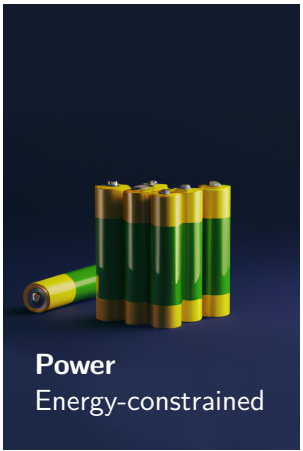
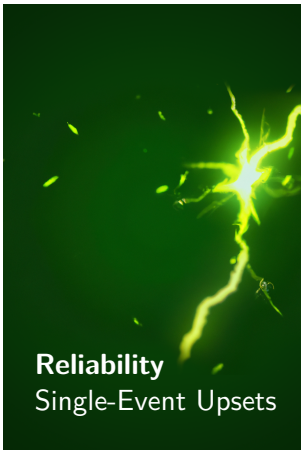
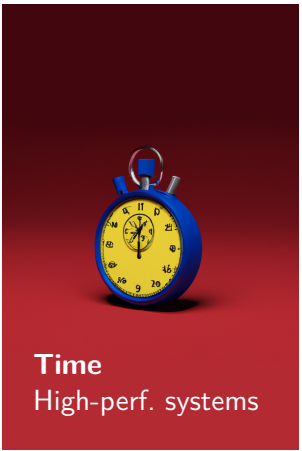
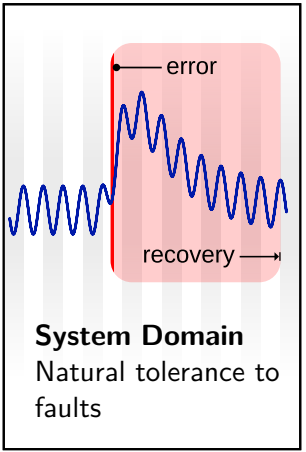
Optimizing for power, reliability, and resource utilization

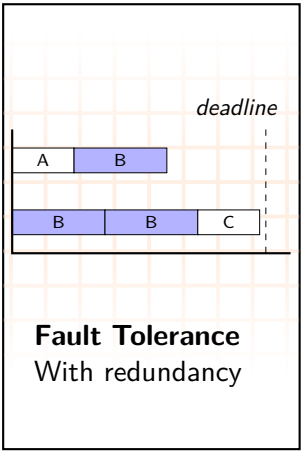




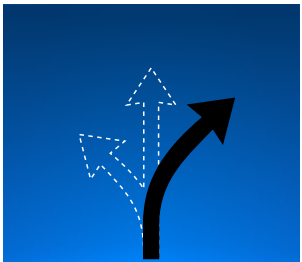


Optimizing for power, reliability, and resource utilization





Optimizing for power, reliability, and resource utilization



Solution
Dynamically change fault-tolerance

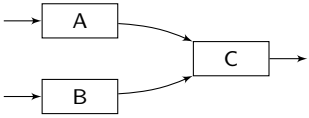
- Dynamically change coverage of fault-tolerance
- Based on resource and task considerations
  - Energy, time, task behavior
- Prepare ahead-of-time

Our method: optimize reliability for any energy budget and time constraint

Evaluated on real-time systems

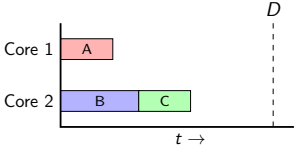
Evaluated on real-time systems

- Typically embedded
- Well-defined sequential task model
- Natural tolerance of control tasks
- Faults recovered on the (software) component level



Towards HPC applications

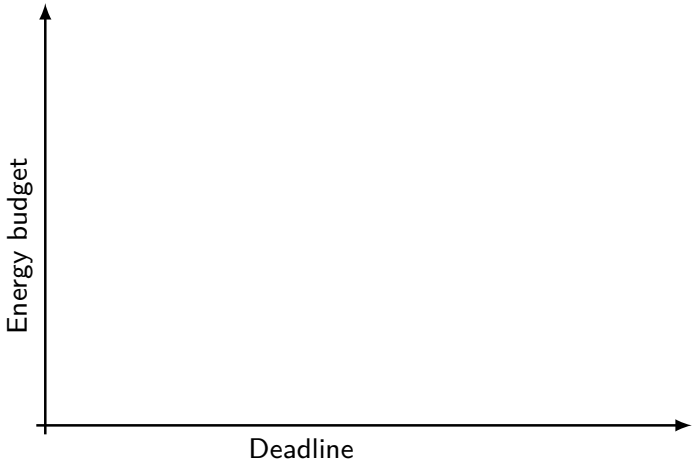
- Typically in multi-node / multi-cluster
- Less defined, streaming task model
- Natural tolerance of ML tasks (approx. computing)
- Faults recovered on the socket / node level



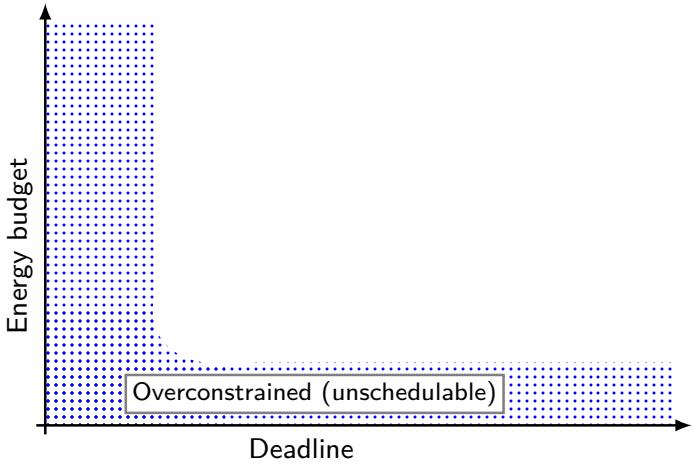
Bonus part

Some results slides for a single-node embedded system

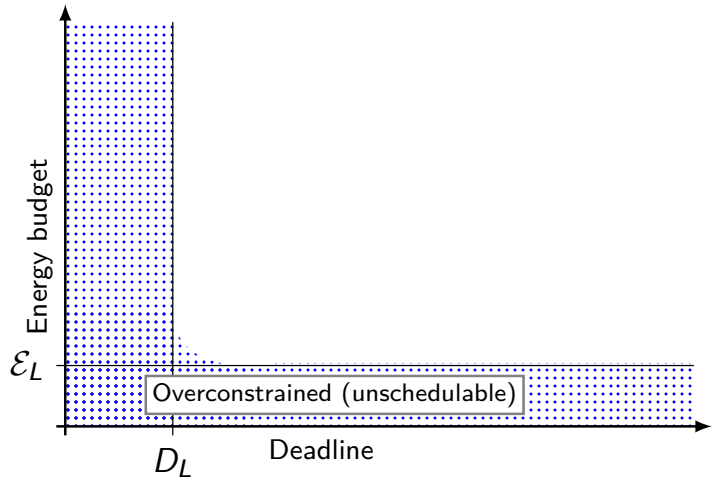
Results: what to expect



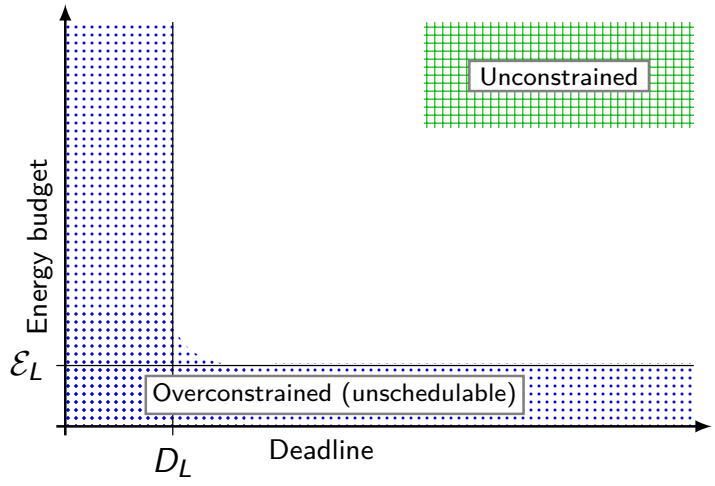
Results: what to expect



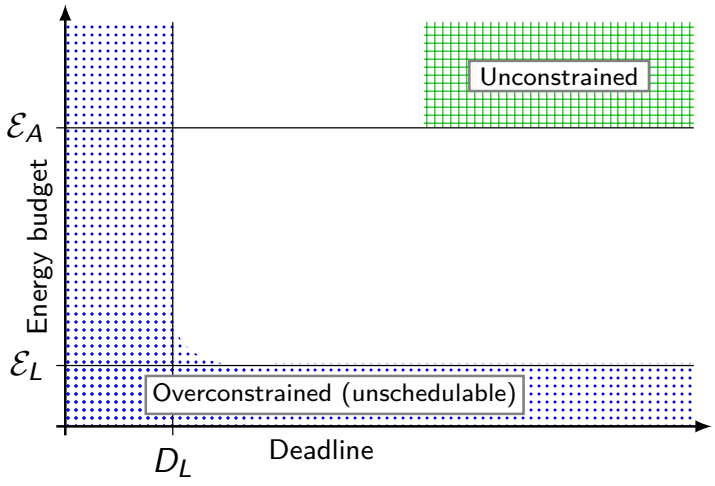
Results: what to expect



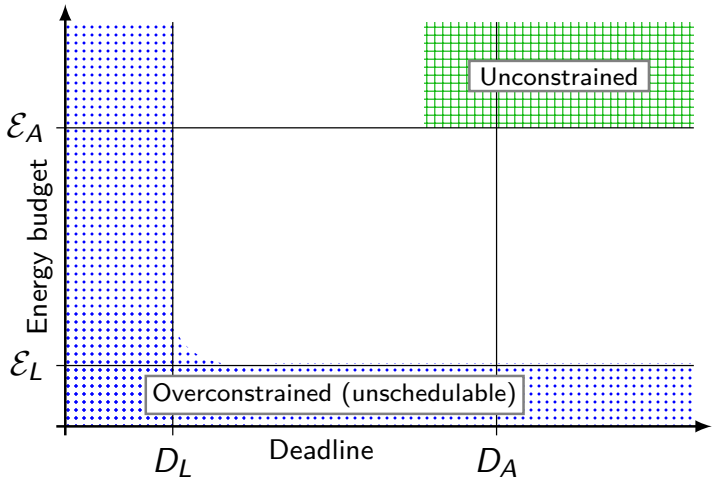
Results: what to expect



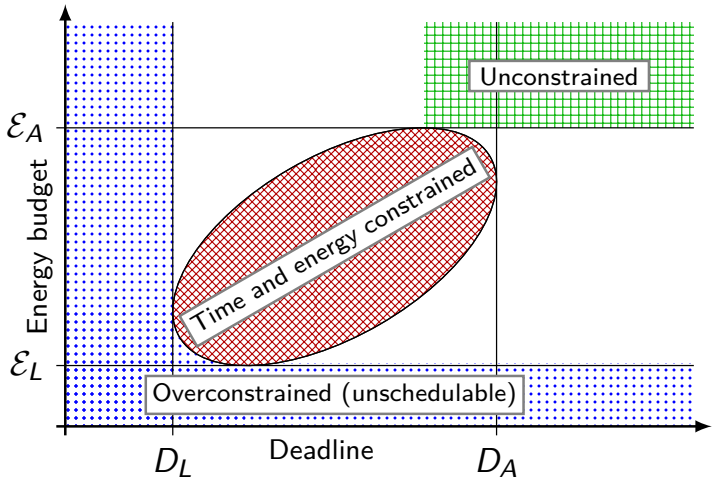
Results: what to expect



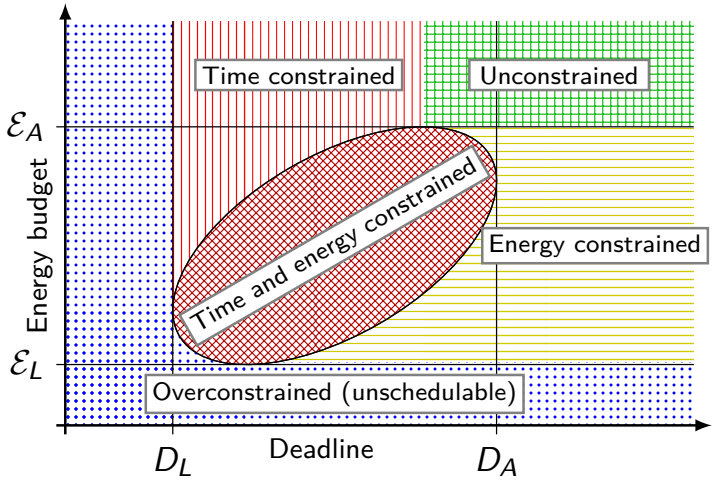
Results: what to expect



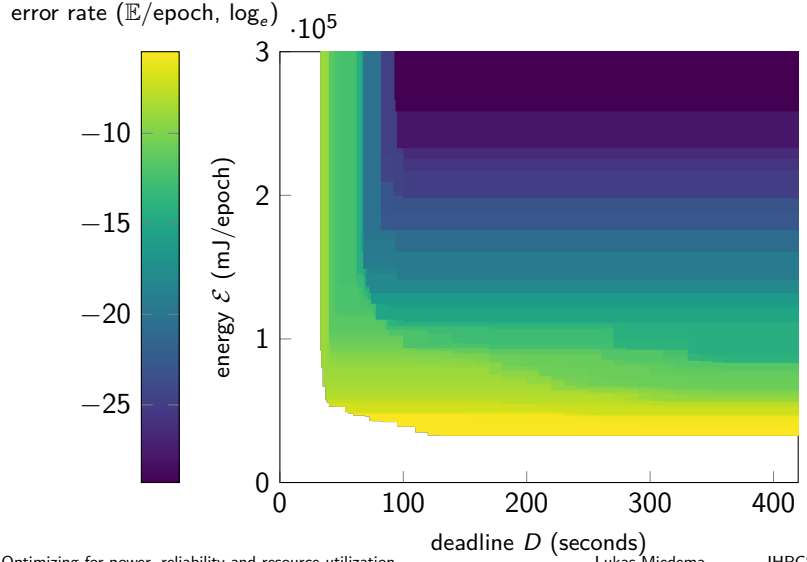
Results: what to expect



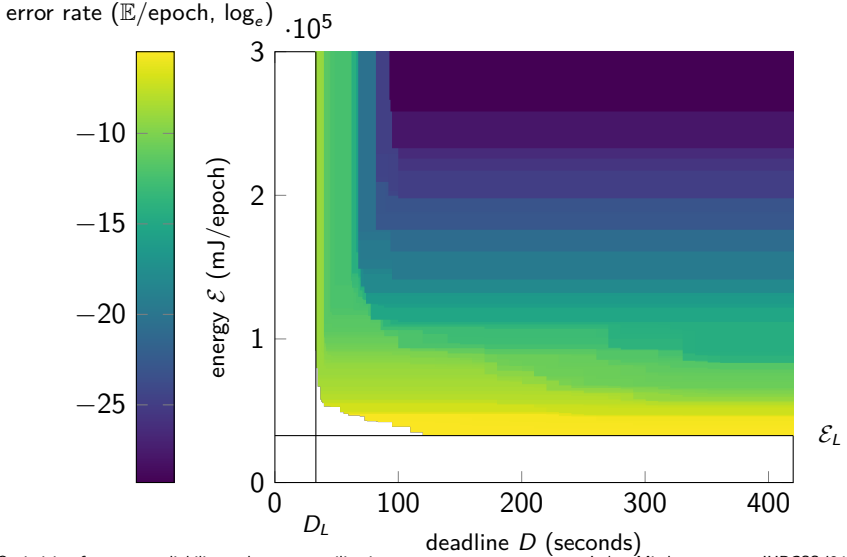
Results: what to expect



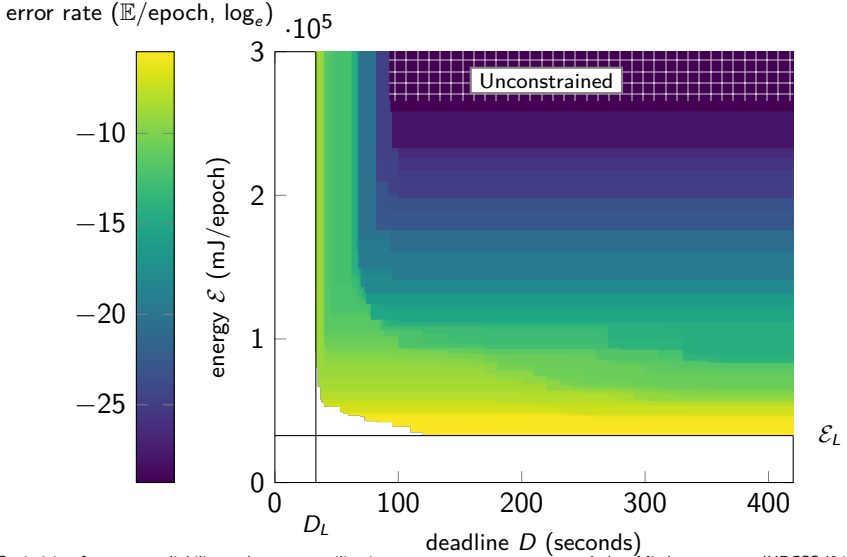
Scenario 2: sweep across deadline and energy budget



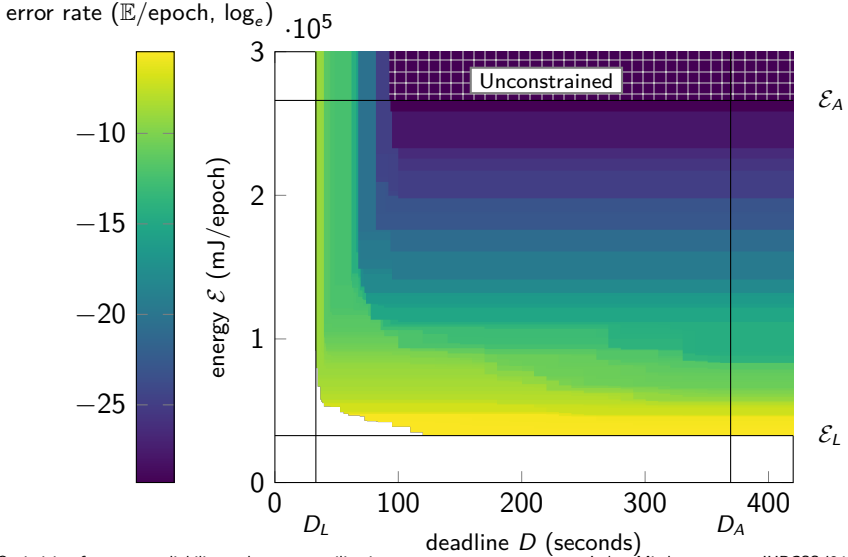
Scenario 2: sweep across deadline and energy budget



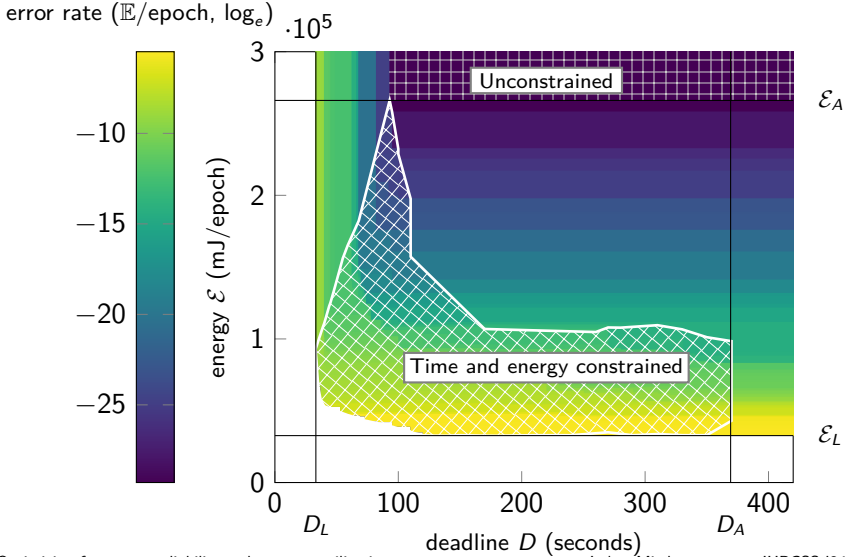
Scenario 2: sweep across deadline and energy budget



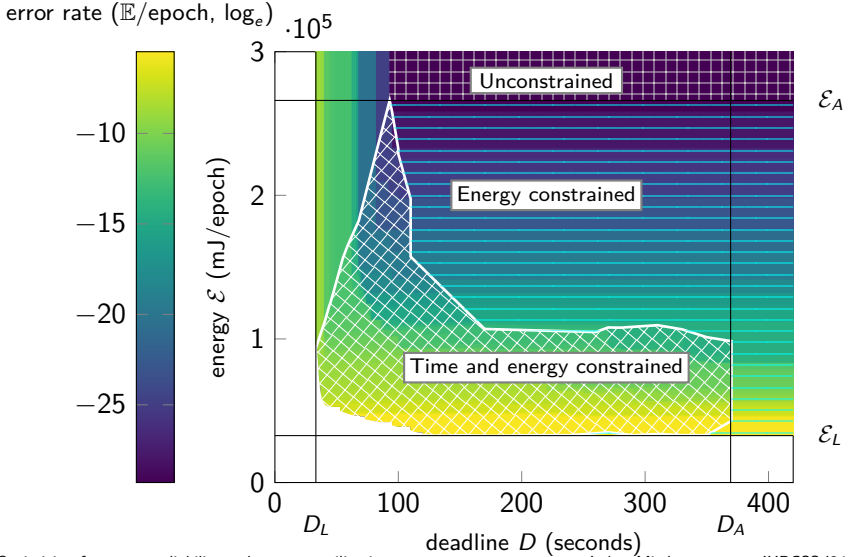
Scenario 2: sweep across deadline and energy budget



Scenario 2: sweep across deadline and energy budget



Scenario 2: sweep across deadline and energy budget



Scenario 2: sweep across deadline and energy budget