential with # of variables
  - Accuracy ➔ Floating point ops
  - High/real-time performance ➔ high throughput

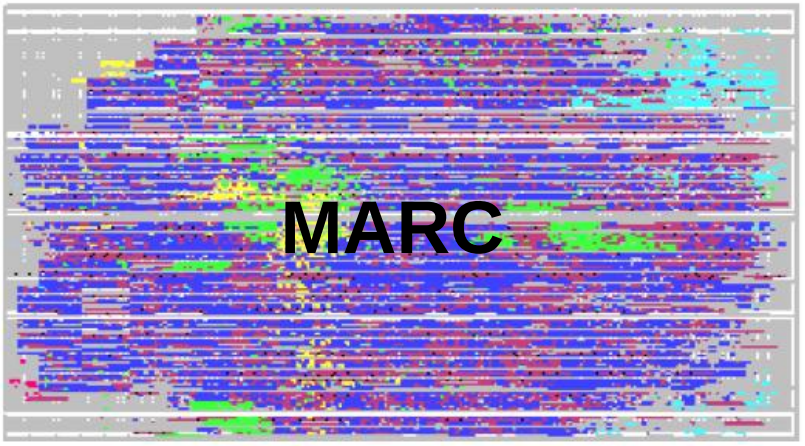
- Baseline solution: BCM [FPGA2009]
  - Stall-free memory accesses
  - Deep pipelining
  - Optimally scheduling processing nodes

Performance Comparison



16 Nodes (a)

Slices: 17650, DSP Blk: 42, B. RAM: 4646



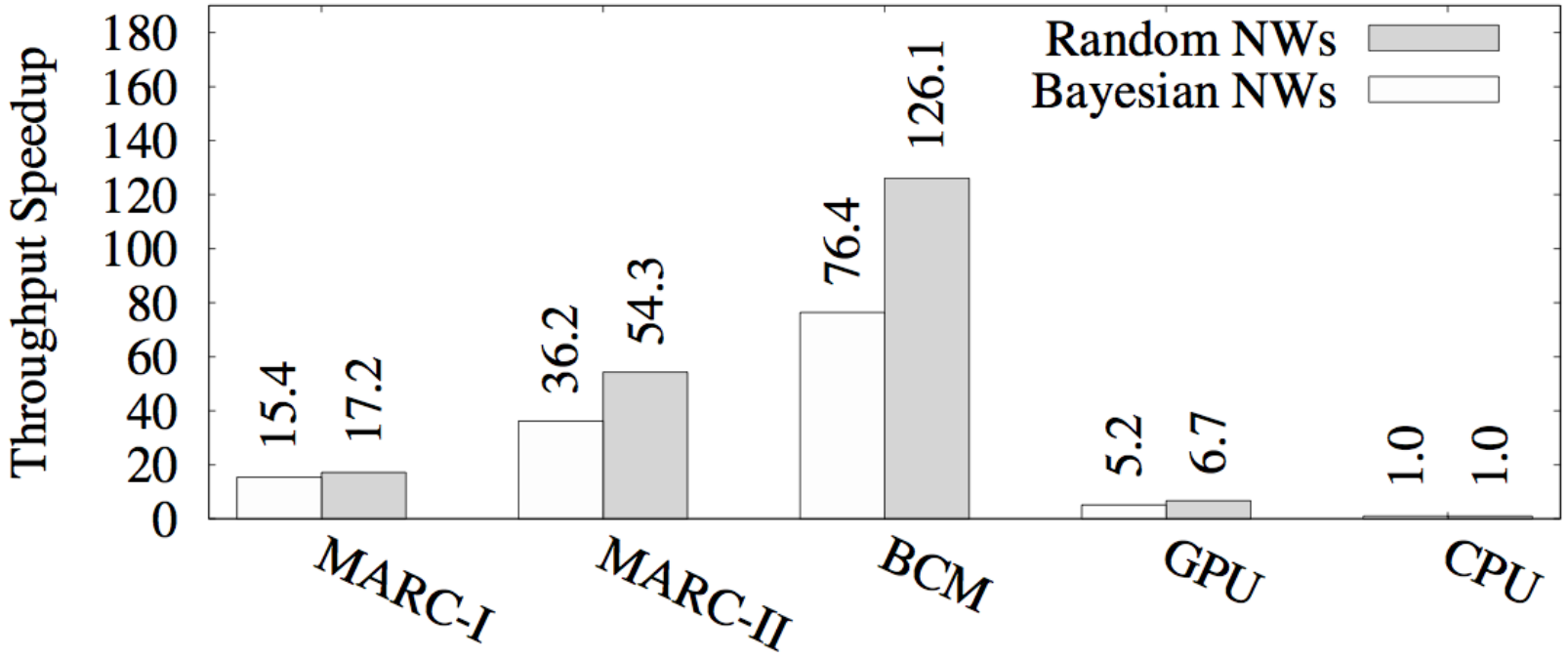
1 C-Core + 32 A-Cores (b)

Slices: 20560, DSP Blk: 76, B. RAM: 5928

Platforms	Benchmarks					
	Random NWs			Bayesian NWs		
	Min.	Max.	Avg.	Min.	Max.	Avg.
MARC-I	1.12	5.55	2.56	0.69	4.28	3.28
MARC-II	**2.92**	**16.66**	**7.83**	**1.61**	**12.4**	**8.13**
BCM	11.25	29.23	18.23	5.78	20.4	17.31
GPU	1.09	0.23	0.96	4.94	0.18	1.15
CPU	0.50	0.12	0.14	0.34	0.11	0.22

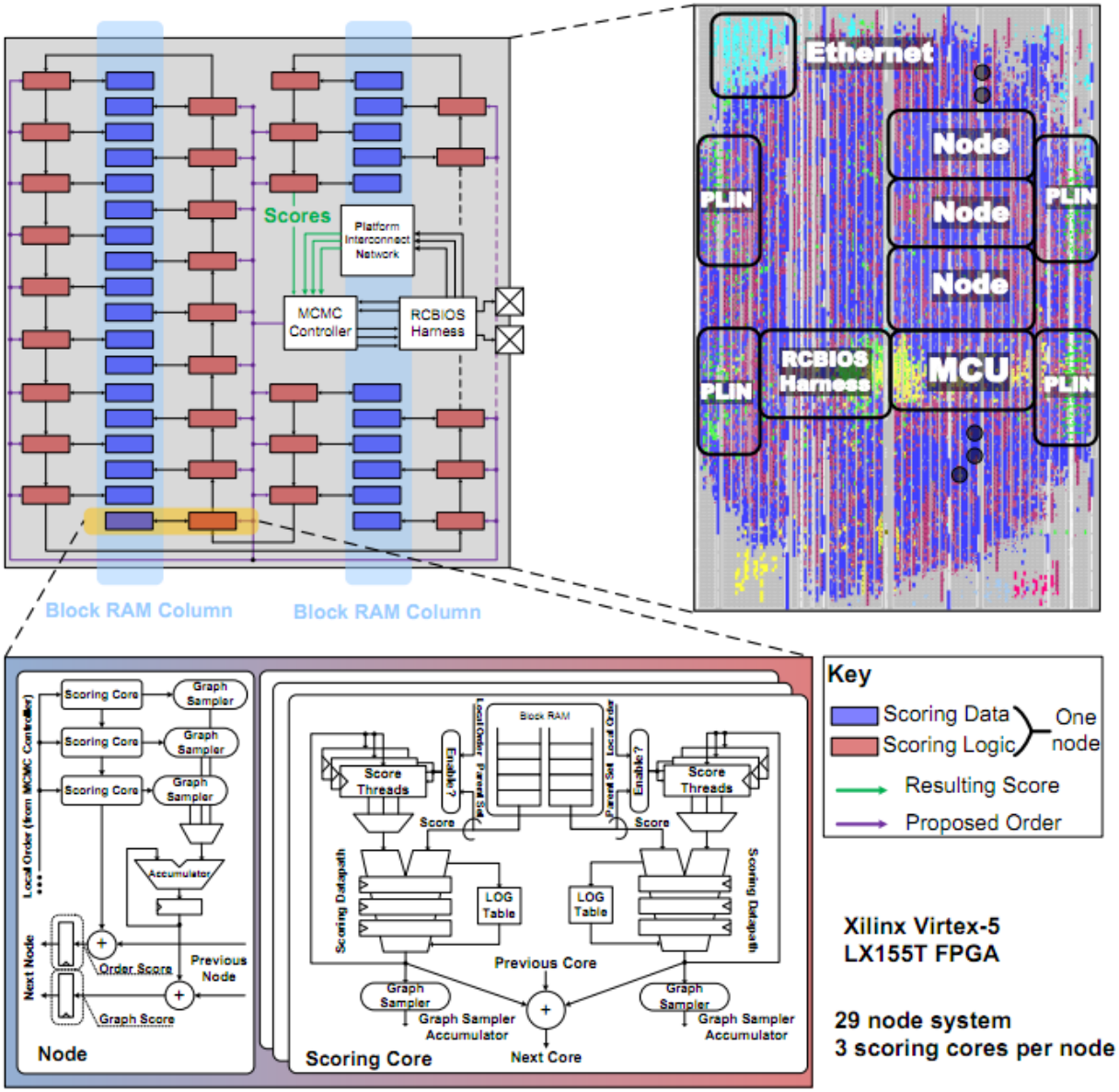
(a)

Bottom Line

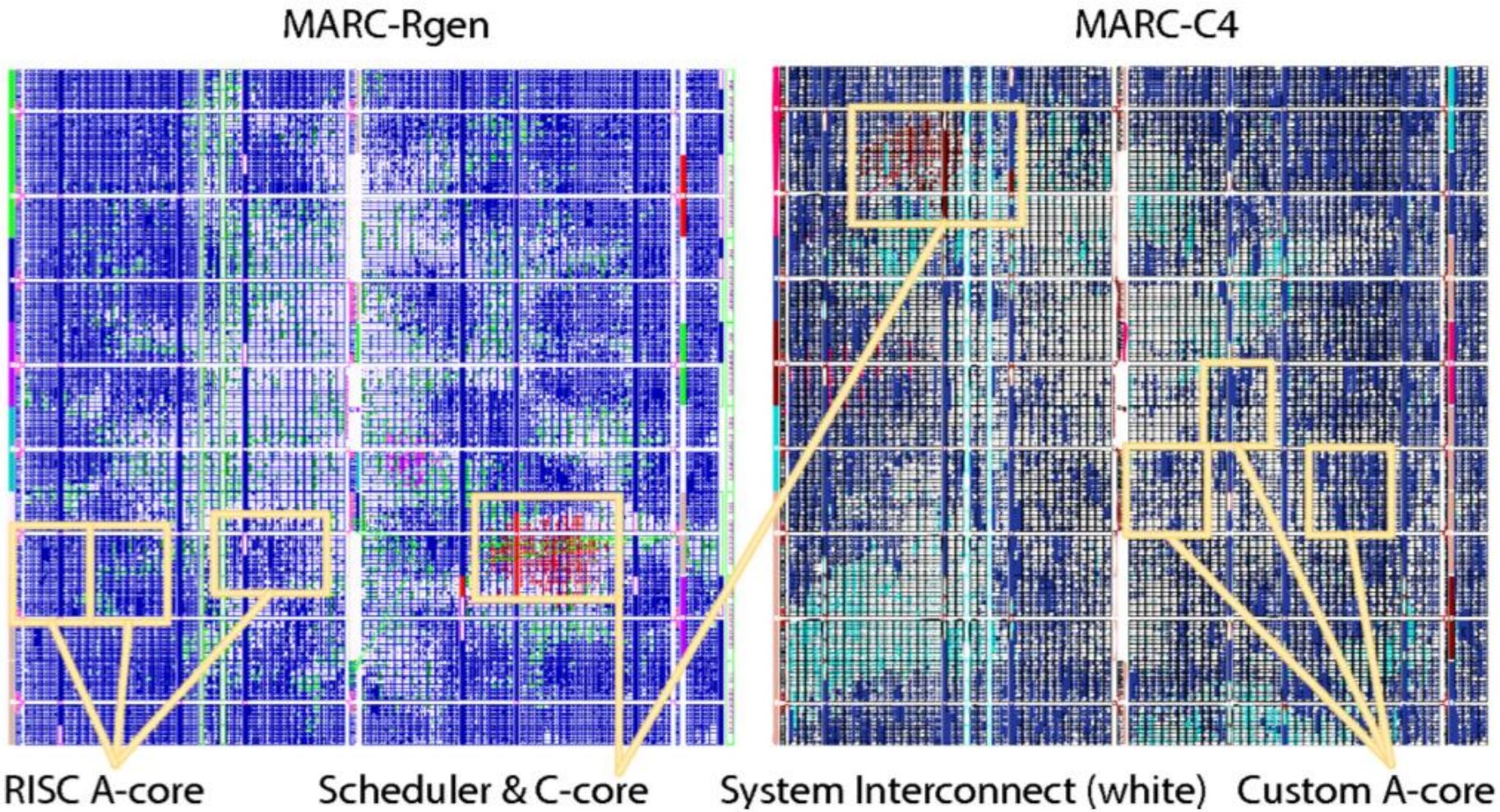


- **BCM**: 6 months, two people
- **MARC**: 2 months, one people; reusable

Case 2: ParaLearn

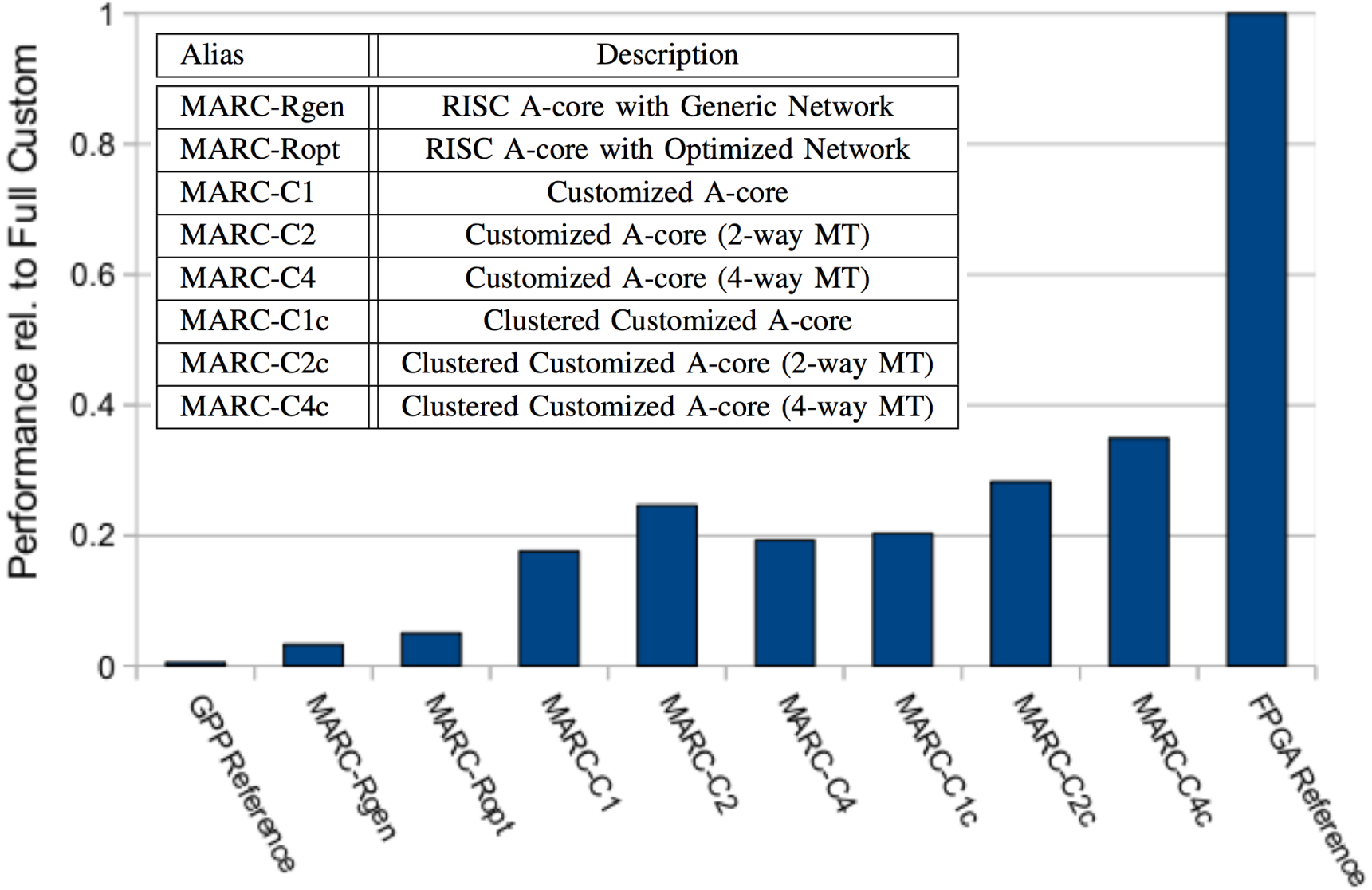


FPGA Layouts after Placement and Routing



Performance Comparison

Alias	Description
MARC-Rgen	RISC A-core with Generic Network
MARC-Ropt	RISC A-core with Optimized Network
MARC-C1	Customized A-core
MARC-C2	Customized A-core (2-way MT)
MARC-C4	Customized A-core (4-way MT)
MARC-C1c	Clustered Customized A-core
MARC-C2c	Clustered Customized A-core (2-way MT)
MARC-C4c	Clustered Customized A-core (4-way MT)



Performance Comparison (cont.)

Configuration	Device Utilization	Execution Time (μs)	Relative Perf.	Relative Area Eff.
GPP Reference	n/a	350	0.0055	n/a
MARC-Rgen	0.9	58.48	0.0327	0.0334
MARC-Ropt	0.84	38	0.0503	0.0551
MARC-C1	0.55	10.89	0.1754	0.2935
MARC-C2	0.63	7.76	0.2462	0.3595
MARC-C4	0.71	9.93	0.1924	0.2493
MARC-C1c	0.46	9.4	0.2033	0.4066
MARC-C2c	0.53	6.77	0.2819	0.4894
MARC-C4c	0.57	5.47	0.3492	0.5636
FPGA Reference	0.92	1.91	1.0000	1.0000

- **ParaLearn**: 12 months, four people
- **MARC**: 3 months, two people; reusable

Conclusion

- MARC can effectively trade between design effort and performance

- But, “bad” examples to be found …
  - Little explicit parallelsim
  - Complex memory access patterns

References

- High-Throughput Bayesian Computing Machine with Reconfigurable Hardware, Mingjie Lin, Ilia Lebedev, and John Wawrzynek, *the 2010 ACM/SIGDA International Symposium on Field Programmable Gate Arrays, Pages: 73-82*
- ParaLearn: a massively parallel, scalable system for learning interaction networks on FPGAs, Narges A., et. al., *ICS '10 Proceedings of the 24th ACM International Conference on Supercomputing*
- MARC: A Many-Core Approach to Reconfigurable Computing, Lebedev, I.; Shaoyi Cheng; Doupnik, A.; Martin, J.; Fletcher, C.; Burke, D.; Mingjie Lin; Wawrzynek, J.; *2010 International Conference on Reconfigurable Computing and FPGAs (ReConFig)*
- Rethinking FPGA Computing with a Many-Core Approach, Mingjie L. and Wawrzynek, J., *The First Workshop on the Intersections of Computer Architecture and Reconfigurable Logic (CARL 2010)*

Acknowledgements

- DARPA, Grant FA8650- 09-C-7907
- NIH, grant 1R01CA130826-01
- NSF Grants #0403427 & #0551739
- Berkeley Wireless Research Center (BWRC)
- Berkeley GateLib team (Greg Gibeling, Chris Fletcher, …)

Thanks!

CAD flow of synthesizing A-Cores