Automated Characterization of Message Passing Application Communication Patterns

- **The Problem**
  - We want a concise way to express application communication demands
  - E.g., “10 Nearest Neighbor, broadcast, and reduce” instead of communication matrices

- **Our Approach**
  - Automated search using a library of patterns to identify collection of parameters
  - Detects best applications communication behavior
  - Adopts ideas from astronomy to scan through known, remove the known to make it easier to identify the unknown

- **Multiple Pattern Detection**
  - Search step recognizes a pattern, scales the recognized pattern as possible, and
  - Uses the scaled pattern to produce a communication matrix containing as yet unclassified communication behavior

### Performance API – PAPI

PAPI (Performance Application Programming Interface) provides the tool designer and application engineer with a consistent interface and methodology for use of the performance counter hardware found in most high performance systems. In addition, it provides access to a collection of components that express performance in a consistent and portable fashion.

- **Simple tool for performance engineers and associated analysis**
- **Add a layer of abstraction over existing performance tools**
- **Automates tedious and error-prone tasks**
- **Uses application and component tools: PAPI, TAG, HPC Toolkit, Open SpeedShop**
- **Setting up the environment for each tool, managing sequential and batch parallel jobs on different architectures**
- **Generating selective profiling configuration based on sampling results**
- **Configuring access to databases (e.g., TAG) and reusing and analyzable esthetics that are easily understandable; comparisons across multiple code versions**

### Autoperf

- **Example:** Studying the effects of optimizations on a G4arnet application (SimplifiedCalorimeter) compiled gcc 4.8 (any two versions can be compared this way).

### Empirical Roofline Toolkit

- **Background**
  - The Roofline model provides a visually intuitive approach to analyzing application performance.
  - Decomposition of application into key performance metrics
  - Principal: primarily oriented toward throughput metrics (GFlop/s vs. GB/s)

- **Our Approach**
  - Explores the type of Scalate tool, function names, and application behaviors by visualizing data on the roofline.
  - Explores by culling out L2, L3, L2+L3, or L3+L4, and adding L2 data to the roofline.

- **To date, applications for Roofline toolkit include:**
  - rwgs hôm: It requires a model of processor architecture that supports the model of executing applications.
  - If a model is not available, then the roofline model is difficult to select suitable benchmarks to represent all the processors.

- **To that end, SUPER and FASTMath have collaborated on developing a Roofline Toolkit to facilitate use of the model.**

### Integral ERT Release

- **Initial ERT release focused on characterizing and visualizing the Flop/MB roofline on CPU architectures.**
- **Peak Flops (using polynomial amenable to FMA instructions)**
- **Active Working Set Size**

### Beyond the Taxebok Model

- **Nominal:** Roofline is a throughput-oriented (streaming) performance model on a single level of memory or cache.
- **In reality:** Architectures have multiple levels of memory and applications have hierarchical working sets.
- **Thus:** Memory, bandwidth, and working set sizes are important metrics in understanding performance.
- **Expanded Roofline to capture performance on a two-level memory as a function of reuse and working set size...**
- **GPU performance is highly dependent on use of shared memory (application with most reasonable use on level 2 cache).**
- **CPU is much faster than GPU is overall.**

### How does performance vary on Kepler GPUs

- **CUDA 5.5 supports Unified Memory (GPU device memory as OS-controlled page cache on CPU memory)**

### GPU Programmers must choose whether...

- **Use CUDA and return to the GPU software stack...**
- **Access to memory is an important consideration for GPU applications.**
- **Page table optimization is critical for performance.**
- **Unified memory brings the system management facilities to the GPU.**