EnergAt: Fine-Grained Energy Attribution for Multi-Tenancy

Hongyu Hè, Michal Friedman, Theodoros Rekatsinas

HotCarbon '23

9 July 2023

ETH zürich

Software Energy

- End of Dennard Scaling

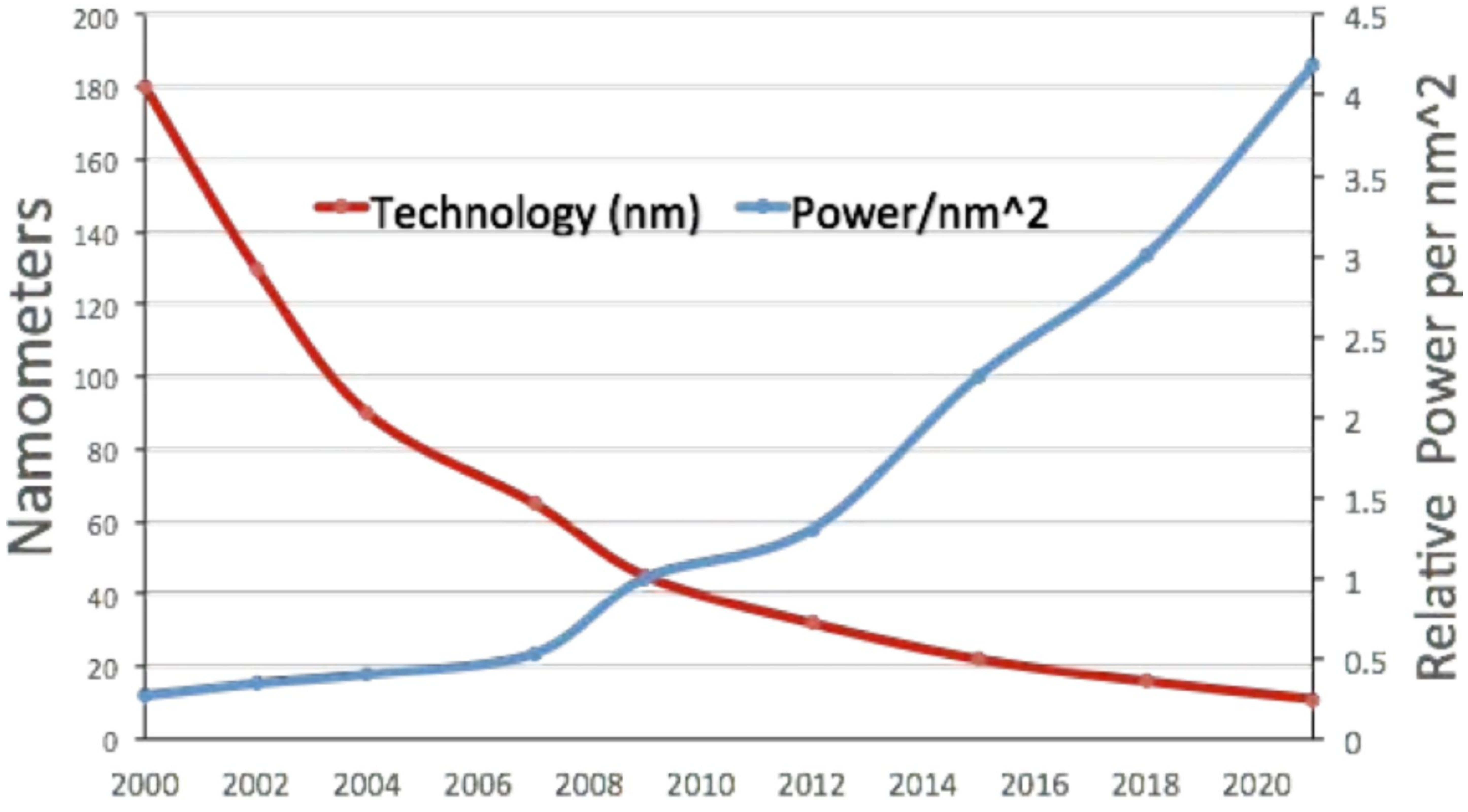
  ⇒ Performance gain comes at energy costs
- Machine learning systems

  ⇒ Huge energy consumers
- Software energy efficiency

  ⇒ More observability
- Can be a sensitive topic

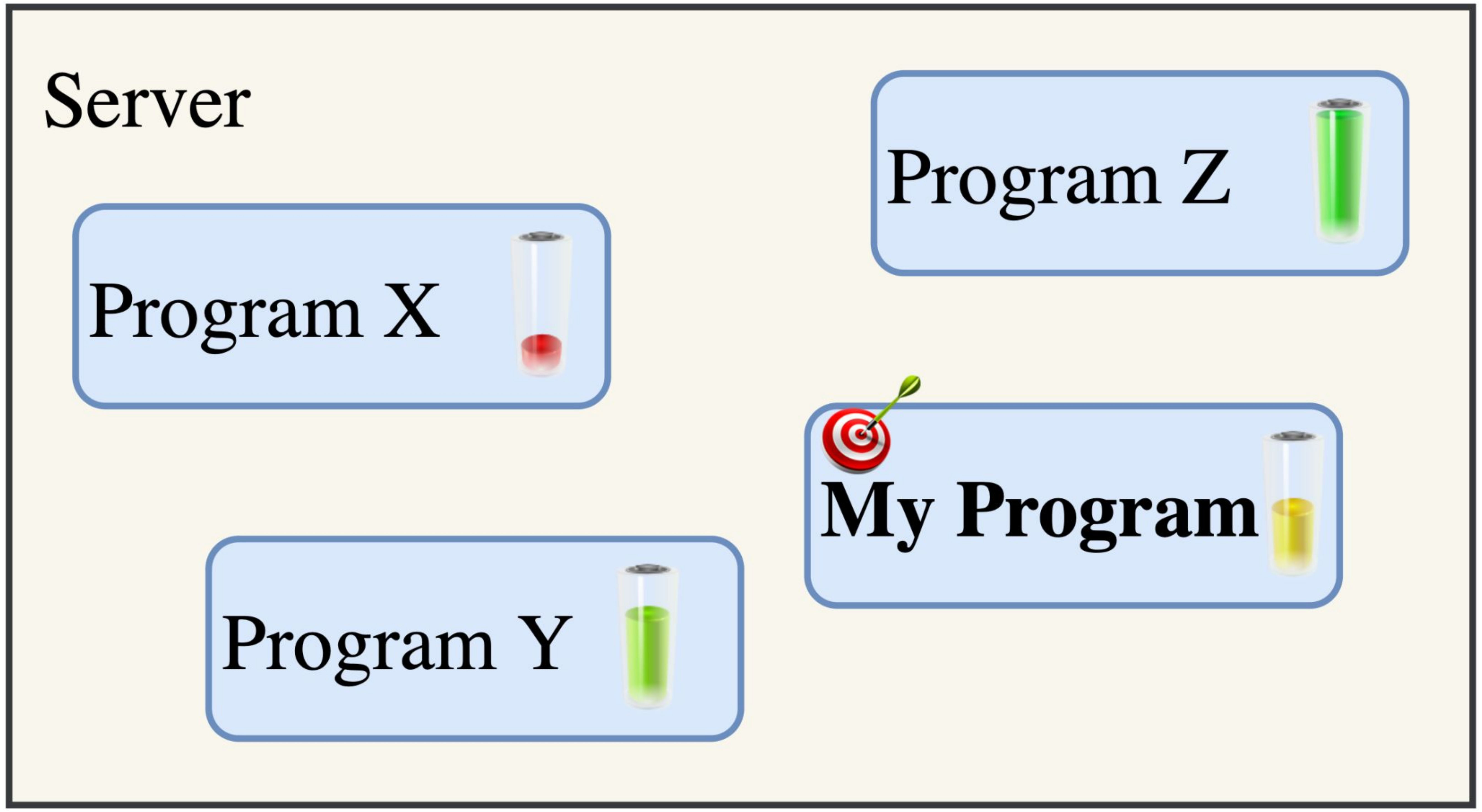
  ⇒ Precise energy provenance

[David Patterson '19]



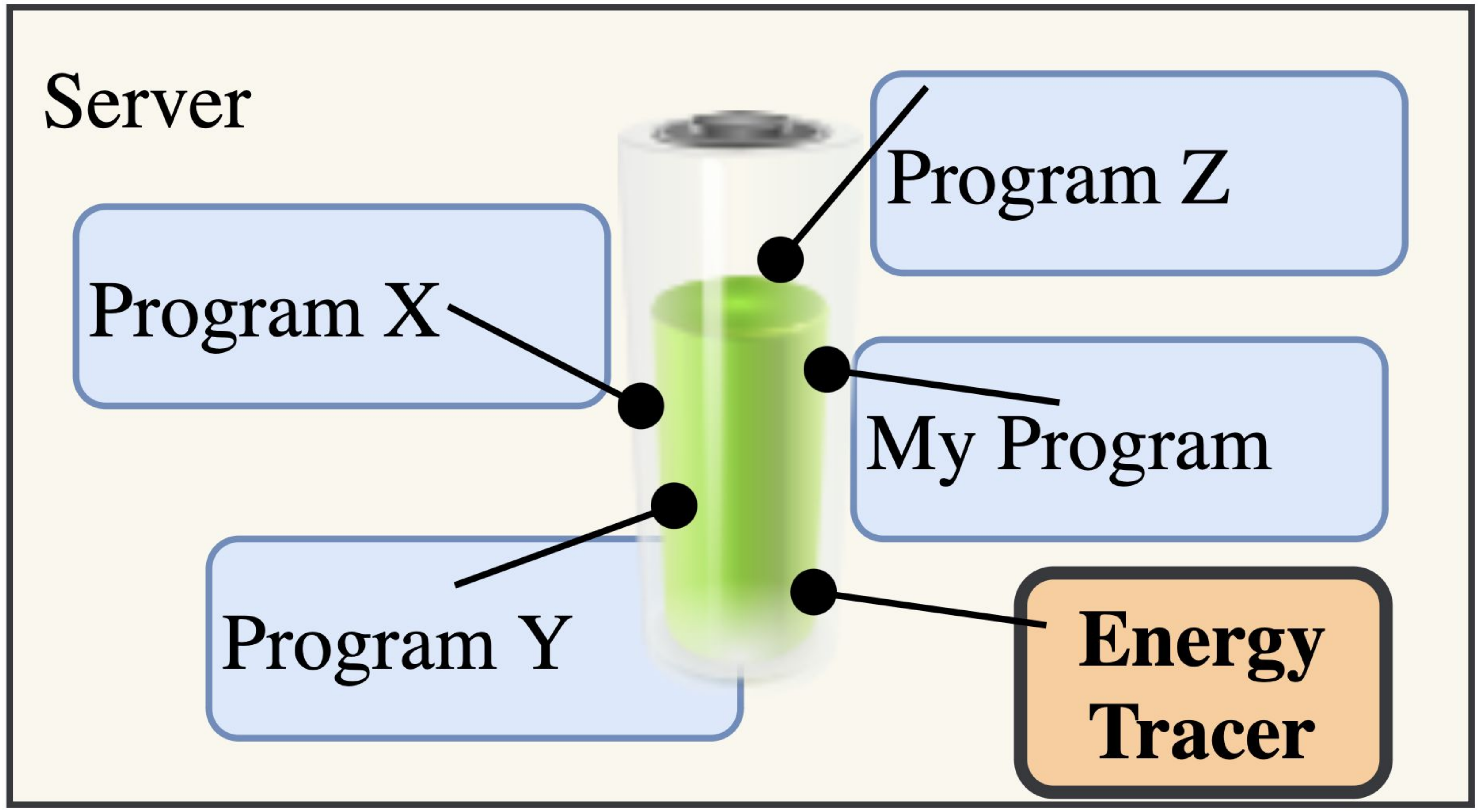
Goal of Fine-Grained Energy Attribution

- Determine the energy of the **target application** and its subtasks (aka. energy provenance)
- **Exclude** the energy used by collocated jobs (aka. "noisy neighbors")



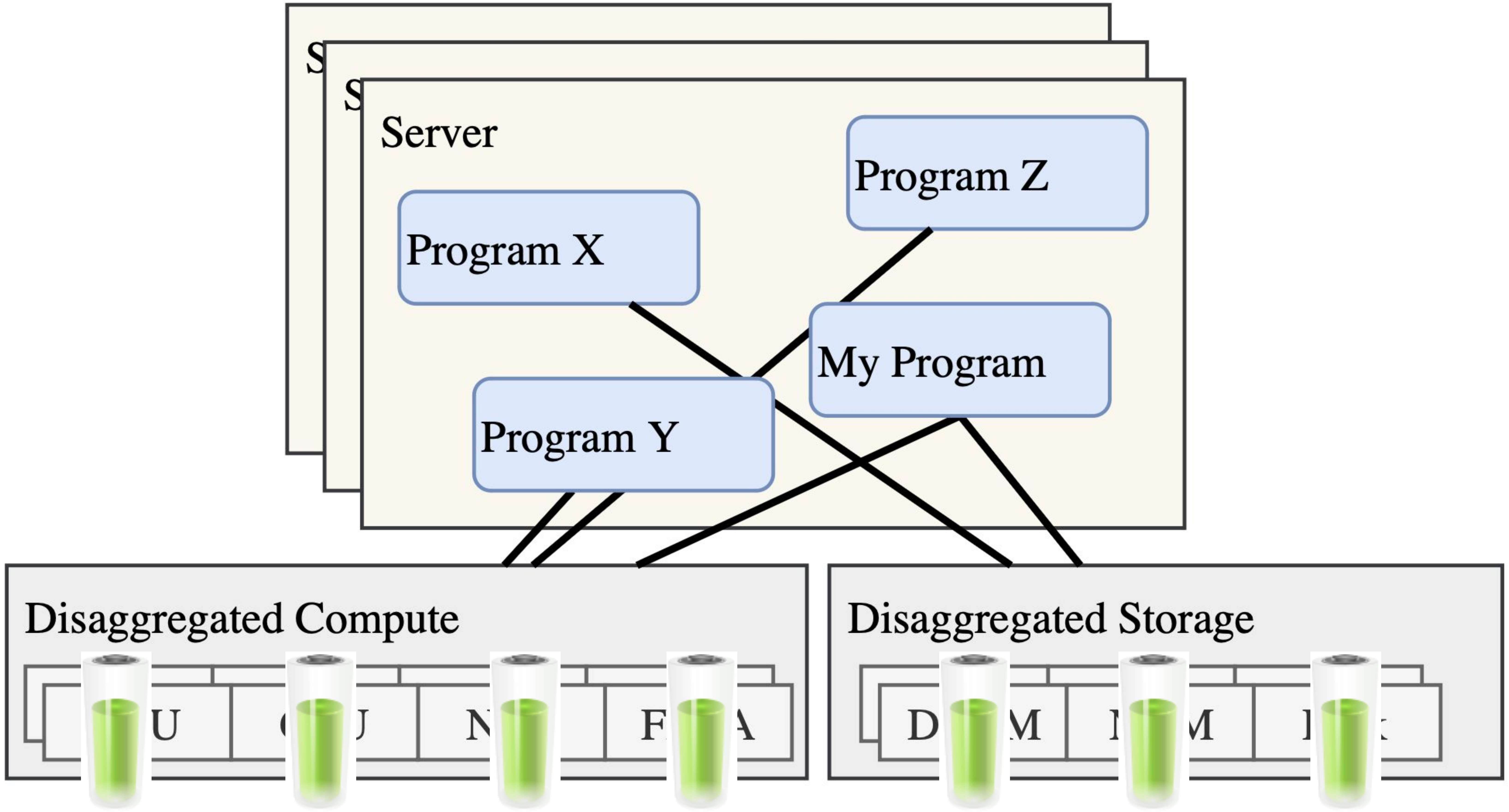
Gaps in Energy Attribution methods

- Many useful tools available
  - E.g., CodeCarbon [Lacoste et al.], Carbontracker [Wolff Anthony et al.], and Experiment Impact Tracker [Henderson et al.]
- Coarse-grained energy attribution



A Key Culprit: Coarse hardware support

- Heterogeneous, disaggregated hardware shared in the cloud
- Only device/user-level power estimation



Hongyu Hè, Michal Friedman, and Theodoros Rekatsinas. "EnergAt: Fine-Grained Energy Attribution for Multi-Tenancy." HotCarbon '23

Our Position

Fine-grained software energy attribution is feasible even with **coarse-grained** hardware support!

Contributions

1. Thread-level, NUMA-aware method for CPU and DRAM energy attribution in a multi-tenant environment
2. Evaluation with open-sourced prototype implementation model validity, effectiveness, and robustness to noisy-neighbor effect
3. Opportunities and challenges towards energy-aware clouds

NUMA-Aware Thread-Level Model for Multi-Tenancy

$$\left(P^{D}_{\text{static}}\right)^{s} = \left(\text{Sample energy value of } D \text{ for } T_{\text{static}}\right)/T_{\text{static}}. \quad (3)$$

$$\left(E^{D}_{\text{static}}\right)^{s} = \left(P^{D}_{\text{static}}\right)^{s} \cdot T_{\text{sample}}. \quad (4)$$

$$\left(E^{\text{CPU}}_{\Delta}\right)^{s} = \left(E^{\text{CPU}}_{\text{total}}\right)^{s} - \left(E^{\text{CPU}}_{\text{static}}\right)^{s}. \quad (5)$$

$$\mathbb{P}^{\text{CPU}}(s \mid a) \approx \left(\int_{t=t'}^{t'+T_{\text{sample}}} \mathbb{1}_{\{a \text{ on } s\}}\, dt\right) \Big/ T_{\text{sample}}, \quad (6)$$

$$\left(T^{\text{CPU}}_{\mathcal{A}}\right)^{s} = \mathbb{E}\left[T^{\text{CPU}}_{\mathcal{A}} \mid s\right] \approx \sum_{a \in \mathcal{A}} \mathbb{P}^{\text{CPU}}(s \mid a) \cdot T^{\text{CPU}}_{a}, \quad (7)$$

$$\left(T_{\text{total}}\right)^{s} \leftarrow \text{Total CPU time (kernel + user) of } s \quad (8)$$

$$\left(C^{\text{CPU}}_{\mathcal{A}}\right)^{s} = \left[\left(T^{\text{CPU}}_{\mathcal{A}}\right)^{s} / \left(T^{\text{CPU}}_{\text{total}}\right)^{s}\right]^{\gamma}, \quad (9)$$

$$E^{\text{CPU}}_{\mathcal{A}} = \sum_{s \in S} \left(E^{\text{CPU}}_{\Delta}\right)^{s} \cdot \left(C^{\text{CPU}}_{\mathcal{A}}\right)^{s} + \left(E^{\text{CPU}}_{\text{static}}\right)^{s}. \quad (10)$$

$$\left(M_{\text{total}}\right)^{s} \leftarrow \text{Total available NUMA memory on } s \quad (11)$$

$$\left(E^{\text{DRAM}}_{\Delta}\right)^{s} = \left(E^{\text{DRAM}}_{\text{total}}\right)^{s} - \left(E^{\text{DRAM}}_{\text{static}}\right)^{s}. \quad (12)$$

$$\mathbb{P}^{\text{DRAM}}(s \mid a) \approx \mathbb{E}\left[\left\{\left(M^{\Delta t}_{a}\right)^{s} \Big/ \left(M^{\Delta t}_{\text{total}}\right)^{s}\right\}^{T_{\text{sample}}}\right], \quad (13)$$

$$\left(M_{\mathcal{A}}\right)^{s} = \mathbb{E}\left[M_{\mathcal{A}} \mid s\right] \approx \sum_{a \in \mathcal{A}} \mathbb{P}^{\text{DRAM}}(s \mid a) \cdot \left(M_{a}\right)^{s}. \quad (14)$$

$$\left(C^{\text{DRAM}}_{\mathcal{A}}\right)^{s} = \left[\left(M_{\mathcal{A}}\right)^{s} / \left(M_{\text{total}}\right)^{s}\right]^{\sigma}, \quad (15)$$

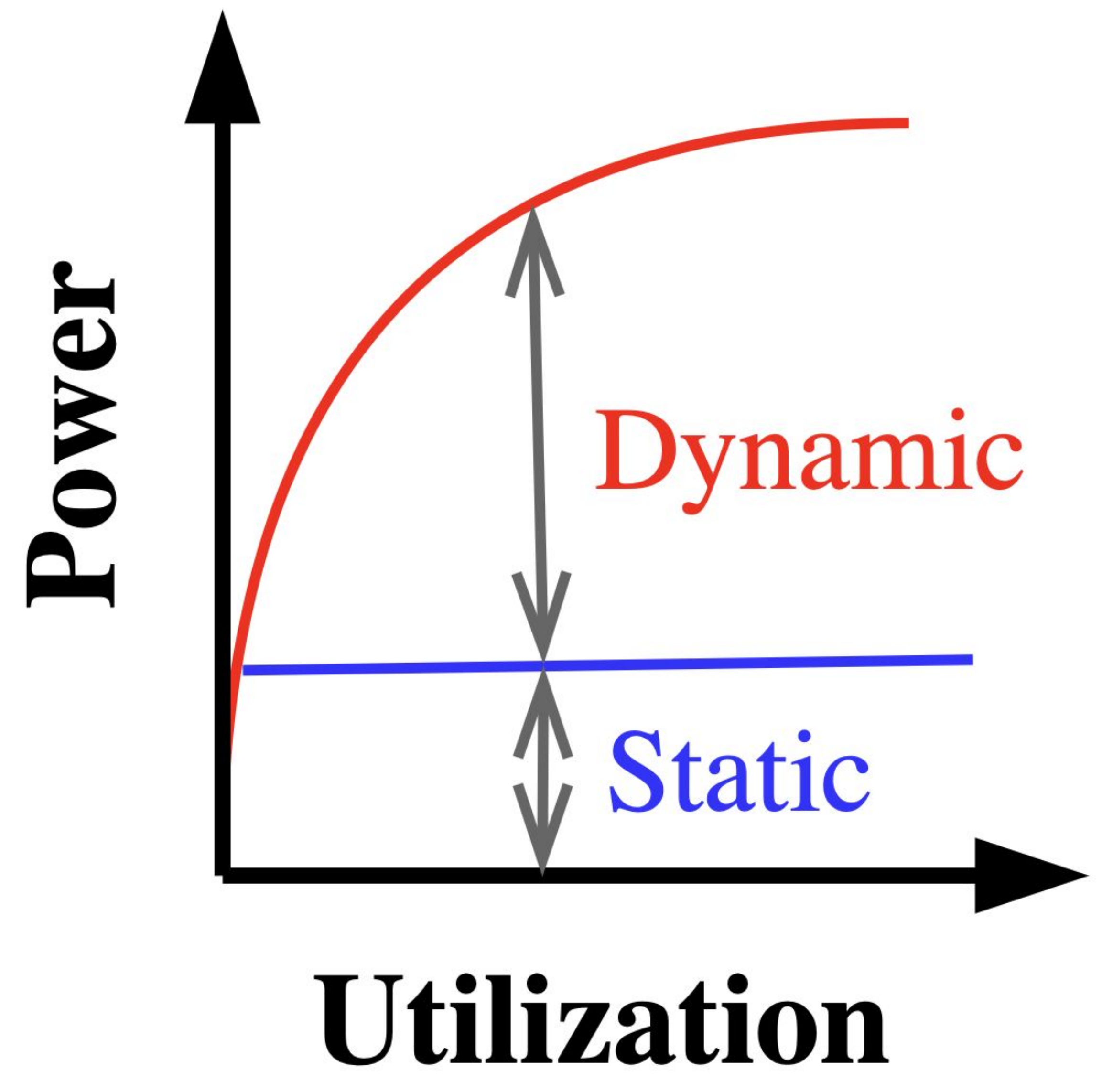
$$E^{\text{DRAM}}_{\mathcal{A}} = \sum_{s \in S} \left(E^{\text{DRAM}}_{\Delta}\right)^{s} \cdot \left(C^{\text{DRAM}}_{\mathcal{A}}\right)^{s} + \left(E^{\text{DRAM}}_{\text{static}}\right)^{s}. \quad (16)$$

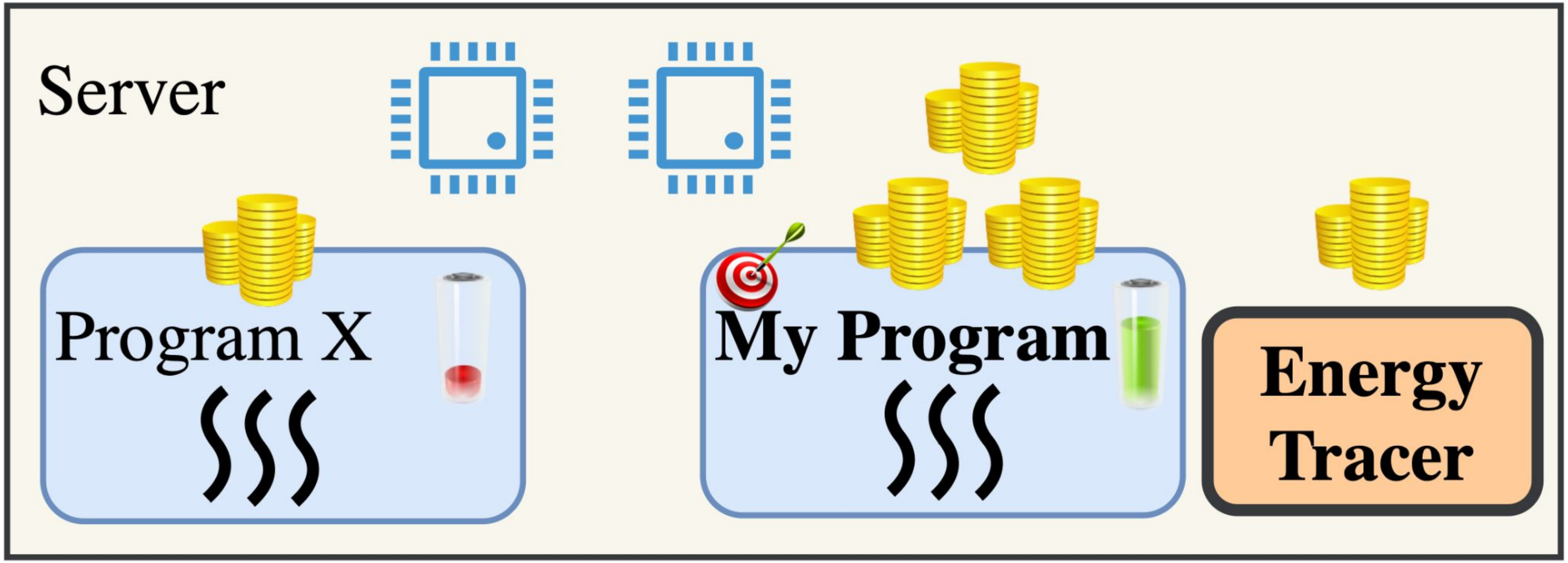
NUMA-Aware Thread-Level Model for Multi-Tenancy

Our model fits on 1 slide

Our Method: High-level principles

1. Take into account **static** vs. **dynamic power**
2. **Thread-level, NUMA-aware** energy attribution
3. **‘Energy credits’** based on exclusive resource usage
4. Separate the energy of the **attribution model itself**



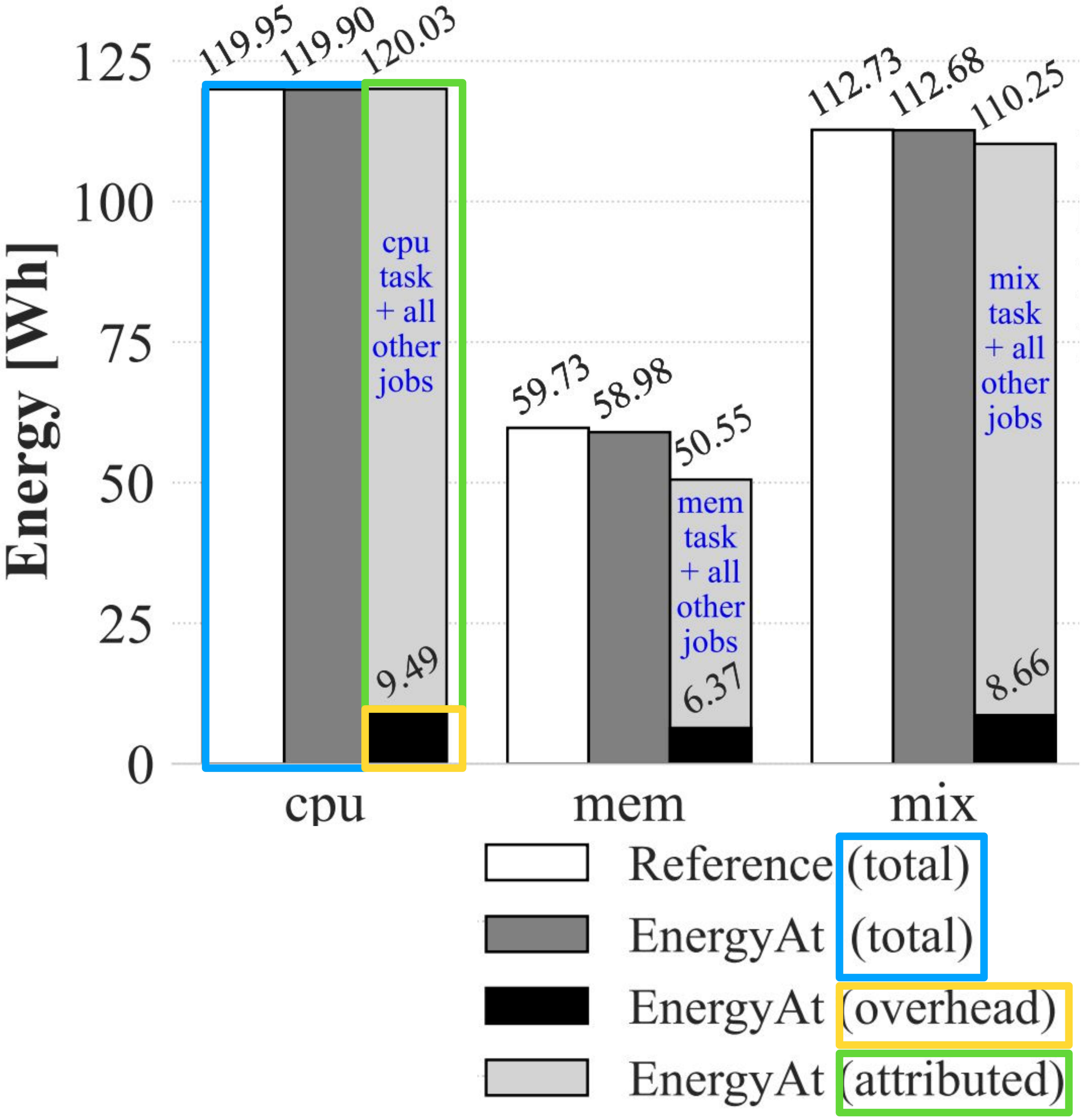


Evaluation: Setup

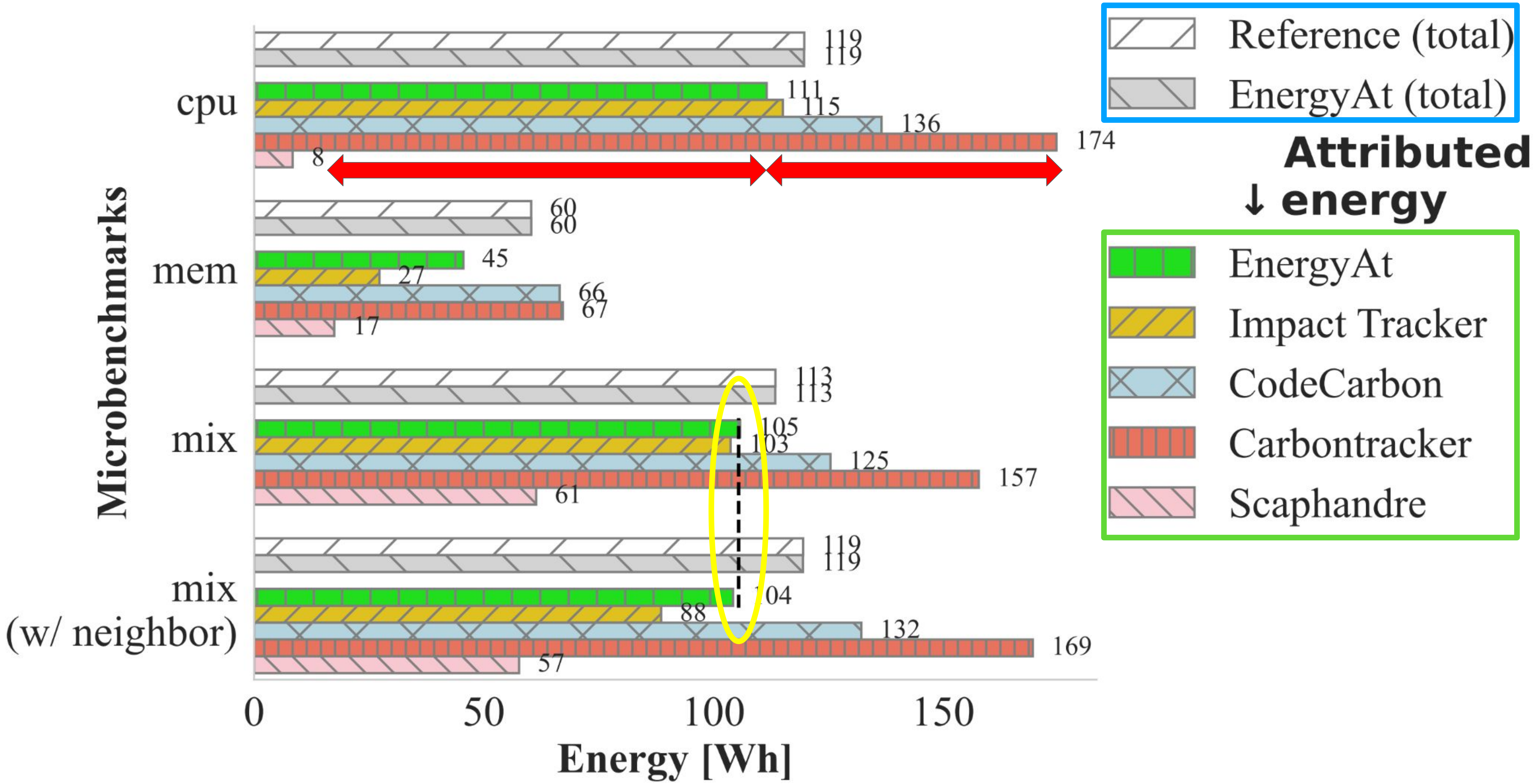
- Prototype implementation: EnergAt
  - https://github.com/HongyuHe/energat
- Microbenchmarks (target applications):
  - `cpu`: CPU utilization 0 → 100% (equal # of threads and processes)
  - `mem`: DRAM usage 0 → 100% (one process)
  - `mix`: Both CPU and DRAM at ~50% (using `cpu` and `mem` methods)
  - `mix(w/ neighbor)`: 2 mix workloads (target and noisy neighbor)
- Objectives:
  - Cover all utilization levels
  - Emulate the noisy-neighbor effect

Evaluation: Model validation

- Methodology
  - Validation by summation [Shen et al. ' 13]
- Reference (total)
  - Modified Firefox plugin
- Observations
  - Total value ≒ Reference value
  - Sum of attributed energies + cost ≒ Total
  - `mem`: underestimation
    - Only private memories are considered



Comparison with Other Tools



Hongyu Hè, Michal Friedman, and Theodoros Rekatsinas. "EnergAt: Fine-Grained Energy Attribution for Multi-Tenancy." HotCarbon '23

Limitations

1. Not considering other pertinent factors
   - E.g., shared memory, I/O, and caches
2. ‘Validation by summation’: No insight into individual accounting
3. Non-negligible energy overhead: up to 9.5% (when tracing all jobs)
4. Evaluation on real workloads

Energy-Aware Heterogeneous Clouds

1. **HW-SW interface** for **secure** and **efficient** energy reporting
2. Energy attribution for **cloud services** on **heterogeneous devices**
    - Energy-based billing in the cloud
3. NUMA-aware energy optimization
4. Revisit traditional algorithms in terms of energy efficiency

Summary

- Position: Fine-grained SW energy attribution is feasible even with coarse-grained HW support
- Contributions
  - Thread-level, NUMA-aware energy attribution for multi-tenancy
  - Validation of validity, effectiveness, and robustness to noisy-neighbor effect
  - Opportunities and challenges towards energy-aware clouds
- Code: https://github.com/HongyuHe/energat
  - `sudo pip install energat`
- Paper:

Big shout out to Shail David for his help in revising the paper!

Email: honghe@inf.ethz.ch
Web: hongyu.nl