Application - Architecture Agnostic Efficiency Optimization using Reinforcement Learning

Akhilesh Raj

Swann Perarnau

Aniruddha Gokhale

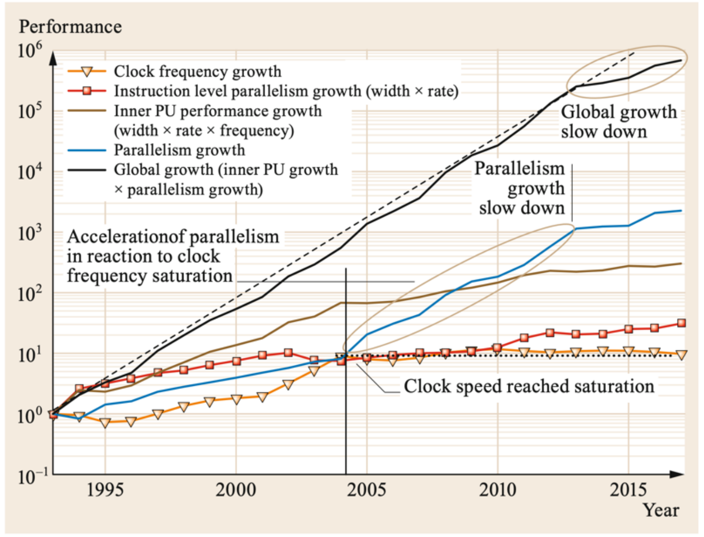
Introduction

- **Exascale Computing:** A milestone for processing power, capable of at least one exaFLOP, or a billion billion calculations per second.
- **Necessity:** Requires advanced High-Performance Compute (HPC) nodes to handle complex computations.
- **Energy Challenge:** HPC systems are notorious for high energy demands.
- **DoE Goal:** Earlier goal was to keep HPC units below 20 megawatts.
    - The goal of Frontier machines at ORNL is to achieve an efficiency above 62.684 GFlops/watts.
- **Areas of Focus:**
    - **Semi-conductor Design:** Crafting energy-efficient chips.
    - **Application Design:** Creating software that utilizes resources prudently.
    - **Workload Scheduling:** Optimal task assignments to minimize energy use.
    - **Power Control:** Dynamic adjustment of energy based on computing needs.
- **Our Contribution: Run-time Energy Management:**
    - **Approach:** Control real-time energy consumption of applications by adjusting power inputs.
    - **Impact:** Long-duration scientific applications necessitate careful energy regulation for days on end.
- **Towards Sustainable Exascale Computing:**
    - **Packaging Experiments:** Providing a toolkit for researchers and practitioners.
    - **Advocating for Green HPC:** Encouraging adoption of practices that reduce the environmental footprint of computing at scale.

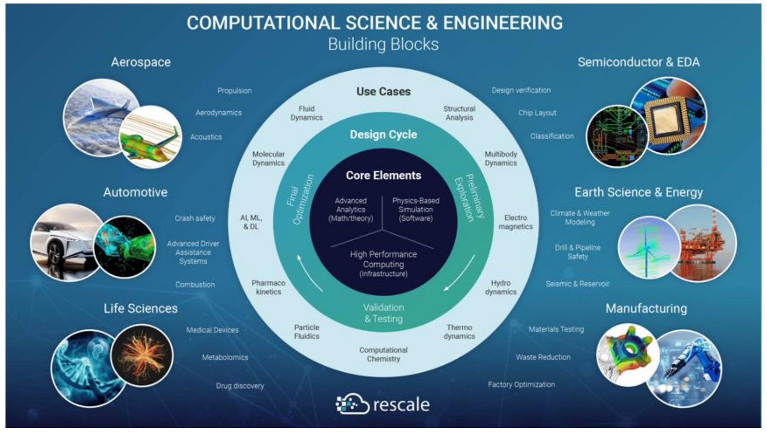
Image from Tech Asia Lab

Application-Architecture Agnostic Approach

- **Understanding Performance Variables:**
  - **Application Characteristics:**
    - Type: Compute-bound vs memory-bound vs I/O bound etc. can influence how an application performs.
  - **System Architecture:**
    - Memory Bandwidth & Processors: Variations in hardware can impact execution speed.
    - Spatial Complexity: Different architectures handle computational tasks differently.
  - **Input Power:**
    - Directly affects the performance and energy consumption.
- **Limitations of Conventional Methods**
  - **Initial Run:** Often used to profile an application's runtime behavior.
  - **Static Power Control:** Adjustments made based on the initial profile, potentially missing dynamic changes.
- **Efficiency Gaps**
  - **Phase Changes:** Applications may evolve over time, rendering initial profiles less accurate.
  - **Performance Fluctuations:** A one-size-fits-all power control cannot adapt to varying run-time requirements.
- **The Case for Closed-Loop Control**
  - **Adaptive Strategy:** Monitors and adjusts to application's needs in real time.
  - **Dynamic Response:** Ensures efficient use of power throughout the varying phases of application execution.



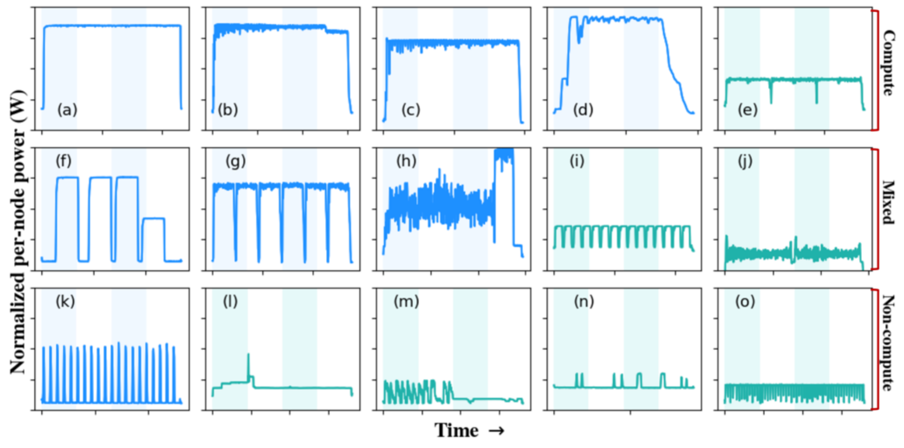
Evolution of Top 20 HPC systems with architectures improving features such as clock speed, parallelism etc. (Image from https://top500.org/)



HPC use cases depicting diverse applications and their classification based on the HPC usage (Image from rescale.com.)

State of the Art

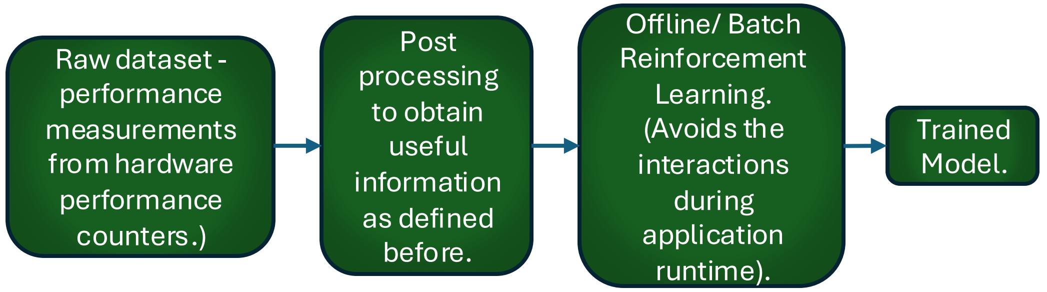
- **Initial Practices: Power Profiling & DVFS**
  - **Power Profiling:** A preliminary run to understand power dynamics.
  - **DVFS (Dynamic Voltage and Frequency Scaling):**
    - Used to regulate power after profiling.
    - Acts as a secondary control mechanism on uncapped nodes.
    - May not handle outlier situations effectively.
    - With a limited number of frequencies to actuate it may not be precise.
- **Limitations of Traditional Methods**
  - **Susceptibility:** Traditional DVFS can struggle with unpredictable, extreme scenarios.
  - **Model Dependence**: Closed-loop controllers traditionally need a specific Power vs. Performance model.
- **Challenges in Modeling**
  - **Variability:** Difficult to create models for every possible application-architecture pairing.
  - **Knowledge Requirement**: Need to establish an optimal performance set point beforehand, which may not always be feasible.
- **Our Approach: Node Power Capping**
  - **Intel Power Actuators**: Utilized to impose power caps directly at the node level.
  - **Proactive Adjustment:** Allows for real-time modifications in power usage, circumventing the limitations of traditional DVFS.



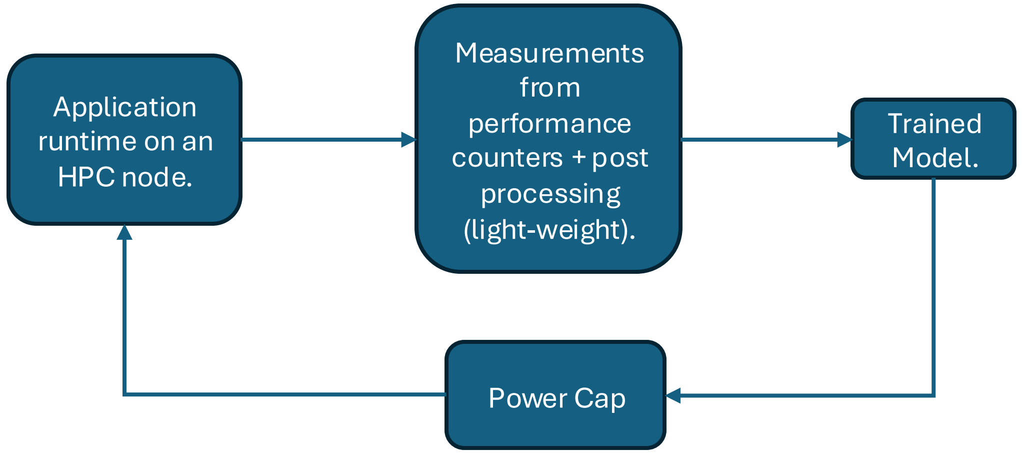
Timeseries of typical HPC workloads running different benchmark application with varying power consumption during run time. (Figure from ORNL repository.)

Our experiments

- **Harnessing Run-Time Data**
  - We collect datasets during application execution for in-depth analysis.
- **Model-Based Approach**
  - **Model Creation:** Develop a mathematical relationship between power and performance from real-time data.
  - **Reinforcement Learning (RL) Agent:**
    - Trains to recognize the generated model.
    - Adjusts power to maintain optimal performance.
  - **Adaptability:** The same RL agent is trained to suit various application and hardware setups.
- **Learning-Based Approach**
  - **Algorithm Utilization:** Implement batch/offline RL algorithms.
  - **Agent Training:**
    - Gains insights from the run-time data.
    - Constructs a comprehensive model that correlates application features, hardware specifications, and power caps (PCAP) to expected performance outcomes.


Raw dataset - performance measurements from hardware performance counters.)
Post processing to obtain useful information as defined before.
Offline/ Batch Reinforcement Learning. (Avoids the interactions during application runtime).
Trained Model.

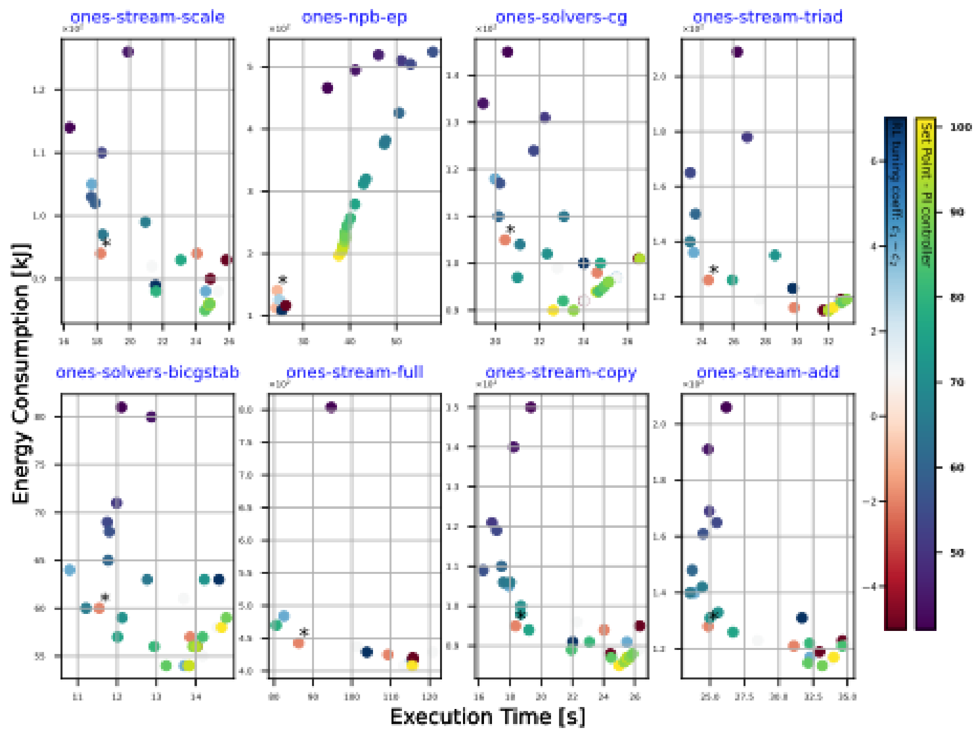
Training phase in learning-based approach


Application runtime on an HPC node.
Measurements from performance counters + post processing (light-weight).
Trained Model.
Power Cap

Testing phase in learning-based approach.

The Importance of Packaging Our Research

- **Maximizing HPC Runtime Efficiency**
  - **Objective:** Achieve peak efficiency in HPC operations—maximal performance per watt.
- **Intelligent PCAP Recommendation**
  - **Observation**: Monitor real-time application demands to suggest optimal Power Cap (PCAP).
  - **RAPL Regulation**: Employ Intel's Running Average Power Limit (RAPL) to manage node power efficiently.
- **Determining Runtime Requirements**
  - **Performance Counters:** Offer insights into the application's operational characteristics.
  - **Application Instrumentation:** Assesses progress towards scientific objectives.
  - **Light-Weight Measures:** Calculate iterations per second for iterative tasks in runtime.
- **Evolution of Our Approach**
  - **Initial Phase:** Create a model correlating application performance with architecture, setting the foundational PCAP.
  - **Later Stage**: Integrate performance counter data to refine the model, enhancing its relevance across diverse application and architectural scenarios.
- **The Need for a Turnkey Solution**
  - **Empower Researchers**: Simplifying the adaptation of our power regulation strategies.
  - **Facilitate Broader Benefit**: Enable seamless integration into a variety of HPC environments.
  - **Encourage Sustainable Practices**: Foster an ecosystem of energy-efficient computation in the scientific community.



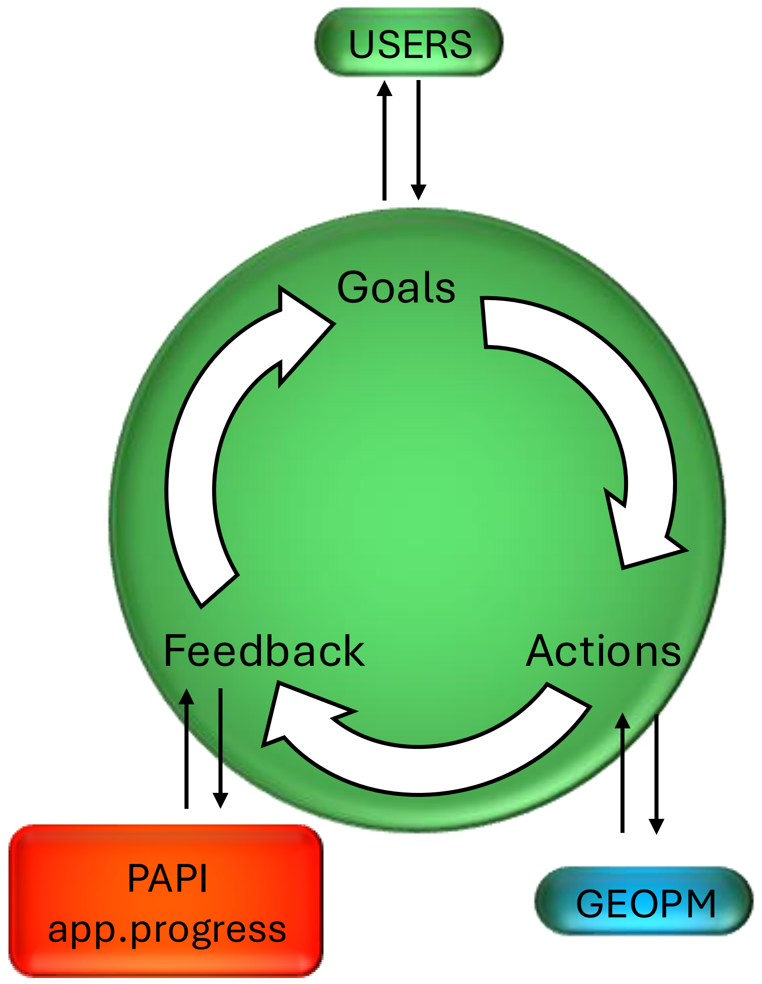
Regulation of power at the maximum efficiency point using RL based controller compared against different operating points for various applications. (Images from our research.)

Experiment requirements

- **Hardware and Control Specifications**
  - **Bare Metal Instance:** Necessary to uphold direct supervisory control over the power actuator.
  - **MSR Access:** Must be able to access Model Specific Registers to retrieve and set power configurations.
  - **Node-Level Deployment:**
    - Power metrics should solely reflect the consumption of the target applications.
- **Exclusion of Containerized Environments**
  - Due to the need for low-level hardware interaction and control, containers are unsuitable for these experiments.
- **Current Packaging Scope**
  - Tailored specifically for Intel processors equipped with RAPL capabilities.
  - **RAPL Overview**:
    - Manages node power to adhere to user-defined thresholds.
    - Provides fine-grained Power and Time controls.
- **Customization Approach**
  - **Power and Time Knobs:** Leveraging these for precise actuation.
  - **Time Knob:** Retained at a constant setting throughout experiments.
  - **Power Knob (PCAP):** Used to impose a ceiling on the allowable power consumption.
- **Software Stack Dependencies**
  - **Customized Software:** Essential for orchestrating the power regulation mechanisms effectively.
  - This custom-made software is a critical part of the experimental setup to ensure seamless interaction with the hardware layer.

Software Stack

- **NRM**: Node Resource Manager is a light-weight process running as a daemon service on the nodes capable of :
  - Job Management,
  - Power Management and
  - Resource Management.
- **PAPI** - Performance Application Programming Interface :
  - Offers a universal interface and methodology for gathering performance counter information.
  - Almost all architectures are compliant with PAPI releases.
- **GEOPM** - Global Extensible Open Power Manager:
  - It supplies all root level MSR data.
  - We in our research uses the MSR data on Energy and Power.
- **Instrumented iterative applications**:
  - Applications need to be made compliant with the NRM.
  - Need to instrument to get progress report apart from the PAPI measurements.
- Apart from the above key software dependencies, there are some provisioning requirements that are needed to be satisfied.



ARGO NRM software stack

How are we packaging the stacks currently.

- **Initially utilized Nix for setup:**
  - Resolved version conflicts.
  - Streamlined polyglot package distribution across nodes.
  - Drawbacks:
    - Managing packages was a burdensome process.
- **Transitioned to using Ansible scripts for:**
  - Reducing code diversity with updated versions.
  - Smoother post-provisioning, including special MSR access.
- **Recognized challenges with interactive installations.**
- **Packaging provides crucial benefits:**
  - Optimizes research phases through parallel data pipeline processing.
  - Facilitates efficient data aggregation post-parallelization.
  - Consolidates software and hardware dependencies into a singular offering.
  - Supports extension to the broader research community.
  - Elevates research efficiency and enhances the scope for collaboration.

Expectations with Chameleon Trovi:

- **Streamlined Deployment for Users**
  - **Provisioned Images**: Chameleon cloud users receive pre-configured images ready for deployment.
- **User-Centric Reprovisioning**
  - **Flexibility:** Users retain the ability to reprovision nodes to accommodate specific software and hardware stack requirements.
- **Cross-Architecture Compatibility**
  - **Installation Variability**: Across different architectures, the installation nuances can vary, but…
  - **Interactive Installation:** Aids researchers to tailor the packaging to their unique experimental needs.
- **Instrumentation and Reproducibility**
  - **Versatile Support:** Emphasis on providing a reproducibility interface that caters to diverse applications.
  - **Ease of Use:** Aims for a user-friendly experience that simplifies the instrumentation process.

Acknowledgement

Results presented here were obtained using the Chameleon testbed supported by the National Science Foundation. Argonne National Laboratory's work was supported by the U.S. Department of Energy, Office of Science, Advanced Scientific Computer Research, under Contract DE-AC02-06CH11357. This research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.

References

- Imes, C., Zhang, H., Zhao, K., & Hoffmann, H. (Year). Handing DVFS to Hardware: Using Power Capping to Control Software Performance. University of Chicago.
- A. Raj, S. Perarnau and A. Gokhale, "A Reinforcement Learning Approach for Performance-aware Reduction in Power Consumption of Data Center Compute Nodes," 2023 IEEE International Conference on Cloud Engineering (IC2E), Boston, MA, USA.
- TOP500.org. (2024). Frontier remains No. 1 in the TOP500 but Aurora with Intel's Sapphire Rapids chips enters with a half-scale system at No. 2. Retrieved from TOP500.org
- Tsafack Chetsa, G. L., Lefèvre, L., Pierson, J.-M., Stolf, P., & Da Costa, G. (2013). A User Friendly Phase Detection Methodology for HPC Systems' Analysis. IEEE International Conference on Green Computing and Communications (GreenCom 2013), Beijing, China, pp. 118-125. doi:10.1109/GreenCom-iThings-CPSCom.2013.43

Thank You