Modular Reinforcement Learning for Self-Adaptive Energy Efficiency Optimization in Multicore System

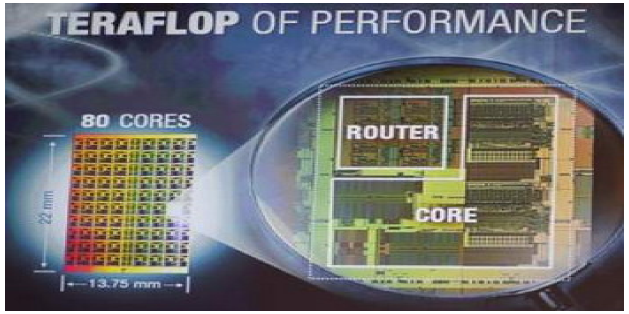
Presenter: Zhongyuan Tian

Co-authors: Zhe Wang, Jiang Xu, *etc.*

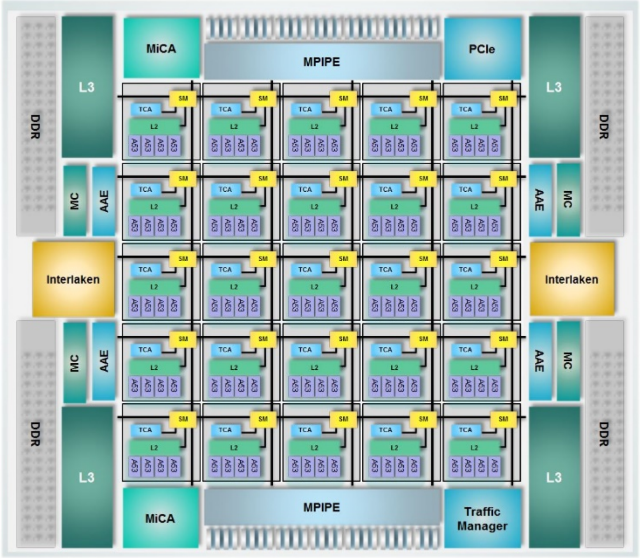
香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Energy Efficient Multi-core System

- Increasing number of processor cores
  - Enabled by **technology scaling**
  - Motivated by the failure of **Dennard scaling**
  - Happen in both high-performance processors (CMP) and embedded systems (MPSoC)
    - Tilera TILE-Pro64 MPSoC (64-core)
    - Intel Xeon Phi 7210 (72-core)
    - Intel Polaris chip (80-core)
    - EZchip/ Tilera MX-100 (100-core)
- Energy efficiency problem of multi-core system
  - Ever-increasing number of cores
  - Complexity of emerging application workloads



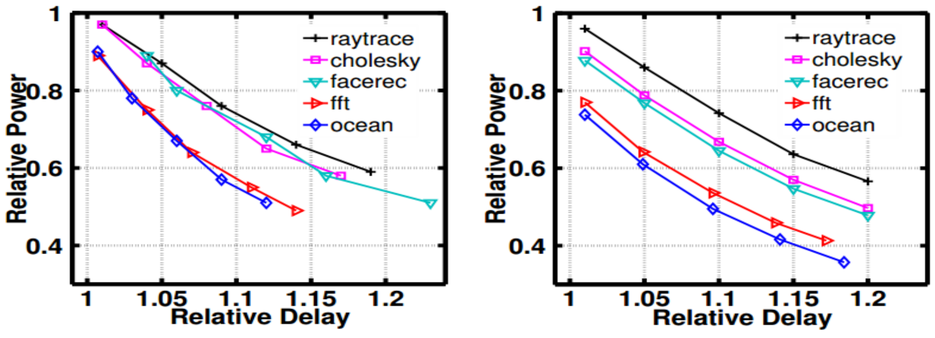
Intel's Polaris chip: 8x10 mesh



Ezchip MX-100: 5x5 mesh (4 core/Tile)

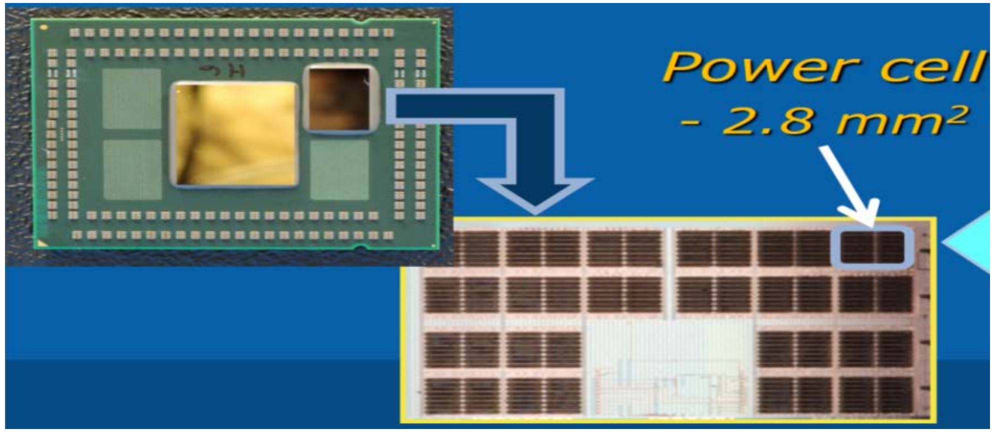
Low Power Techniques

- Power dissipation sources
  - Static power
  - Dynamic power: $P_{dyn} = \alpha C V^2 f$
- Low power techniques
  - Power gating
  - Dynamic voltage and frequency scaling (DVFS)
- DVFS control and schedule
  - Adaptively tune operating points (V/ F level) for each core based on runtime workload conditions
  - DVFS schedule is NP-hard problem
  - Solutions:
    - Reactive ways: e.g. Linux OS on-demand
    - Proactive ways: e.g. Heuristics and Learning based methods



(a) Chip-Wide DVFS (b) Per-Core DVFS

Performance vs. Power of two DVFS granularity [Kim, HPCA'08]



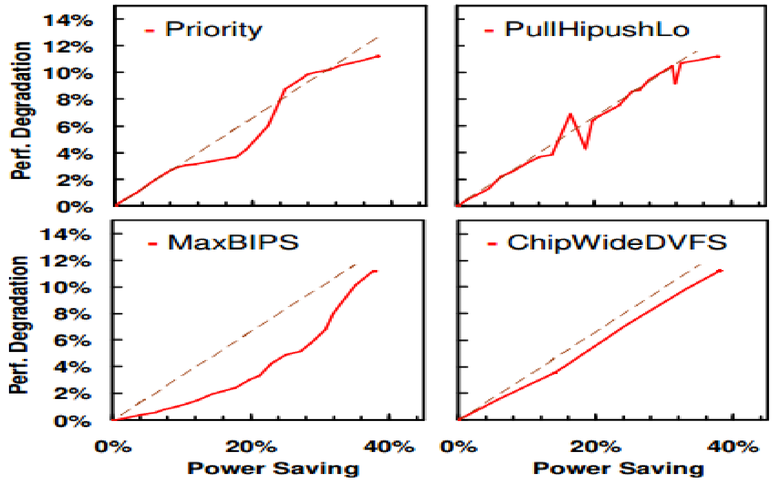
Intel Haswell processor with on-chip regulators [DiBene II, APEC'10]

Outline

- Related work and motivation
- Reinforcement learning (RL) based power management
- Modular RL for multicore energy efficiency optimization
- Experimental results
- Conclusion and future work

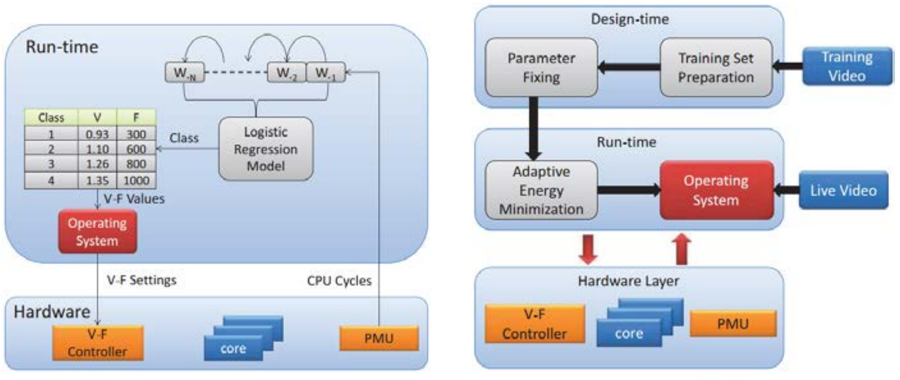
Previous Works on Power Management

- Ad-hoc and heuristic methods
  - Workload phase-based VF control in awareness of DVFS latency. [1]
  - Developed and analyzed some DVFS algorithms based on per-core or chip-wide DVFS. [2][3]



Power saving vs. perf degradation for the three policies [2]

- Supervised learning-based methods
  - Expert-based online allocation method based on online decision-tree algorithm. [4]
  - Supervised learning based on Bayesian classifier to predict system state and select actions. [7]
  - Multinomial logistic regression algorithm to predict the best VF-level based on workload features under workload uncertainties. [10]

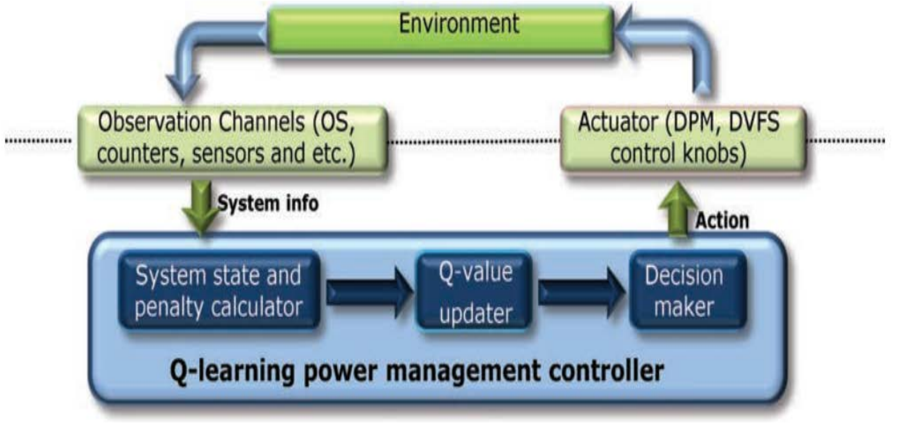


(a) Adaptive energy minimization (b) Design methodology

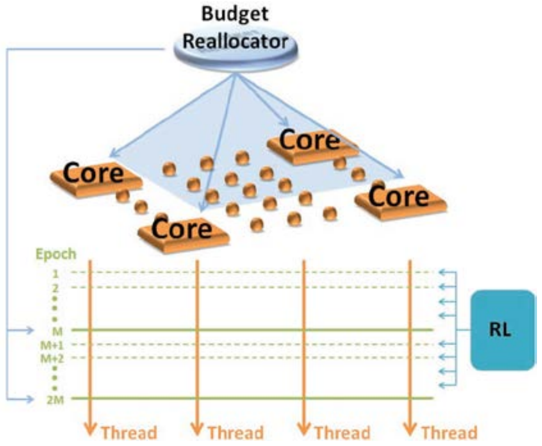
DVFS control strategy proposed by [10]

Previous Works on Power Management

- Reinforcement learning-based power management for single-core systems
  - Model-free Q-learning algorithm for dynamic power management. [6]
  - Add a second learning layer to get the parameters more accurately for traditional QL. [9]
  - Temporal Difference RL and a Bayesian classifier to improve state-prediction. [12]
- Reinforcement learning-based power management for multi-core systems
  - Challenge:
    - Complexity of the environment increases exponentially with number of cores.
  - Solutions:
    - Learning transfer among cores by sharing Q-table. [11]
    - Use a neural network to approximate the Q-table. [14]
    - Each core run Q-learning independently.[15][16]



QL power management diagram [9]



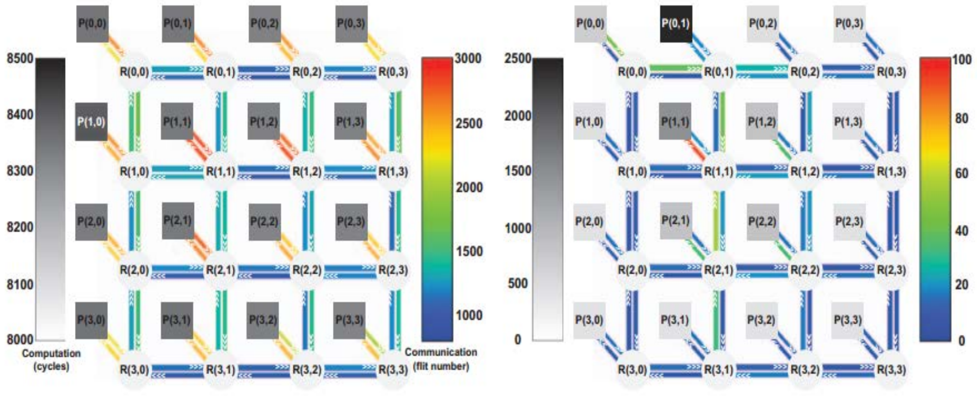
Local QL and global controller [15]

Motivation Example

- Impacts of inter-core relationship and dependency
  - Complicated execution causality for emerging multi-task/ thread applications.
  - Locally learned policy might not benefit the global system energy-efficiency.

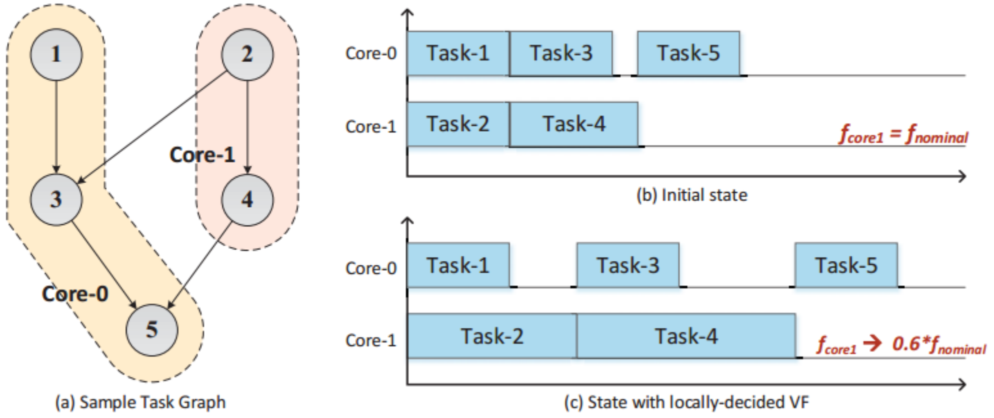
Modular reinforcement learning (MRL) based DVFS control strategy

  - Consider the inter-core relationship
  - Incurring polynomial amount of overhead



(a) FFT-1024_complex (b) RS-32_28_8_dec

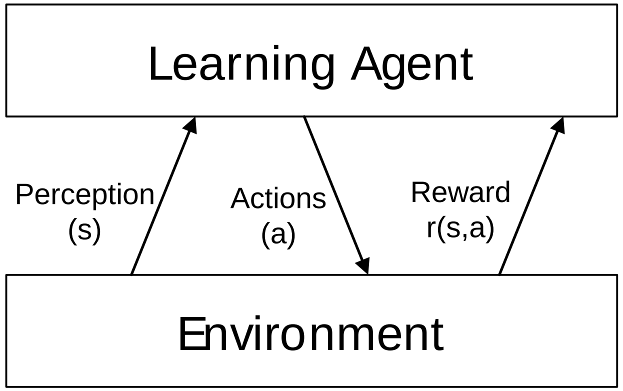
Inter-core communication of two workloads [21]



Sample motivation example showing the impact of task dependency

Reinforcement Learning Basics

- Reinforcement learning
  - Learns appropriate behavior by trail-and-error method while interacting with the dynamic environment.
  - Key elements
    - **Agent**: action-space.
    - **Environment**: state-space.
    - **Reward** function for action-state pairs.
  - Reward feedback for agent to learn the effect of its behavior.
  - Finds an appropriate **policy** to achieve a certain goal.



Reinforcement learning flow

Q-Learning Algorithm

- Q-learning (QL) background
  - One of the most popular algorithm in RL.
  - Solve a RL problem without having to know the state-transition model of the environment
- QL basics
  - Q-value for each state-action pair stored in a table
  - Q-value updating rule:

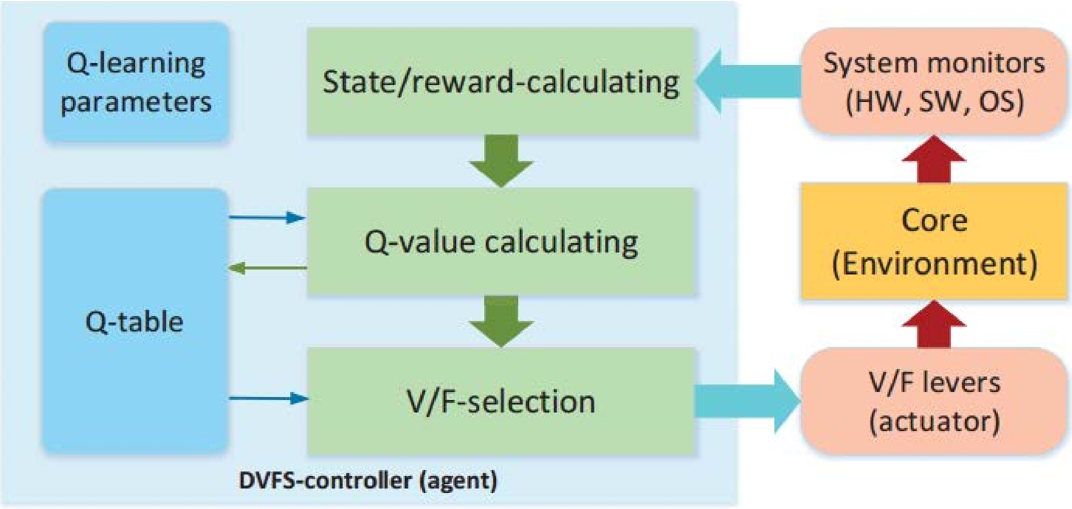
$$\boldsymbol{Q_{t+1}(s,a) \leftarrow Q_t(s,a) + \alpha_t(s,a) \cdot \left[\left(R_t(s,a) + \gamma \cdot \max_a Q(s',a)\right) - Q_t(s,a)\right]}$$

- Exploration vs. exploitation
  - Tradeoff between convergence speed and system performance

Notations	Description
s / s'	Last epoch state / current epoch state
a	Last epoch action
$Q_t(s,a)$	Q-value for state s and action a at last epoch t
$R_t(s,a)$	Reward for state s and action a for last epoch t
$\alpha_t(s,a)$	Learning rate for last epoch t
γ	Discount factor

DVFS Control Problem Formulation

- QL-based DVFS control
  - Control knob: Per-core DVFS
  - Agent: system power controller
  - Environment: processor core system
- System formulation
  - State-space: 2D tuple $s_t = (h_t, \mu_t)$
  - Action-space: available V/ F levels
  - Reward function: $r_t = \frac{h_t}{energy_t}$
  - Exploration vs. Exploitation:
    - ϵ-greedy
  - α and ϵ will decay with the visiting times of the state



Overview of QL based DVFS control

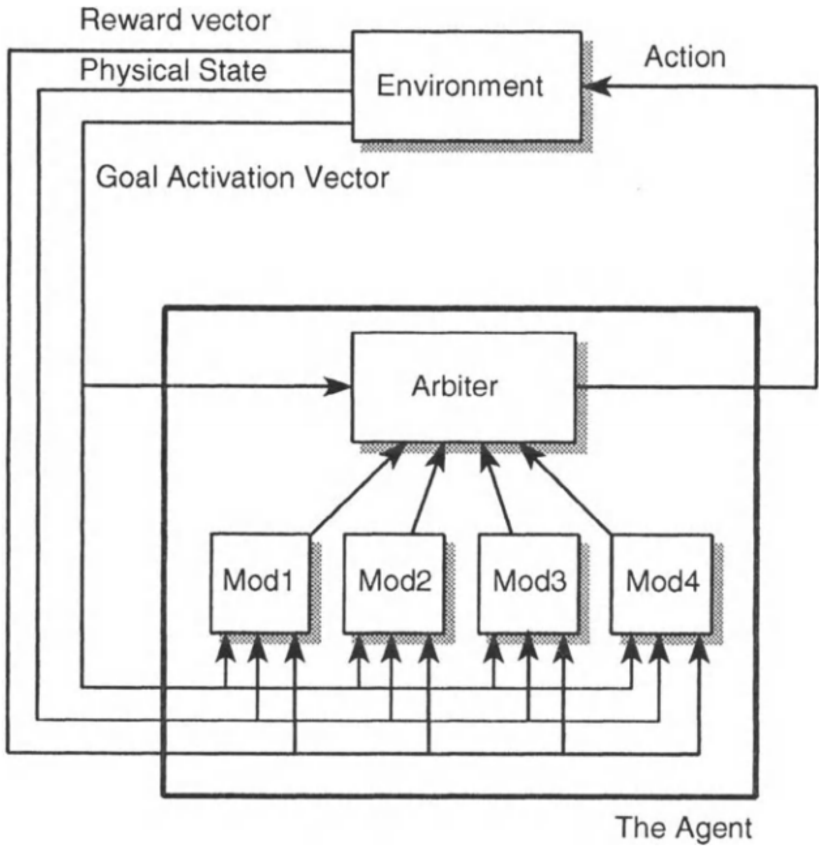
- At each epoch, generate a random number a ranged in (0,1);
- If $a < \varepsilon$, choose an action randomly (exploration);
- Otherwise, choose the best action (exploitation).

$$h_t = \frac{num\_busy\_cycles_t}{time\_elapsed_t}$$

$$\mu_t = 1 - \frac{num\_cycles\_L1\$\_stalled_t}{num\_busy\_cycles_t}$$

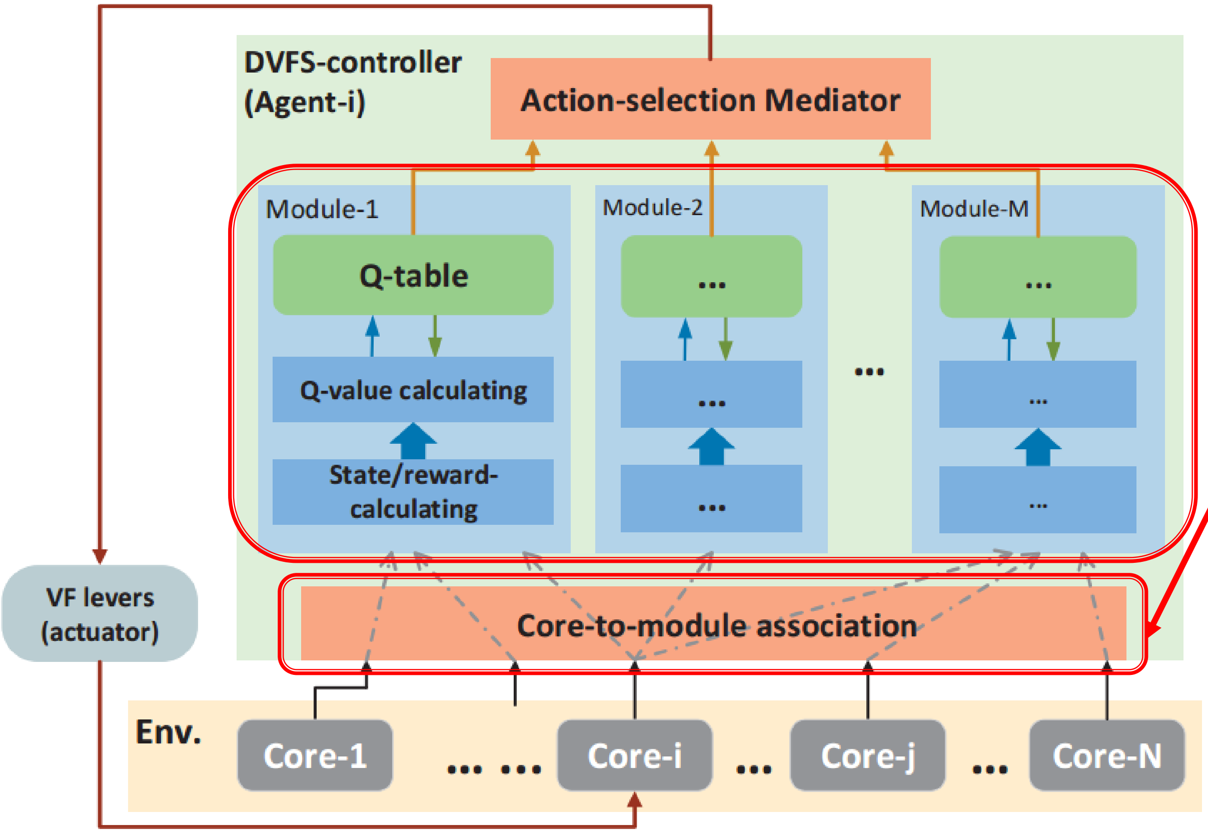
Modular Reinforcement Learning

- Difficulty of applying QL for multicore system
  - Monolithic QL -> **state-space explosion** with exponentially increased Q-table size $O(|S|^N \cdot |A|^N)$
  - Independent/ local QL -> ignoring inter-core relationship results in deteriorated the quality of the policy learned
- MRL [17] background
  - Target multi-aim or multi-agent optimization problem
  - Polynomial memory overhead depending on modularity
  - A balance between memory overhead and learning quality
  - Intuitively fitting the power management problem in multicore system



Modular architecture overview [17]

MRL for Multicore System Power Management

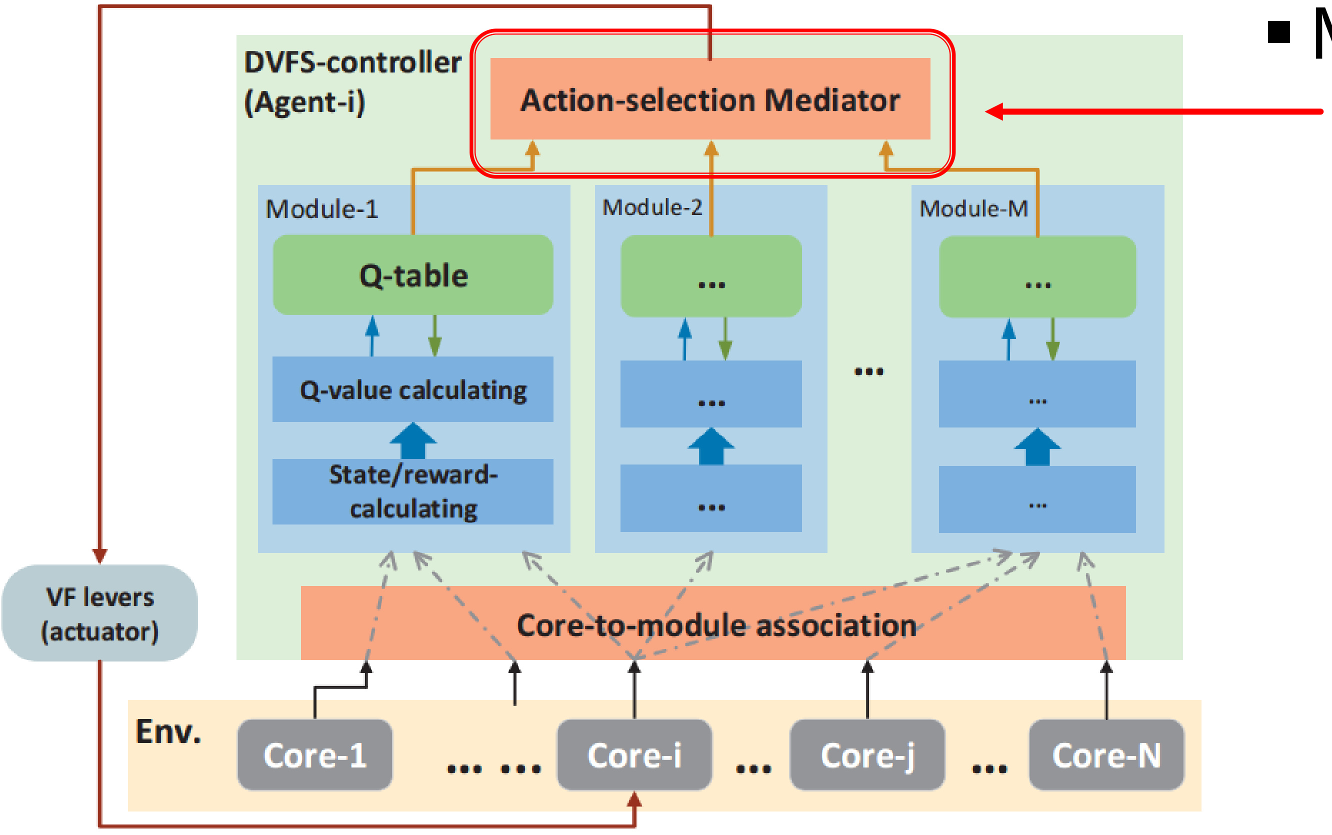


Overview of MQL based adaptive DVFS control mechanism

- Modularity (modular structuring)
  - The number of modules for each agent : 2;
  - State space of each module
    - Module-1: state of itself;
    - Module-2: state of the most relavate core;

Structure Name	Description	Total Table Size
Monolithic	Joint state of all cores	$O(\|S\|^N \cdot \|A\|^N)$
Individual	States of Local-core and every other core	$O(N \cdot \|S\| \cdot \|A\|)$
This work	Local state and one most relevant core state	$O(N \cdot \|S\| \cdot \|A\|)$

MRL for Multicore System Power Management



Overview of MQL based adaptive DVFS control mechanism

- Mediation strategy
  - Coordinates the learned policy of different modules;
  - Requirement: simple for online control
  - Two widely used algorithm:
    - Top-Value -> winner-take-all

$$a_t^i = \underset{a^i \in A^i}{\operatorname{argmax}} \{ \max_j Q^{ij}(s_t^{ij}, a^i) \}$$

    - Greatest-Mass (GM) -> majority-voting

$$a_t^i = \underset{a^i \in A^i}{\operatorname{argmax}} \sum_{j=1}^{M} Q^{ij}(s_t^{ij}, a^i)$$

MRL for Multicore System Power Management

Algorithm 1 Proposed Modular Q-Learning Based DVFS control algorithm.

Require: Q-table $QT^{ij}(s,a)$,
Q-Learning parameters α, γ, ϵ,
Input: μ_t, h_t, e_t

```
// main learning loop.
when reaching a learning-epoch t do
    for all agent-i in parallel do
        for all module-j in agent-i do
            s_t = calculate_state_id(μ_t^i, h_t^i)
            r_t = calculate_reward(h_t^i, e_t^i)
            QT^ij(s_last, a_last) =
              update_q_value(QT^ij(s_last, a_last), s_t, r_t, α, γ)
        end for
        rand_action = ε-greedy(ε)
        if rand_action is true then
            a_t^i = random_action(A)
        else
            a_t^i = great_mass(QT^ij(s, a))
        end if
        issue_action(a_t^i)
    end for
end when
```

a) Information from environment

$$\mu_t = 1 - \frac{num\_cycles\_L1\$\_stalled}{num\_busy\_cycles}$$

$$h_t = \frac{num\_busy\_cycles}{time\_elapsed}$$

e_t : energy consumption

$$r_t = \frac{h_t}{energy_t}$$

b) Calculate states and rewards for each module in each agent.

c) Update Q-value:

$$Q_{t+1}(s,a) \leftarrow Q_t(s,a) + \alpha_t(s,a) \cdot \left[\left(R_t(s,a) + \gamma \cdot \max_a Q(s',a)\right) - Q_t(s,a)\right]$$

d) Exploration vs. exploitation (ε-greedy algorithm):

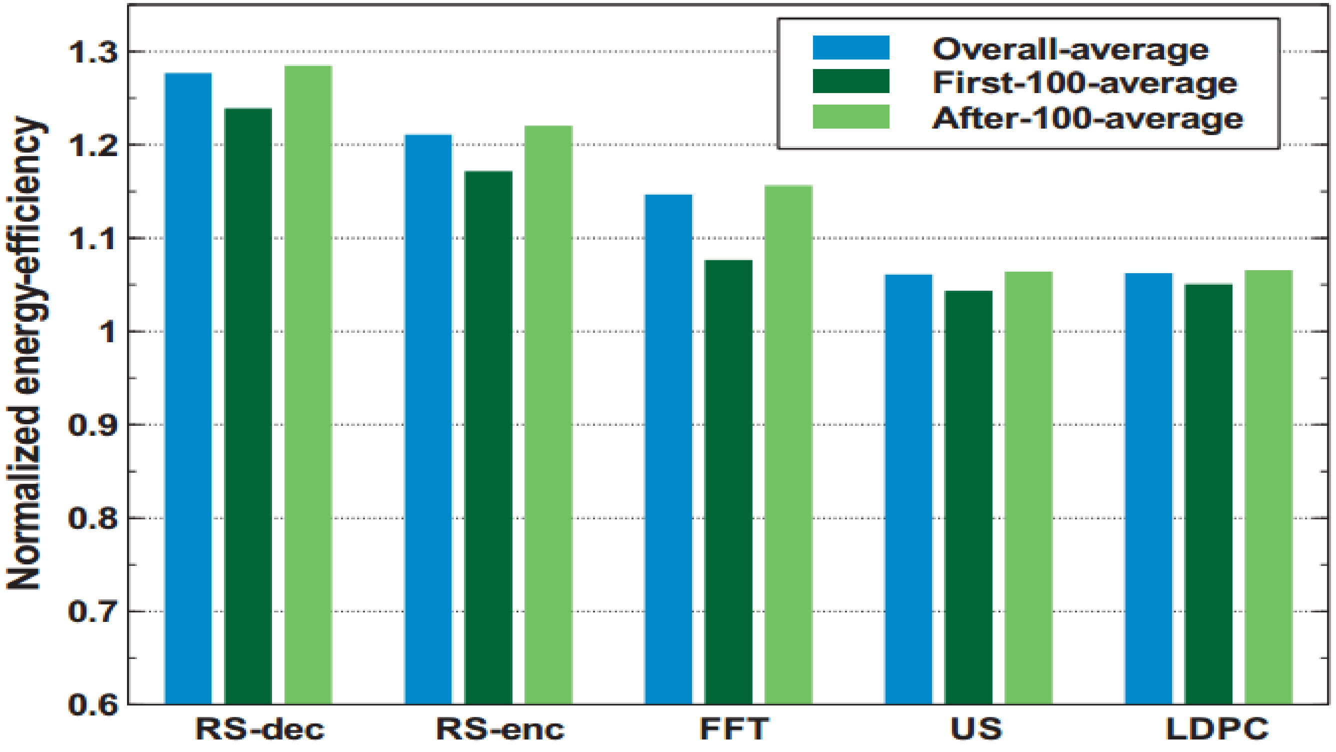
- Exploration: choose an action randomly;
- Exploitation: mediation among multiple modules.

Experiment Results and Analysis

- Setups
  - Homogeneous MPSoC with mesh-based NoC, JADE simulator [19]
  - Five real applications from COSMIC benchmark suit[21]
    - RS-dec, RS-enc, FFT, US, LDPC
  - Power model based on McPAT [20]
  - System assumptions
    - Five operating VF levels:
      0.55V/ 1.4GHz, 0.5V/ 1.2GHz, 0.45V/ 1GHz, 0.4V/ 800MHz and 0.35V/ 600MHz
- Evaluation:
  - Metrics: energy efficiency: $r_t = \frac{h_t}{energy_t}$ $(\propto \frac{1}{\mathrm{EDP}})$;
  - Compared with individual learning method.

Energy Efficiency Improvement

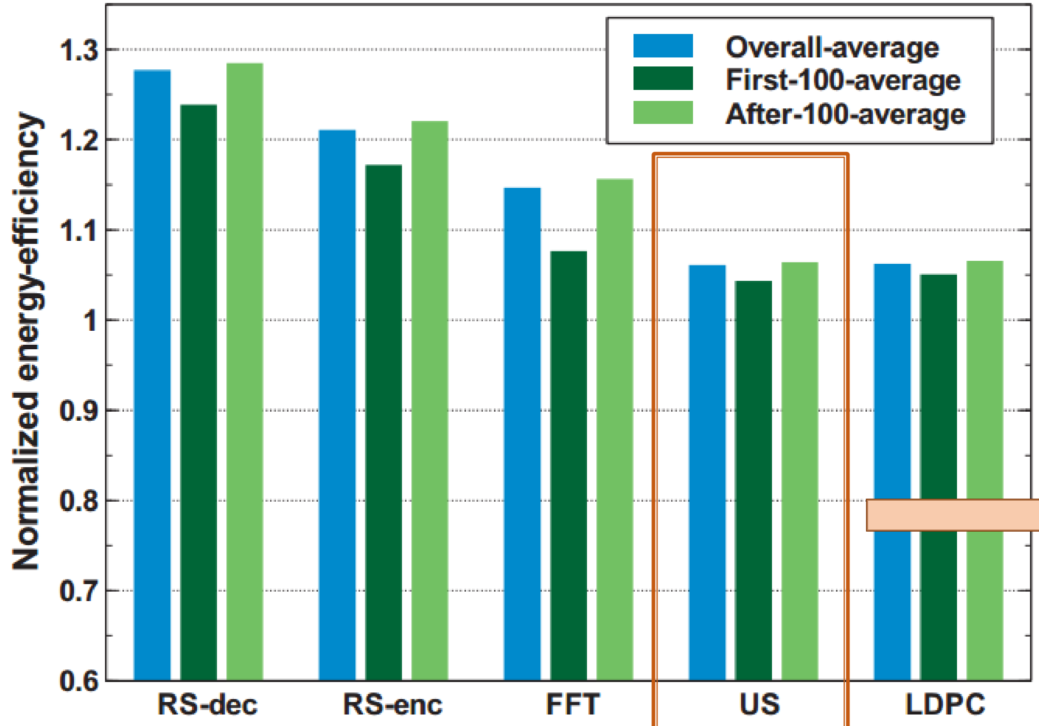
- Energy-efficiency improvement for different applications
  - On-average **14%** energy efficiency improvement



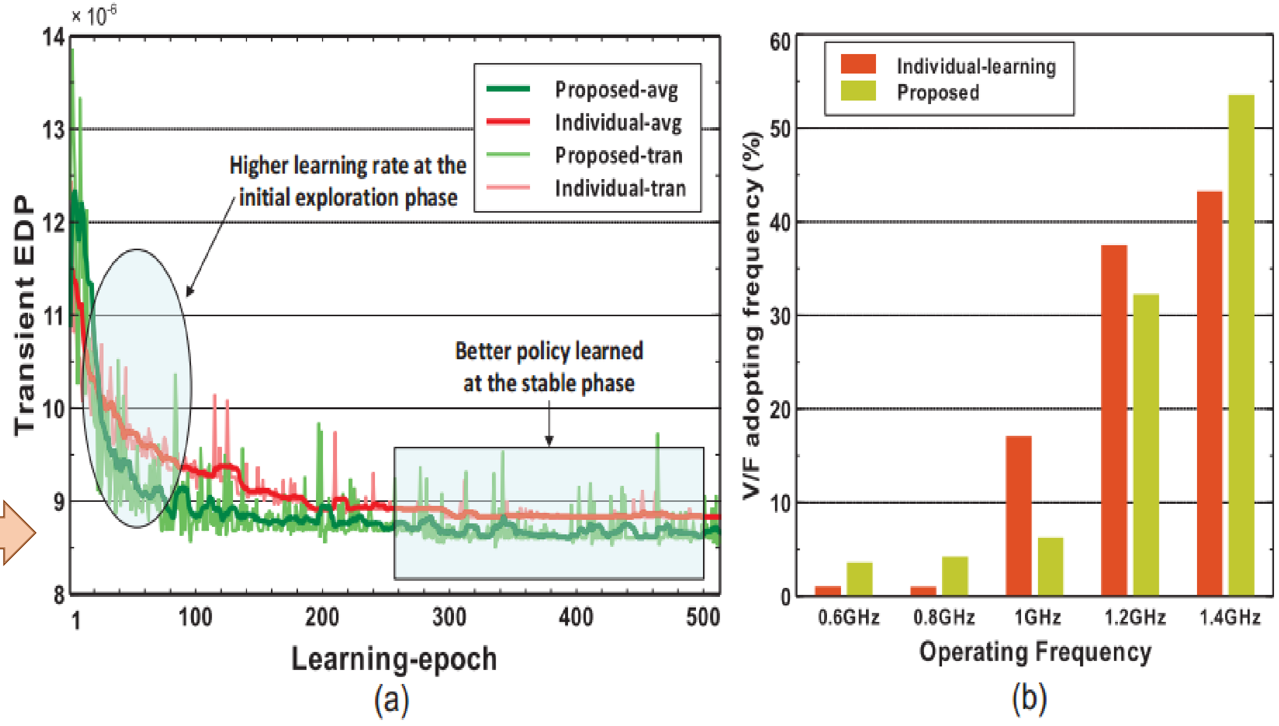
Energy-efficiency improvement in 32-core system

Energy Efficiency Improvement

- Energy-efficiency improvement for different applications
  - Faster convergence and better final policy learned



Energy-efficiency improvement in 32-core system

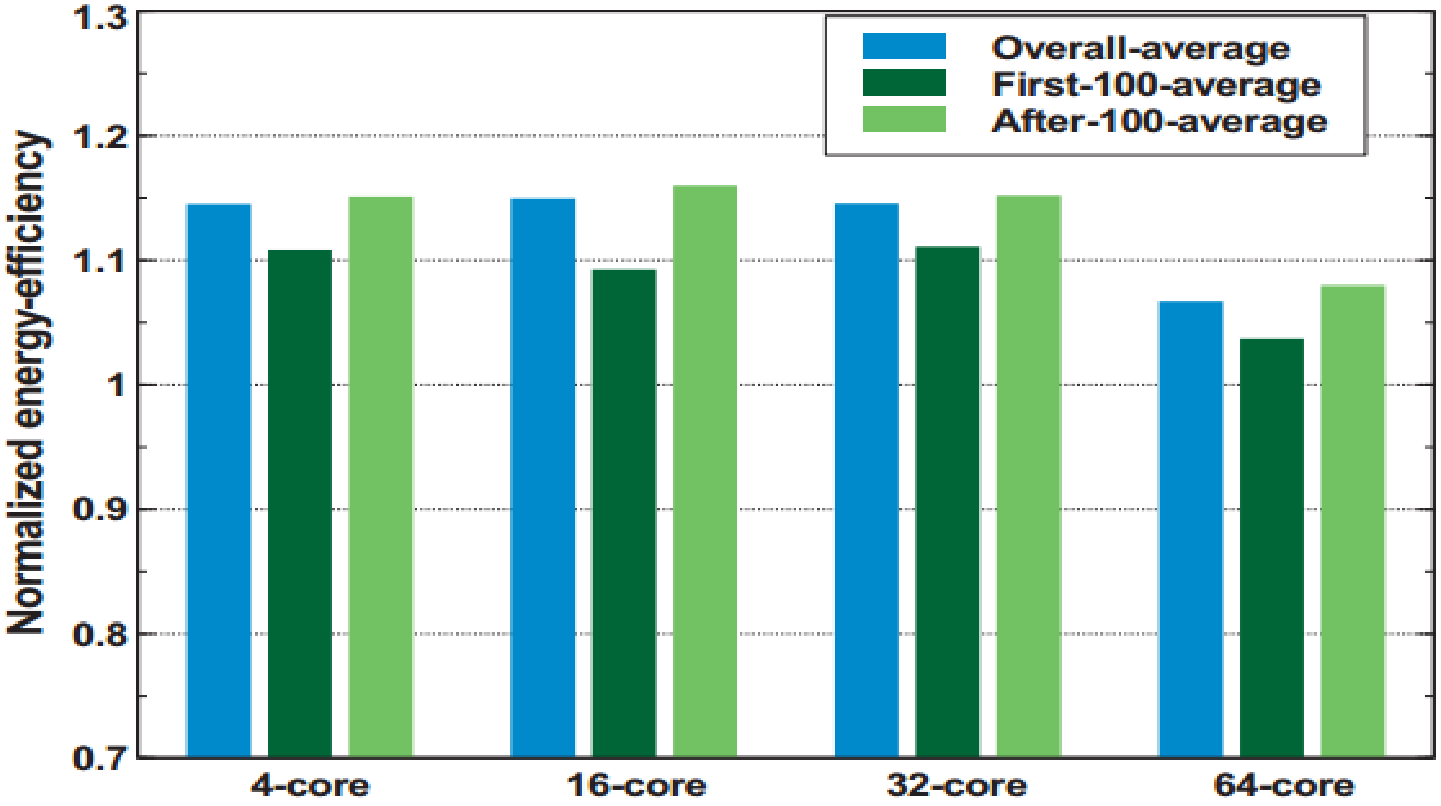


EDP transient analysis for US application

V/F adopting percentage

Scalability Evaluation

- Energy-efficiency improvement for systems of different scales
  - On-average **12.6%** better energy-efficiency over all four scales



Energy-efficiency improvement for different system scales

Conclusion and Future Work

- Conclusion
  - Propose a Modular Reinforcement Learning based framework for DVFS control in multicore system to improve system energy-efficiency.
  - Achieve globally optimized DVFS control policy with incurring reasonable amount of overhead.
  - Experimental results shows the effectiveness and advantage of the proposed method over the individual local RL scheme.
- Future work
  - Exploration of different modular structures and mediation strategies.
  - Adaptively constructing modular structures based on application knowledge and OS scheduling information.

Reference

[1] Z. Lai, K. T. Lam, C. L. Wang, J. Su, Y. Yan, and W. Zhu, "Latency-aware dynamic voltage and frequency scaling on many-core architectures for data-intensive applications," in *Cloud Computing and Big Data (CloudCom-Asia), 2013 International Conference on*, 2013, pp. 78–83.
[2] C. Isci, A. Buyuktosunoglu *et al.*, "An analysis of efficient multicore global power management policies: Maximizing performance for a given power budget," in *2006 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'06)*, 2006, pp. 347–358.
[3] S. Herbert and D. Marculescu, "Analysis of dynamic voltage/frequency scaling in chip-multiprocessors," in *Low Power Electronics and Design, 2007 ACM/IEEE International Symposium on*.
[4] G. Dhiman and T. S. Rosing, "Dynamic power management using machine learning," in *Proceedings of the 2006 IEEE/ACM International Conference on Computer-aided Design*, ser. ICCAD '06.
[5] G. Theocharous, S. Mannor, N. Shah, P. Gandhi, B. Kveton, S. Siddiqi, and C.-H. Yu, "Machine learning for adaptive power management," *Intel Technology Journal*, vol. 10, no. 4, Nov. 2006.
[6] Y. Tan, W. Liu, and Q. Qiu, "Adaptive power management using reinforcement learning," in *2009 IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers*.
[7] H. Jung and M. Pedram, "Supervised learning based power management for multicore processors," *Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol. 29, no. 9, Sep. 2010.
[8] W. Liu, Y. Tan, and Q. Qiu, "Enhanced q-learning algorithm for dynamic power management with performance constraint," in *2010 Design, Automation Test in Europe Conference Exhibition (DATE)*.
[9] H. Shen, Y. Tan, J. Lu, Q. Wu, and Q. Qiu, "Achieving autonomous power management using reinforcement learning," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 18, no. 2, Apr. 2013.
[10] A. Das, A. Kumar *et al.*, "Workload uncertainty characterization and adaptive frequency scaling for energy minimization of embedded systems," in *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*, ser. DATE '15, 2015.
[11] R. A. Shafik, S. Yang, A. Das, L. A. Maeda-Nunez *et al.*, "Learning transfer-based adaptive energy minimization in embedded systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 6, pp. 877–890, 2016.
[12] Y. Wang and M. Pedram, "Model-free reinforcement learning and bayesian classification in system-level power management," *IEEE Transactions on Computers*, vol. PP, no. 99, 2016.
[13] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, 1st ed. Cambridge, MA, USA: MIT Press, 1998.
[14] R. Ye and Q. Xu, "Learning-based power management for multicore processors via idle period manipulation," *IEEE Transactions on ComputerAided Design of Integrated Circuits and Systems*, vol. 33, no. 7, pp. 1043–1055, 2014.
[15] Z. Chen and D. Marculescu, "Distributed reinforcement learning for power limited many-core system performance optimization," in *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*, ser. DATE '15, 2015.
[16] M. Otoom, P. Trancoso, H. Almasaeid, and M. Alzubaidi, "Scalable and dynamic global power management for multicore chips," in *Proceedings of the 6th Workshop on Parallel Programming and Run-Time Management Techniques for Many-core Architectures*.
[17] S. Whitehead, J. Karlsson, and J. Tenenberg, *Learning Multiple Goal Behavior via Task Decomposition and Dynamic Policy Merging*. Boston, MA: Springer US, 1993, pp. 45–78.
[18] N. Ono and K. Fukumoto, *A modular approach to multi-agent reinforcement learning*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 25–39.
[19] R. K. V. Maeda, P. Yang, X. Wu *et al.*, "Jade: A heterogeneous multiprocessor system simulation platform using recorded and statistical application models," in *Proceedings of the 1st International Workshop on Advanced Interconnect Solutions and Technologies for Emerging Computing Systems*, ser. AISTECS '16, 2016.
[20] S. Li, J. H. Ahn *et al.*, "Mcpat: An integrated power, area, and timing modeling framework for multicore and manycore architectures," in *Microarchitecture, 2009. MICRO-42. 42nd Annual IEEE/ACM International Symposium on*, Dec 2009, pp. 469–480.
[21] Z. Wang, W. Liu, J. Xu *et al.*, "A Case Study on the Communication and Computation Behaviors of Real Applications in NoC-based MPSoCs," in *IEEE Computer Society Annual Symp. VLSI*, 2014.

Thank You!