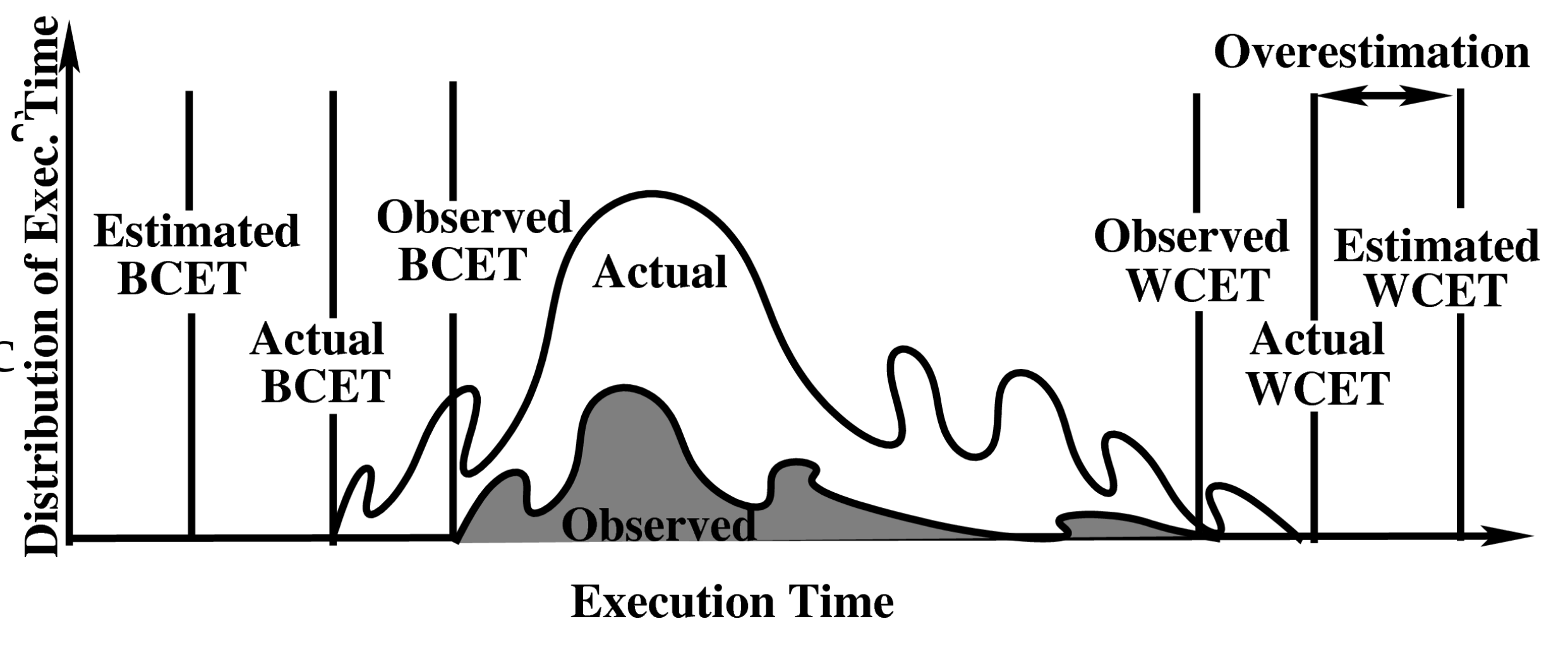
Department of Computer Science and Automation

Worst Case Execution Time Analysis (WCET):

• Critical for building Real Time Systems.
• Static WCET: Estimating WCET of a program before it is run.
• Dynamic WCET: Estimating Worst Case Completion Time of a program once it starts running.

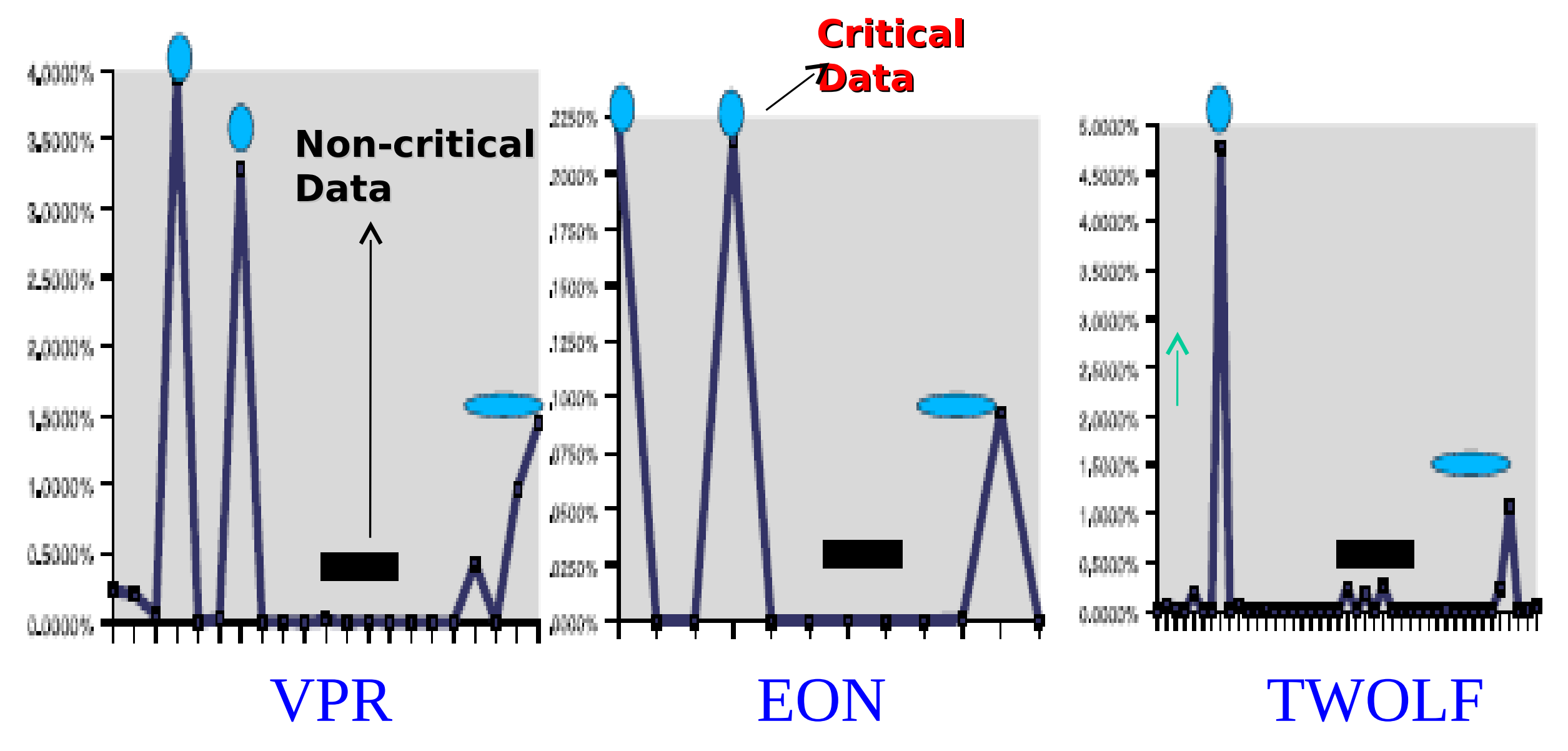
Special Focus:

• Hybrid Timing Approach: Estimating both Static and Dynamic WCET that uses program properties to help in arriving at an accurate estimate.
• Highly Useful for Soft Real Time Systems.



Worst Case Execution Time Analysis

Image-Courtesy: The Compiler Design Handbook – Optimizations and Machine Code Generation; Second Edition; CRC Press; Edited by Y.N.Srikant and Priti Shankar



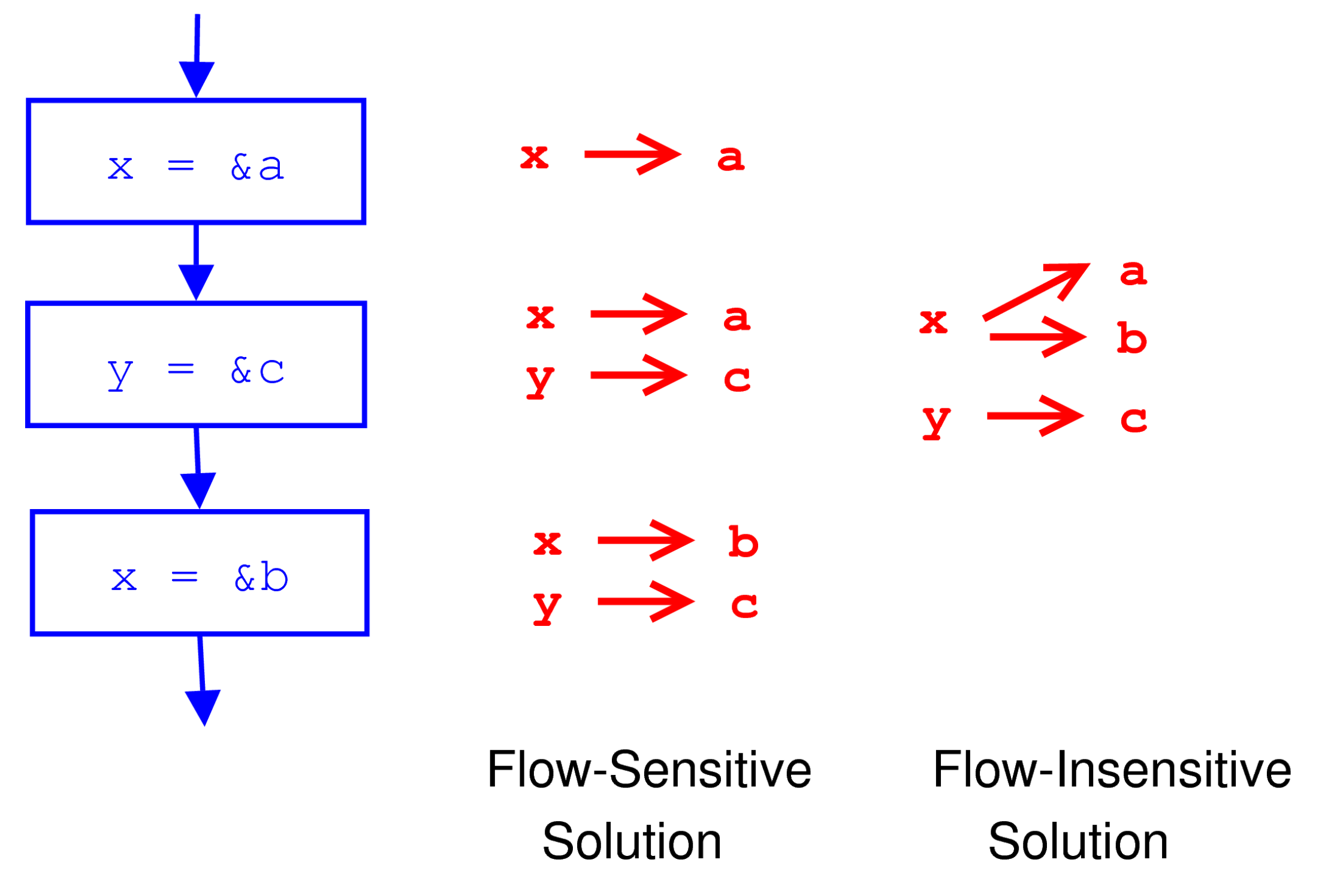
Critical vs Non-Critical Data in Programs

Critical Path Analysis for Energy Efficient Computing:

• Critical execution path in a program – The longest path in the dynamic event precedence graph in an execution.
• Events: Fetches (Instruction/Data), Decode, Execution, Commits, Write-backs, etc.
• Why are some paths Critical? - Memory latencies, Resource dependencies, Control dependencies and other bottlenecks
• Parallel paths co-exist in an execution

Special Focus:

• Data on Critical execution path
• Questions:
  •Does this data belong to a subset of data structures?
  •Metric for criticality?
  •Can this information be used for achieving energy efficiency?
• Exploiting a Heterogeneous Cache Architecture: Identify non-critical data accesses, and route them to a low power cache.



Program Analysis: Tradeoffs

Balancing Scalability and Precision of Program Analysis

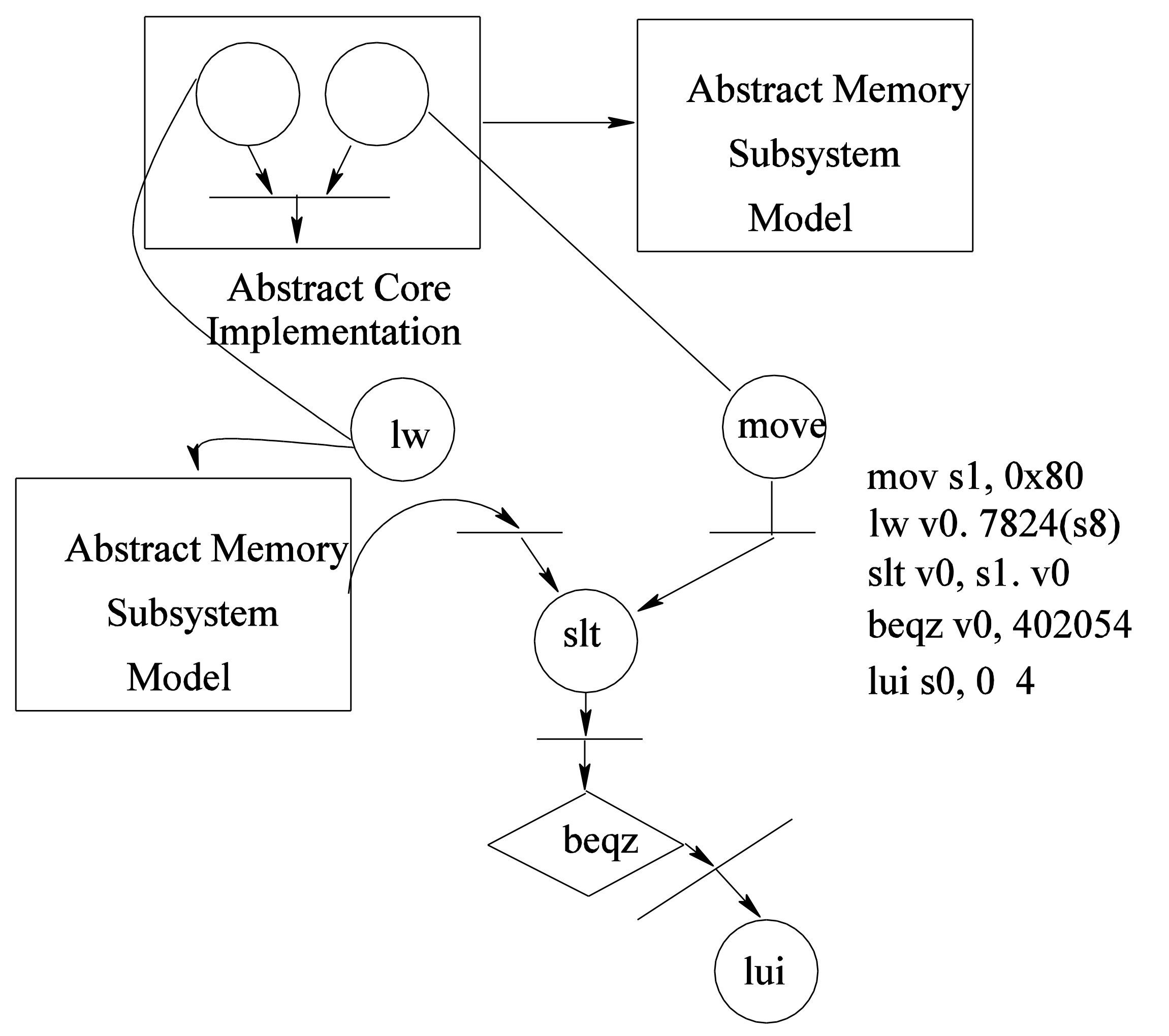
● Precision – Essential for Effective Compiler Optimizations
● Flow Insensitive Program Analysis – Less Expensive, but less precise
● Flow Sensitive Program Analysis – Precise, but not Scalable

Improved Precision at Critial-Points at Reduced Cost:

•Increased Precision at Specific Program Points
•Collapsing Non-Critical Nodes for Scalability
•Reduced Flow Graph
•Result: More Scalability (than a fully Flow Sensitive Analysis) and More Precision (than a Flow Insensitive Analysis)

Improved Flow Insensitive Solution for Non-Separable Data Flow Problems:

•Dependence among Data Flow Facts exacerbate Imprecision for Flow Insensitive Analyses
•Improvement to Standard Flow Insensitive Analysis: Generalization of Dominator Relation
•Resulting Solution: Very Close to Flow-Sensitive Solution


Abstract Core Implementation
Abstract Memory Subsystem Model
Abstract Memory Subsystem Model
lw
move
slt
beqz
lui

```
mov s1, 0x80
lw v0. 7824(s8)
slt v0, s1. v0
beqz v0, 402054
lui s0, 0  4
```

Accelerating Multi-Core Simulators

Accelerating Multi-Core Simulators

•Simulators – Critical Tool in assessing benefits of micro-architectural designs and choices
•Useful for Design Space Exploration also
•Major drawback: Orders of magnitude slower than real machines
•Worse for Multi-Core Simulators
•Major Trade-off: Speed (Functional) and Accuracy (Detailed) of Simulators

Special Focus:

•_Abstraction of Core Implementation
• Use of these abstract models for estimation of execution time

Dynamic Voltage and Frequency Scaling:

● Energy and Power: Limit Performance, Affect System Reliability, A Concern for Environment
● Dynamic Power: Proportional to the Square of Operating Voltage and Frequency
● Static Power: At-least linear in Operating Voltage
● Dynamic Voltage and Frequency Scaling – Alter Operating Voltage and Frequency at Run Time
● Meet Performance Constraints

Special Focus:

● Software Based Frequency and Voltage Scaling for
  ● Multiple Clock Domain Micro-architectures
  ● Multi-core Systems
● Uses Petri net based Program Performance Models for estimating performance of program regions with different voltage and frequency setings
● Chooses the least setting that meets performance constraints

Some Recent Publications:

• The Hot Path SSA Form: Extending the Static Single Assignment Form for Speculative Optimizations (CC 2010)
• Accelerating Multi-Core Simulators (SAC 2010)
• Profiling k-Iteration Paths: A Generalization of Ball-Larus Path Profiling Algorithm (CGO 09)
•Compiler Directed Frequency and Voltage Scaling for a Multiple Clock Domain Micro-architecture (Computing Frontiers 2008)
• Improved Precision at Reduced Cost at Critical Points (SAC 2008)
• Partial Flow Sensitivity (HiPC 07)
• Compiler Assisted Instruction Decoder Energy Optimization for Clustered VLIW Architectures (HIPC 07)
• Probabilistic Model of Data Cache Behaviour (EMSOFT 2009)
•Register File Optimization for Snooping Based Clustered VLIW Architectures (SBAC-PAD 07)
• Pragmatic Integrated Scheduling for Clustered VLIW Architectures (SPE 07)
• INTACTE: An Interconnect Area, Delay and Energy Estimation Tool for Microarchitectural Explorations (CASES 07) [With VLSI Circuits & Systems Lab, ECE]
•WCET Estimation for Executables in Presence of Data Caches (EMSOFT 07)
• Executable Analysis using Abstract Interpretation with Circular Linear Progressions (MEMOCODE 07)
• Micro-architecture Sensitive Empirical Models for Compiler Optimizations (CGO 07) [With Computer Architecture Lab]
• Preferential Path Profiling: Compactly Numbering Interesting Paths (POPL 07)

Contact:
NAME: Prof. Y. N. Srikant
Name of Lab: Compiler Lab
srikant@csa.iisc.ernet.in