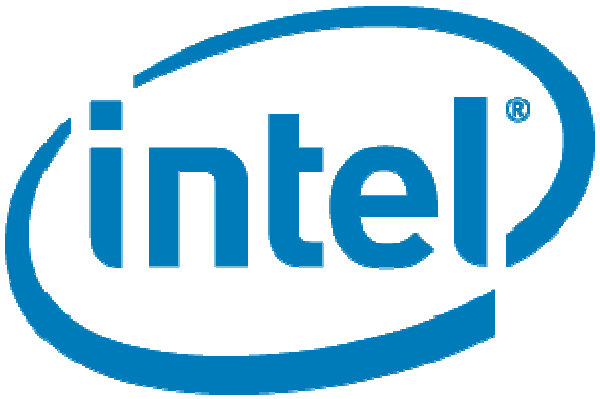
The Intel® Adaptive Spike-Based Solver: Using Software Adaptation to Achieve Better Performance

Henry A. Gabb, David Wong, Ping Tak Peter Tang, David J. Kuck
Intel Corporation

Ahmed H. Sameh, Murat Manguoglu
Purdue University

Eric Polizzi
University of Massachusetts Amherst



Proposition: Having a repertoire of algorithms is better than a single, highly-tuned algorithm.

Agenda

Banded Linear Systems

Spike-Based Decomposition

Software Adaptation

Accuracy and Performance

Intel® Adaptive Spike-Based Solver is a trademark or registered trademark of Intel Corporation or its subsidiaries in the United States or other countries

Band Matrices

Band matrices have all of their nonzero elements on or near the diagonal.

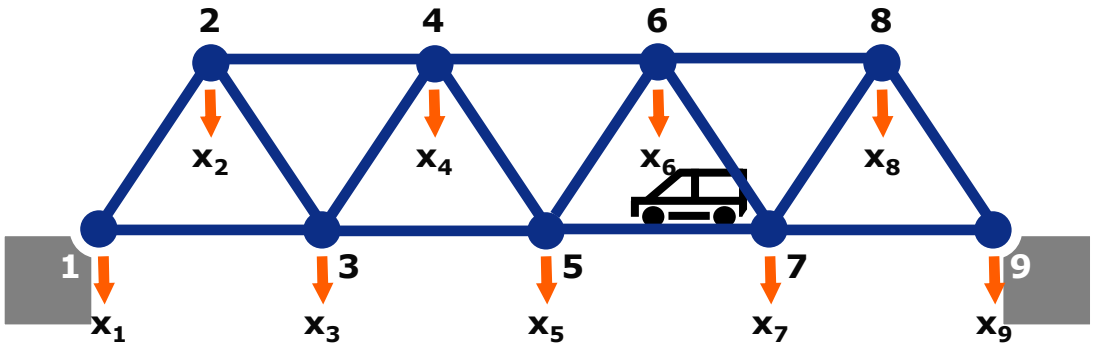
$$\begin{pmatrix} 6 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 6 & -1 & -1 & 0 & 0 & 0 & 0 \\ -1 & -1 & 6 & -1 & -1 & 0 & 0 & 0 \\ 0 & -1 & -1 & 6 & -1 & -1 & 0 & 0 \\ 0 & 0 & -1 & -1 & 6 & -1 & -1 & 0 \\ 0 & 0 & 0 & -1 & -1 & 6 & -1 & -1 \\ 0 & 0 & 0 & 0 & -1 & -1 & 6 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 & 6 \end{pmatrix}$$

Where do they come from?

- Reordered sparse matrices
- Stiffness matrices from structural mechanics

Banded Linear Systems

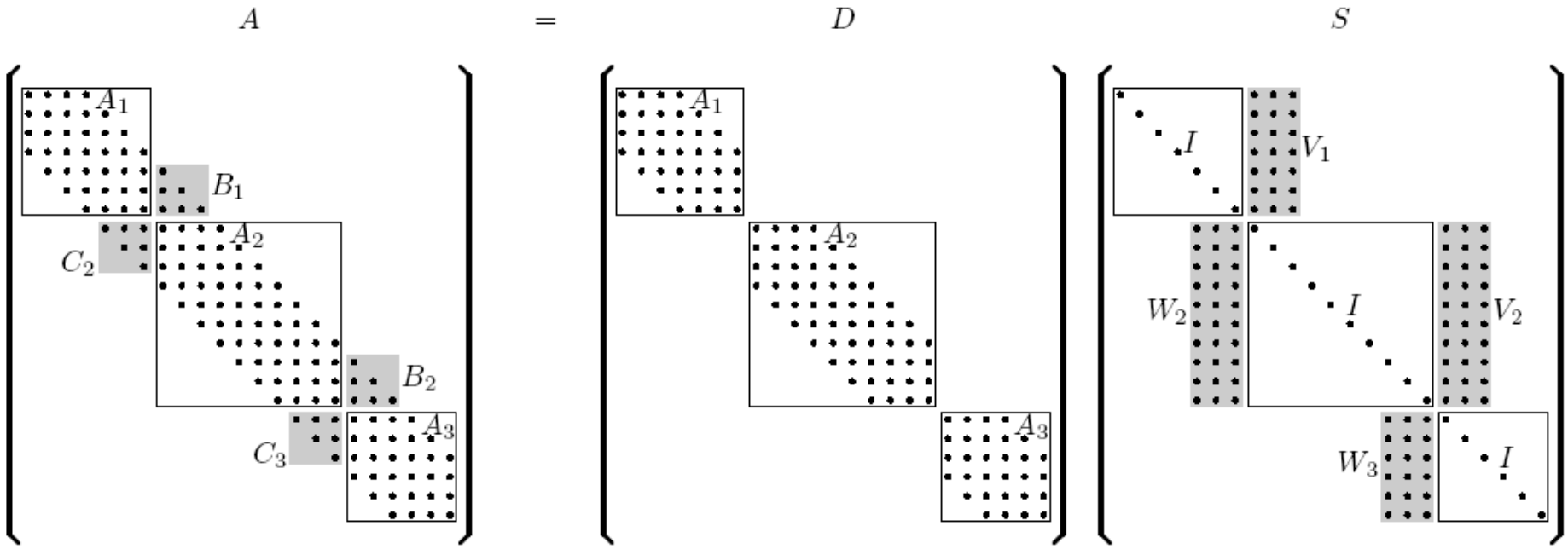
$$AX=B$$



$$A_{coef} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & & & & & & \\ a_{21} & a_{22} & a_{23} & a_{24} & & & & & \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & & & & \\ & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} & & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & & \\ & & & & & & a_{86} & a_{87} & a_{88} & a_{89} \\ & & & & & & & a_{97} & a_{98} & a_{99} \end{pmatrix}$$

Adapted from Richard O. Hill Jr., Elementary Linear Algebra, Academic Press Inc., 1986.

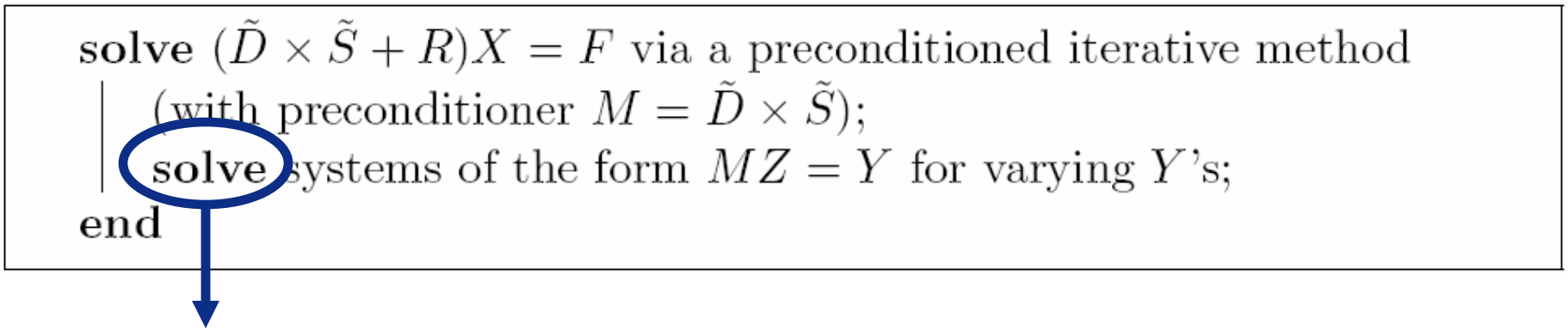
Spike-Based Decomposition



$$A = D\,S \qquad S = D^{-1}A$$

- A.H. Sameh and D.J. Kuck, "On stable parallel linear system solvers" *J. ACM*, 25:81-91, 1978.
- E. Polizzi and A.H. Sameh, "Spike: A parallel environment for solving linear systems" *Computers & Fluids*, 36:113-120, 2007.

Solving the Reduced System



Step 1: D G = F

- **D consists of decoupled systems, no sync**

Step 2: S Y = G

- **Except for junctions near identity blocks, this system is also decoupled**

Step 3: Compute R and apply corrections (iterative refinement, GMRES, BiCGStab)

Creating the Reduced System

solve $(\tilde{D} \times \tilde{S} + R)X = F$ via a preconditioned iterative method
  (with preconditioner $M \equiv \tilde{D} \times \tilde{S}$);
  solve systems of the form $MZ = Y$ for varying Y's;
end

Reduced system strategy	**Truncated** **Explicit** **Recursive** **On-the-fly**
Diagonal factorization strategy	**LU with pivoting** **LU without pivoting** **LU and UL without pivoting** **Alternate LU and UL without pivoting**

Selecting an Optimal Spike Strategy

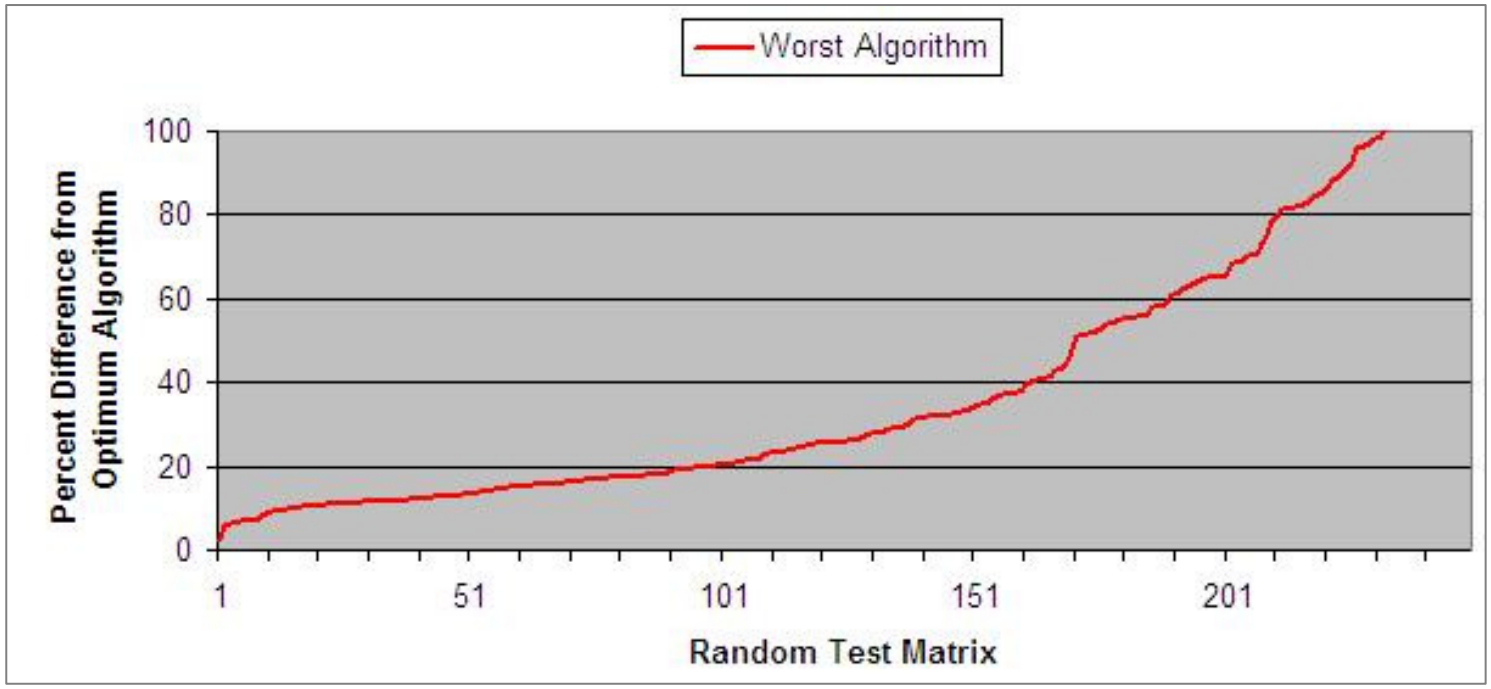
Combining the different reduced system, diagonal factorization, and outer iteration strategies yields dozens of possible solution schemes.

Simple heuristics can guide selection if the user has some information about the coefficient matrix, e.g.:

- Truncating the Spikes only works well for diagonally dominant matrices
- Pivoting should be used for ill-conditioned systems

However, even Spike experts have trouble predicting the optimum solution scheme.

Why Is Spike_Adapt Necessary?



Few users will be SPIKE experts

Selecting suboptimal strategy has serious performance ramifications

Boeing Design Explorer*

Developed by Boeing Phantom Works and Rice University

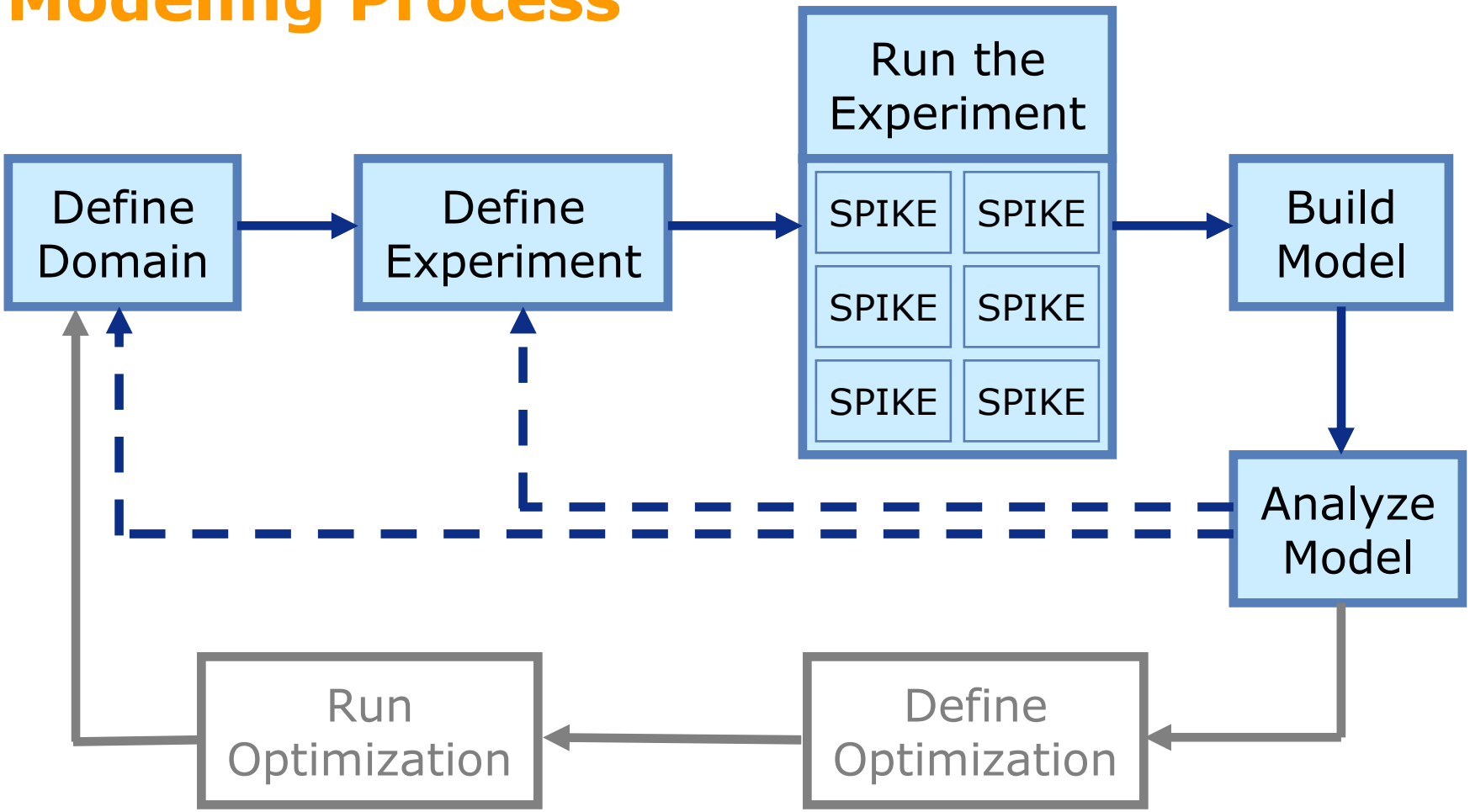
Used by Boeing for design optimization and to reduce design cycle time

- Design and analysis of computer experiments
- Automation framework
- Data modeling
- Design optimization

Commercially available from Phoenix Integration Inc.

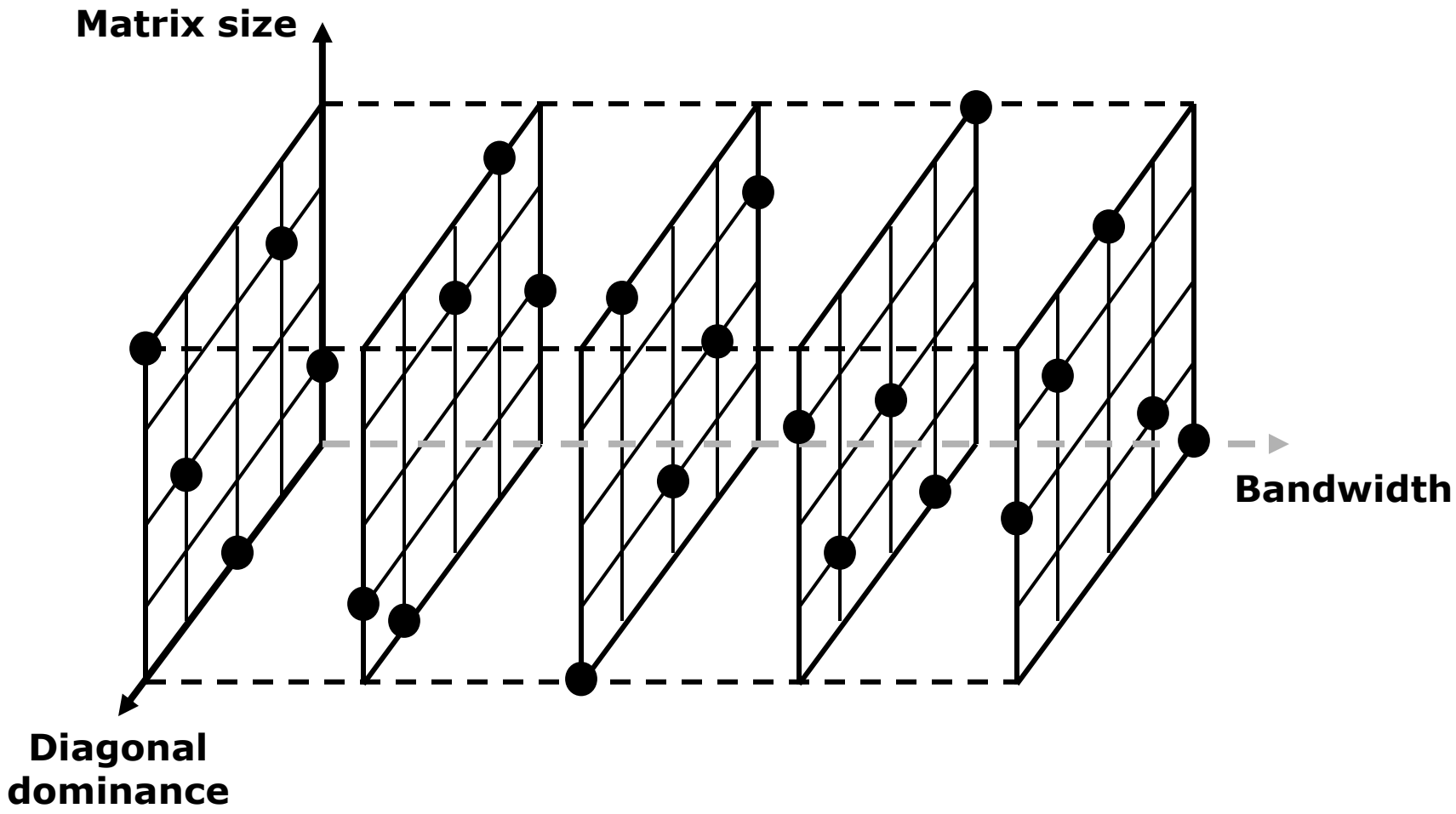
* Other brands and names may be claimed as the property of others.

Modeling Process



Adapted from Andrew Booker *et al.*, Design Explorer Training Manual, Mathematics & Computing Technology, Boeing Phantom Works, December 2004.

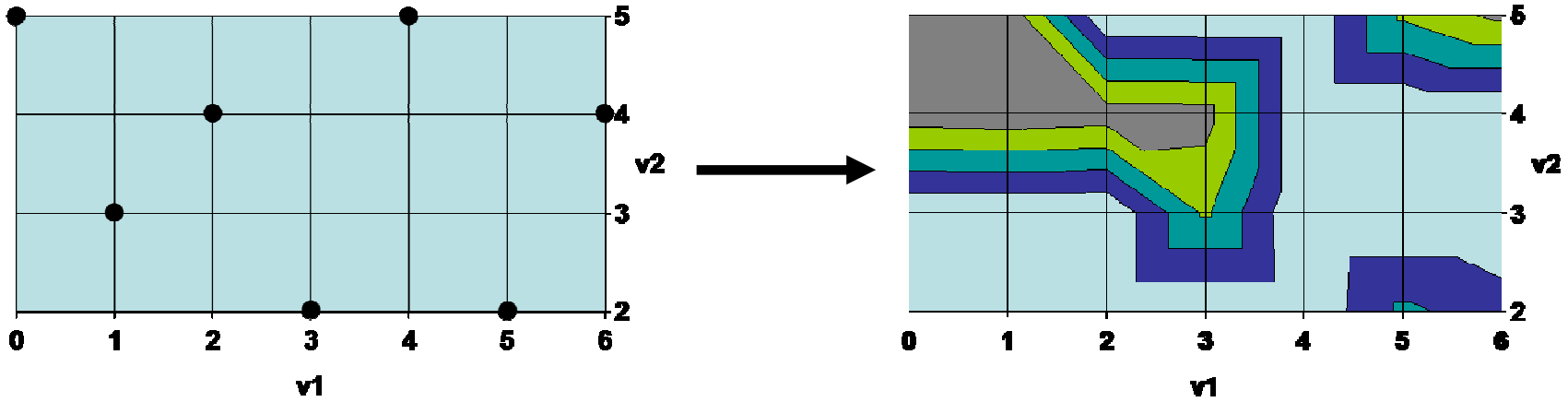
Space-Filling with Orthogonal Arrays



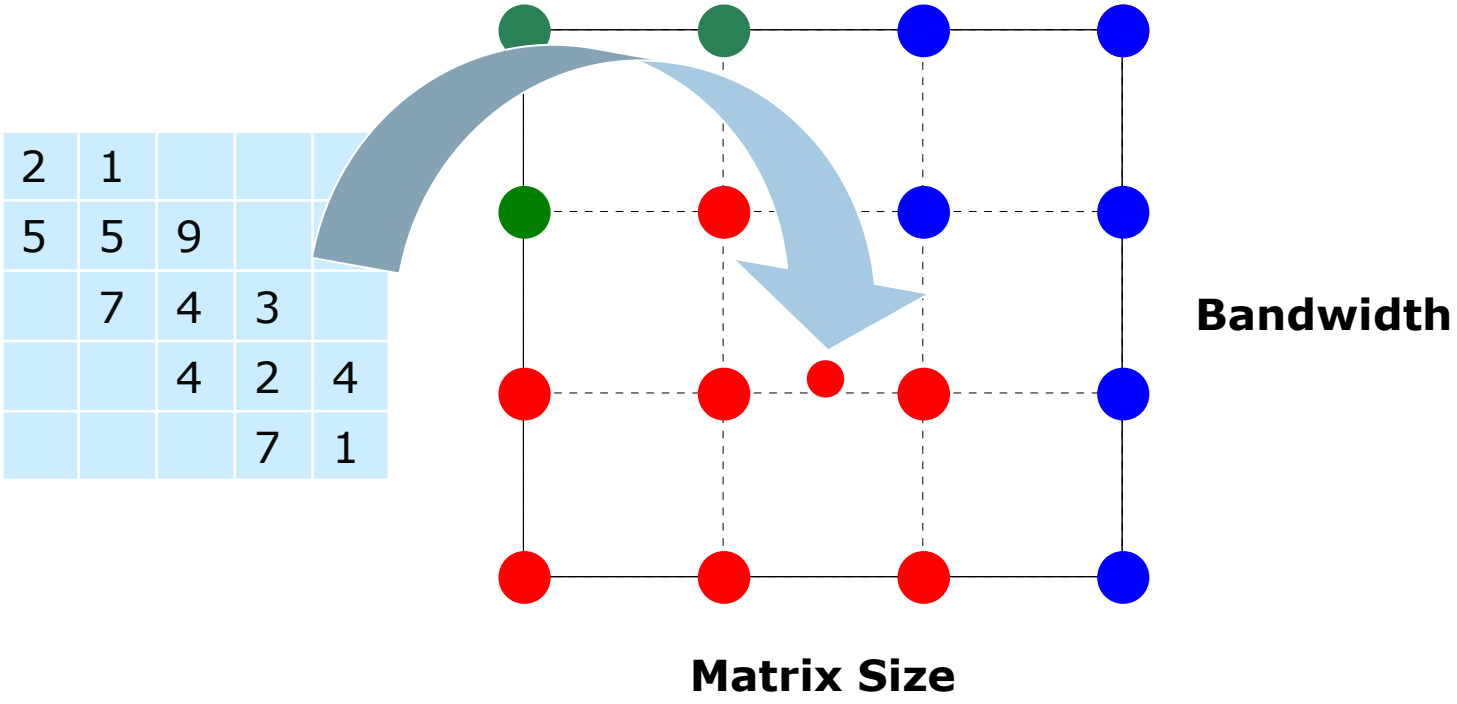
A.B. Owen, "Orthogonal arrays for computer experiments, integration, and visualization" *Statistica Sinica*, 2:439-452, 1992.

Building the Performance Models

1. Define the modeling domain in terms of independent variables: matrix size, bandwidth, diagonal dominance, sparsity, number of processors
2. Use orthogonal arrays to design an experiment
3. Run the experiment and build a kriging model
4. Assess model accuracy, refine experiment, and recalibrate model

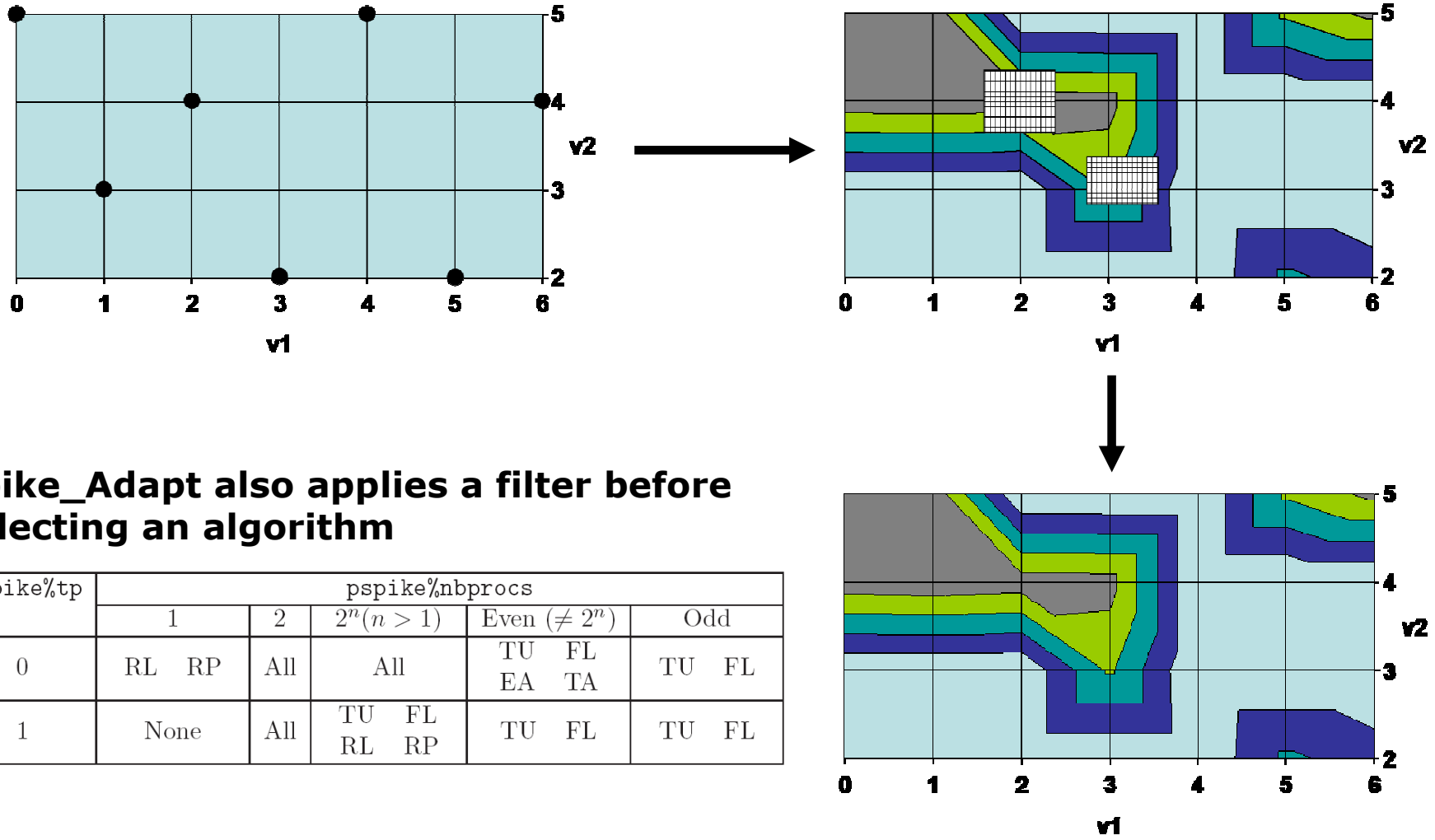


Grid-Based Performance Models



Colors represent optimum SPIKE algorithm at each grid node
SPIKE algorithm selected based on proximity to grid nodes
Accurate but computationally expensive

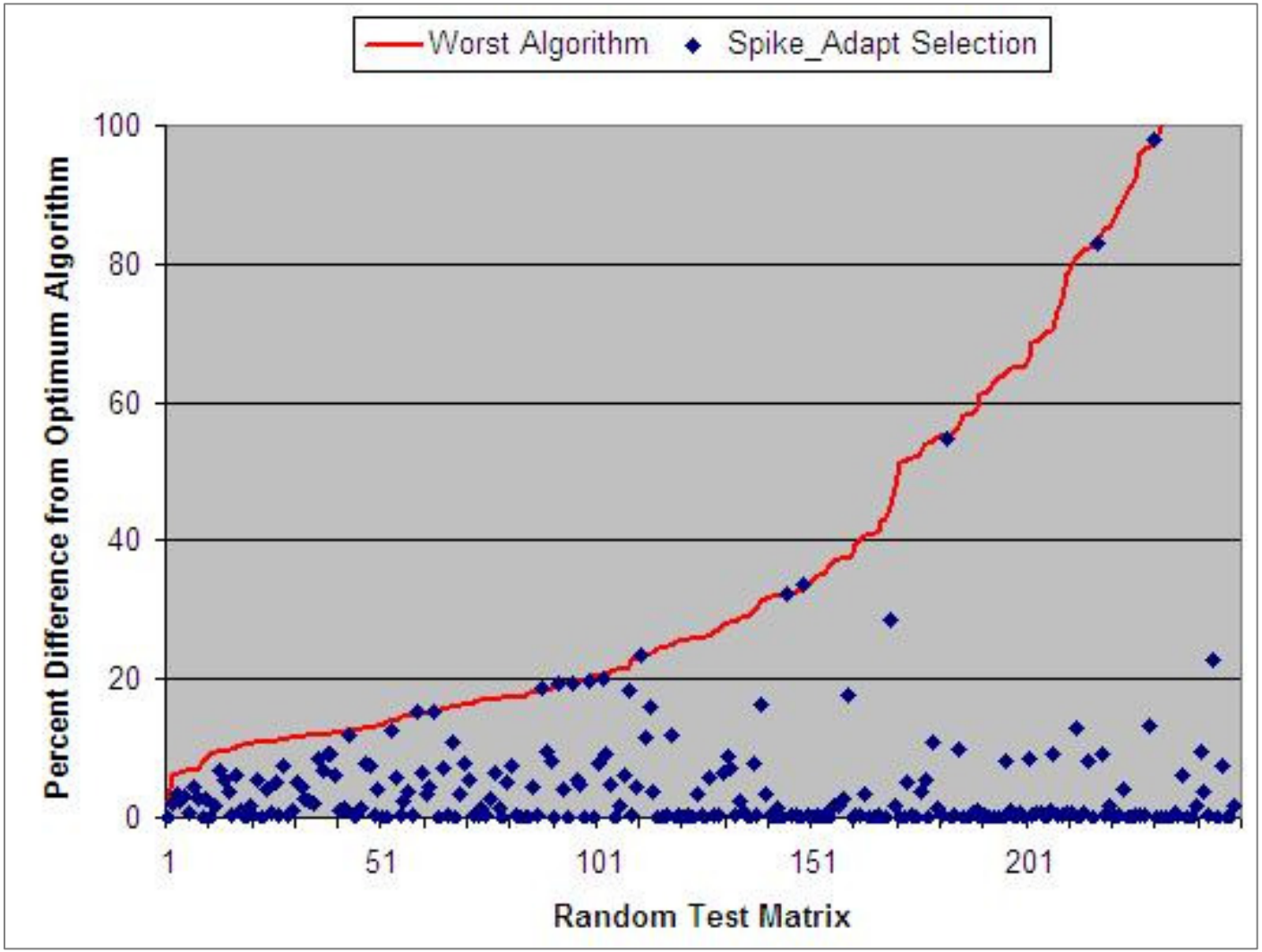
Spike_Adapt: Putting It All Together



Spike_Adapt also applies a filter before selecting an algorithm

pspike%tp	pspike%nbprocs				
	1	2	$2^n (n > 1)$	Even ($\neq 2^n$)	Odd
0	RL RP	All	All	TU FL EA TA	TU FL
1	None	All	TU FL RL RP	TU FL	TU FL

Spike_Adapt Accuracy



Future Directions

Include more independent variables (e.g., the number of RHS, CPU speed, memory bandwidth)

Train Spike_Adapt to select a good preconditioner

Train Spike_Adapt for specific applications areas (e.g., reservoir modeling)

Apply adaptation to the Intel® MPI Library

Summary

- The Intel® Adaptive Spike-Based Solver achieves better performance than other banded solvers
- The Spike_Adapt layer makes the Intel® Adaptive Spike-Based Solver easier to use

Sources for Additional Information

- Henry Gabb (henry.gabb@intel.com)
- http://whatif.intel.com – Intel® Adaptive Spike-Based Solver