Discovering the Runtime Structure of Software with Probabilistic Generative Models
Scott Richardson, Michael Otte, Michael Mozer, Amer Diwan, Dan Conners

**Designing a better microprocessor**
- Accurate profiling tools, needed to design better microprocessors, are very slow.
- Particularly if the microarchitectural design is emulated.
- Benchmarking suites exist to profile a microprocessor (e.g., SPEC2000).
- Need to speed up microarchitectural analysis.

**Simulation Points**
- Perform cycle accurate emulation only for simulation points.
  - A simulation point is an interval of program execution.
  - Random Sampling of the CPI (Cycles Per Instruction) trace for the program GZIP.

**Can we do better?**
- Strategically select simulation points.
  - Don’t choose simulation points at random.
  - Exploit a program’s phase behavior.
  - A phase is a distinct pattern of program behavior (e.g., CPI over time).

**Discovering phases**
- Infer phases from an executed basic block sequence.
  - Basic block trace is cheap to collect.
- Procedure
  - Periodically record Basic Block Vectors (BBVs) as the program executes.
  - A BBV is a histogram summarizing each period of basic blocks.
  - Cluster BBVs into phases.

**Related Work**
- SimPoint (Sherwood, Perelman, Harmerly, et al.)
  - k-means clustering of BBVs
- Multinomial mixture model (MMM)
  - Probabilistic interpretation of BBV distribution

**Likelihood based model evaluation**
- The BBV sequence is not significantly more likely under the HMMs than the mixture models.

**Model likelihood correlates with error in runtime profile estimates**
- The simulation point selection technique MaxLikelihood is most strongly correlated.

**Estimating the runtime profile: A comparison of the algorithms**
- No model outperforms SimPoint.

**Conclusions**
- SimPoint (k-means) does very well.
- To our surprise, the temporal constraints of the HMM did not improve the quality of the models.
- Dimensionality reduction does not have a significant impact on model performance.
- Phases are both code and data dependent.

**Future work**
- If observations are made more quickly, the temporal context may become more important.
  - Additionally, this requires that a smaller percentage of the program be functionally simulated.

**Discovering phases from an executed basic block sequence.**
- Basic block trace is cheap to collect.

**Discovering the temporal structure of programs**
- The HMM’s transition distribution allows us to analyze structure in programs.

**Code vs. data dependent phases**
- Code dependent phases
  - A given basic block is associated with a single phase.
- Data dependent phases
  - A given basic block may behave differently based on the context in which it is invoked.
- 36% of the phases are heavily code dependent.
- However, a significant portion of the basic blocks can appear in multiple phases.

**Phases are both code and data dependent.**
- Significant impact on model performance.
- Dimensionality reduction does not have a significant impact on model performance.
- Phases are both code and data dependent.

**Our approach**
- Our space of models varies over 5 dimensions
  1. HMM vs. mixture model
  2. Multinomial vs. Gaussian output
  3. What is the better interpretation of a BBV?
  4. BBVs vs. Reduced Dimensionality BBVs
  5. Does random projection corrupt the data?
  6. Techniques for selecting simulation points

**Our model of phases varies over 5 dimensions**
- HMMs transition to a different phase after 36% of the observations.
- The HMM predicts the next phase more accurately than an algorithm that assumes:
  - Next phase → Current phase
  - Next phase → Most frequent

**Discovering the phases**
- Infer phases from an executed basic block sequence.
- Basic block trace is cheap to collect.

**Procedure**
- Periodically record Basic Block Vectors (BBVs) as the program executes.
- A BBV is a histogram summarizing each period of basic blocks.
- Cluster BBVs into phases.

**Describing phases**
- A phase is a distinct pattern of program behavior.
  - Random Sampling of the CPI (Cycles Per Instruction) trace for the program GZIP.

**Applying an HMM**
- Observation at each time step corresponds to a BBV.
- Hidden state corresponds to a program’s latent phase.
- Alternative HMM emission distributions: multinomial and a Gaussian.

**Our model of phases varies over 5 dimensions**
- HMMs transition to a different phase after 36% of the observations.
- The HMM predicts the next phase more accurately than an algorithm that assumes:
  - Next phase → Current phase
  - Next phase → Most frequent

**Limitation with current models**
- SimPoint
  - Why treat a BBV as a continuous vector?
  - A BBV is composed of discrete blocks.
  - A better interpretation is multinomial.
- MMMs
  - BBV is randomly projected in a way that disrupts its interpretation as a count vector.
  - The BBV’s temporal structure is ignored.

**Our approach**
- Our space of models varies over 5 dimensions
  1. HMM vs. mixture model
  2. Multinomial vs. Gaussian output
  3. What is the better interpretation of a BBV?
  4. BBVs vs. Reduced Dimensionality BBVs
  5. Does random projection corrupt the data?
  6. Techniques for selecting simulation points

**Discovering the phases**
- Infer phases from an executed basic block sequence.
- Basic block trace is cheap to collect.

**Procedure**
- Periodically record Basic Block Vectors (BBVs) as the program executes.
- A BBV is a histogram summarizing each period of basic blocks.
- Cluster BBVs into phases.

**Describing phases**
- A phase is a distinct pattern of program behavior.
  - Random Sampling of the CPI (Cycles Per Instruction) trace for the program GZIP.

**Applying an HMM**
- Observation at each time step corresponds to a BBV.
- Hidden state corresponds to a program’s latent phase.
- Alternative HMM emission distributions: multinomial and a Gaussian.

**Our model of phases varies over 5 dimensions**
- HMMs transition to a different phase after 36% of the observations.
- The HMM predicts the next phase more accurately than an algorithm that assumes:
  - Next phase → Current phase
  - Next phase → Most frequent