Detecting Behavior Phases in Utility Programs

Presenter: Xipeng Shen

Joint work with C. Ding,
S. Dwarkadas, and M. L. Scott

University of Rochester

Introduction

- Complex program analysis has evolved from static code analysis to behavior analysis
- Behavior analysis
  - To discover common behavior patterns for all executions through training executions
  - The patterns enable behavior prediction on any execution

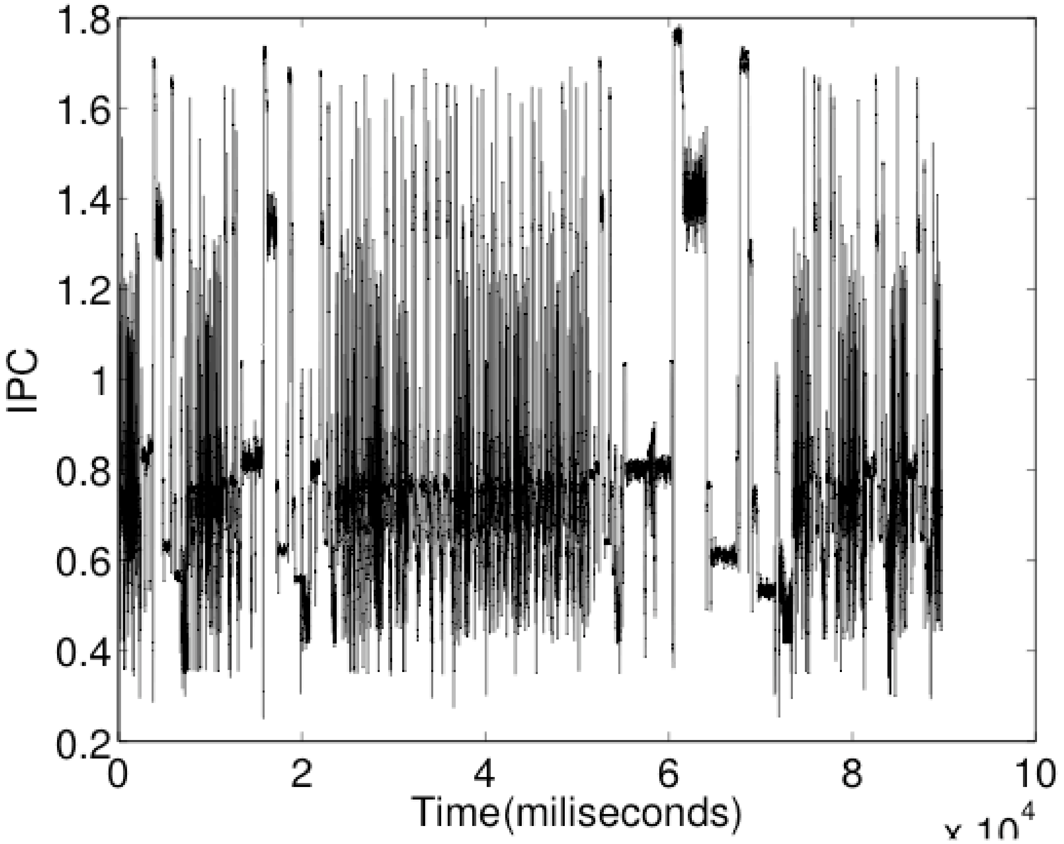
Behavior Phase

- Definition
  - A unit of the recurring behavior
- Captures high-level behavior patterns
  - Enables coarse-grain memory control and program parallelization
- Provides program behavior prediction
  - Guides dynamic software and hardware optimizations

Utility Programs

- Take a group of requests as inputs and serve them one by one
  - GCC: compile function by function
  - Compilers, compressions, transcoding utilities, interpreters, servers, ...
- Big challenges for phase analysis
  - Dynamic data and control structures
  - Strongly input-dependent behavior

Instruction Per Cycle(IPC) of GCC

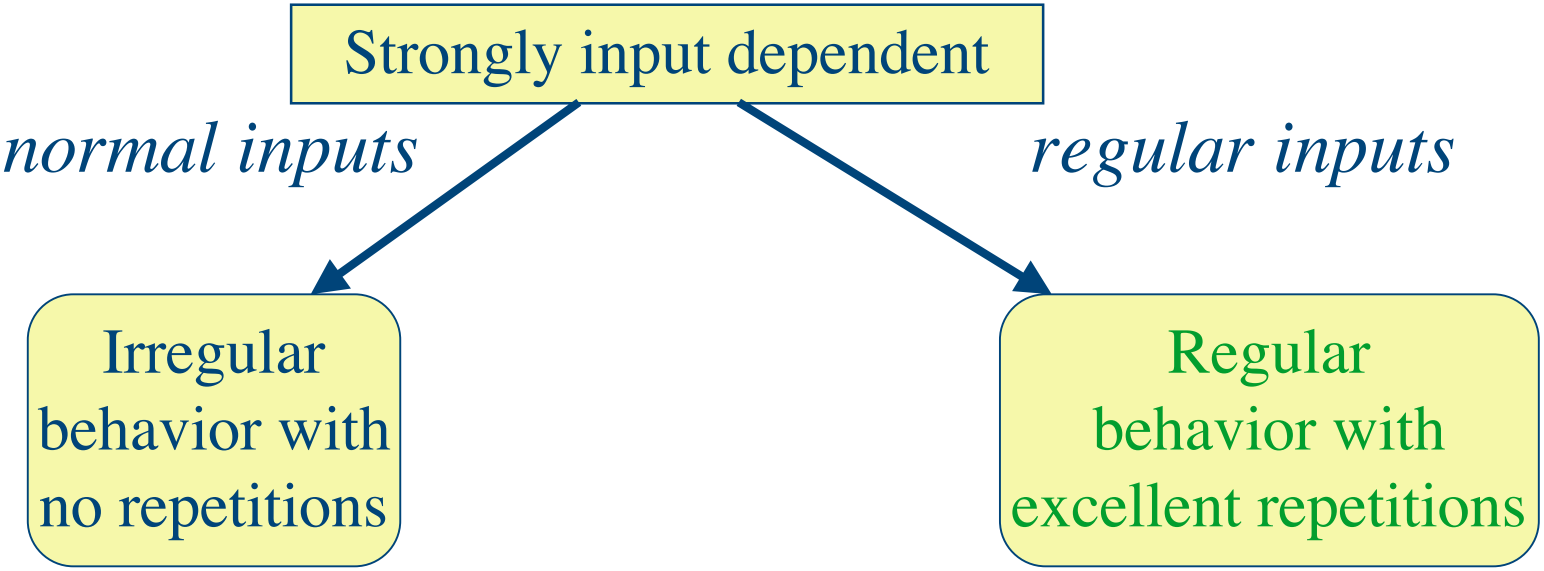


Outline

- Introduction
- Technology
  - Active profiling
  - Regularity filtering
  - Consistency filtering
- Evaluation
- Related work
- Conclusions

Active Profiling

- Converts challenges to opportunities



Strongly input dependent

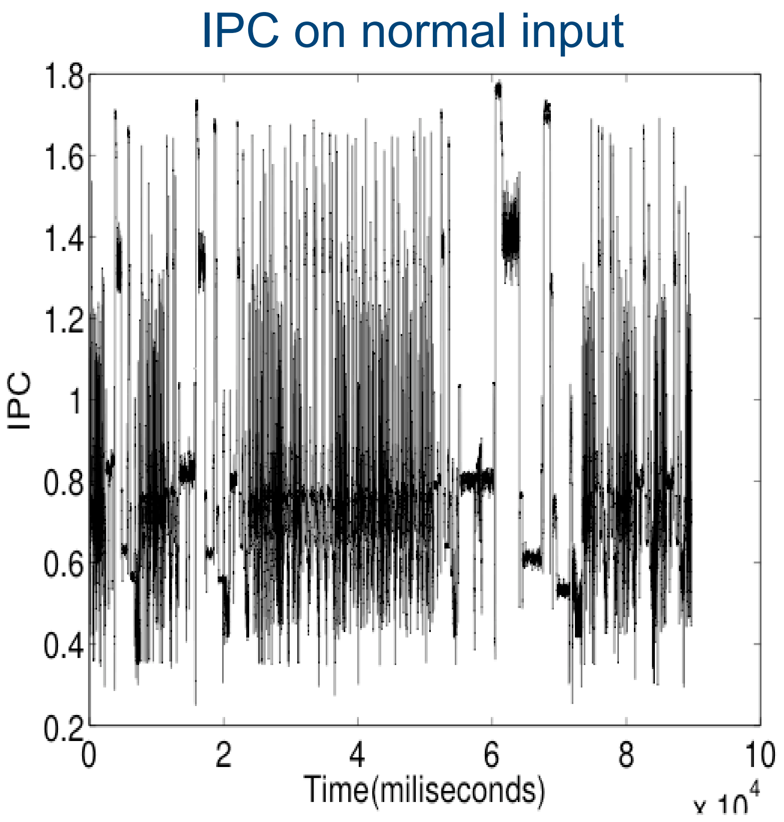
normal inputs

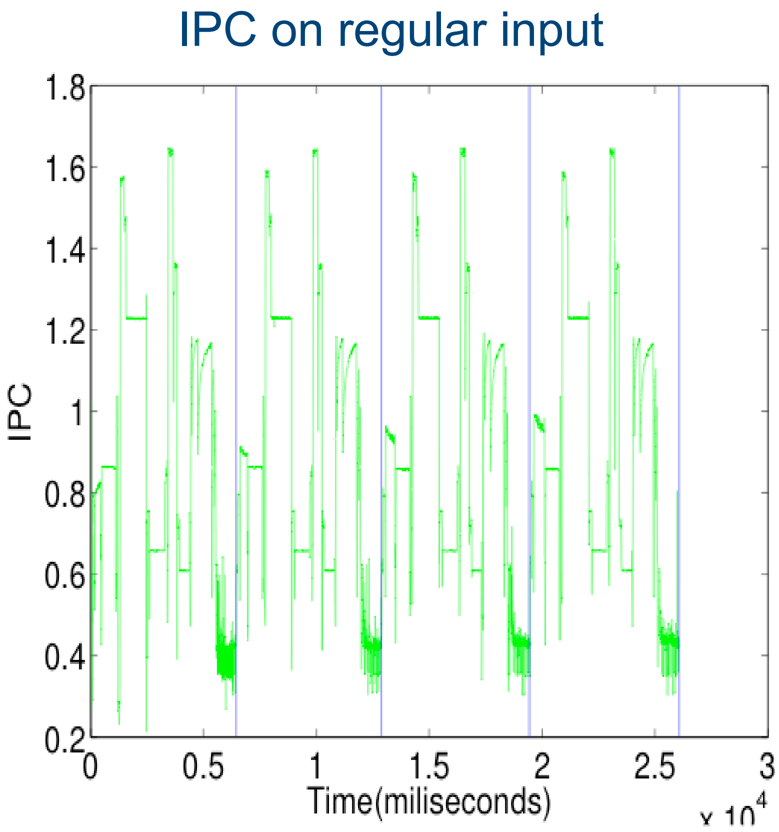
regular inputs

Irregular behavior with no repetitions

Regular behavior with excellent repetitions

GCC Normal & Regular IPC Graph





Regularity Filtering

- Filtering on dynamic basic block trace
  - Frequency-based filtering
    - Keep block *b* only if *freq(b)* equals the number of requests
  - Distance-based filtering:
    - Keep block *b* only if it has the similar recurring distance pattern as the majority

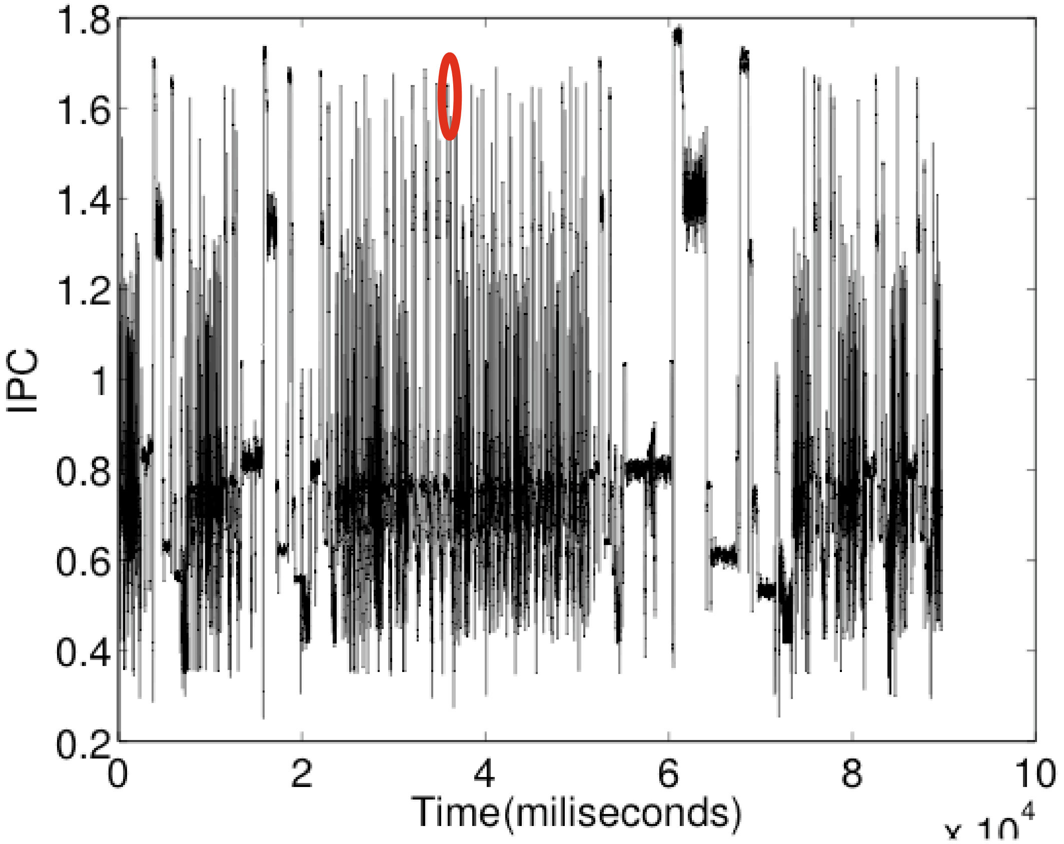
Consistency Filtering

- Profiling on a normal input
  - Check consistency of the markers
  - Find phase markers common to most normal request handling

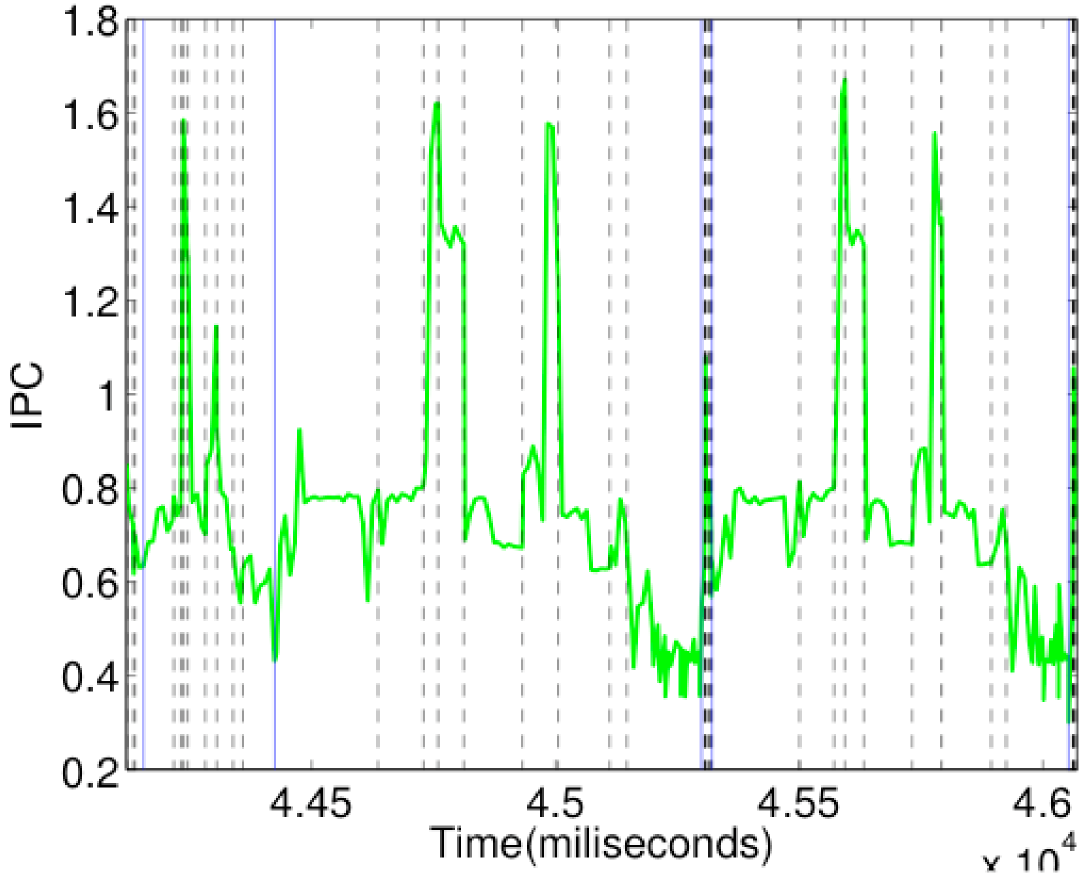
Evaluation

- Five SPEC95 and SPEC2k integer benchmarks
  - GCC, Compress, LI, Vortex, Parser
- Detection: Digital Alpha machines
  - ATOM: Binary code instrumentor
- Test: IBM POWER4 pSeries
  - PMAPI: hardware performance counter

Regularity across Request Handling (GCC)

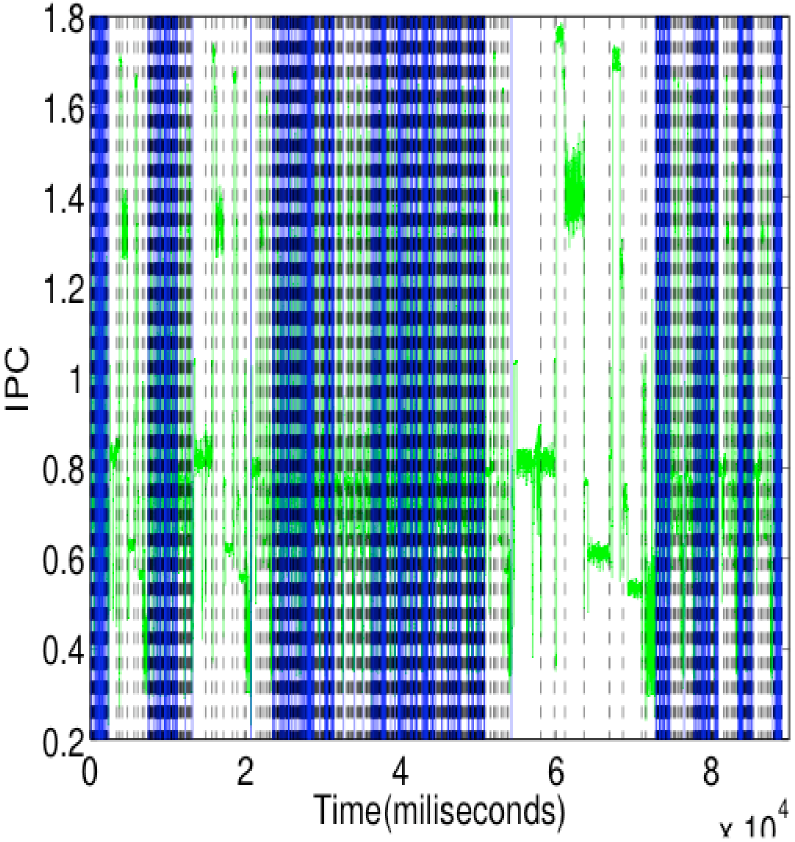


Regularity across Request Handling (GCC)

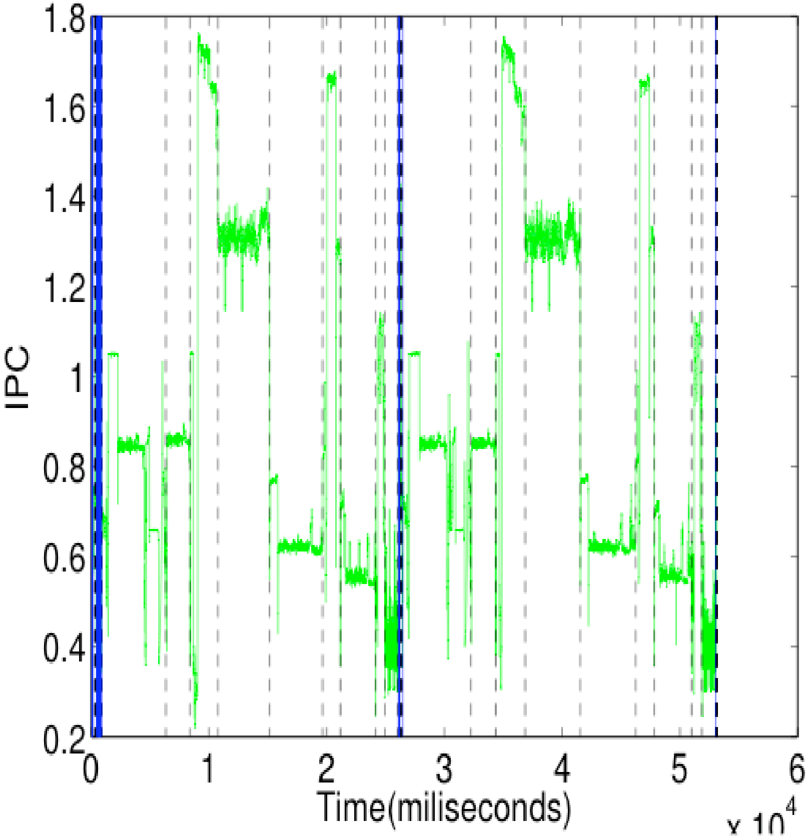


Regularity across Executions on Different Inputs (GCC)

scilab



166



Phase Behavior Consistency

- Consistency is the base for prediction
- Comparison to subroutine phases
  - Behavior phases have 2.6 to 21 times smaller variations in cache hit rates
- Comparison to ideal interval phases
  - For GCC and Compress, behavior phases have 1.7 to 4.3 times smaller variations
  - For Vortex, LI, Parser, both kinds of phases have very low variations (0.3% to 1.6%)
  - Unlike interval phase analysis, behavior analysis requires no thresholds

Uses

- Preventive memory management
  - 44% speedup (LI)
- Behavior-based coarse-grain parallelization
  - 2x speedup on 4-CPU Xeon machines, 8x on 16-CPU Sunfire machines (GZip & Parser)
- Phase-based memory monitoring
  - Predict memory demand
  - Memory leak detection

Related Work

- Locality phases
  - Not working for utility programs
- Code structure-based phases
  - Rely on static program structure
- Interval phases
  - Run-time overhead
  - Hard to determine the good interval length

Conclusions

- An active profiling based approach to analyze utility programs' behavior phases
- Captures coarse-grain behavior regularities
- Enables new program improvement techniques
  - Preventive garbage collection
  - Behavior-based parallelization
  - Memory usage monitoring and memory leak detection
- For more info, see our technical report:

  http://www.cs.rochester.edu/u/xshen/TR848.pdf

The End

Thanks!