Machine Learning Methods for Performance/Power/Thermal Optimization of Signal Processing Systems

Marilyn Wolf, Georgia Tech

Shuvra Bhattacharyya, University of Maryland

Tutorial Presentation at the IEEE Workshop on Signal Processing Systems

Cape Town, South Africa

October 21, 2018

Outline

- Non-functional design parameters and system-level methodologies
  - Modeling complexity
  - System design space
- Modeling techniques:
  - WCET
  - Thermal
  - Reliability
- Statistical models for non-functional parameters:
  - Markov decision processes
  - Solution algorithms

Complex application requirements

- Real-time (soft and hard) performance.
  - Throughput and latency.
- Power consumption.
- Thermal performance.
- Complex functionality.

Causes of modeling complexity

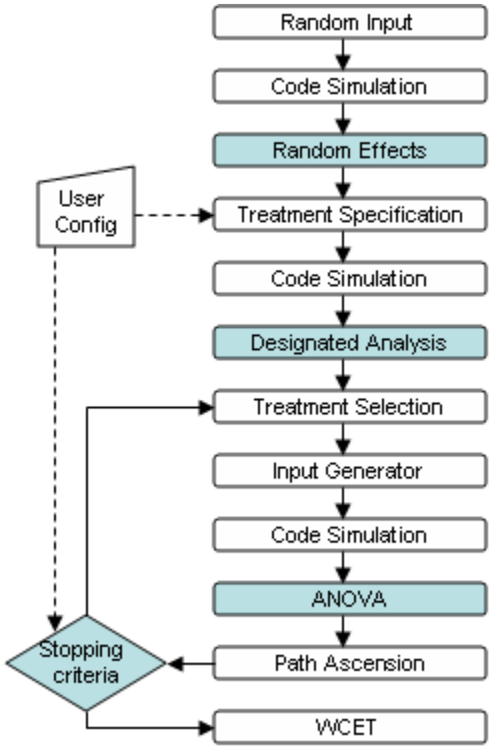
- Complex architecture, microarchitecture, logic design.
- Complex physics and multi-physics.
- Process variation.
- Aging-induced variation.

Software performance analysis

- Worst-case execution time (WCET):
  - Worst-case under any inputs or system state.
- Sometimes interested in best-case execution time.
- Software performance analysis:
  - Execution time = f(program path, path timing).
- Path timing is hard:
  - Pipelining.
  - Caches for single process.
  - Multi-tasking cache behavior.

Iterative DOE Methodology

- Initially include all inputs
- Random effects analysis to screen for significant inputs
- Create factors to treat the inputs
- Test these for significance
- Begin iterative process
  - Apply factors to inputs to generate test cases
  - Simulate set
  - Evaluate statistical trends to seek increasing times
  - Test stopping criteria, and repeat



Experimental Setup

- Matlab used to develop analytical tools, and control entire process
- SimpleScalar used for simulation
- Perl scripts parse output, and collect data

- Initially tested on well analyzed sorting algorithms
  - Heap, quick, insert
- Inputs: fixed size arrays

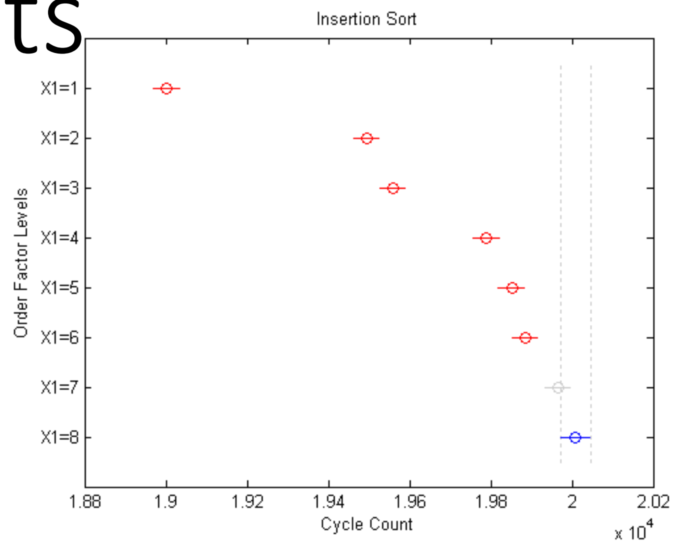
Initial Random Effects Analysis

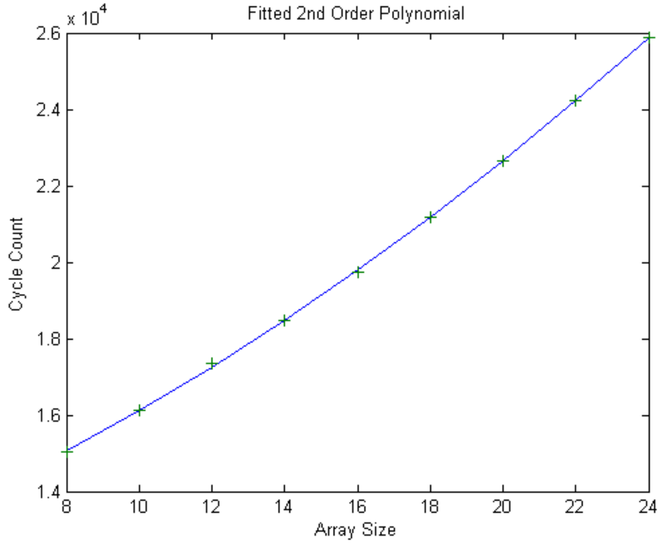
Source	Sum Sq.	d.f	Mean Sq.	F	Prob>F
Insertion	6987112	63	110906.5	48.26	0
Error	294179.3	128	2298.3		
Total	7281292	191			
Heap	1456967	63	23126.5	12.6	0
Error	234969.3	128	1835.7		
Total	1691936	191			
Quick	2074438	63	32927.6	17.42	0
Error	241889.3	128	1889.8		
Total	2316327	191			

Treatment Factors Random Effects Analysis

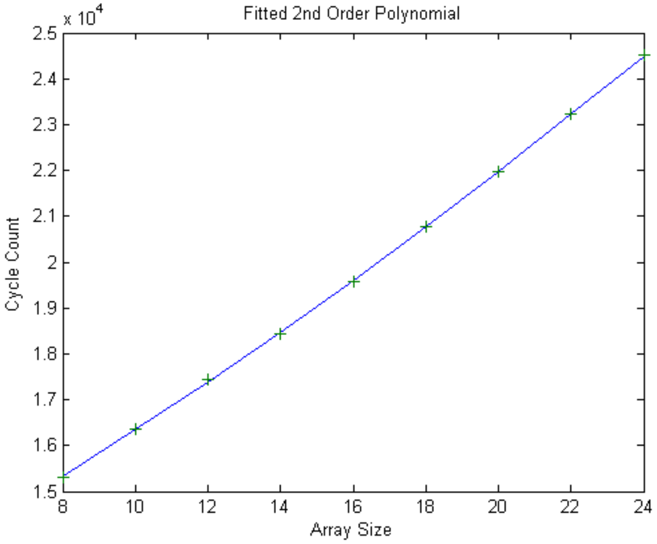
Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F	
order	161049.8	4	40262.4	19.06	0.0072	Insertion
scale	1428.3	1	1428.3	0.68	0.4571	
order*scale	8447.5	4	2111.9	1.2	0.3401	
error	35091.3	20	1754.6			
total	206017	29				
order	8611.1	4	21652.8	11.43	0.0184	Heap
scale	1241.6	1	1241.6	0.66	0.4636	
order*scale	7577.5	4	1894.4	0.54	0.7048	
error	69544.7	20	3477.2			
total	164975	29				
order	33991.1	4	8497.78	19.51	0.0069	Quick
scale	43.2	1	43.2	0.1	0.7685	
order*scale	1741.8	4	435.45	0.27	0.8913	
error	31787.3	20	1589.37			
total	67563.5	29				

Results



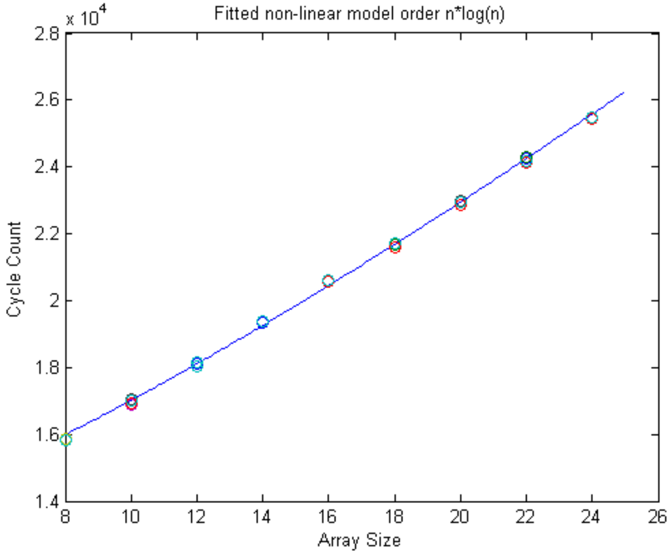


Fitted 2nd-order Polynomial for



Fitted 2nd-order Polynomial for
measured Quick WCETs

- Agree with proven analysis
- Scalable
- Tight bounds



Fitted *nlog*(*n*)-order Polynomial for

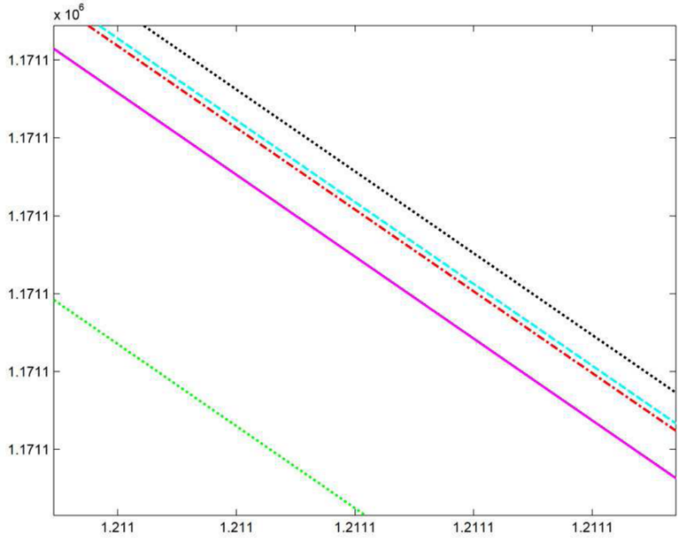
Further Experiments

Analysis of Variance

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
variance	55543902956988.484	254	218676783295.2	8.92201e+015	0
order	26694894.1875	4	6673723.5	2.72288e+011	0
variance*order	49851549.40625	1016	49066.5	2.00191e+009	0
Error	0.0625	2550	0		
Total	55543979503432.148	3824			

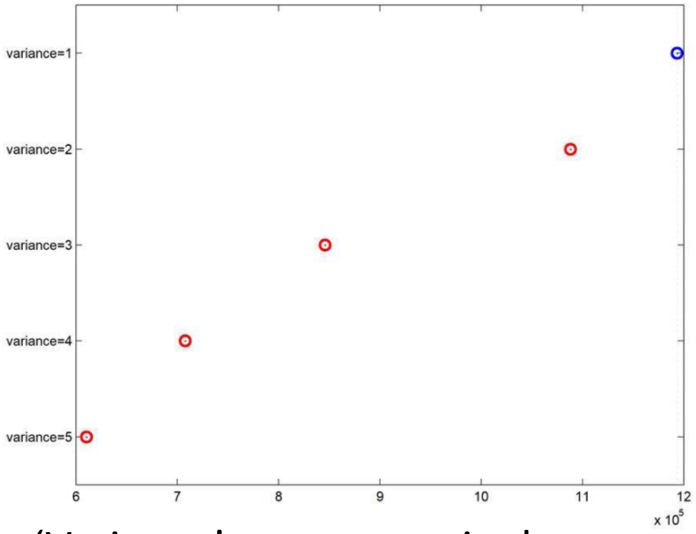
Constrained (Type III) sums of squares.

Huffman ANOVA

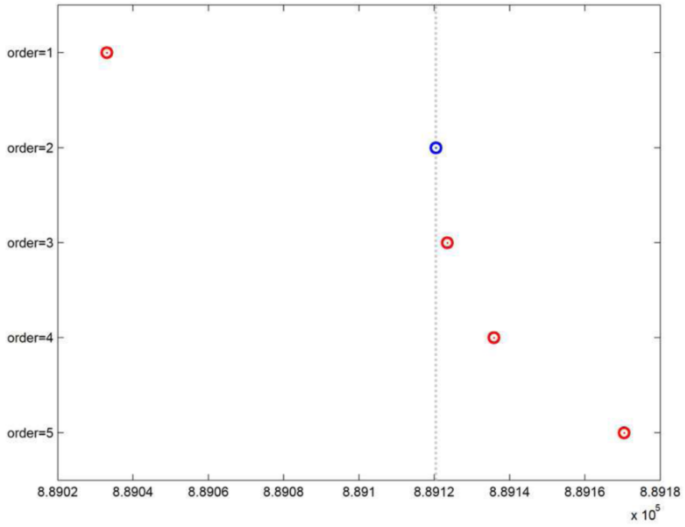


Huffman factor interactions

- Performed on Huffman code
- Dealt with significant cross-factor interactions



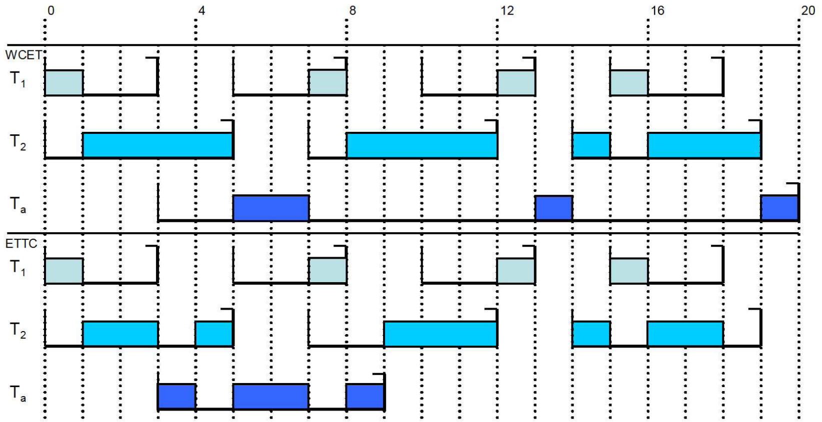
'Variance' group marginal means



'Order' group marginal means

Expected Time To Completion
Introduction

- ETTC proposes to estimate the expected remaining execution time for active tasks
- Potential benefits
  - Reclaim system resources
  - Accommodate more sporadic tasks
  - Schedule for power
  - Efficient partitioning



Expected Time To Completion
Approach

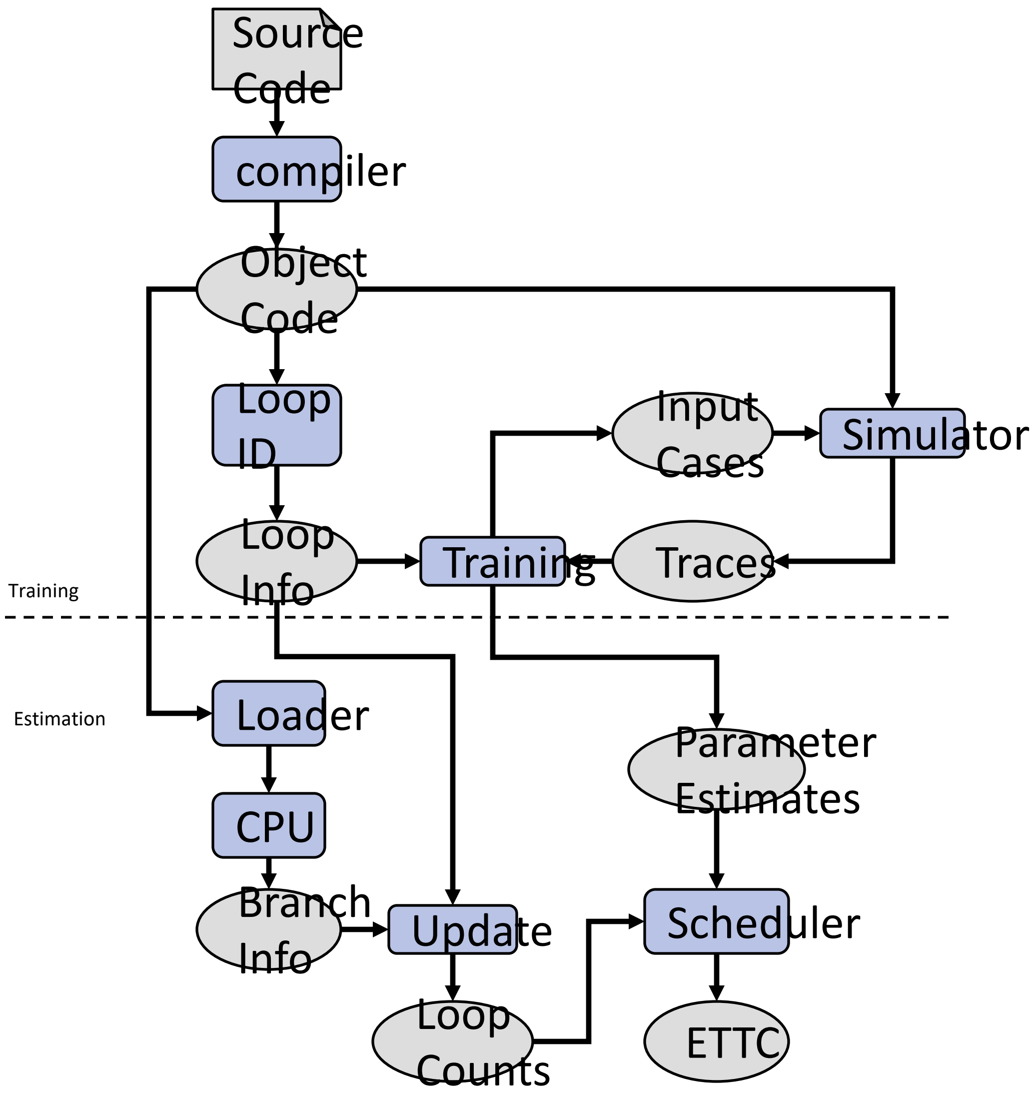
- Based on actual measurements from system
- Methods are platform independent
- Performance improves
- Can be carried out dynamically
- Employs the use of multivariate statistics

The Statistical Tools

- Factor Analysis
  - Interdependence technique
  - Simplifies links amongst separate factors
- Multivariate Discriminant Analysis
  - Intra-dependence technique
  - To simplify relationships amongst observations
  - Means of performing classification

ETTC Framework

- Consider applications to be advancing through series of states
  - States not defined, unobservable
  - Need observable events
- Use loop counts as indicator for progress



Portion of Execution Time Spent in Loops

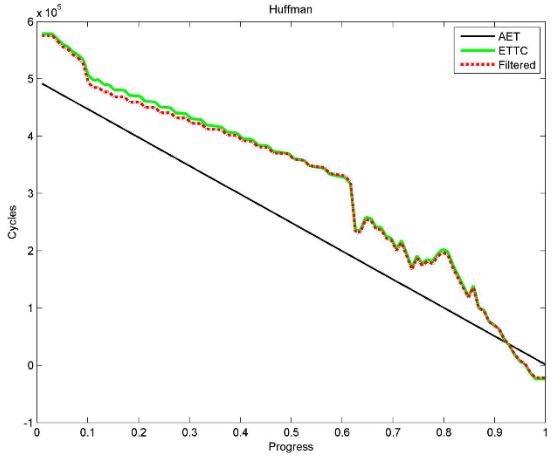
Benchmark	Percentage
Huffman	9[illegible].40%
FIR	95.70%
LZW	93.30%
Fourstep	83.70%

Computing the ETTC
Regression

- A set of loops $\Lambda_i = (\lambda_1, \lambda_2, \ldots, \lambda_L)$ is identified for each task T_i, with L loops
- During training each observation at time t consists of
  - Number of iterations for λ_j, for $j=1,\ldots,L$
  - Actual time remaining until completion, τ_t
- Coefficients of $Y_t = \beta_0 + \beta_1 X_{1,t} + \ldots + \beta_L X_{L,t}$, are then calculated, where Y_t is time remaining, and $X_{i,t}$ is count for loop i, all at time t.
- The β_i are then used in prediction

Computing the ETTC

Regression



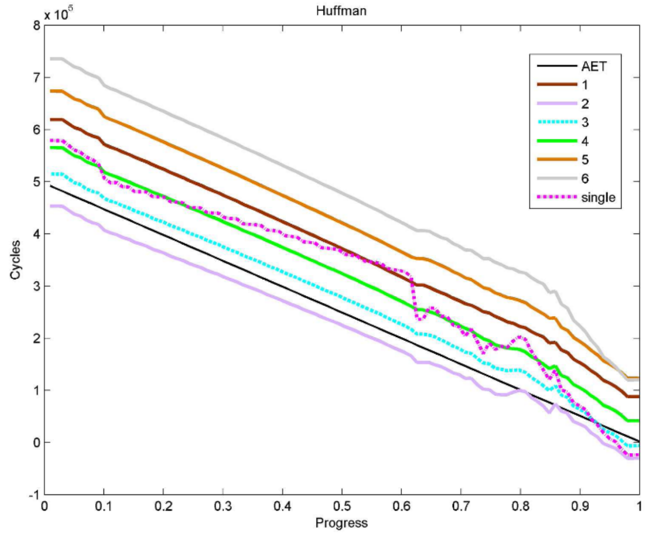
- Data possibly too varied
- Consider execution scenarios
  - Cluster training data into S clusters based on total execution time
  - Now have S separate models,

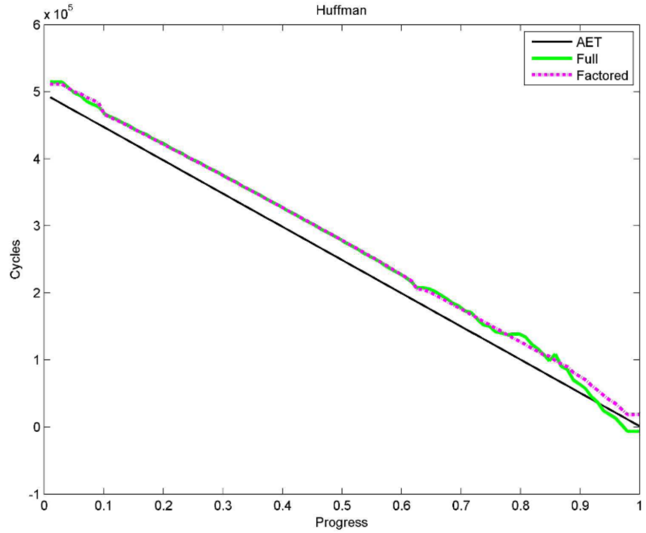
$$Y_{t,s} = \beta_{0,s} + \beta_{1,s} X_{1,t}^{(s)} + \ldots + \beta_{L,s} X_{L,t}^{(s)}$$

- Select proper scenario with following heuristic
  - Scheduler runs at fixed time intervals, Φ
  - For (t_i, t_{i+1}), let $\delta_s = \mathrm{Y}_{t_{i,s}} - \mathrm{Y}_{t_{i+1,s}}$, for $s = 1, \ldots, S$
  - Select scenario s which satisfies $min(\Phi - \delta_s)$

Computing the ETTC
Regression

- Scenarios improve the prediction
- Employ factor analysis to uncover latent loop dependencies
  - Reduces memory storage demands (here by 70%)
  - No performance penalty
  - Removes under prediction near end





Computing the ETTC

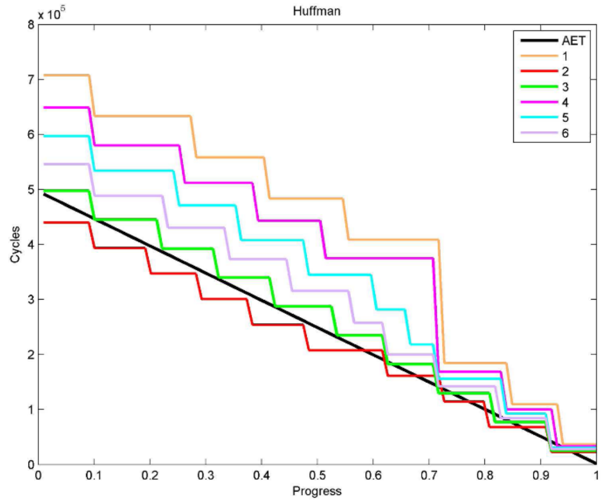
Discriminant Analysis

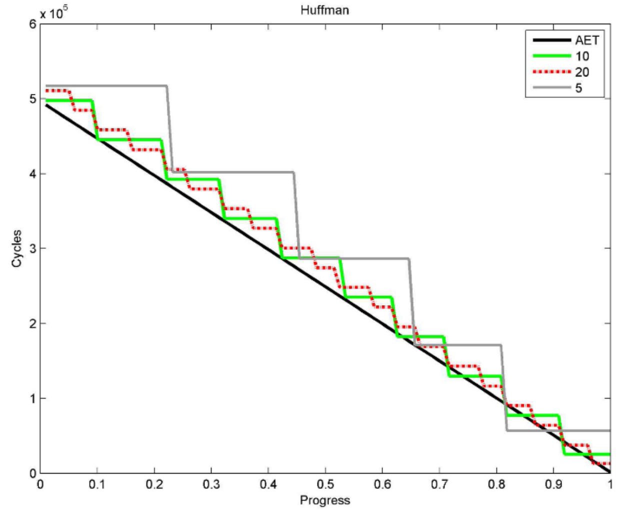
- Using same observations as prior method
- Create database by grouping similar observations together
- Associate each group with an AET
- For training data, with G groups, each run
  - Has N observations
  - These are divided into G equal sized groups
  - Successive observations grouped together
  - Multivariate normal distributions are fit to each group
- In making an ETTC prediction,
  - Observations are compared to all groups
  - Classified with best fitting group, and given the associated ETTC

Computing the ETTC

Discriminant Analysis

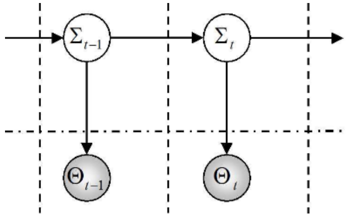
- No need for scenario heuristic
- Predictions are discrete
- Can add groups
    - Reduces quantization
    - Slower, more memory
    - Does not guarantee better performance





Computing the ETTC

Kalman Filter

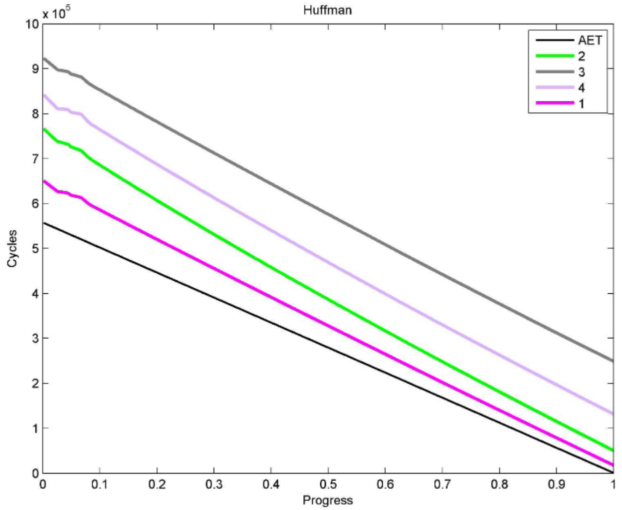


- Want to deduce hidden state of task from observable loop counts,
- Allows for continuous-time inputs, and continuous-valued hidden states,
- current state depends on previous state $\Sigma_t = A\Sigma_{t-1} + w_{t-1}$, A is state transition matrix, w_{t-1} is process noise $\sim N(0, Q_t)$
- To measure current state, $\Theta_t = H_t \Sigma_t + v_t$
- Initial conditions are critical, hence segmentation is necessary

Results

	Regression			Segmented Regression			Discriminant Analysis			Kalman Filter		
Process	¼	½	¾	¼	½	¾	¼	½	¾	¼	½	¾
Huff	14.60%	24.90%	35.30%	3.30%	4.60%	8.60%	2.30%	1.90%	1.70%	22.80%	20.60%	22.40%
FIR	2.80%	2.50%	1.70%	0.52%	0.45%	0.38%	0.64%	0.58%	0.65%	10.40%	13.40%	22.50%
LZW	14.70%	18.50%	25.20%	3.80%	5.60%	10.30%	3.60%	3.90%	8.30%	18.70%	20.40%	29.40%
4step	9.50%	13.20%	25.20%	1.30%	1.70%	3.20%	1.70%	0.69%	0.93%	13.70%	12.40%	16.10%

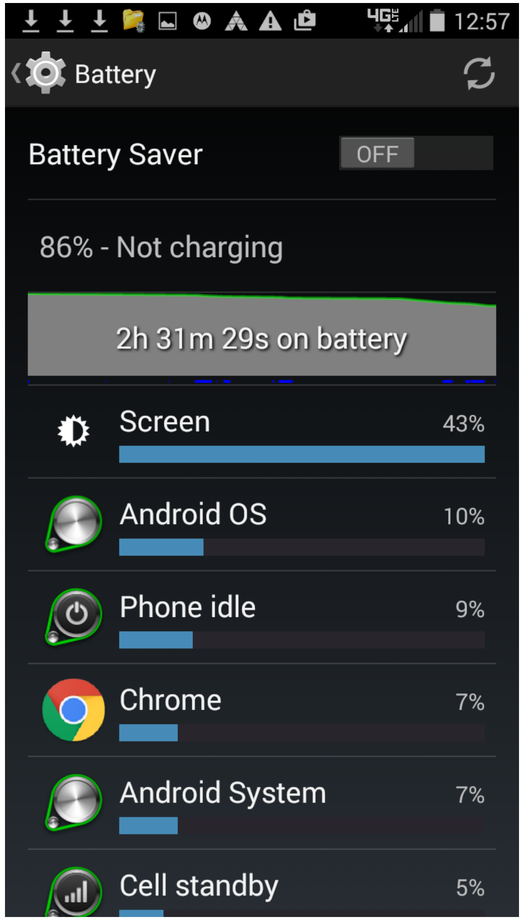
- Scenarios are crucial
- D.A. accurate, but expensive/slower
- KF has potential to perform better in complex heavily loaded systems, with noisy observation



Computer system energy consumption

Component	Power (W)
CPU	100-200
Memory	25
Disk	10-15
Board	40-50
Power/fans	30-40
Total	200-350

server



smartphone

Current requirements

- Intel Xeon E7-8800:
  - $I_{CC\_MAX} = 120\ A$
  - Operating voltage of 1.3V
- Craftsman Arc Welder:
  - $I = 60\ A$
  - Operating voltage of 120V.

Welder draws higher power but CPU has impressively high current density.

Heat transfer mechanisms

- Conduction.
  - Molecular motion.
- Radiation.
  - Electromagnetic energy.
- Convection.
  - Bulk fluid motion.

Heat carried through a solid.

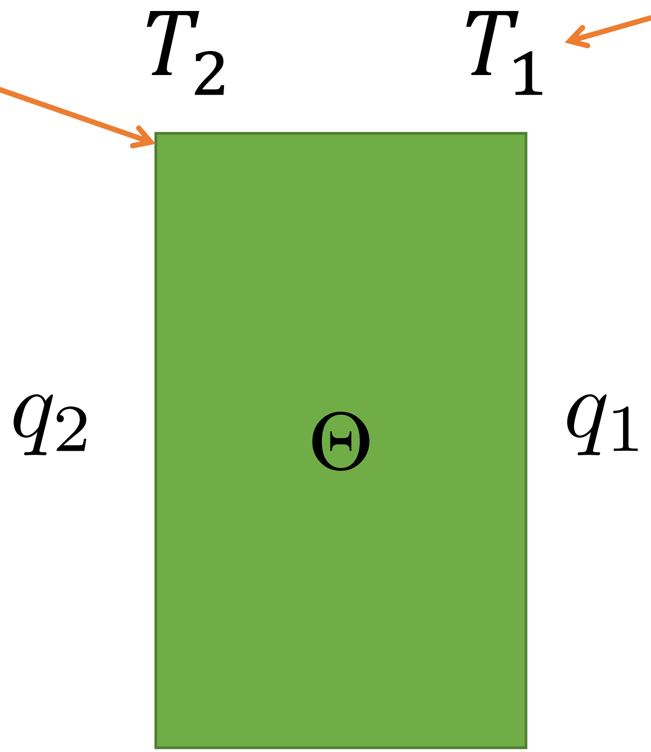
Can be transmitted in a vacuum.

Air or water flow.

Θ: thermal resistance

T: temperature

$$\theta = \frac{T_2 - T_1}{q_2}$$



q: heat flow

Key thermal ratings

- Transistor junctions must be kept below maximum junction temperature:
  - $T_{J,max} = 85°\mathrm{C}$
  - Temperature at which heat damages the transistor structures.
- Chip specifies thermal design power (TDP).
  - Amount of operating heat that its cooling system must be able to dissipate.

Thermal drives performance

TDP $\Rightarrow$ P $\Rightarrow$ f

Physical properties

- Specific heat:
  - Relationship between heat input/output and temperature.
  - Measured in Joules/kilogram-Kelvin.
- Thermal conductivity:
  - Relationship between temperature difference and heat flow per unit time.
  - Measured in Watts/meter-Kelvin.

material	specific heat (J/kg K)	thermal conductivity (W/m K)	density (kg/m^3)
silicon	710	149	2.3×10^{-3}
ceramic (aluminum nitride)	740	150	3.3×10^{-3}
carbon steel	620	41	7.9×10^{-3}

Thermal properties of objects

- Thermal resistance R:
  - Thermal conductivity for a specific shape and size of material.
  - $R = \frac{l}{kA}$, length l, area A, thermal conductivity k.
- Thermal capacitance C:
  - Specific heat for a specific shape and size of material.
  - $C = mC_m$, mass m, specific heat C_m.

Thermal/electrical analogy

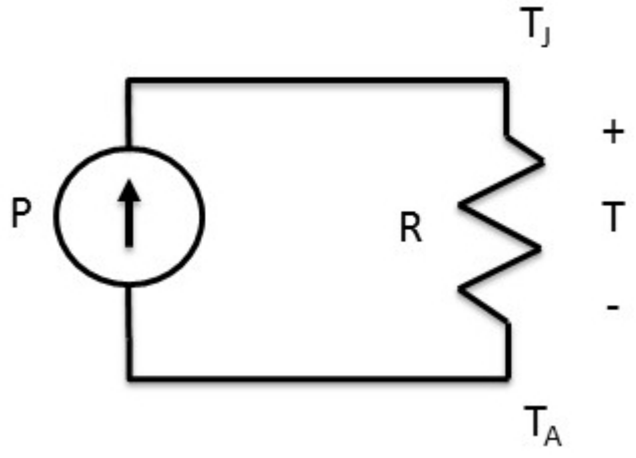
electrical	thermal
charge Q	thermal energy Q
current I	heat flow P
voltage V	temperature T
resistance R	thermal resistance R
capacitance C	thermal capacitance C

Physical laws of thermal behavior

- Fourier's Law of Heat Conduction:
  - $T = PR$
- Newton's Law of Cooling:
  - $P = C\frac{dT}{dt}$

Steady-state temperature

- Use thermal resistance to determine steady-state temperature.
  - Calculate temperature difference from ambient to junctions.
- Thermal circuit has a heat source P, thermal resistance R.
- Output temperature is measured across thermal resistance.
  - $T_J = T_A + PR$



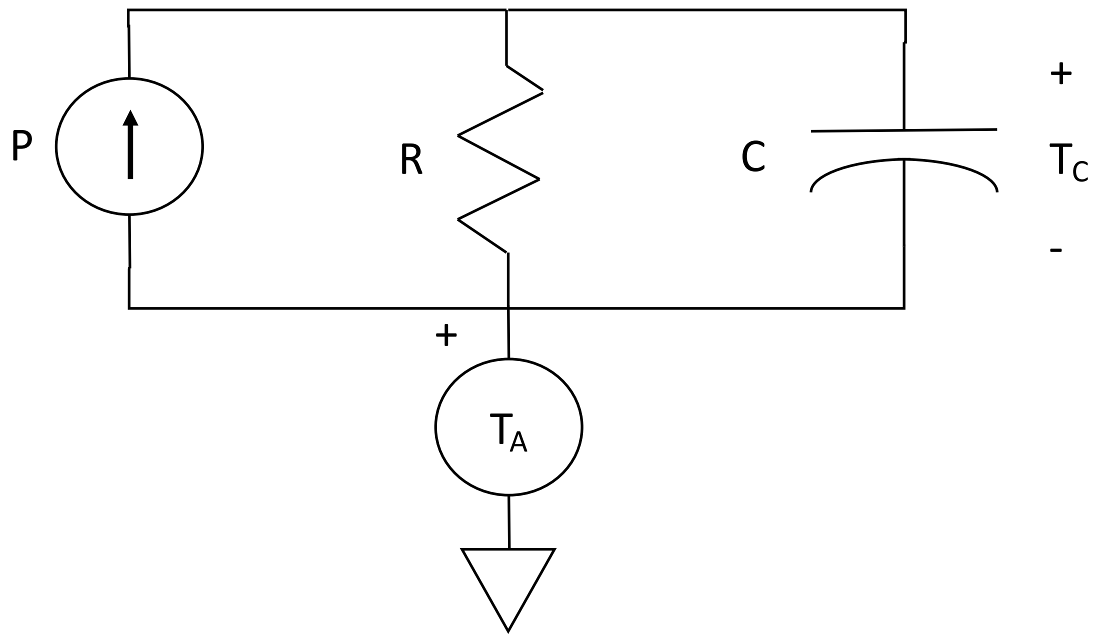
Example: heat sink performance

- Computer power $P = 20\ W$.
- Ambient temperature 20°C.

- Case 1---no heat sink:
  - $T_{none} = 20 + 20W \cdot 10\frac{℃}{W} = 220℃$
- Case 2---heat sink:
  - $T_{sink} = 20 + 20W \cdot 1.5\frac{℃}{W} = 50℃$

Thermal transient analysis

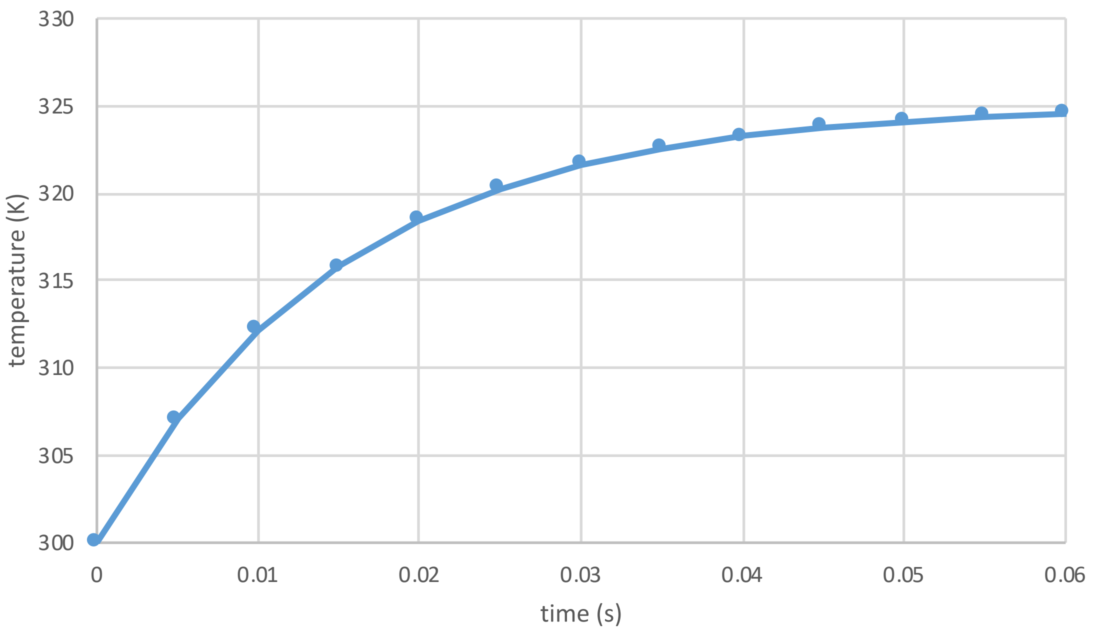
- Chip and heat sink form a thermal RC circuit.
  - Measure chip temperature relative to ambient.
- Temperature as a function of time:
  - $T(t) = (T_0 - PR)e^{-t/RC} + PR + T_A$
  - Temperature above ambient at $t = \infty$ is PR.



Example: thermal RC model of chip temperature

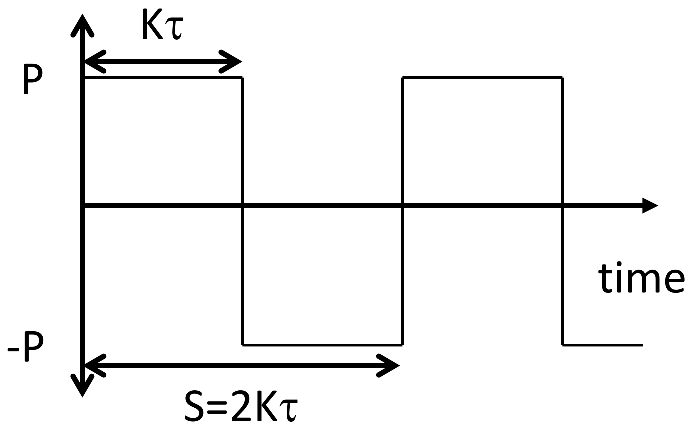
parameter	value
R	$0.5\ K/W$
C	$0.03\ J/K$
P	$50\ W$
T_0	$0\ K$
T_A	$300\ K$

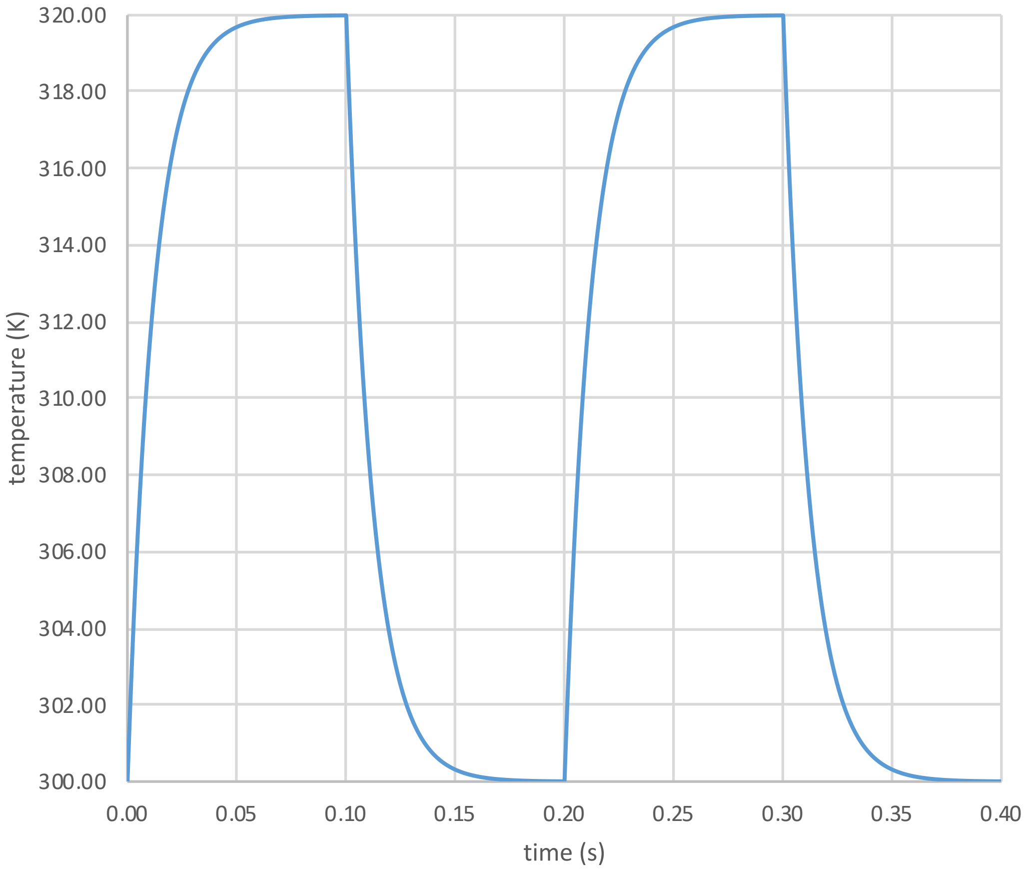
- $T(t) = 325 - 25e^{-\frac{t}{0.015}}$
- Thermal time constant $0.015\ sec$.
- Steady-state temperature $325\ K$.



Thermal square wave

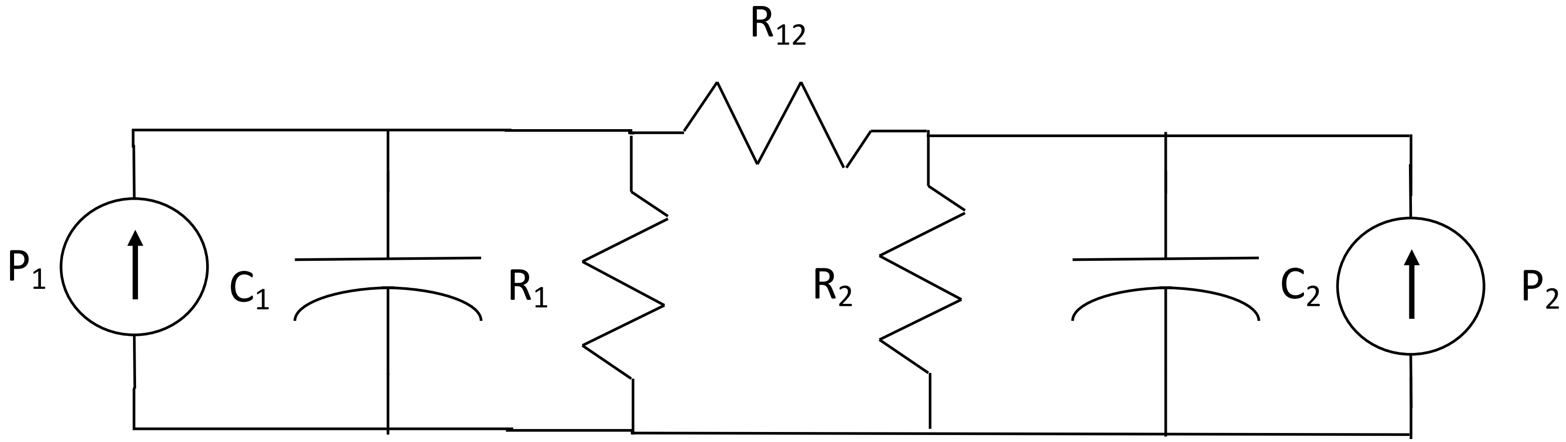
- Chip runs periodically:





Dual-core processor

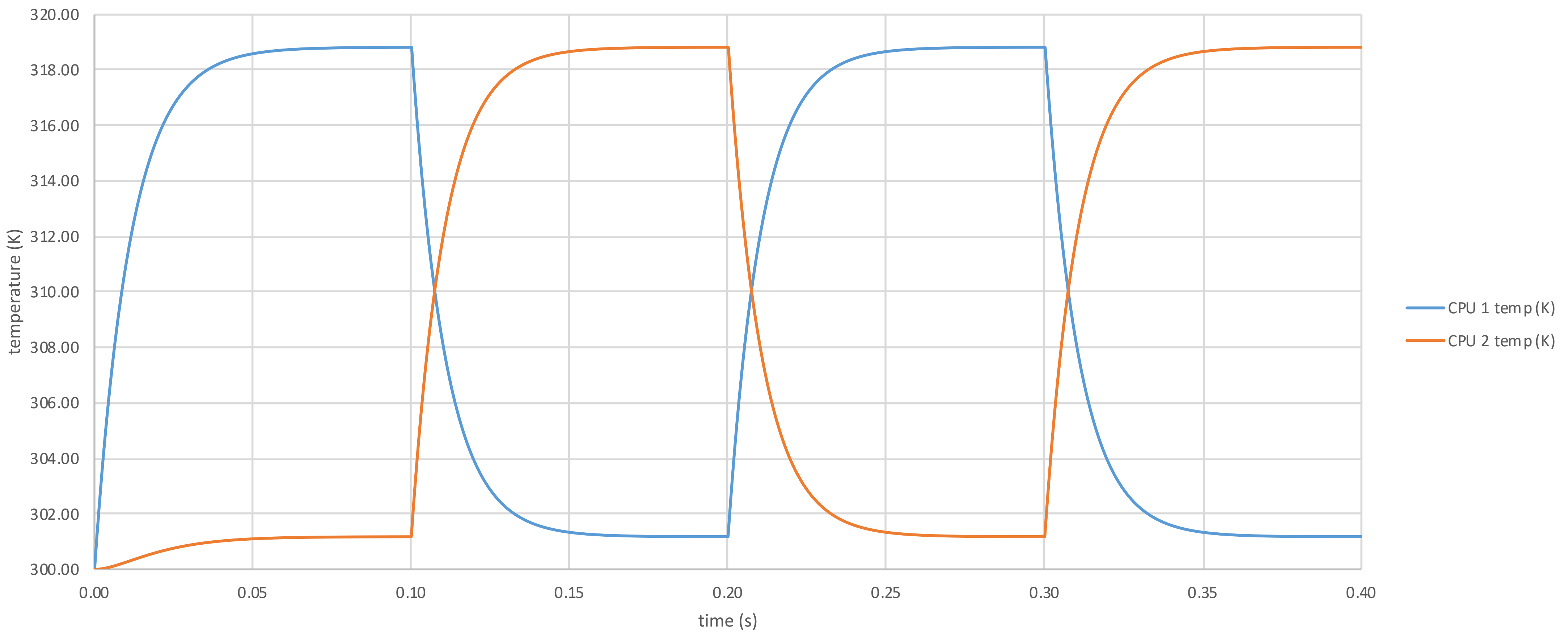
- Two processors alternate between run, stop.
- Cores are connected by thermal resistance.



Dual-core thermal analysis

- Can borrow a result from electrical circuits:
    - Assume 50% duty cycle, period $S = 2K\tau$.
- Upward and downward temperature waveforms:
    - $T^u(t + t_0) = \left(-T_p - P\right)e^{-t/RC} + P$
    - $T^d(t + t_0) = \left(T_p + P\right)e^{-t/RC} - P$
- Temperature cycles between $T_p, -T_p$:
    - $T_p = \left(-T_p - P\right)e^{-K} + P = \left(T_p + P\right)e^{-K} - P$
- So $\frac{T_p}{P} = \frac{1-e^{-K}}{1+e^{-K}}$

Dual-core thermal behavior



Heat and reliability

- Heat contributes to aging.
  - Higher temperatures cause chips to fail earlier.
- Arrhenius' equation describes the relationship between energy and the rate of physical processes:
  - $r = Ae^{-E_a/kT}$
  - Activation energy E_a is determined by energy required to promote electrons to high orbits.
  - Arrhenius prefactor A is measured experimentally.

Electromigration

- Electromigration is a common temperature-related failure mechanism.
  - Heat causes some molecules in wire to release free atoms.
  - Current flowing through wire causes free atoms to move.
  - Destructive feedback---thinner wire segments heat more, causing more rapid failure.
- Failure rate modeled using Black's equation:
  - $MTTF = AJ^{-n}e^{E_a/kT}, 1 \leq n \leq 3.$

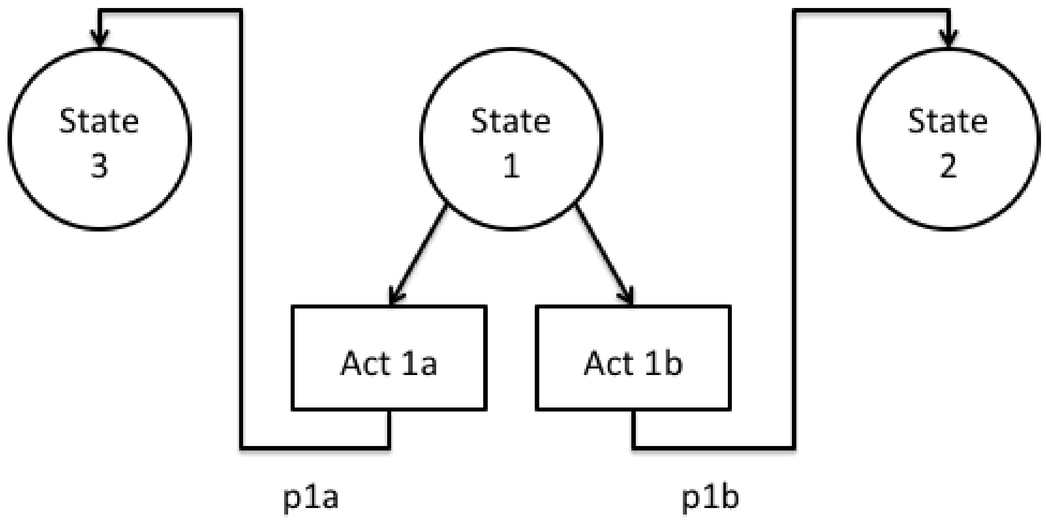
Lifetime analysis

- Chip temperature often varies over time based on use case and computing activity.
- We can model aging as a function of temperature:
    - $R(t) = \frac{1}{kT(t)} e^{-E_a/kT(t)}$
    - $\varphi_{th} = \int_0^t \frac{1}{kT(t)} e^{-E_a/kT(t)}$
- Chip-level reliability engineering:
    - Minimize hot spots.
    - Use operating system to spread workload across cores.

Thermal management

- A combination of hardware and software is used to manage thermal behavior.
- On-chip temperature measured using band gap reference circuit.
- Processor may provide a software interface to on-chip temperature sensors.
    - Intel Thermal Monitor 1 turns the clocks off and on at a duty cycle chosen for the processor type, typically 30%-50%.
    - Intel Thermal Monitor 2 uses dynamic voltage and frequency scaling mechanisms to reduce both the clock speed and power supply voltage of the processor.

Markov decision processes



- Probabilistic transitions combined with inputs.
  - Given an input at a state, next state is chosen probabilistically.
- A policy π defines the actions in each state *s*.
  - Optimal policy maximizes rewards.

MDP model

- States S.
- Actions S.
- Probability that action a in state s gives transition to s’ $P_a(s, s')$.
- Reward for action$R_a(s, s')$.
- Discount factor γ.

- Find policy π to maximize time-discounted reward:
  - $\sum_{t \geq 0} \gamma^t R(s_t, s_{t+1})$

Value iteration

- $V_{i+1}(s) = \max_a [\sum_{s'} P_a(s, s')\{R_a(s, s') + \gamma V_i(s, s')\}]$
- Value at each step is maximum over all possible actions.
- Iterate until converged.

Policy iteration

1. Find policy $\pi(s) = \arg max_a \, P_a(s, s')\{R_a(s, s') + \gamma V_i(s, s')\}$
2. Iterate until converged $V \quad (s) = \sum_{s'} P_{\pi(s)} \, (s, s')\left\{R_{\pi(s)}(s, s') + \gamma V_i(s, s')\right\}$
3. Repeat 1-2 until converged.

Reinforcement learning

- Identify transition probabilities using random search.