Roughness of Microarchitectural Design Topologies and Implications for Optimization

Benjamin C. Lee, David M. Brooks

www.seas.harvard.edu/∼bclee
School of Engineering and Applied Sciences
Harvard University

International Symposium on High-Performance Computer Architecture
19 February 2008

Optimization Challenges

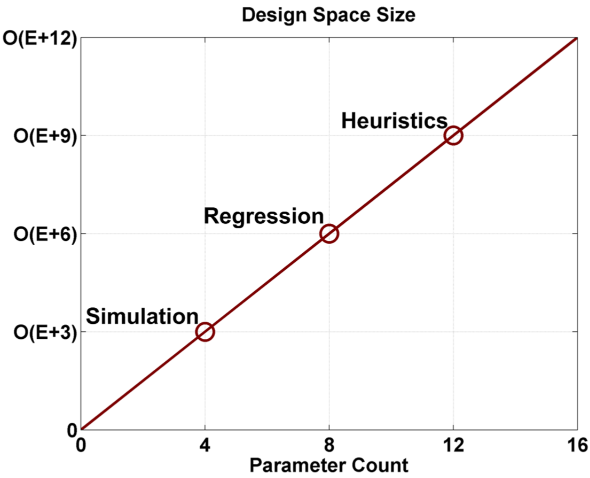
- **Design Diversity**
  - Diversity of interesting, viable designs
  - Ex :: Power 6, Core 2, UltraSPARC T2
- **Metric Diversity**
  - Differentiated market segments and metric priorities
  - Ex :: latency, throughput, power, temperature
- **Comprehensive Optimization**
  - Mechanisms to identify representative workloads
  - Models to estimate design metrics
  - Heuristics to traverse design topology

Statistical Inference

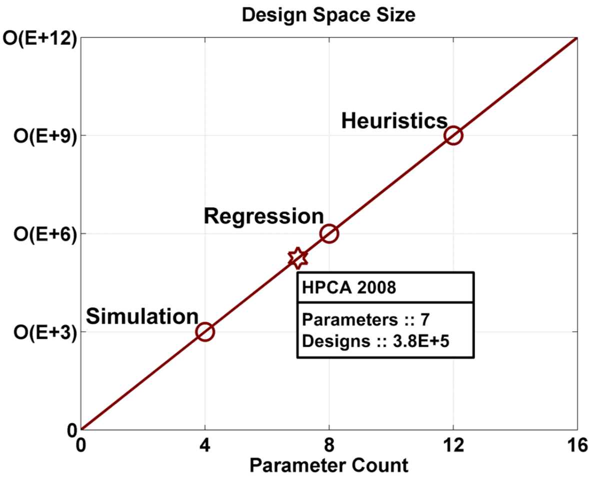
- **Simulation Paradigm**[1]
  - Comprehensively understand design space
  - Selectively simulate modest number of designs
  - Efficiently leverage simulation data with inference
- **Regression Models**
  - Low formulation costs (1K samples from 1B designs)
  - Accurate inference ($5 - 7\%$ median error)
  - Efficient computation (100's of predictions per second)

[1] Lee+[ASPLOS'06]

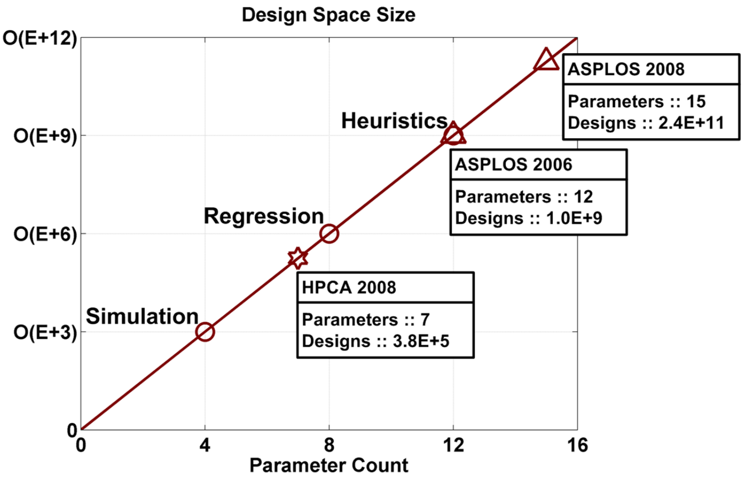
Qualitatively New Capabilities



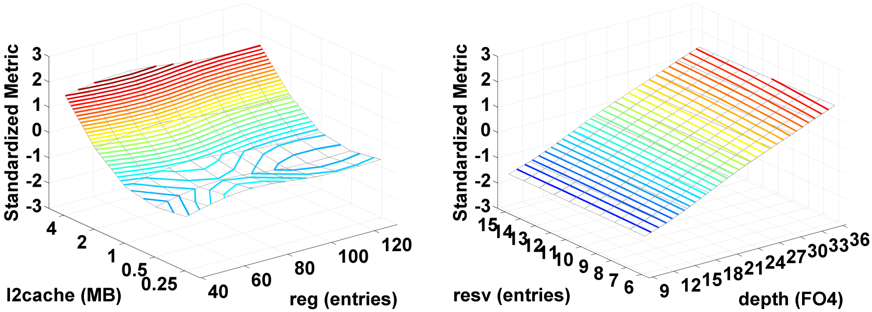
Qualitatively New Capabilities



Qualitatively New Capabilities

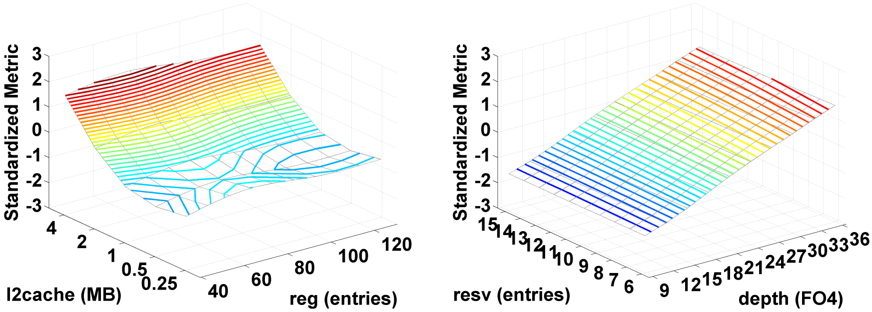


Effective Analysis and Optimization



- Identify interesting regions of design space
- Identify appropriate heuristic, implementation

Roughness Impact



- Rough topologies more challenging to model, optimize
- Rough topologies more likely to contain local optima

Outline

Modeling
- Regression Models
- Roughness Metrics

Visualization
- Contour Maps
- Contours & Roughness

Optimization
- Gradient Ascent
- Heuristic Effectiveness
- Optimization & Roughness

Outline

Tools and Benchmarks

- **Simulation Framework**
  - Turandot :: a cycle-accurate trace driven simulator
  - PowerTimer :: power models derived from circuit analyses
  - Baseline simulator models POWER4/POWER5 architecture
- **Benchmarks**
  - SPECcpu :: compute-intensive benchmarks
  - SPECjbb :: Java server benchmark
- **Statistical Framework**
  - R :: software environment for statistical computing
  - Hmisc and Design packages[2]

[2] Harrell [Springer,'01]

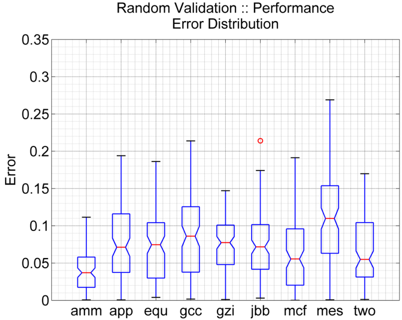
Design Space

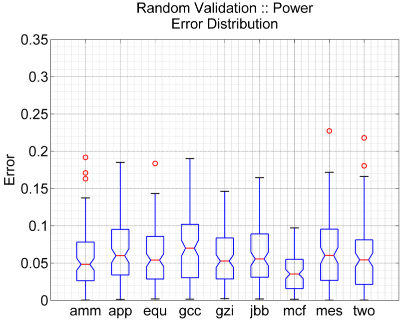
	Set	Parameters	Measure	Range	|S|
S_1	Depth	depth	FO4	9::3::36	10
S_2	Width	width	insn b/w	4,8,16	3
		L/S reorder queue	entries	15::15::45	
		store queue	entries	14::14::42	
		functional units	count	1,2,4	
S_3	Physical Registers	general purpose (GP)	count	40::10::130	10
		floating-point (FP)	count	40::8::112	
		special purpose (SP)	count	42::6::96	
S_4	Reservation Stations	branch	entries	6::1::15	10
		fixed-point/memory	entries	10::2::28	
		floating-point	entries	5::1::14	
S_5	I-L1 Cache	i-L1 cache size	log_2(entries)	7::1::11	5
S_6	D-L1 Cache	d-L1 cache size	log_2(entries)	6::1::10	5
S_7	L2 Cache	L2 cache size	log_2(entries)	11::1::15	5

Model Accuracy I

- **Approach**
  - Formulate models with $n = 1,000$ samples
  - Obtain 100 additional random samples for validation
  - Quantify percentage error, $100 * |\hat{y}_i - y_i|/y_i$
- **Comparison**
  - Simulator-reported performance, power
  - Regression-predicted performance, power

Model Accuracy II





Roughness Metrics

- $f(x)$ :: regression model
- $x_1, \ldots, x_d$ :: design parameters
- Derived for spline-based regression models[3]
- Computed numerically

$$
\begin{aligned}
R_1 &= \int_x \left(\frac{\delta^2 f}{\delta x_1^2} \right)^2 dx \\
R_2 &= \int_{x_2} \int_{x_1} \left\{ \left(\frac{\delta^2 f}{\delta x_1^2} \right)^2 + 2 \left(\frac{\delta^2 f}{\delta x_1 x_2} \right)^2 + \left(\frac{\delta^2 f}{\delta x_2^2} \right)^2 \right\} dx_1 dx_2 \\
R_d &= \int_{x_d} \ldots \int_{x_1} \sum \frac{m!}{v_1! \ldots v_d!} \left(\frac{\delta^m f}{\delta {x_1}^{v_1} \ldots \delta {x_d}^{v_d}} \right)^2 dx_1 \ldots dx_d
\end{aligned}
$$

[3] Green+[Monographs Stat & Applied Prob.]

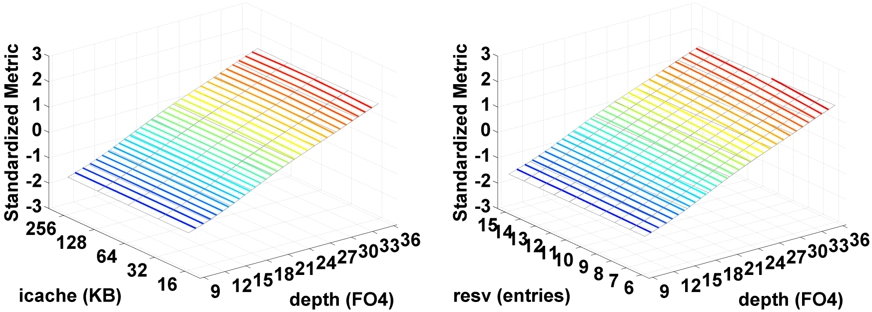
Outline

- Modeling
  - Regression Models
  - Roughness Metrics
- Visualization
  - Contour Maps
  - Contours & Roughness
- Optimization
  - Gradient Ascent
  - Heuristic Effectiveness
  - Optimization & Roughness

Contour Maps

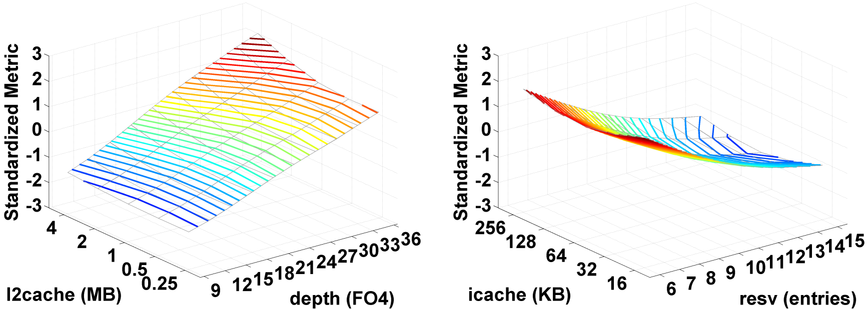
- **Applications**
  - Reveal bottlenecks
  - Characterize workloads
- **Approach**
  - Select 2-dim slice of p-dim design space
  - Plot performance, power topology with regression models
  - Iterate for $\binom{p}{2}$ contours
- **Contours & Roughness**
  - Rank contours by R_2 roughness
  - Roughness metrics corroborate observed variability

Low R_2 Contours



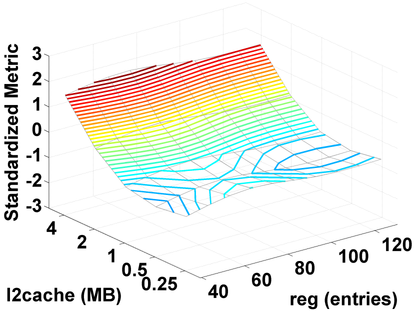
- Roughness metrics corroborate observed variability
- Contours ranked 20, 21 of 21 (*mcf*, $bips/w$)

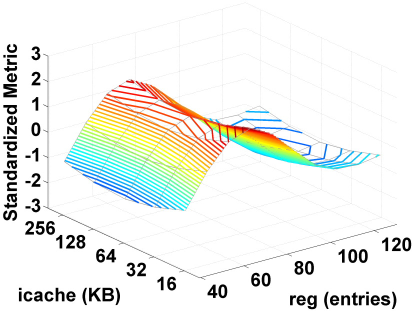
Medium R_2 Contours



- Roughness metrics corroborate observed variability
- Contours ranked 10, 11 of 21 (*mcf*, $bips/w$)

High R_2 Contours

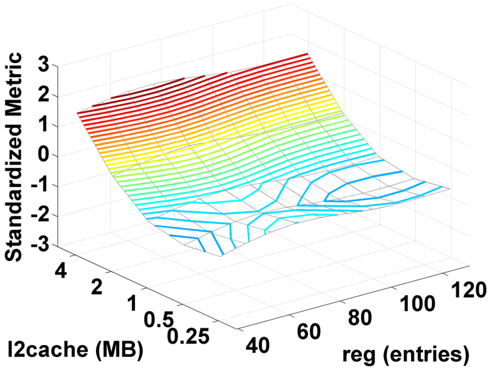




- Roughness metrics corroborate observed variability
- Contours ranked 1, 2 of 21 (*mcf*, $bips/w$)

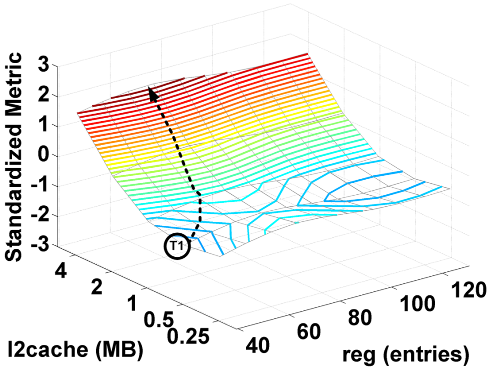
Outline

Gradient Ascent :: Heuristic



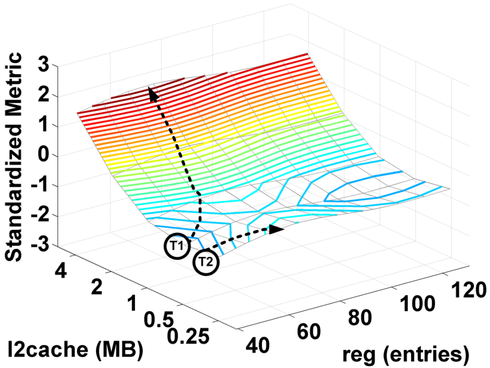

```
for(t = 1 to T)
    begin @ random starting point
    while(~ Converged)
        evaluate all neighbors using model
        step in direction of gradient
```

Gradient Ascent :: Heuristic



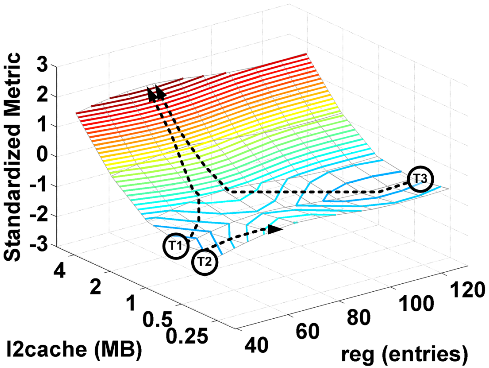

```
for(t = 1 to T)
    begin @ random starting point
    while(~ Converged)
        evaluate all neighbors using model
        step in direction of gradient
```

Gradient Ascent :: Heuristic



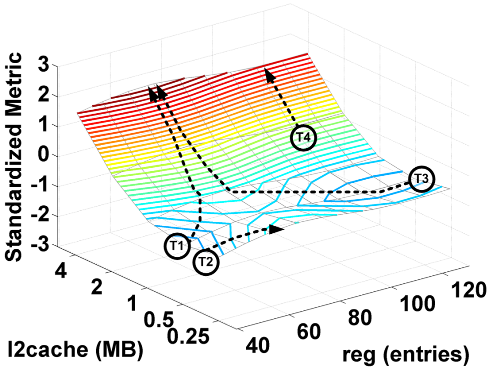

```
for(t = 1 to T)
    begin @ random starting point
    while(~ Converged)
        evaluate all neighbors using model
        step in direction of gradient
```

Gradient Ascent :: Heuristic



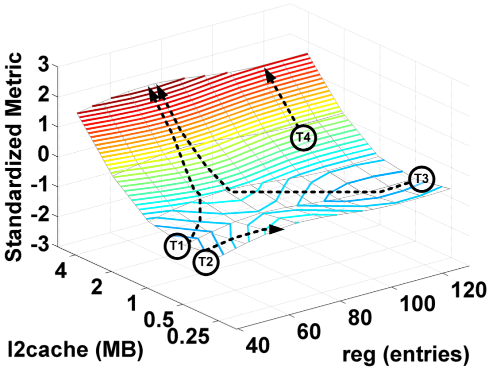

```
for(t = 1 to T)
    begin @ random starting point
    while(~ Converged)
        evaluate all neighbors using model
        step in direction of gradient
```

Gradient Ascent :: Heuristic



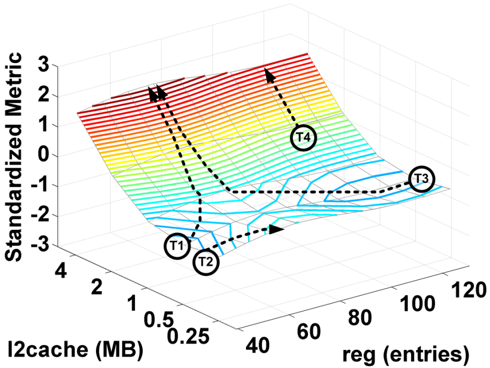

```
for(t = 1 to T)
    begin @ random starting point
    while(~ Converged)
        evaluate all neighbors using model
        step in direction of gradient
```

Gradient Ascent :: Computational Cost



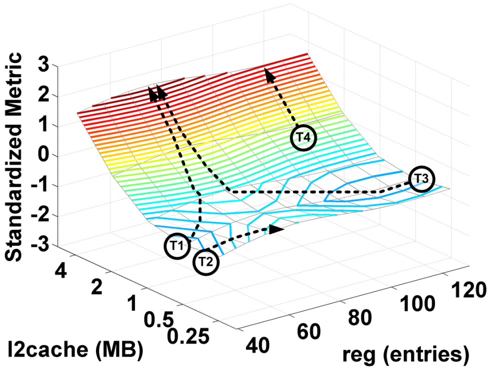

```
for(t = 1 to T)
    begin @ random starting point
    while(~ Converged)
        evaluate all neighbors using model
        step in direction of gradient
```

Gradient Ascent :: Computational Cost



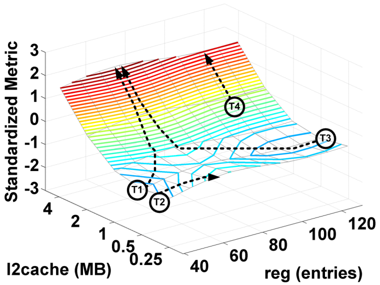

```
for(t = 1 to T)
    begin @ random starting point
    while(~ Converged)
        evaluate all neighbors using model
        step in direction of gradient
```

Gradient Ascent :: Computational Cost



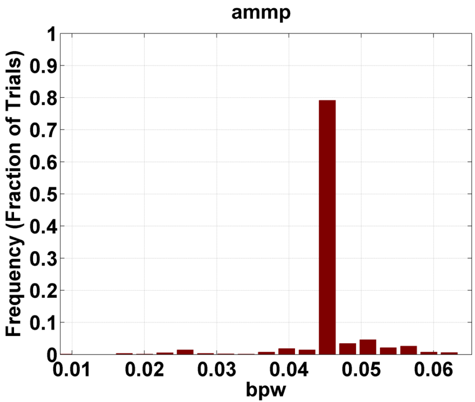

```
for(t = 1 to T)
    begin @ random starting point
    while(~ Converged)
        evaluate all neighbors using model
        step in direction of gradient
```

Gradient Ascent :: Definitions



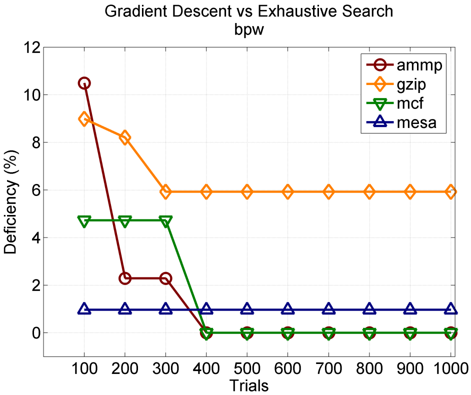
- Trials :: No. of random starting points
- Iterations :: No. of path steps before convergence
- Deficiency :: Difference between global, local maximum

Trial Consistency



- 79% of 1K *ammp* trials report sub-optimum
- 14% of 1K *ammp* trials better than mode

Deficiency



- Deficiency of gradient ascent results w.r.t. global optimum
- Additional trials reduce deficiency

R_7 and Optimization

Benchmarks	Rough	Deficiency	Rough	Iterations
ammp	4	1	4	6
applu	2	3	2	2
equake	3	9	3	1
gcc	5	7	5	8
gzip	8	8	8	9
jbb	7	6	7	3
mcf	1	2	1	7
mesa	9	4	9	4
twolf	6	5	6	5
roughness correlation	1.00	0.35	1.00	0.20

- Rank benchmarks by roughness (*e.g.*, 1 $\rightarrow$ least rough)
- Rank benchmarks by effectiveness (*e.g.*, 1 $\rightarrow$ lowest deficiency)

R_7 and Optimization :: Deficiency

Benchmarks	Rough	Deficiency	Rough	Iterations
ammp	4	1	4	6
applu	2	3	2	2
equake	3	9	3	1
gcc	5	7	5	8
gzip	8	8	8	9
jbb	7	6	7	3
mcf	1	2	1	7
mesa	9	4	9	4
twolf	6	5	6	5
roughness correlation	1.00	0.35	1.00	0.20

- Roughness and deficiency correlated ($\rho = 0.35$)
- Mitigate roughness with additional trials, stochastic variants

R_7 and Optimization :: Iterations

Benchmarks	Rough	Deficiency	Rough	Iterations
ammp	4	1	4	6
applu	2	3	2	2
equake	3	9	3	1
gcc	5	7	5	8
gzip	8	8	8	9
jbb	7	6	7	3
mcf	1	2	1	7
mesa	9	4	9	4
twolf	6	5	6	5
roughness correlation	1.00	0.35	1.00	0.20

- Roughness and iterations correlated ($\rho = 0.20$)
- Mitigate roughness with generous convergence criteria

Outline

Conclusion

- Paper Details
- Conclusion
- Future Directions

Also in the paper...

- **Gradients**
  - Background
  - Applications (*e.g.*, sensitivity)
- **Roughness**
  - Mathematical motivation
  - Numerical approximations
  - Implications for model accuracy
- **Contours**
  - Bottleneck analysis
  - Workload characterization

Conclusion

- **Simulation Paradigm**
  - Comprehensively understand design space
  - Selectively simulate modest number of designs
  - Efficiently leverage simulation data with inference
- **Roughness and Optimization**
  - Define, compute roughness metrics
  - Rough topologies are more interesting
  - Rough topologies require more robust optimization
- **ASPLOS 2008 Tutorial**
  - Learning and inference tutorial (LIT)

Future Directions

- **Optimization Heuristics and Applications**
  - Comparing, contrasting optimization heuristics
  - Genetic search for efficiency limits of hardware adaptivity[4]
- **Chip Multiprocessors**
  - Scalable accounting for shared resource contention
  - Larger parameter space (*e.g.*, in-order vs out-of-order)
- **Integration with Circuit Models**
  - Device sizing, supply/threshold voltage
  - Implications for process variation

[4] Lee+[ASPLOS'08]

Roughness of Microarchitectural Design Topologies and Implications for Optimization

Benjamin C. Lee, David M. Brooks

www.seas.harvard.edu/∼bclee
School of Engineering and Applied Sciences
Harvard University

International Symposium on High-Performance Computer Architecture
19 February 2008

Further Reading

www.seas.harvard.edu/∼bclee

- B.C. Lee and D.M. Brooks.
  Efficiency trends and limits from comprehensive microarchitectural adaptivity
  ASPLOS-XIII: International Conference on Architectural Support for Programming Languages and Operating Systems, March 2008.

- B.C. Lee and D.M. Brooks.
  Roughness of microarchitectural topologies and its implications for optimization
  HPCA-14: International Symposium on High Performance Computer Architecture, Feb 2008.

- B.C. Lee and D.M. Brooks and B.R. de Supinski and M. Schulz and K. Singh and S.A. McKee.
  Methods of inference and learning for performance modeling of parallel applications
  PPoPP'07: Symposium on Principles and Practice of Parallel Programming, March 2007.

- B.C. Lee and D.M. Brooks.
  Illustrative design space studies with microarchitectural regression models
  HPCA-13: International Symposium on High Performance Computer Architecture, Feb 2007.

- B.C. Lee and D.M. Brooks.
  Accurate, efficient regression modeling for microarchitectural performance, power prediction.
  ASPLOS-XII: International Conference on Architectural Support for Programming Languages and Operating Systems, Oct 2006.