Performance Characterization and Benchmarking for High Performance Systems and Applications

Erich Strohmaier

NERSC/LBNL

Estrohmaier@lbl.gov

Co-sponsored by DOE/OSC and NSA

General Approach

- Develop a new quantitative characterization of algorithms and codes focusing on performance aspects.
- Avoid using any specific hardware models or concepts for this characterization (as much as possible).
- Develop synthetic performance probes and benchmarks testing these characteristics.
- Test the relation between benchmark performance and application code performance.
- Our focus is initially the performance influence of global data-access.

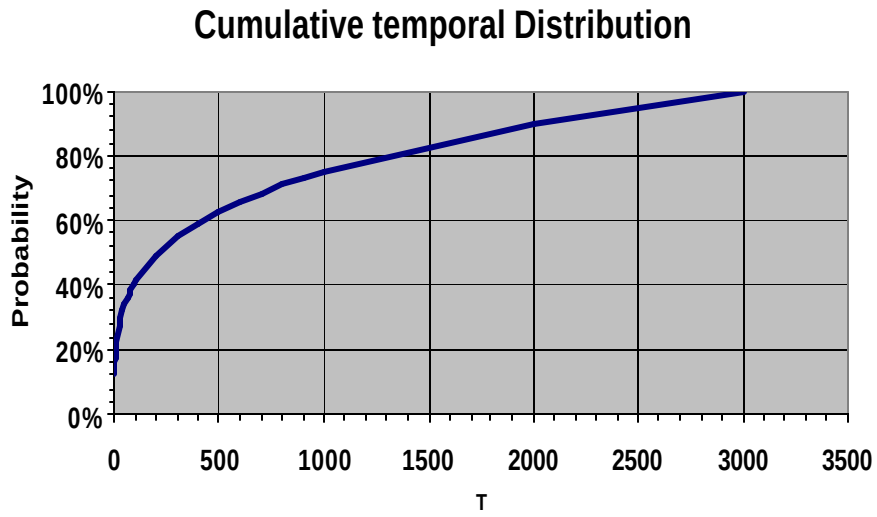
Data Access Characteristics

Data access pattern: What do we want to capture?

- *Re-use* of data – *Temporal locality.*
  - Hierarchical block-structured or recursive algorithms.
  - Hard to define hardware independent.
- Limitations of message sizes or vector-length – *Granularity.*
  - Limited by data-dependencies, etc.
  - Becomes particularly important in parallel context.
- Access to contiguous memory location - “Spatial Locality” – *Regularity.*
  - To characterize data-structures,
  - stride 1 access, etc.

Temporal Locality

- How can we *quantitatively* describe data re-use?
- Look at temporal distribution function:
  - The probability with which I have used my next data item within the last t accesses.
- Approximate the temporal distribution function of codes by a simple generic function with 1 parameter.



Temporal distance is similar to reuse distance, stack distribution, stack distance).

Granularity

Limitation of message sizes and vector-length due to data-dependencies.

- The amount of “pre-computable” addresses.
  - Access can be irregular (‘indirect’) or regular.
  - Limits the amount of dynamic reordering such as gather-scatter or message assembly.
- Granularity becomes very important for parallel version with explicit communication.
  - It (severely) limits message sizes.

Regularity

- Indirect (or “irregular”) data access becomes more and more important for many codes and is usually not avoidable.
- If irregular data access is present in a code it is likely to become the performance bottleneck (Amdahl’s Law).
- Characterizing the influence of indirect data access is essential for deriving proper bounds for achievable performance.
  - Irregular data access is “*our focus*”.

Benchmark – TOP500

- We develop a synthetic benchmark program based on non-uniform random data access with the same control parameters as our characterization.
- Select and fix a few sets of parameters which characterize different application domains.
- Use benchmark results for these parameter sets to complement TOP500 for these application domains.