ITAP: Idle-Time-Aware Power Management for GPU Execution Units

Mohammad Sadrosadati

Hajar Falahati

Arash Tavakkol

Lois Orosa

Hamid Sarbazi-Azad

Seyed Borna Ehsani

Rachata Ausavarungnirun

Mojtaba Abaee

Yaohua Wang

Onur Mutlu

SAFARI

ETH Zürich

Sharif University of Technology

IPM

Institute for Research in Fundamental Sciences

Carnegie Mellon University

Executive Summary

- **Motivation**: GPU execution units are frequently idle
  - Full-lane and partial-lane idleness
  - High static power
- **Problem**: static energy represents a high percentage of the energy consumption on execution units
- **Key idea**: Employ the most efficient power management mode for each idle period
- **Mechanism**: Predict the idle period length and apply power-gating and different levels of voltage-scaling
- **Results:**
  - 27.6% more static energy savings than the state-of-the-art mechanisms
  - Less than 2.1% performance overhead

Outline

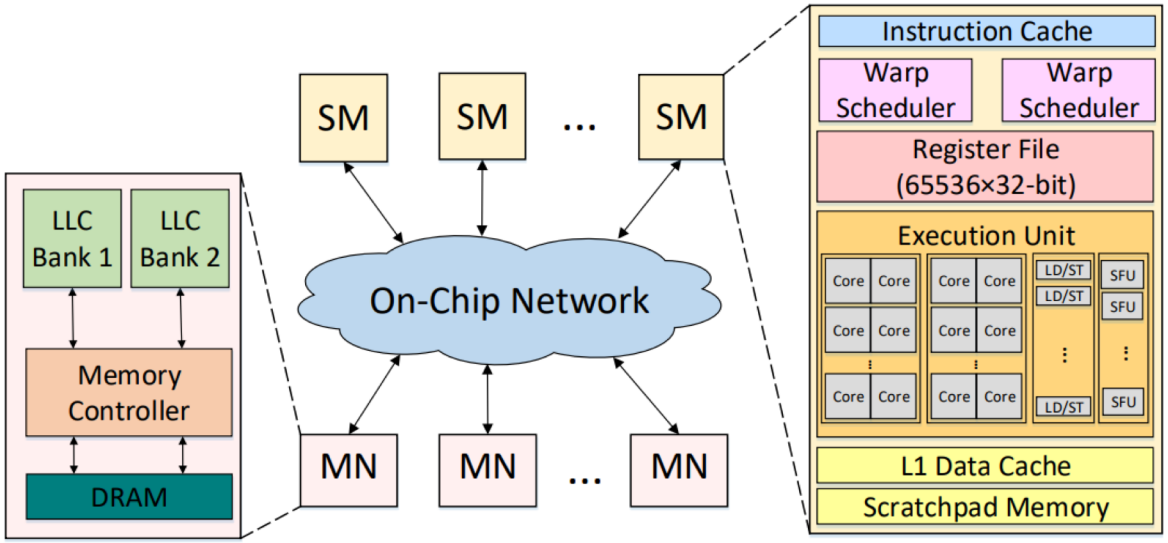
- Background
- Motivation
- Inefficiency of Prior Works
- ITAP
- Evaluation
- Conclusion

Outline

- Background
- Motivation
- Inefficiency of Prior Works
- ITAP
- Evaluation
- Conclusion

GPU Background

- Groups of threads are assigned to a **Streaming Multiprocessor (SM)**
  - Single Instruction Multiple Data (SIMD) execution units
- Threads within the SMs are divided into **warps**
- Threads inside a warp are executed in parallel **lock-step** manner
  - All threads execute the same instruction
- SIMD units are **time-multiplexed** between different warps
- A SIMD **lane** executes the **instructions of a thread**



Static Power Reduction Techniques

- The contribution of the **static power** to total power consumption **is significant** (e.g. 20% on an NVIDIA GTX 480 [Leng+ ISCA'13])

- **Sleep mode:** reduces the power supply
  - The execution units do not function properly

- **Wake-up:** the power supply is switched back to full in order to gain functionality
  - T_{wake_up} : wake-up time
  - E_{wake_up} : wake-up energy

- T_{break_even} : minimum idle time for that the energy savings can break even with the energy overhead of wake-up

Sleep Modes

Two main sleep modes:

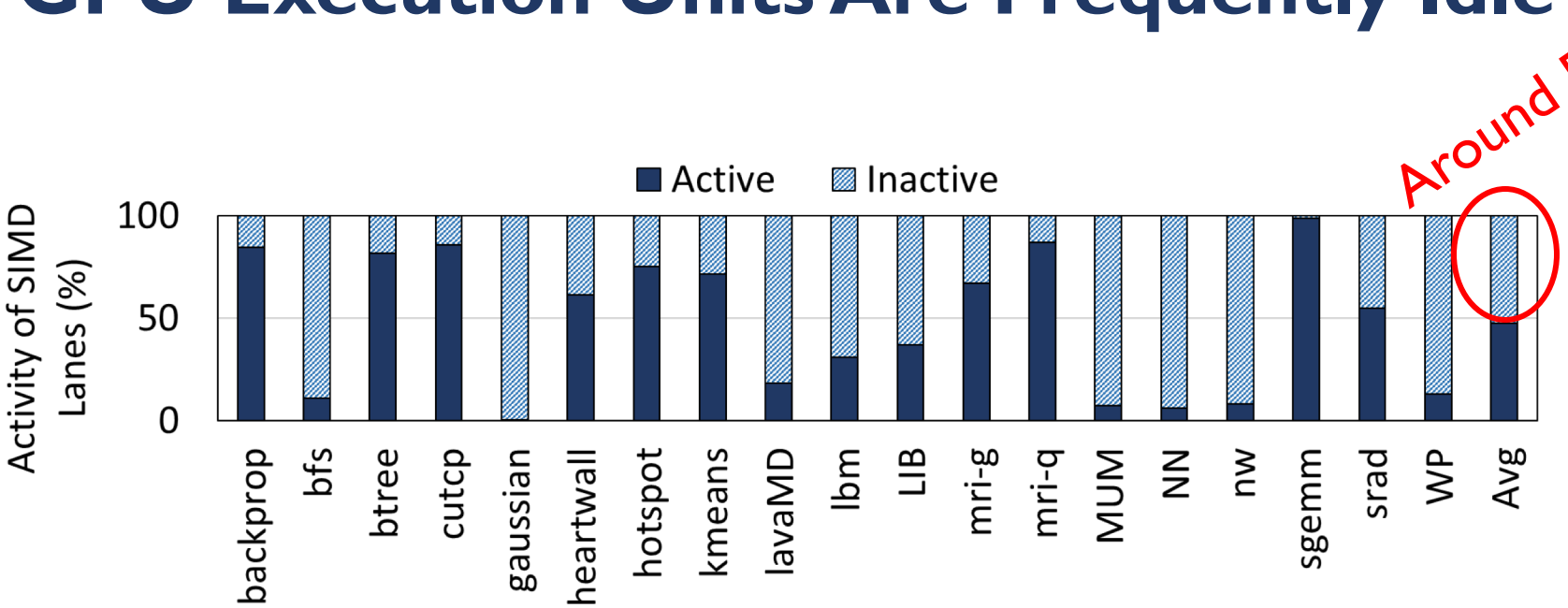
1. **Power-gating (PG)**
   - **Cuts off** the supply voltage entirely
   - **Sleep transistor** between the voltage supply line and the pull up network
   - T_{break_even} = 14 cycles

2. **Voltage-scaling (VS)**
   - Uses voltage regulators
   - T_{break_even} = 1-2 cycles ($VS_{0.3}$-$VS_{0.5}$)

Outline

- Background
- Motivation
- Inefficiency of Prior Works
- ITAP
- Evaluation
- Conclusion

GPU Execution Units Are Frequently Idle

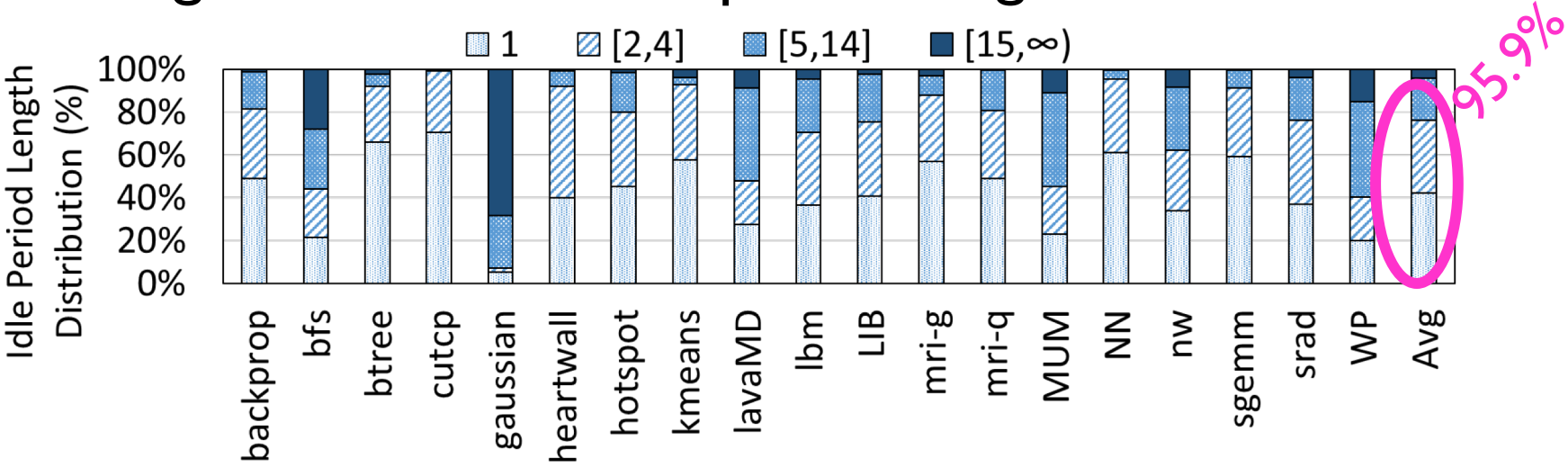


Why?

1. Partial-lane idleness
   - Branch divergence: when threads on a warp take different control flow paths
2. Full-lane idleness
   - There are no active warps to be scheduled for execution (e.g., memory accesses)

What Is The Main Challenge?

- Dealing with different idle period lengths



- **PG is not effective**:
  - The length of the idle periods is shorter than T_{break_even} in most of the cases
- Prior works **do not cover** all idle period sizes

Goal: cover 100% of idle periods in an efficient way

Outline

- Background
- Motivation
- **Inefficiency of Prior Works**
- ITAP
- Evaluation
- Conclusion

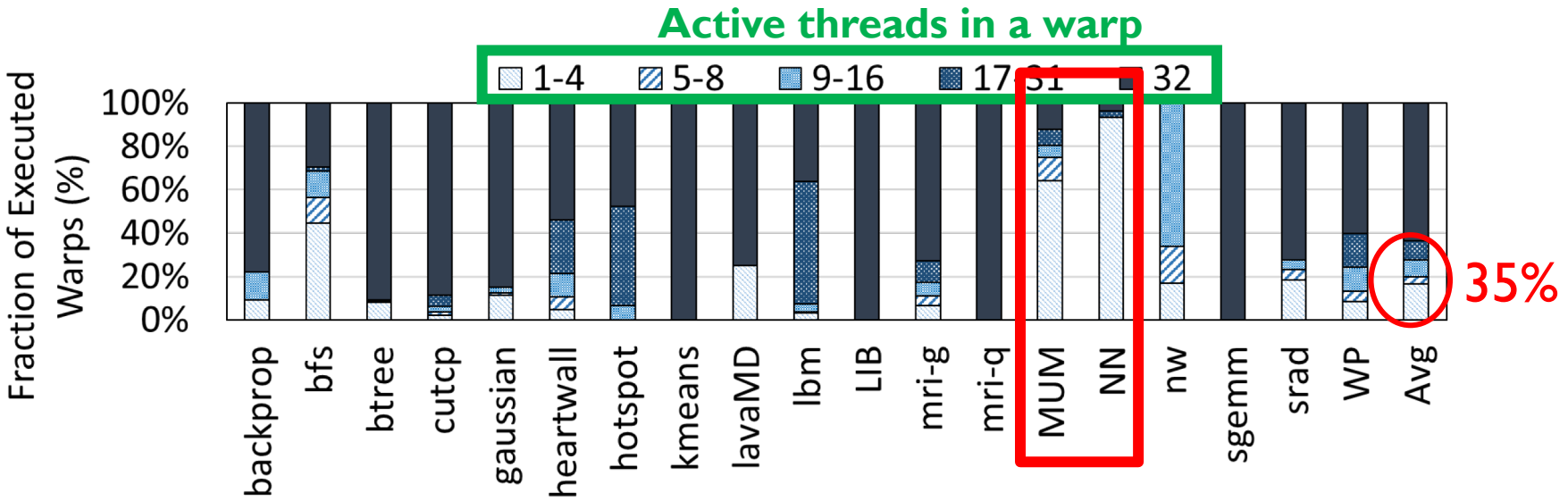
Prior Works

We will show the inefficiency of prior works in two cases:

1. Partial-lane idleness

2. Full-lane idleness

Partial-Lane Idleness



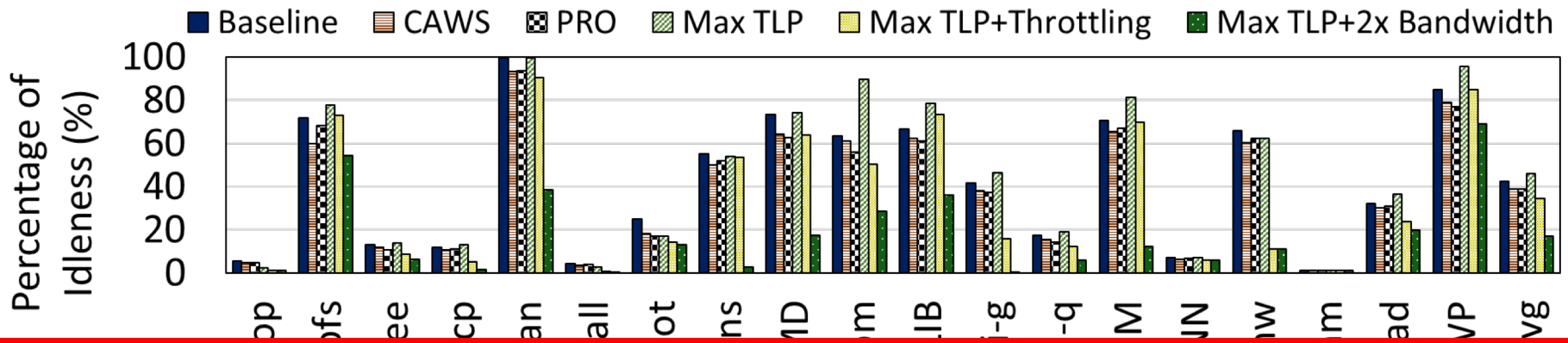
35% of warps are under-utilized

NN & MUM usually have 1-4 active threads

Prior solutions

- Reducing lane size [Vaidya+ ISCA'13] → performance overhead
- Thread block compaction [Fung+ HPCA'11]→ limited opportunity

Full-Lane Idleness



42% full-lane idleness for the baseline GPU

35% full-lane idleness even after using state-of-the-art warp schedulers (CAWS & PRO)

Even the ideal techniques have significant full-lane idleness

Reducing leakage power of idle execution lanes is very important

Outline

- Background
- Motivation
- Inefficiency of Prior Works
- ITAP
- Evaluation
- Conclusion

ITAP: Idle-Time-Aware Power Management

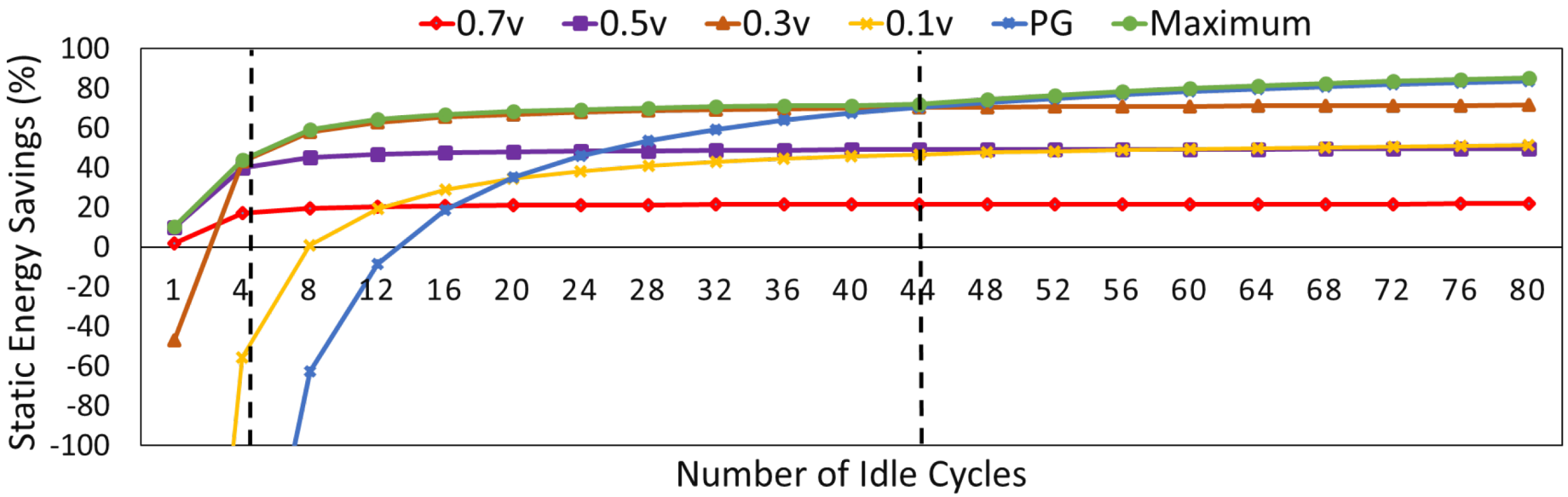
Observation:

- The **best static power reduction** technique depends on the length of the **idle period**

Key ideas:

- **Classify the idle periods on types** (i.e., short, medium, or long periods)
- **Assign** each type to the most effective **power reduction technique**
  1. **Short** idle periods → Voltage-scaling to 0.5VDD
  2. **Medium** idle periods → Voltage-scaling to 0.3VDD
  3. **Long** idle periods → Power-gating
- **Predict idle period** types
- Use a **peek-ahead approach** for improving performance

Power Reduction Techniques



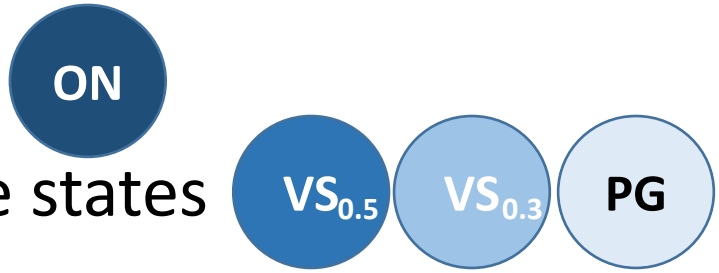
0.5v is the best for idle periods "<=4"

0.3v is the best for idle periods "5-44"

PG is the best for idle periods ">44"

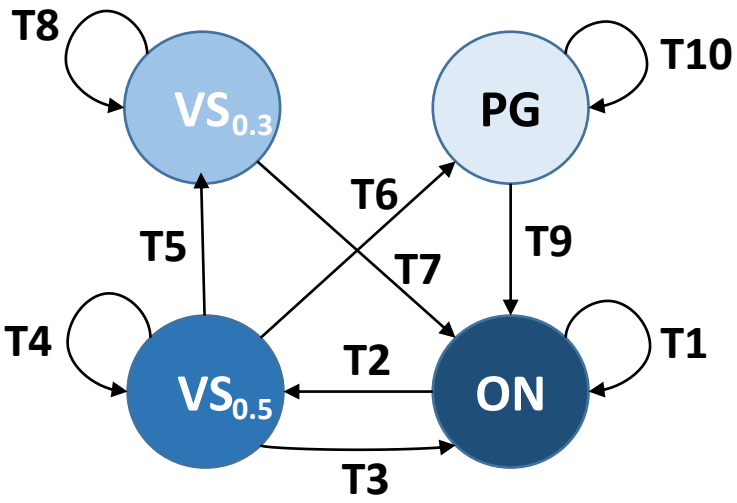
Predicting Idle Period Type

- **Finite State Machine (FSM)** with four states to predict the idle period type
  - One active state
  - Three sleep mode states



- **Two counters:**
  - Mode-change counter (**counter$_{mode}$**):
    - Keeps or changes the mode
    - **Thr$_{switch}$** is the threshold for switching the mode
    - **Incremented** when the idle time is > 8 cycles
  - Confidence Counter (**counter$_{conf}$**):
    - Differentiates between medium and long periods
    - **Thr$_{long}$** is the threshold for switching to a long period
    - **Decremented** if the idle time is < 48 cycles

ITAP's FSM



T1: The lane is active
T2: The lane is idle
T3: The lane is active
T7: The lane is active
T9: The lane is active

T8: The lane is idle
T10: The lane is idle
T4: lane idle for **less** than **4 cycles** **or** $\mathbf{counter_{mode} < Thr_{switch}}$
T5:

1. lane idle for more than **4 cycles**
2. $counter_{mode} > \mathbf{Thr_{switch}}$
3. $counter_{conf} < \mathbf{Thr_{long}}$

T6:

1. lane idle for more than **4 cycles**
2. $counter_{mode} > \mathbf{Thr_{switch}}$
3. $counter_{conf} > \mathbf{Thr_{long}}$

When a lane changes from idle to active, ITAP always exits the sleep modes

ITAP Issues

- **Our mechanism has two issues**

  1. ITAP **always selects $VS_{0.5}$ mode first**, no matter the length of the idle period
     - Room for optimization

  2. ITAP can not figure out when an execution lane should be **woken up ahead of time**
     - Performance overhead due to T_{wake_up}

- To solve these issues, we propose a **peek-ahead technique**

Peek-Ahead Window

- **Goal:** know the state of the lane in the near future

- **Implementation:**
  - A 3-cycle peek-ahead window
  - Slightly modify the round-robin warp scheduler
  - Determine two future warps to schedule in addition to the currently-selected warp

- **Result:** hides the wake-up latency of each power reduction technique
  - **One** cycle for 0.5v
  - **Two** cycles for 0.3v
  - **Three** cycles for PG

- More details in the paper

ITAP Granularity

ITAP can be applied to each lane individually **or** use the same reduction mode for all of idle lanes

- Fine-grain implementation
  - Apply power reduction technique for each idle lane individually
  - Different lanes can be in different power reduction modes
  - Higher accuracy
  - Larger overhead (due to per-lane voltage regulators)

How to select the power reduction mode for all idle lanes?

ITAP Coarse-grain implementation

- The ITAP algorithm to select the power mode is applied to every single lane

If any idle execution lane is selected to be in **$VS_{0.5}$** mode
then the selected power mode for all idle lanes is **$VS_{0.5}$**

If no idle execution lane in **$VS_{0.5}$** and at least one idle execution lane in **$VS_{0.3}$**
then the selected power mode for all idle lanes is **$VS_{0.3}$**

If all lanes are selected to be in **PG** mode
then the selected power mode for all idle lanes is **PG**

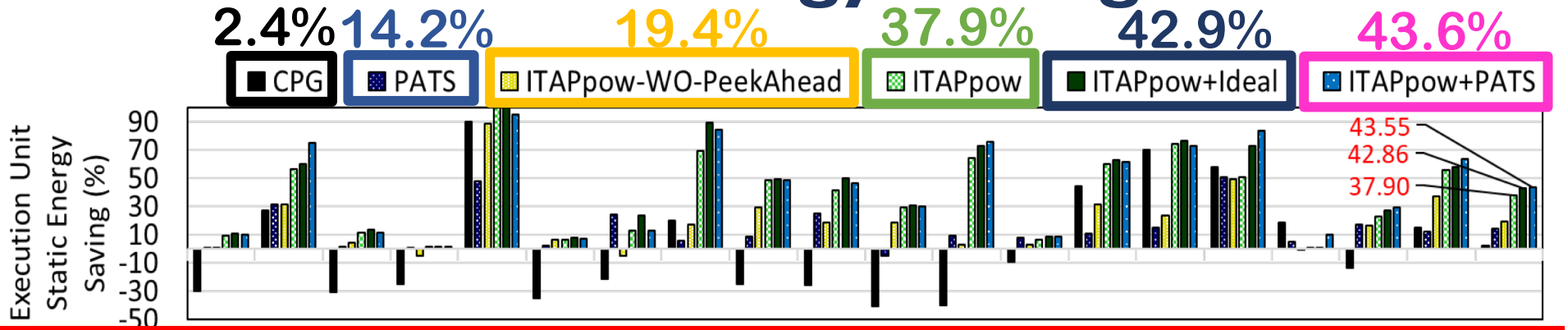
Outline

- Background
- Motivation
- Inefficiency of Prior Works
- ITAP
- **Evaluation**
- Conclusion

Evaluation Methodology

- **Simulator:** GPGPU-Sim [Bakhoda+ ISPASS'09] modeling NVIDIA Pascal
- **Energy:** GPUWattch [Leng+ ISCA'13] and HSPICE
- **19 Workloads from** Rodinia, Parboil and ISPASS benchmark suites
- **Comparison points:**
  - Conventional Power-Gating (**CPG**)
  - Pattern-aware warp scheduling (**PATS**) [Xu+ PACT'14]
    - State-of-the-art scheduler-aware PG technique
  - Variants of **ITAP**
    - Ideal
    - With and without peek-ahead
    - Power aggressive and performance aggressive
  - **ITAP+PATS**

Static Energy Savings



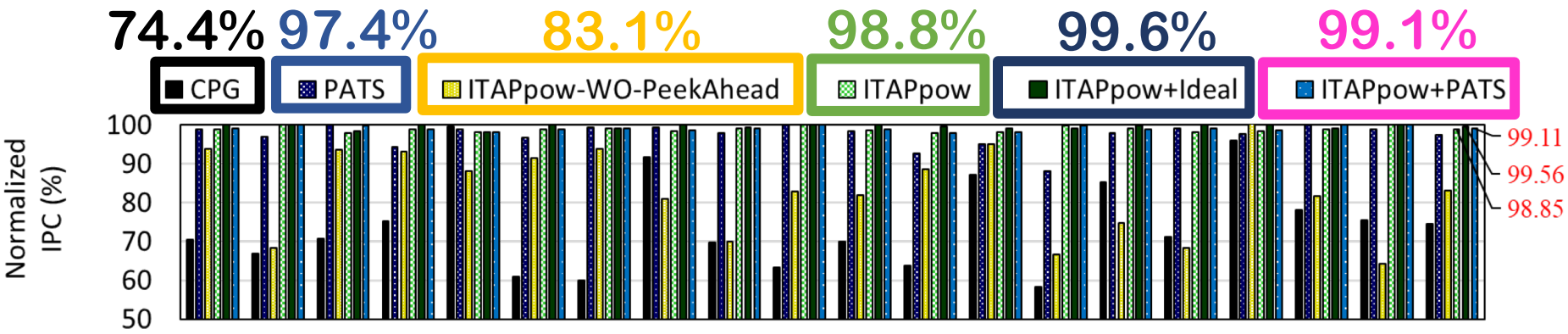
ITAP reduces the static energy by 36.3% compared to CPG and by 27.6% compared to PATS

Peek-ahead improves the ITAP energy savings by 24.3%

PATS defragments the idle periods and improves the opportunity of applying more powerful power reduction modes in ITAP

ITAP+PATS: additional 9.1% compared to ITAP

Performance Overhead



ITAP has low performance overhead compared to CPG and PATS

ITAP has up to 2% performance overhead compared to no static power reduction

CPG has up to 41% performance overhead

PATS has up to 12% performance overhead

Also in the paper…

- Different optimization goals of ITAP
  - Power-aggressive
  - Performance-aggressive
- Detailed analysis of hardware overheads
- Sensitivity analysis on
  - ITAP's **prediction parameters**
  - SIMD **lane size**
  - ITAP **granularity**
  - Effect of $\mathbf{T_{wake\_up}}$ and $\mathbf{T_{break\_even}}$

Outline

- Background
- Motivation
- Inefficiency of Prior Works
- ITAP
- Evaluation
- Conclusion

Executive Summary

- **Motivation**: GPU execution units are frequently idle
  - Full-lane and partial-lane idleness
  - High static power
- **Problem**: static energy represents a high percentage of the energy consumption on execution units
- **Key idea**: Employ the most efficient power management mode for each idle period
- **Mechanism**: Predict the idle period length and apply power-gating and different levels of voltage-scaling
- **Results:**
  - 27.6% more static energy savings than the state-of-the-art mechanisms
  - Less than 2.1% performance overhead

ITAP: Idle-Time-Aware Power Management for GPU Execution Units

Mohammad Sadrosadati

Hajar Falahati

Arash Tavakkol

Lois Orosa

Hamid Sarbazi-Azad

Seyed Borna Ehsani

Rachata Ausavarungnirun

Mojtaba Abaee

Yaohua Wang

Onur Mutlu

SAFARI

ETH Zürich

Sharif University of Technology

IPM

Institute for Research in Fundamental Sciences

Carnegie Mellon University