What is the common, fundamental prerequisite for program optimizations?

Prediction of how the program would behave.

**Program Behaviors**
(calling freq, locality, loop tripcount, ...)

---

**Our Goal**

A new paradigm: **input-centric** program behavior analysis.

Include program inputs into the focus.
Outline

• Why input-centric?
• How to exploit inputs for program optimizations?

What are inputs?

• All the data that are not generated but accessed by the program
  • command arguments
  • input files
  • ... ...

Why input-centric?

Strong and predictive correlations between inputs and behaviors.

Better behavior analysis.

Better prediction.

Better optimizations.

Qualitative View

\[ \text{Prog Beh} = \text{Code} + \text{Inputs} + \text{Running Environments} \]

Input is the only deciding factor for a given program in a given environment.
Quantitative Evidence

- Reuse distance histograms of lucas [Zhong+:TOPLAS'09]

![Histogram of Reuse Distance](image)

- Small (6K data, 5M access)
- Medium (41K data, 40M access)
- Large (21M data, 644B access)
- Predicted large (after first 0.4%)

95% accuracy
93% accuracy for 15 programs

Current Treatments to Inputs

- Static compilation: code only.
- Offline profiling: not adapt to input changes.
- Runtime sampling: no explicit treatment to inputs, hence loses **proactivity** in prediction and optimizations.

Adaptivity-Proactivity Dilemma

- Adaptivity: dynamic optimization
- Proactivity: offline profiling, static compilation

Xipeng Shen @ William and Mary
**Drawbacks of Reactivity**

- Delays in optimizations
- Inferior decisions for local view
- Limited analysis & transformation

**Potential**

- JVM [CGO'09]: 21%
- Parallelization [ICPADS'09]: 50%
- GC [VEE'09]: 12%
- GPU [IPDPS'09]: 2.8X
- CMP Sched. [PACT'08]: 14%

**Opportunities from Inputs**

- Inputs come early
- Strong predictive input-behavior correlations lead to proactive prediction.
- The prediction is meanwhile cross-input adaptive.

**Outline**

- Why inputs?
- How to exploit inputs for program optimizations?
Overview

Input-centric evolvable programming system
input-centric adaptation
input-behavior modeling
input characterizations

Input Characterization

To extract important features from raw inputs

Challenges

- Input attributes rather than values matter
  - e.g., data distribution
- Complex input syntax & semantics
  - e.g., a graph or a tree or a signal
- Interplay among input components
  - overshadow, equivalence, default values, etc.

Domain knowledge needed; automatic solutions are difficult.

Specification-Based Solution [CGO’09]

- eXtensible Input Characterization Language
Automatic Solution

Seminal-Behavior Analysis

• Key observation: correlations in a program.

Strong correlations exist from loops to loops and to other types of behaviors.

Intuition

Prog Beh = Code + Inputs + Running Environments

Input is the only deciding factor for a given program in a given environment.
Seminal Behaviors

- Definition (informal)
  - Behaviors that can lead to accurate prediction of all behaviors of interest, and appear early in a run.
- Reflection of critical program input features.
- Implication
  - Enable proactive & adaptive optimizations.
  - Remove the needs for explicit input characterizations.

Candidate Seminal Behaviors

- Loop trip-counts
- Interface behaviors
  - values directly obtained from program inputs.
  - ignore massive file content
    - include corresponding loop trip-counts

Recognition of Sem Beh

1. Prog & inputs
2. Behavior collection
3. value sets of candidates

Affinity list construction

Affinity lists
Behavior Affinity List

Header can predict body accurately.

Affinity List of mcf

- Incremental construction
  - start with interface behaviors
  - iteratively find headers from the rest based on their predictive capability.

Predictive Capability

- Regression models
  - LMS (Least Mean Square)
  - Regression Trees
- 10-fold cross-validation

Recognition of Sem Beh
Refinement based on both predictive capability and earliness

Modeling and Adaptation

- Modeling --- construct predictive models
  Target Behaviors = f (Seminal Behaviors)
  - Machine learning problem
  - Classification (e.g., for optimization levels)
  - Regression (e.g., for calling frequencies)
- Adaptation
  - Runtime version selection
  - JIT
  - Dynamic speculation
  - ...

Evaluation

- Predictive capability of seminal behaviors
- Potential for program optimizations
## Conclusions

- Inputs strongly correlate with program behaviors and are beneficial to exploit.

- Input-centric behavior analysis is a promising solution.

## Acknowledgment

- Students
  - Yunlian Jiang, Eddy Z Zhang, Kai Tian, Yixun Liu, Feng Mao, Malcom Geathers

- C. Ding at Univ of Rochester offered the behavior equation and the initial insights on inputs.

- Compiler support from the TPO group at IBM Toronto Lab.

- Some program inputs from J. Amaral’s group at Univ of Alberta.

- Equipments from IBM, DoE Jefferson Lab, NVIDIA.

- NSF & IBM provided funding support.
Thanks!

Questions?