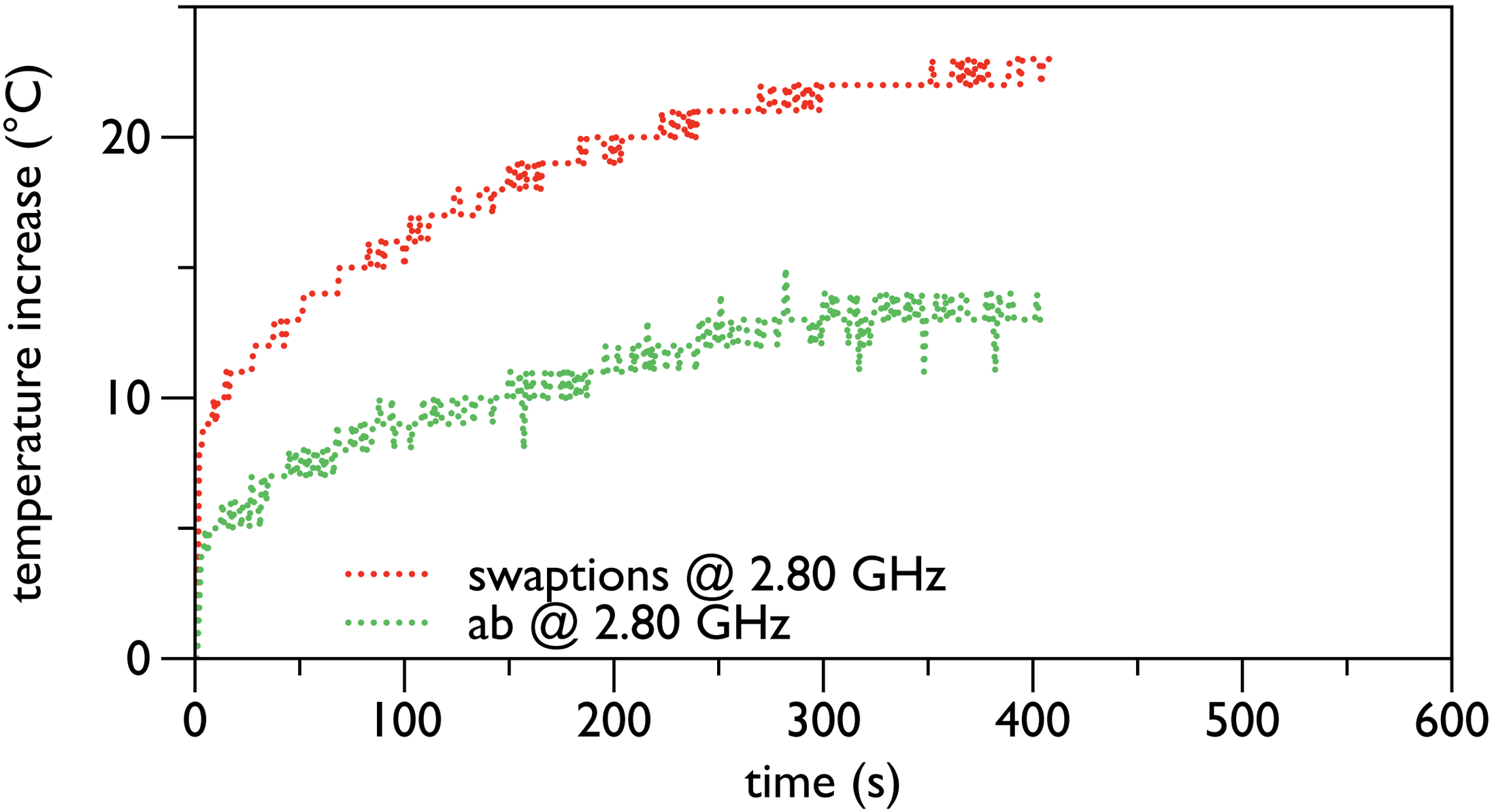
22nd International Conference on Parallel Architectures and Compilation Techniques (PACT-22), 2013
September 9, 2013
Edinburgh, Scotland, UK

ThermOS

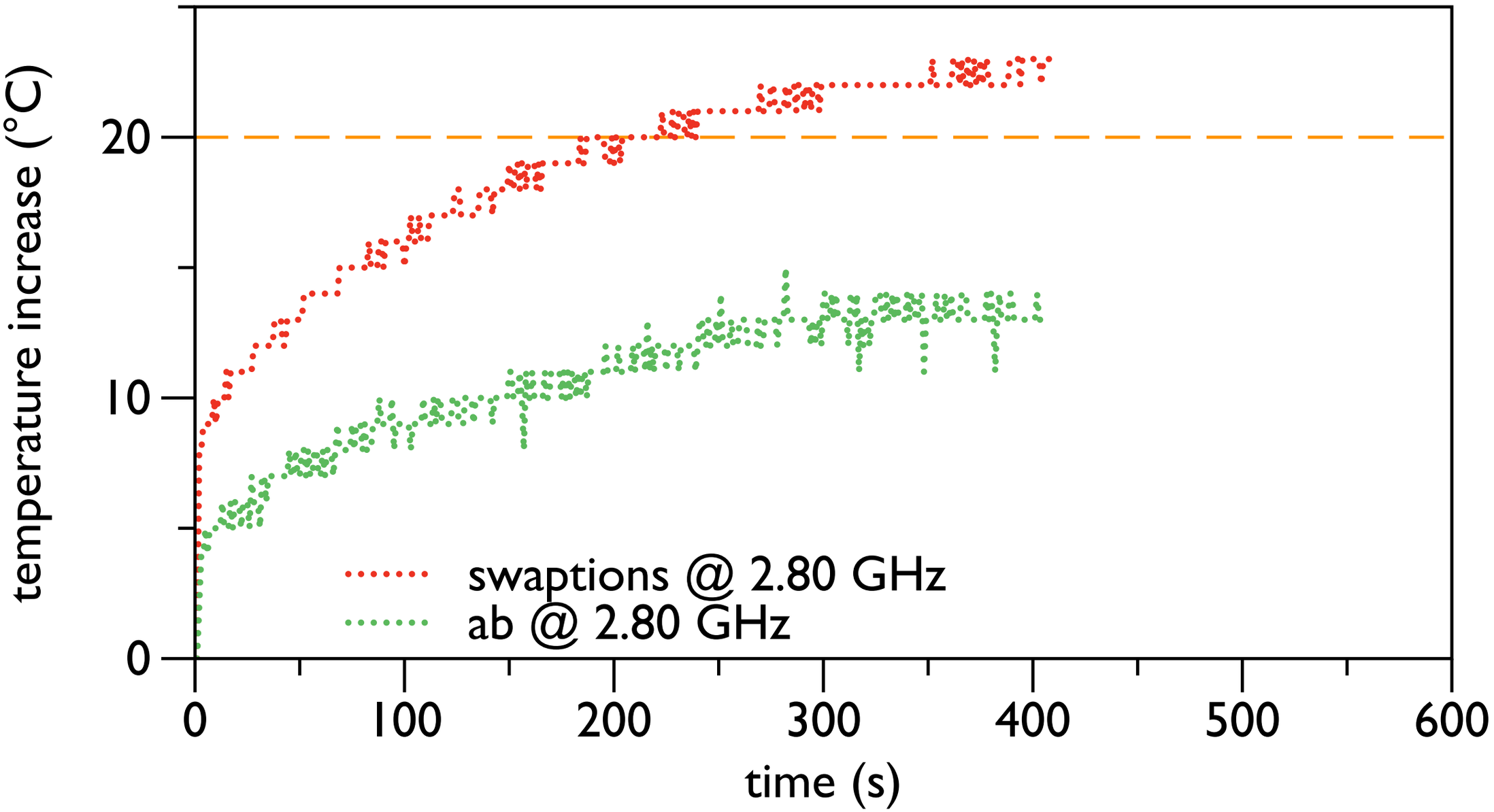
System Support for Dynamic Thermal Management of Chip Multi-Processors

Filippo Sironi (sironi@elet.polimi.it)
Martina Maggio, Riccardo Cattaneo, Giovanni F. Del Nero
Donatella Sciuto, Marco D. Santambrogio

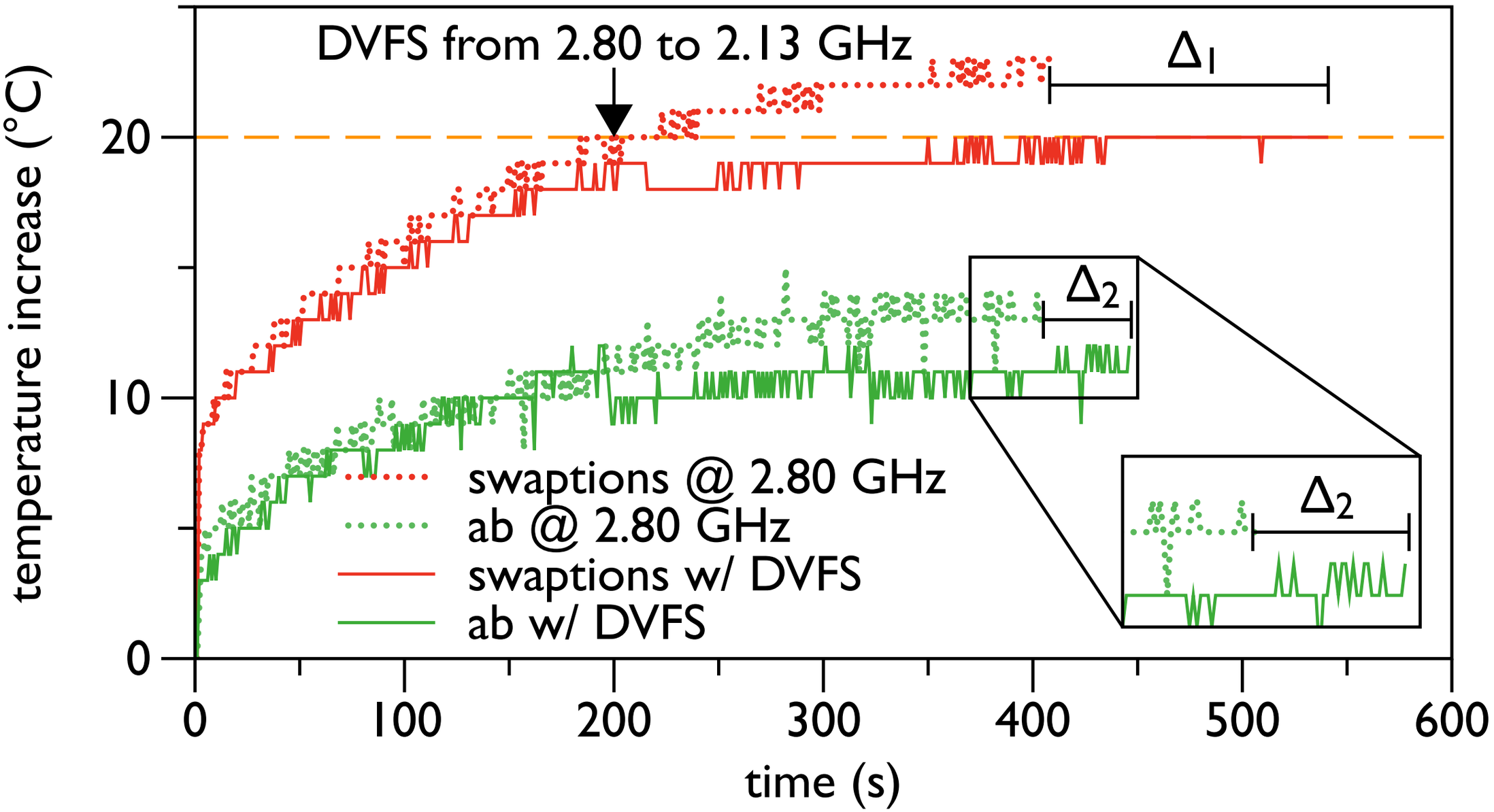
DVFS is dangerous!
(I know this is scary)



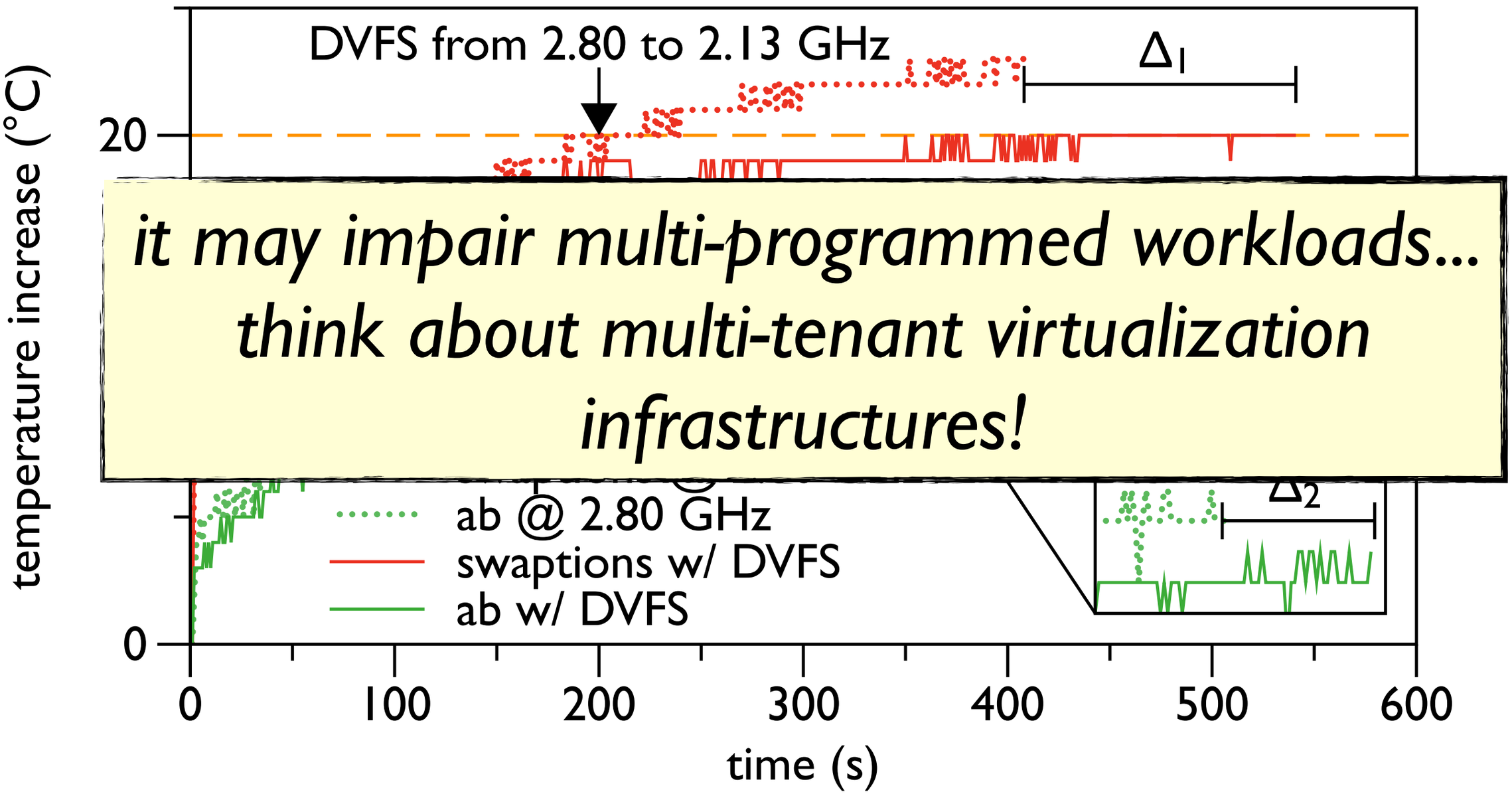
DVFS is dangerous!
(I know this is scary)



DVFS is dangerous!
(I know this is scary)

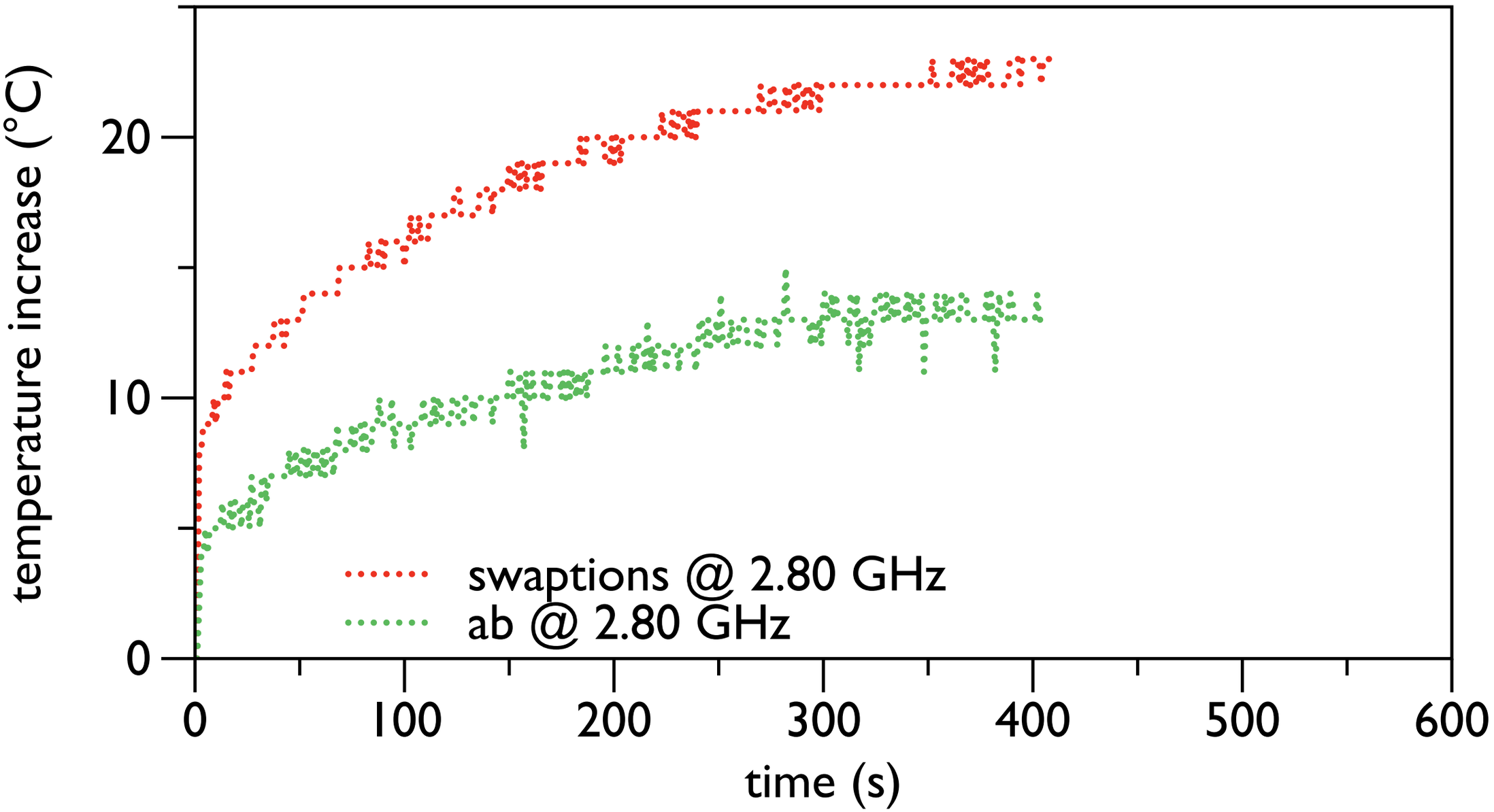


DVFS is dangerous!
(I know this is scary)

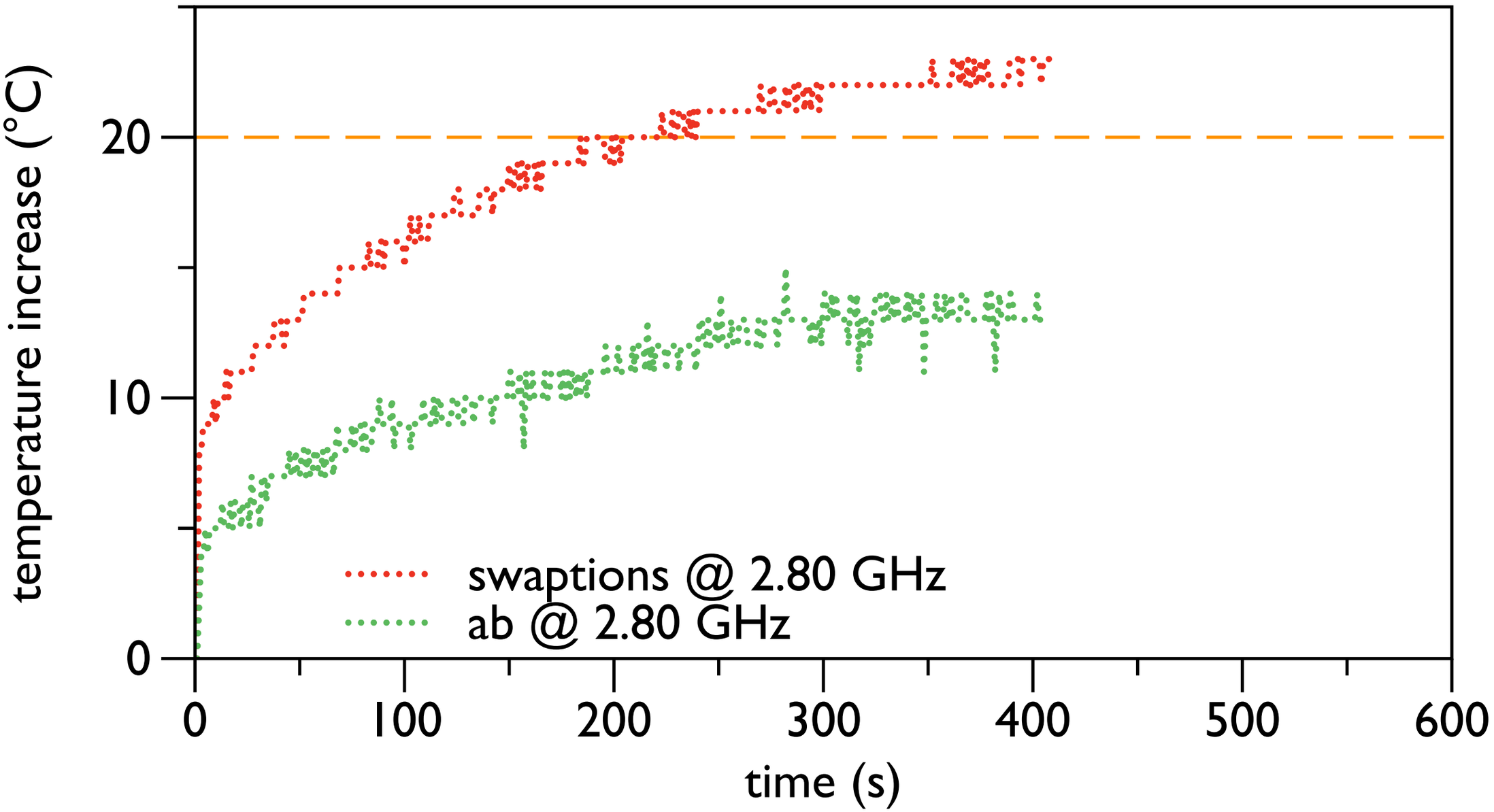


it may impair multi-programmed workloads...
think about multi-tenant virtualization infrastructures!

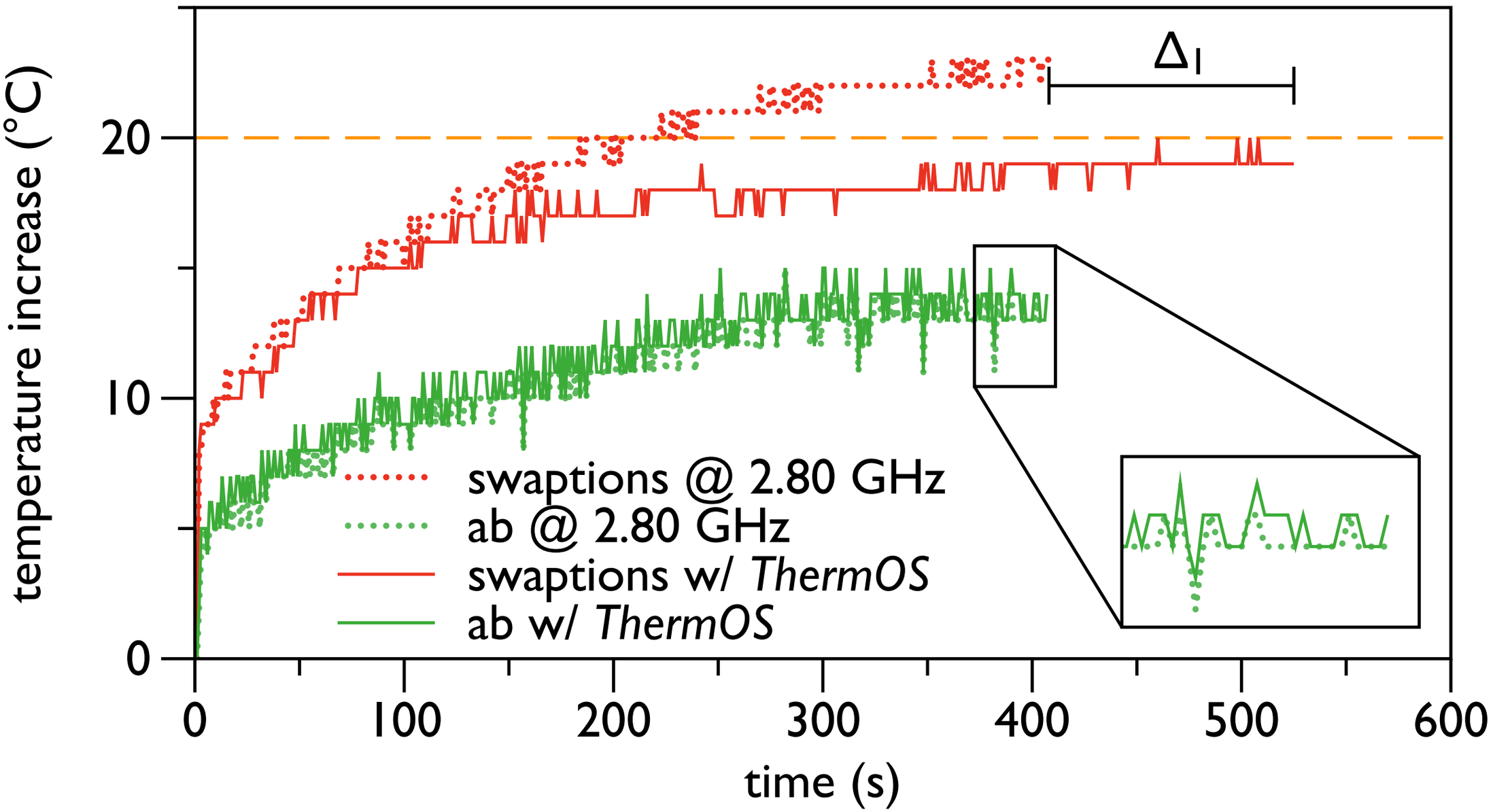
Idle cycle injection improves!



Idle cycle injection improves!



Idle cycle injection improves!



Outline

- Why DTM
- DTM in commodity CMPs
- ThermOS
- Related work
- Conclusions and Future work

Why DTM

- Transistors per unit of area are still increasing (Moore's law)
- Power density is getting worse as lithography advances (failure of Dennard's law)
- High temperature impairs performance, energy efficiency, and reliability (Srinivasan et al. in ISCA'04 [3])

DTM in commodity CMPs

- Commodity CMPs exploits DVFS
- DVFS has chip-wide side effects
  - DVFS with core-wide side effects becomes costly as soon as the core count overcomes 2 (Kim et al. in HPCA'08 [8])
  - Intel Haswell supports per-core DVFS but integrated voltage regulators may cause high temperature
- Side effects are especially bad in shared environments (e.g., multi-tenant virtualized infrastructures)

DTM in commodity CMPs

- Commodity CMPs exploits DVFS
- DVFS has chip-wide side effects
  - DVFS with core-wide side effects becomes costly as soon as the core count overcomes 2 (Kim et al. in HPCA'08 [8])
  - Intel Haswell supports per-core DVFS but integrated voltage regulators may cause high temperature
- Side effects are especially bad in shared environments (e.g., multi-tenant virtualized infrastructures)

software-driven DTM of CMPs

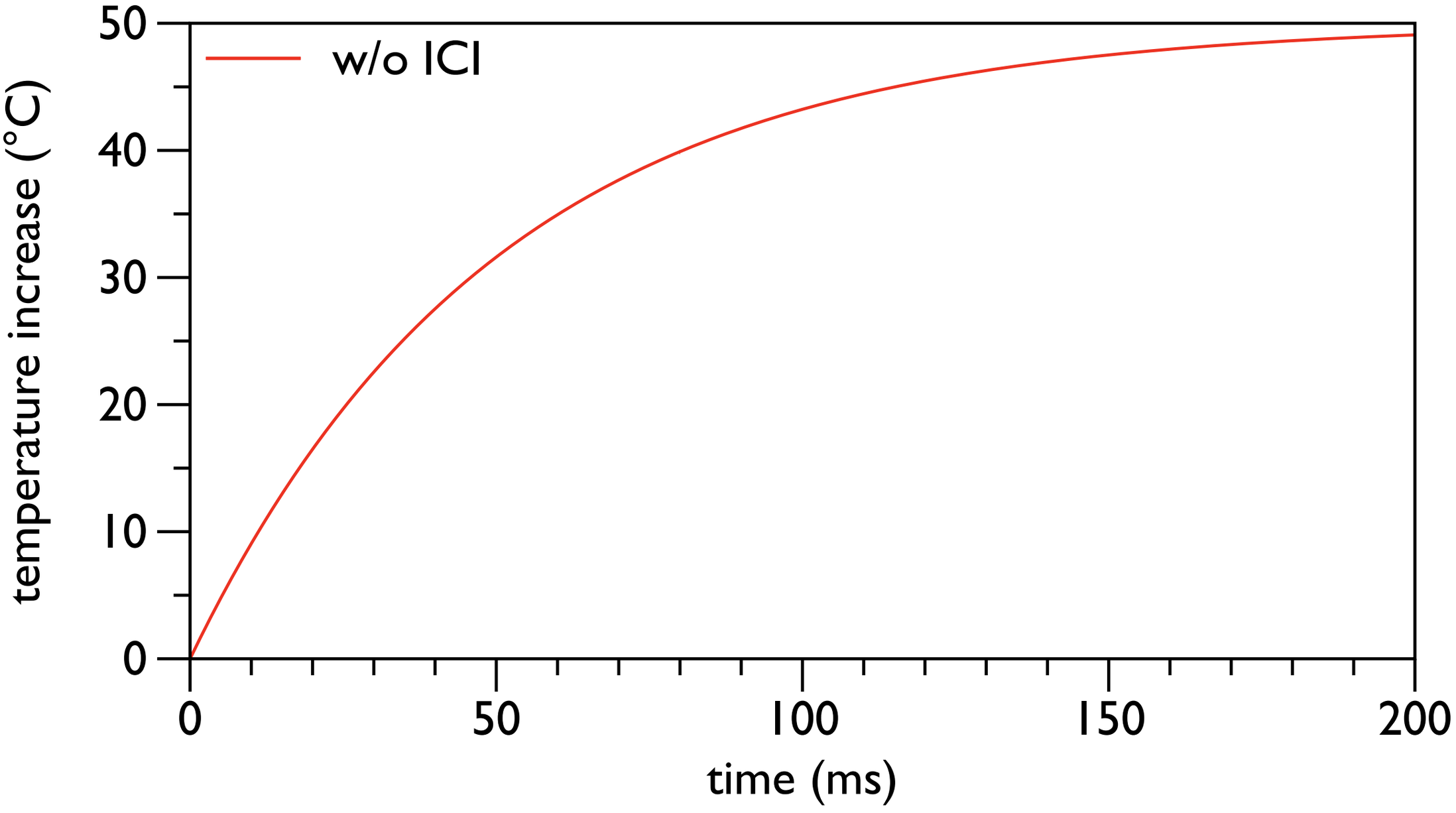
ThermOS

- Linear discrete-time modeling of temperature dynamic
- Commodity solution to measure temperature (i.e., DTSs and MSRs)
- Formal feedback control for idle cycle determination
- Idle cycle injection via operating system scheduling

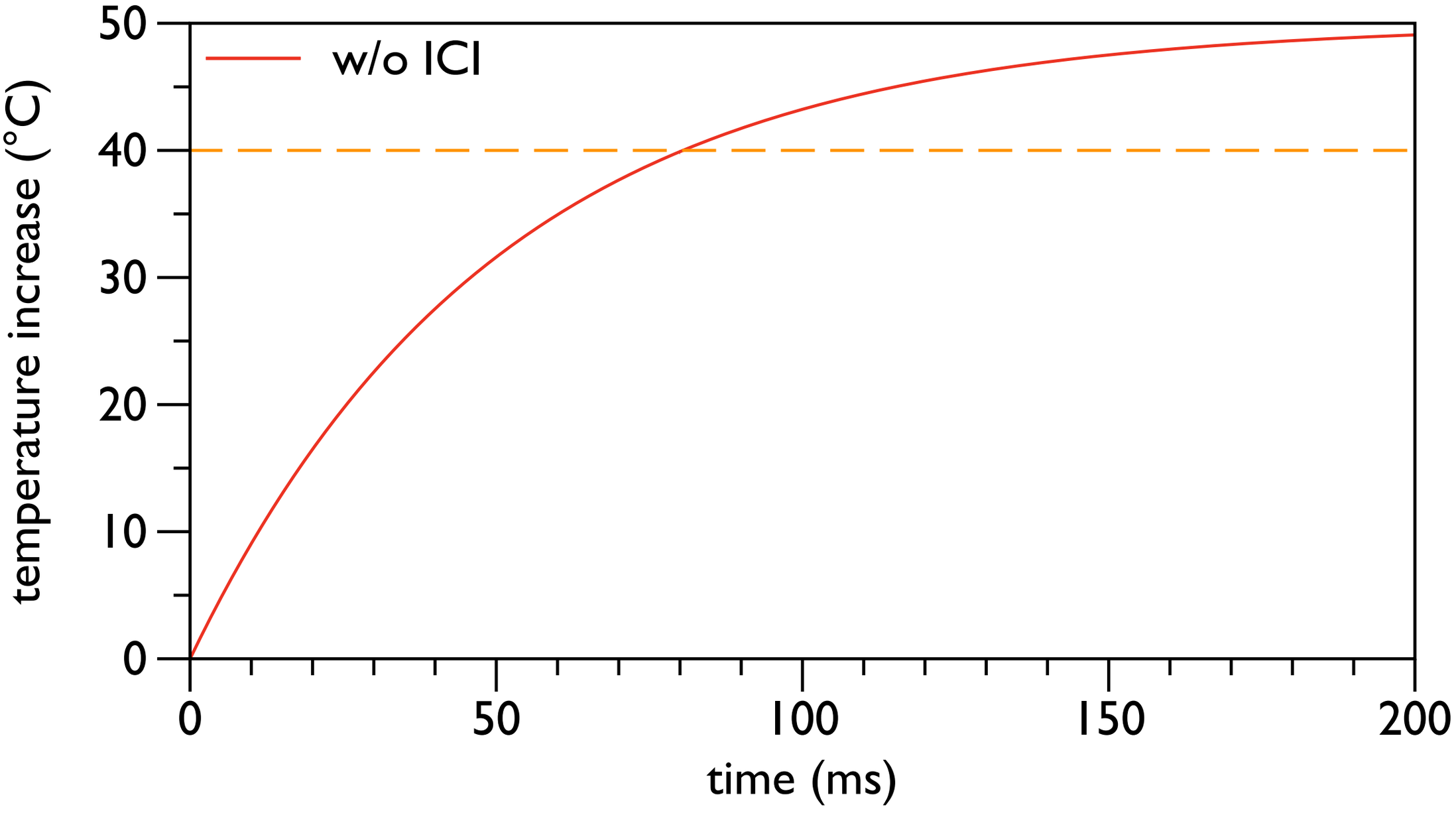
Modeling of temperature dynamic

- Modeling approaches either have shortcomings (Wattch, Brooks and Martonosi in HPCA'01 [14]) or require too many information and become impractical (HotSpot, Skadron et al. in TACO'01 [1])
- No need to understand the full temperature dynamic: we need the dynamic near the temperature threshold

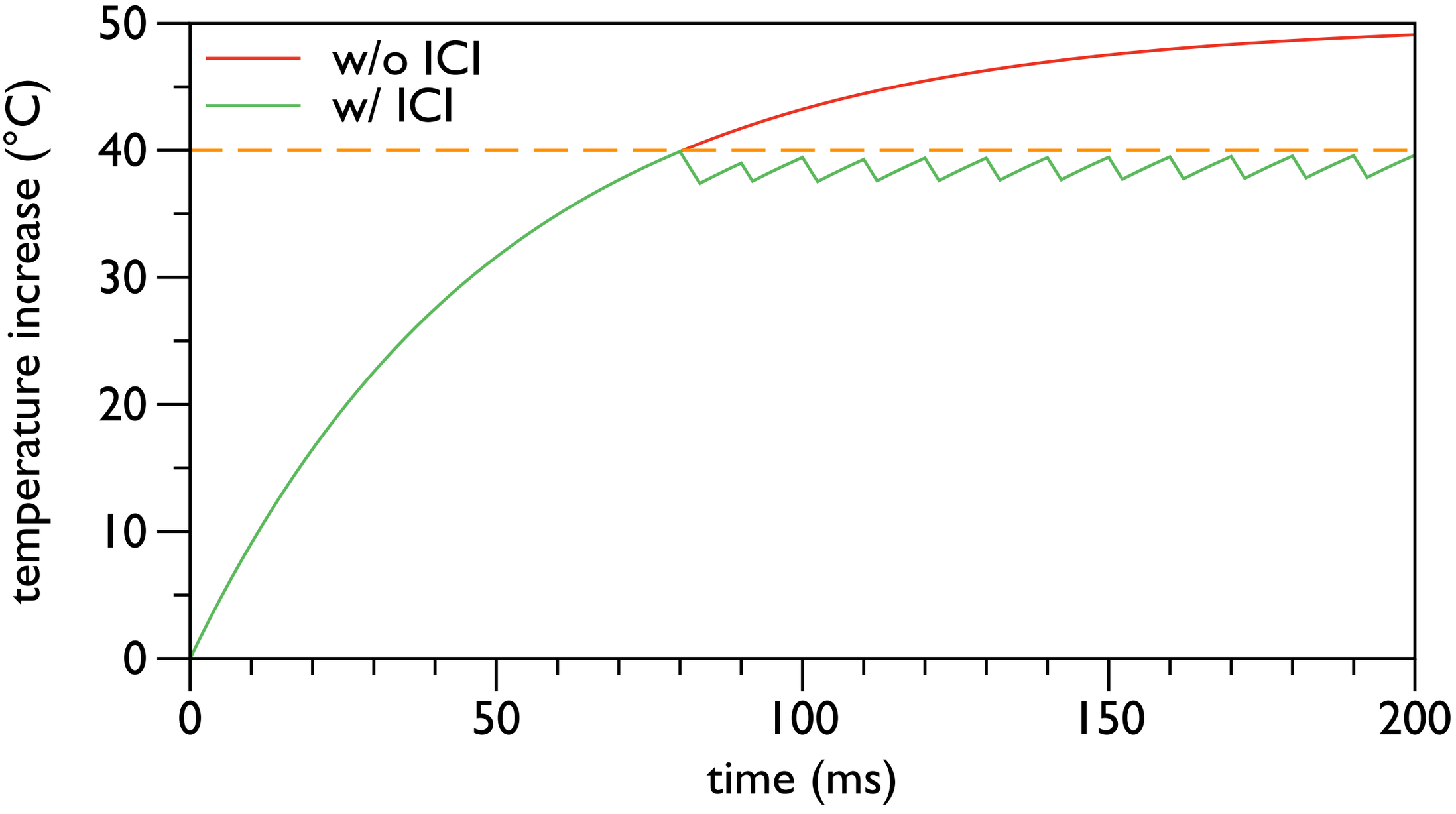
Modeling of temperature dynamic



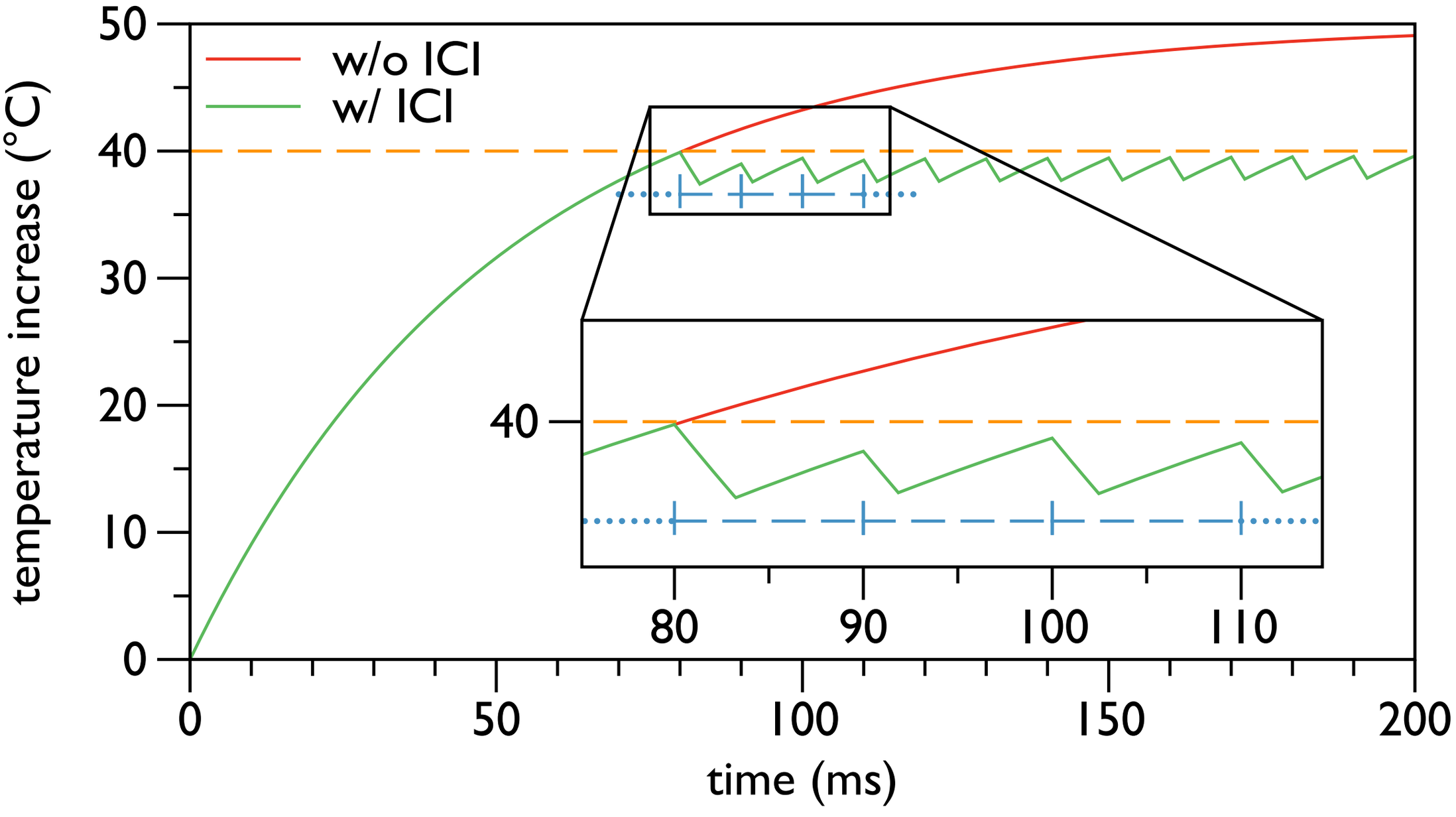
Modeling of temperature dynamic



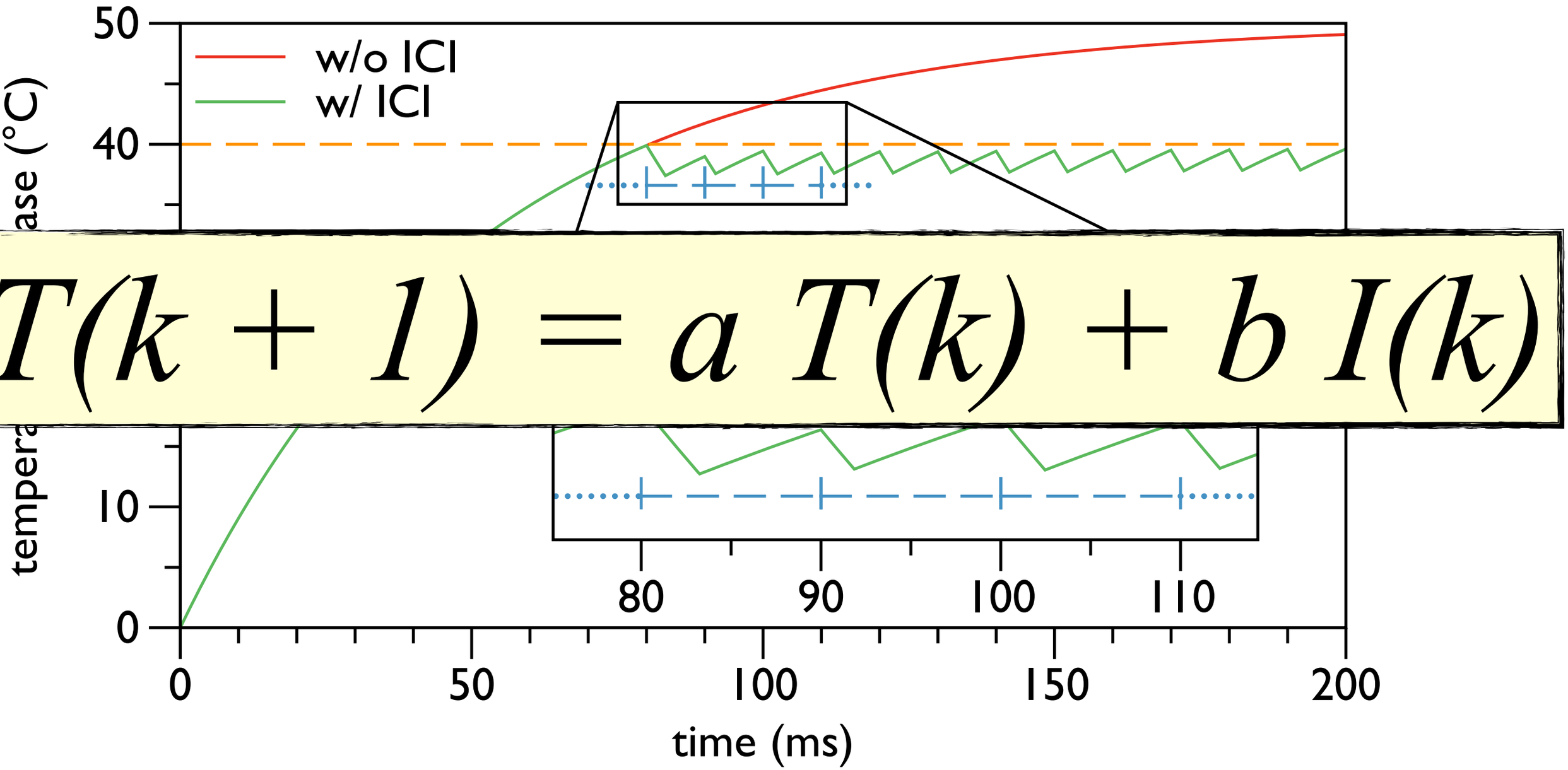
Modeling of temperature dynamic



Modeling of temperature dynamic



Modeling of temperature dynamic



$$T(k+1) = a\,T(k) + b\,I(k)$$

Linear discrete-time thermal model: offline estimation

- Low overhead but requires the model to be conservative
- Linear regression over 70% of a dataset of over 1.5 million of {temperature_next, temperature, idle} tuples; different regressions yields 95% prediction accuracy over the remaining 30% of the dataset
- Estimated variances of a and b parameters is almost negligible

Formal feedback control

- Proportional-Integral (PI) controller
  - proportional term to capture the dependency from the current error (i.e., expected minus current temperature)
  - integral term to get the dependency from past errors
- Synthesis of a “stable by definition” controller
- Robust to estimation errors of the b parameter

Formal feedback control

$$idle = previous\ idle + A\ current\ error - B\ previous\ error$$

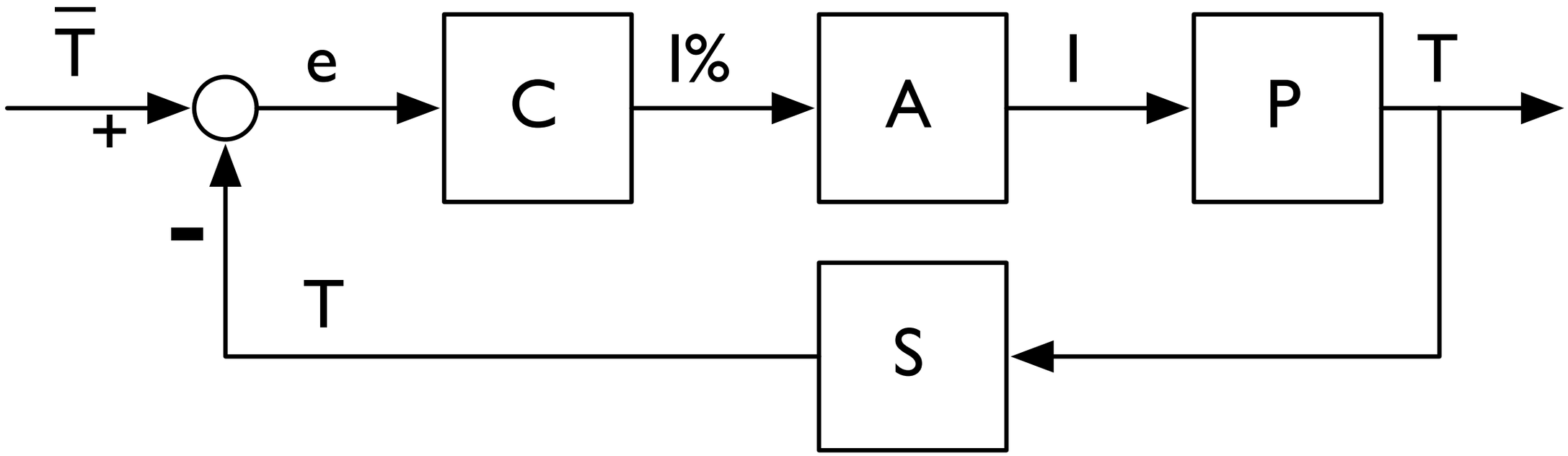
Formal feedback control

$$I(k) = I(k - 1) + e(k) (1 - p) / b - e(k - 1) a (1 - p) / b$$

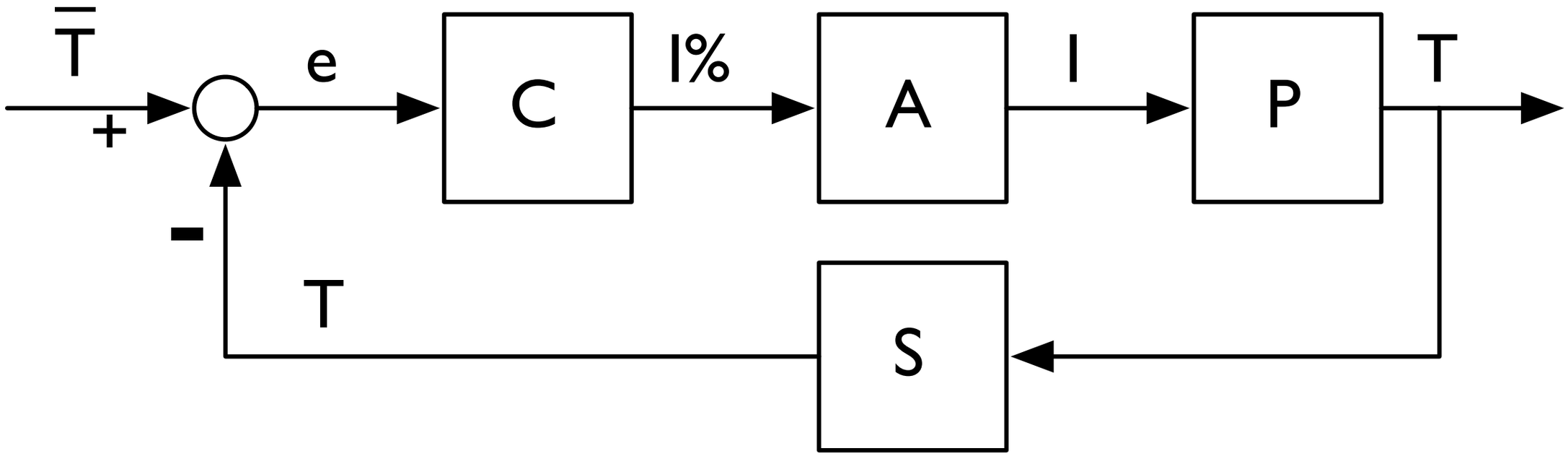
Idle cycle injection

- Do not affect the scheduling of high-priority and vital tasks (e.g., real-time task and kernel tasks)
- Exploit task scheduling and cpuidle (Pallipadi et al. in Linux Symposium'07 [10]) and is not invasive thanks to the use of the dynamic tick code
- Alternative solutions are suboptimal from either a software engineering or an effectiveness stand point

ThermOS



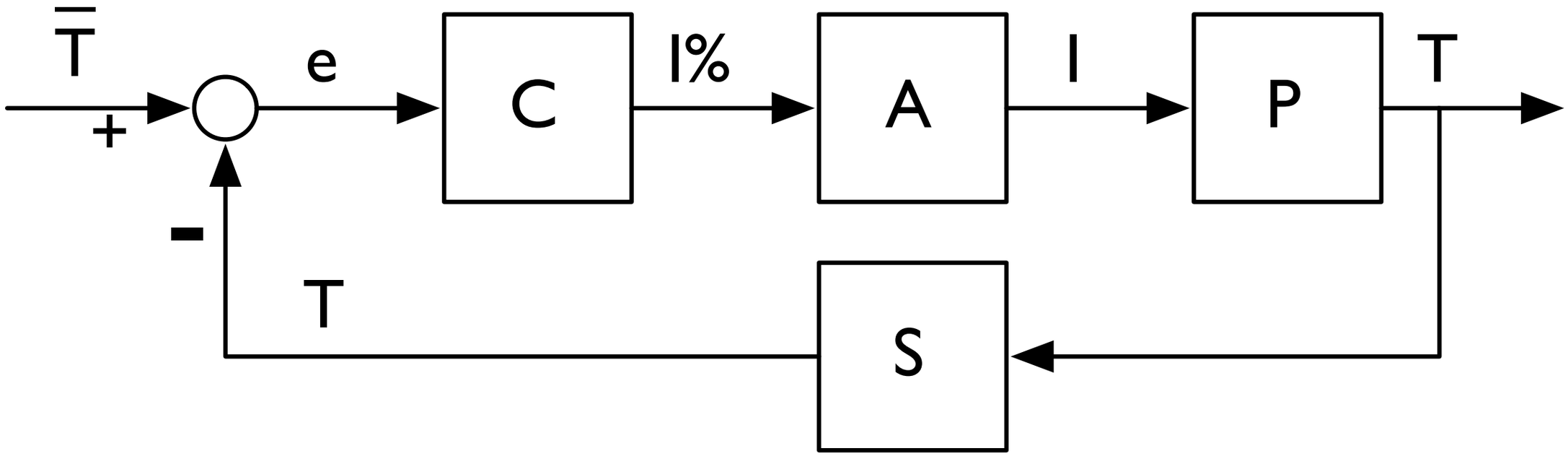
ThermOS



one feedback controller per core

ThermOS

10 ms of control period

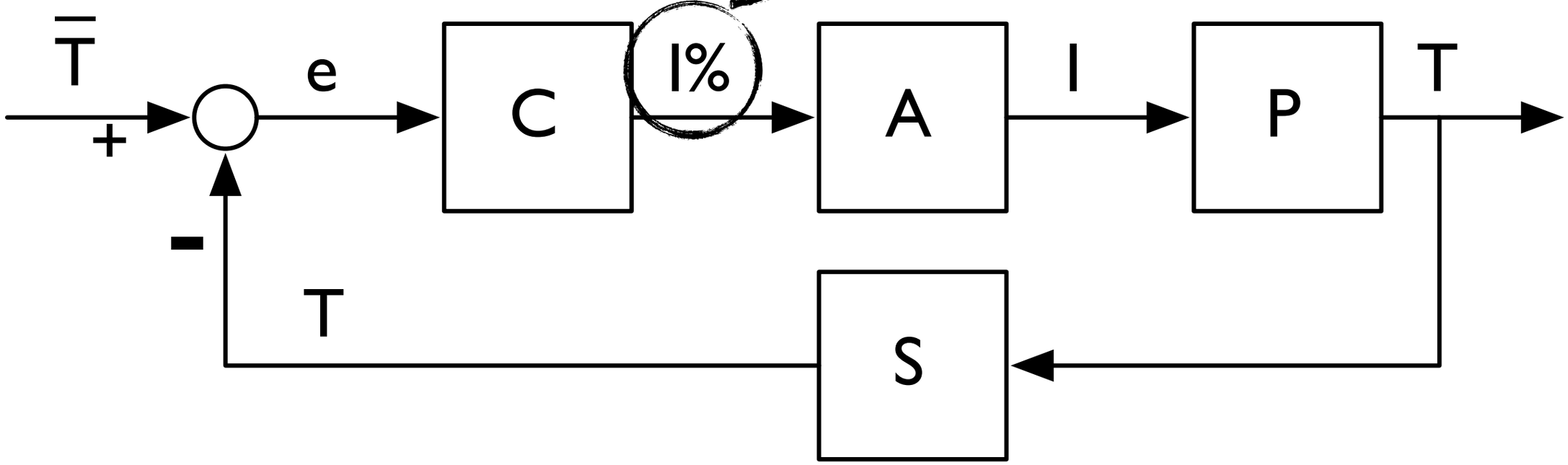


one feedback controller per core

ThermOS

max. 80% of control period

10 ms of control period

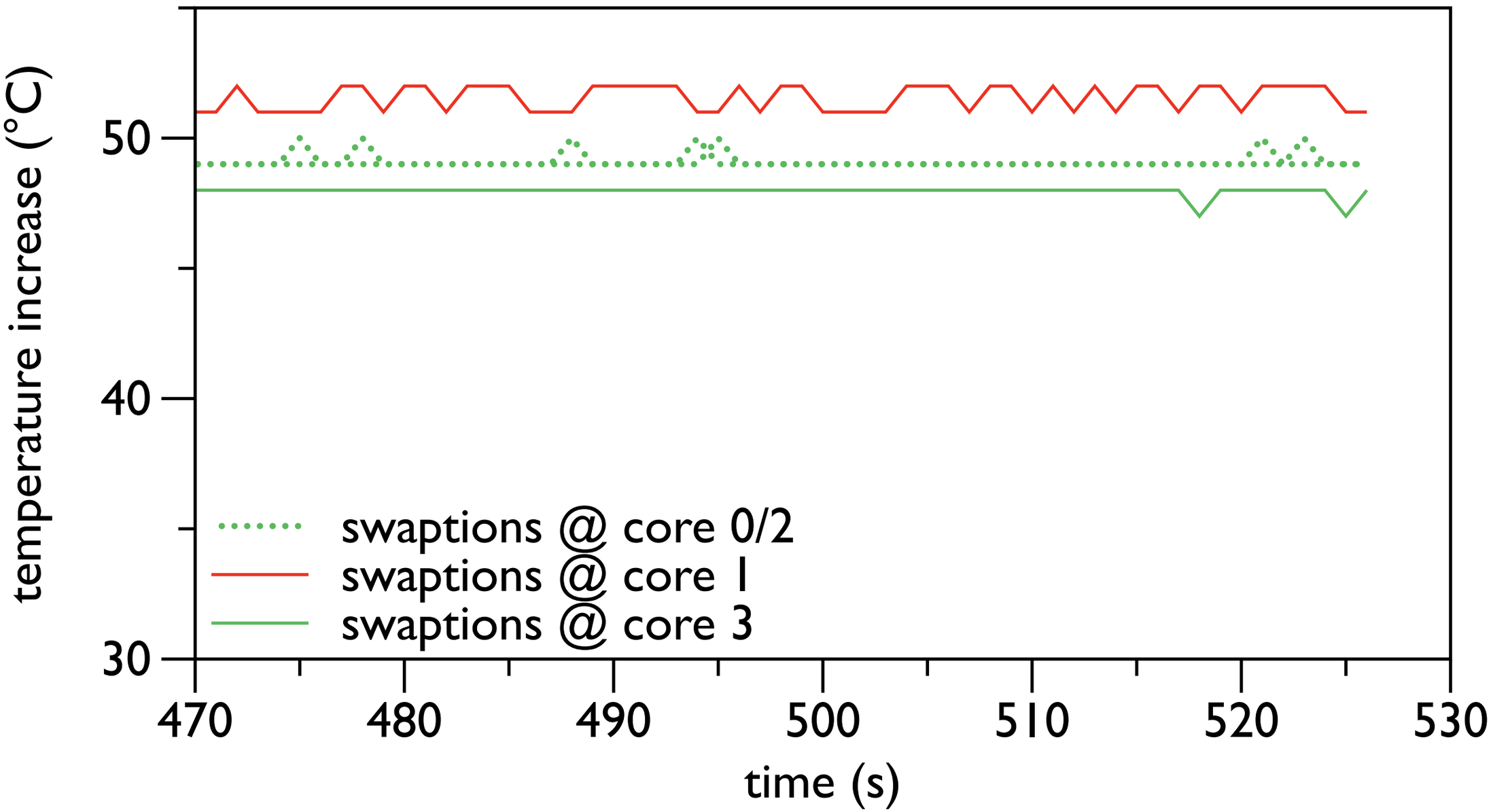


one feedback controller per core

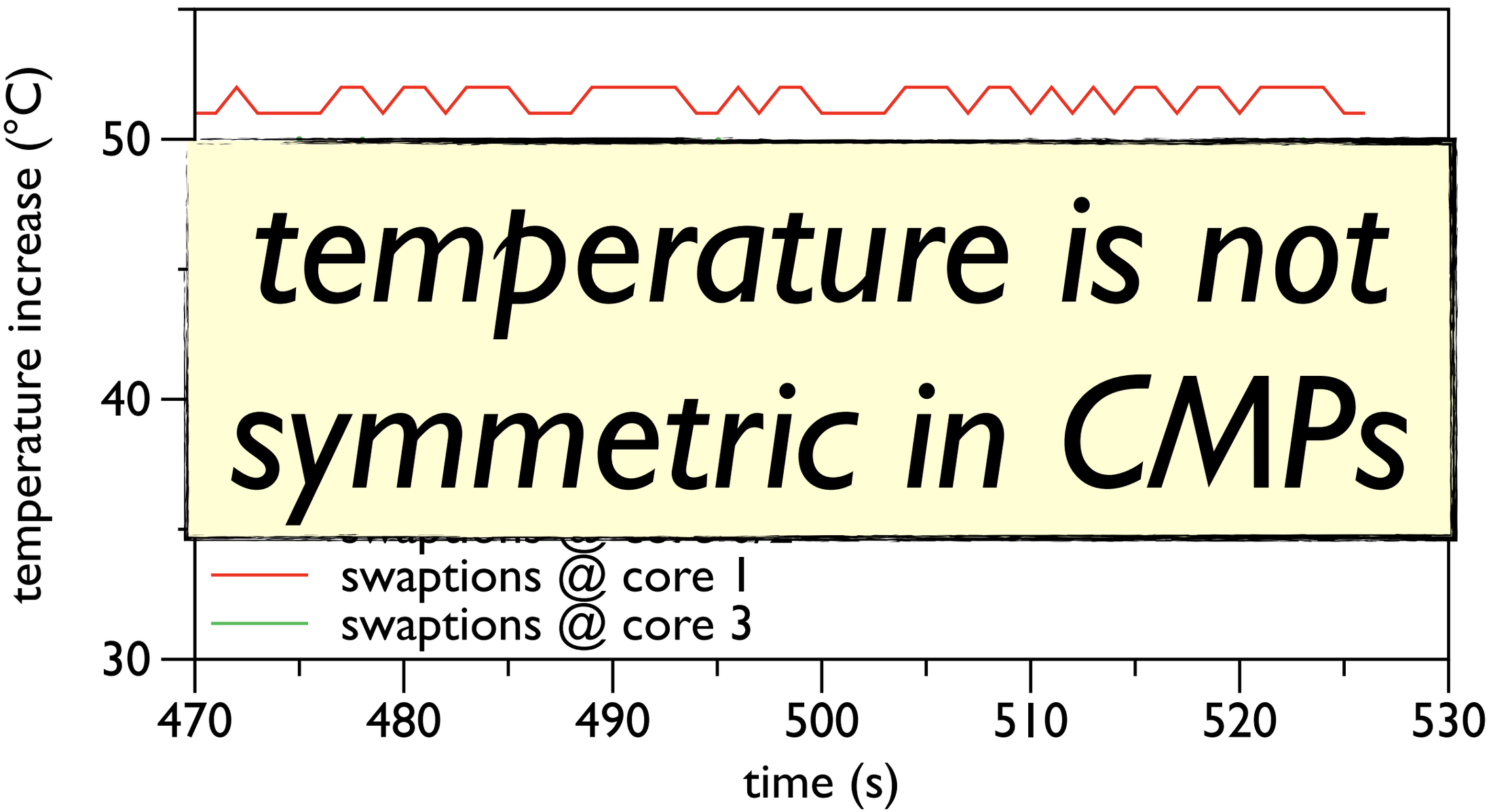
Evaluation platform

- 4-core Intel Xeon (Nehalem)
- From 1.60 GHz to 2.8 GHz
- C0, C1E (3 us latency), C3 (20 us latency plus other overheads), and C6 (200 us latency plus many other overheads) C-states
- Ambient temperature about (20 Celsius plus/minus 1)
- Idle temperature about (28-32 Celsius plus/minus 1 depending on the core)
- Modified Linux kernel 3.4
- PARSEC 2.1 benchmark

Thermal profile



Thermal profile

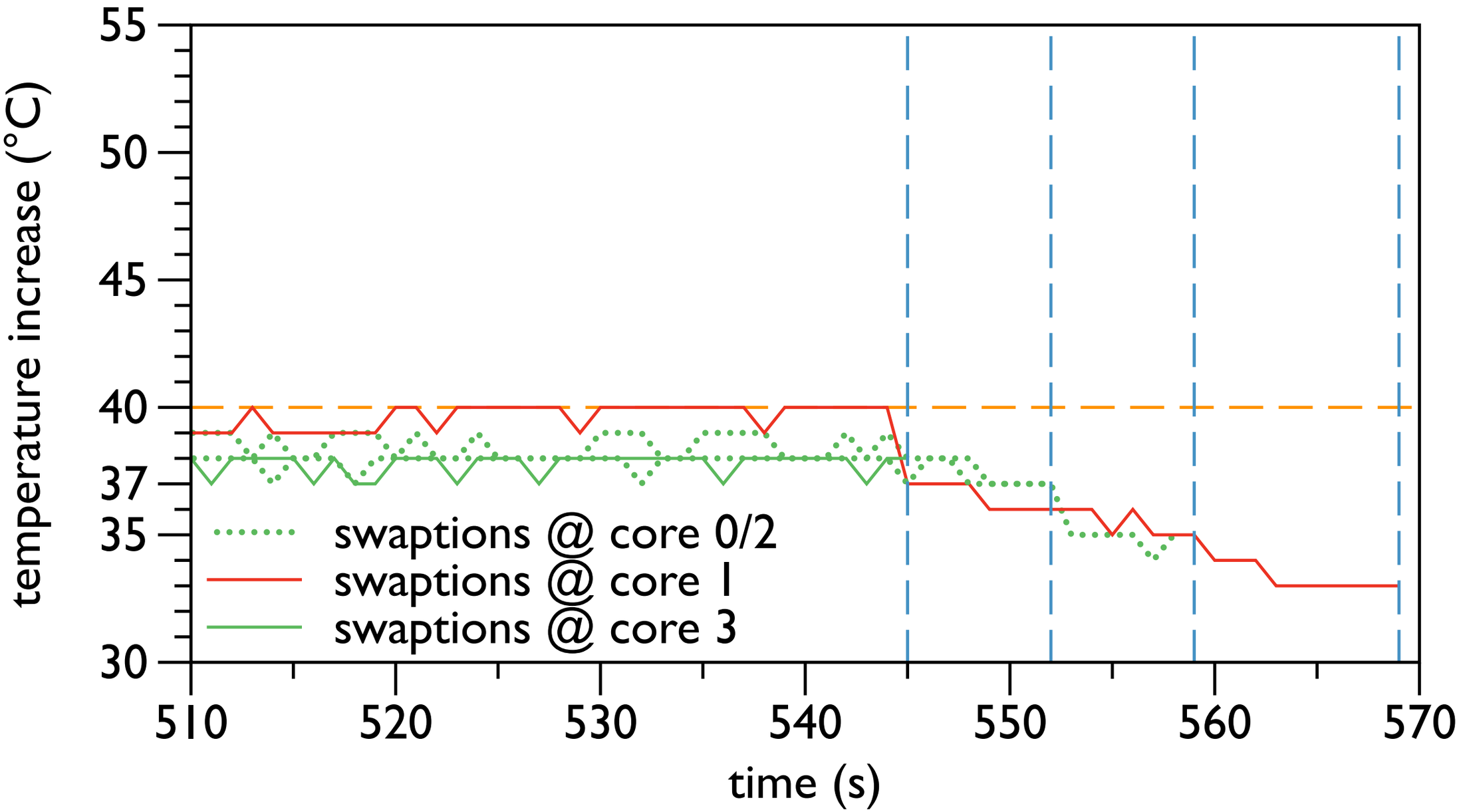


temperature is not symmetric in CMPs

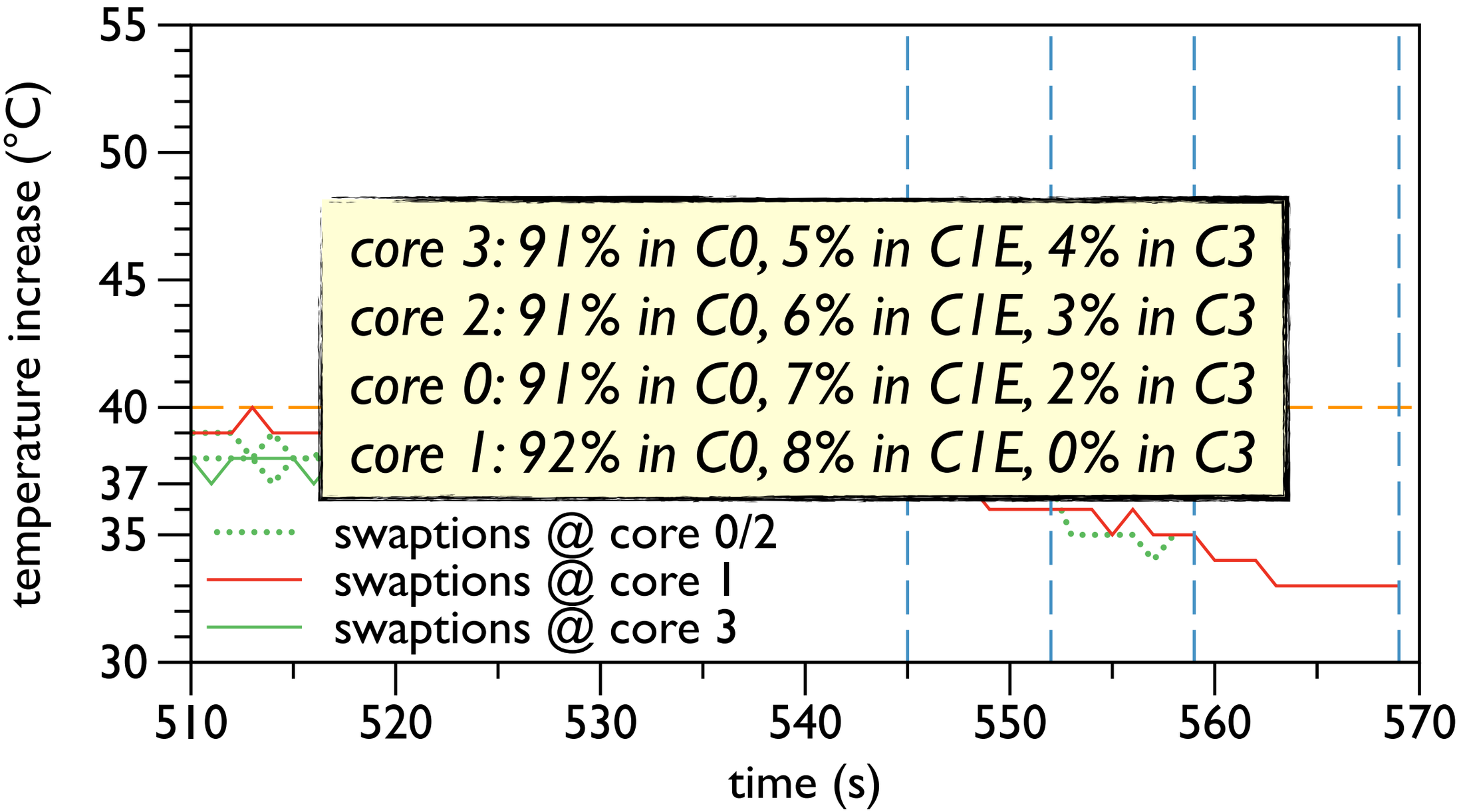
Research questions

- Can ThermOS constraint the temperature and selectively affect applications in a multi-programmed workload?
- How much ThermOS is efficient w.r.t. *state of the art* solutions?

Management of multi-programmed workloads



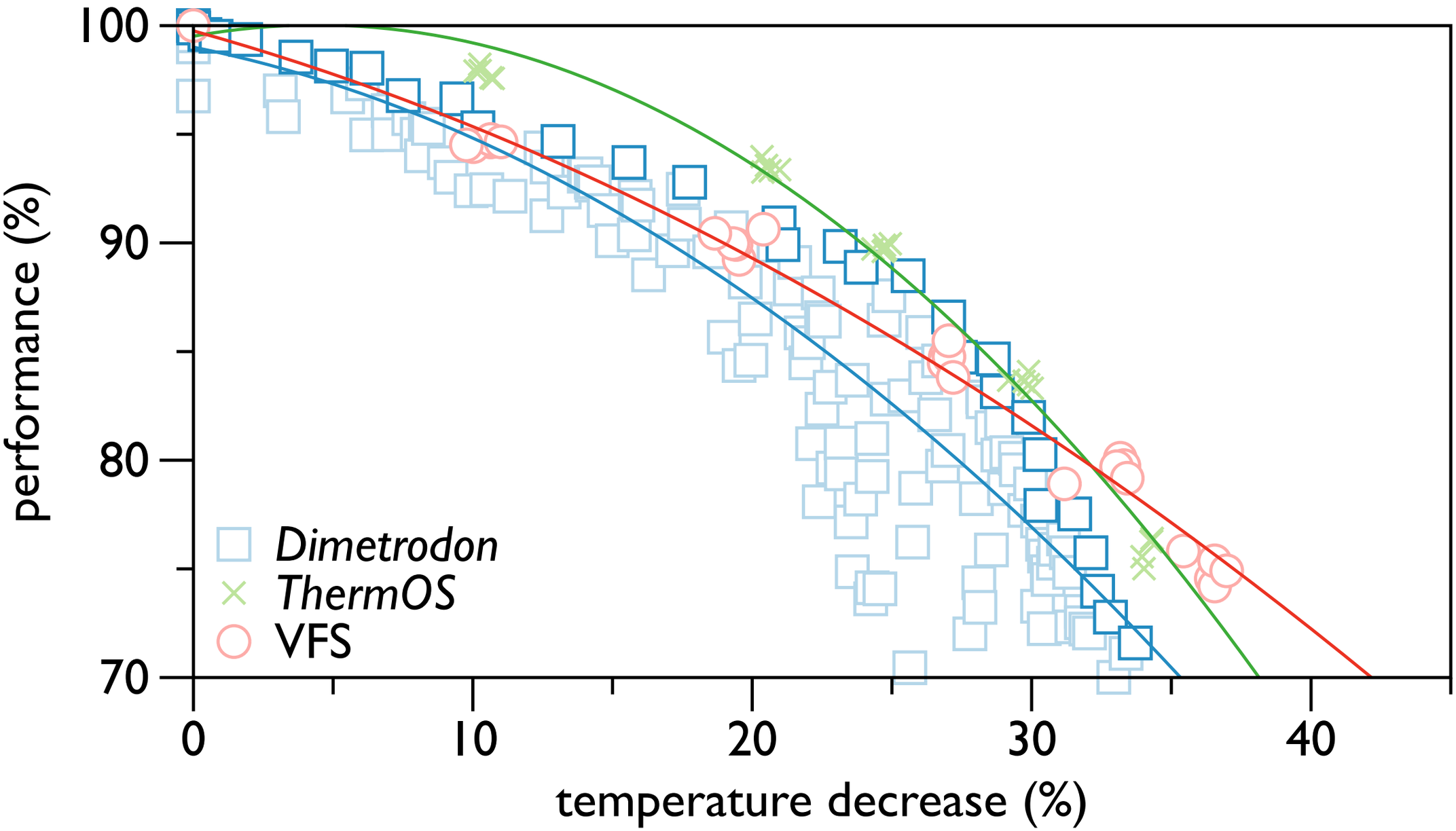
Management of multi-programmed workloads



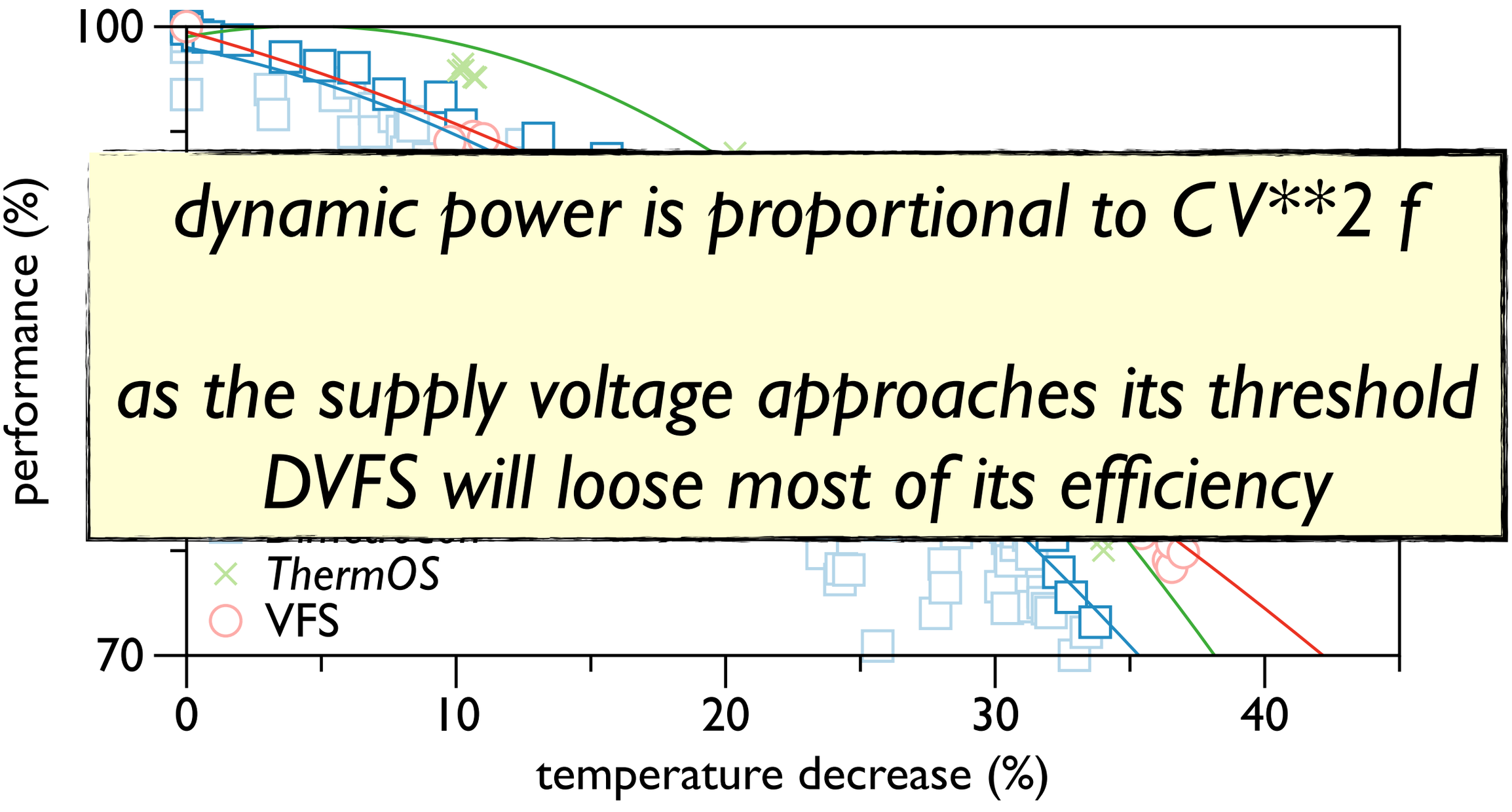
State of the art solutions

- Dimetrodon (Bailis et al. in DAC'11 [4])
  - Probabilistic feedforward control inside the FreeBSD 7.2 task scheduler
  - We swipe the idle quantum/probability configuration space
- VFS
  - We statically select the following frequencies (and the associated voltages): 2.79, 2.66, 2.53, 2.39, 2.26, 2.13 GHz

Efficiency with multi-programmed workloads



Efficiency with multi-programmed workloads



*dynamic power is proportional to CV**2 f*

as the supply voltage approaches its threshold DVFS will loose most of its efficiency

Related work

- Architectural solutions like clock/power gating, NTV/STV designs, conservation cores (Venkatesh et al. in ASPLOS'10 [23])
- Micro-architectural solutions like instruction fetch toggling (Brooks and Martonosi in HPCA'01 [14], Skadron et al. in TACO'04 [1]), instruction issue width resizing (Jayaseelan and Mitra in DAC'09 [25]), activity migration (Heo et al. in ISLPED'03 [24])

Related work
Software solutions

- Orthogonal approaches (thermal-aware scheduling)
    - Workload placement in data centers (Moore et al. in ATC'05 [26])
    - Heat and Run (Powell et al. in ASPLOS'04 [20])
    - ThreshHot (Zhou et al. in TACO'10 [15])
- Similar approaches
    - HybDTM (Kumar et al. in DAC'06 [27])
    - kidled (Google) and PowerClamp (Intel)
    - Dimetrodon (Bailis et al. in DAC '11 [4])

Conclusions and Future work

- ThermOS positively answers our research questions
- The changes to Linux 3.4 will be available soon

Questions... a few suggestions

- Are there overheads beside the injection of idle cycles?
- Your model and controller are conservative, can you extend ThermOS so as to precisely tailor its behavior to workload?
- How do you plan on supporting multi-threaded applications? And what about SMT core?
- Can you comment on energy efficiency?