# CSCI 1300

# Artificial Intelligence Lecture

Mike Mozer

December 4, 2003

# Computer Science

**Operating Systems**

**Programming Languages**

**Networking**

**Security**

**Theory**

**Artificial Intelligence**

# Artificial Intelligence

Natural Language Understanding

Speech Recognition

Computer Vision

Robotics

Reasoning

Planning

Machine Learning

# Machine Learning

## Supervised Learning

- spam filters (hotmail.com)
- ALVINN (autonomous vehicle navigation)

## Unsupervised Learning

- collaborative filtering (amazon.com)
- fault monitoring

## Reinforcement Learning

- td-gammon (champion backgammon playing program)
- elevator controller
- adaptive home lighting/heating control

# Reinforcement Learning: A Simple Example

**Suppose you are in one of two *states***

hungry

sleepy

**Suppose you can take one of two *actions***

go to Turley's

lie on bed

**Reward contingencies**

| | |
|---|---|
| hungry -> go to Turley's | reward |
| hungry -> lie on bed | no reward |
| sleepy -> go to Turley's | no reward |
| sleepy -> lie on bed | reward |

**Reward depends on what action you take in a given state.**

# Reinforcement Learning: A Simple Example

**How do you *learn* to take the correct action?**

**Trial and error!**

**Through *experience*, system can learn to predict the reward that will be obtained for some action given the current state:**

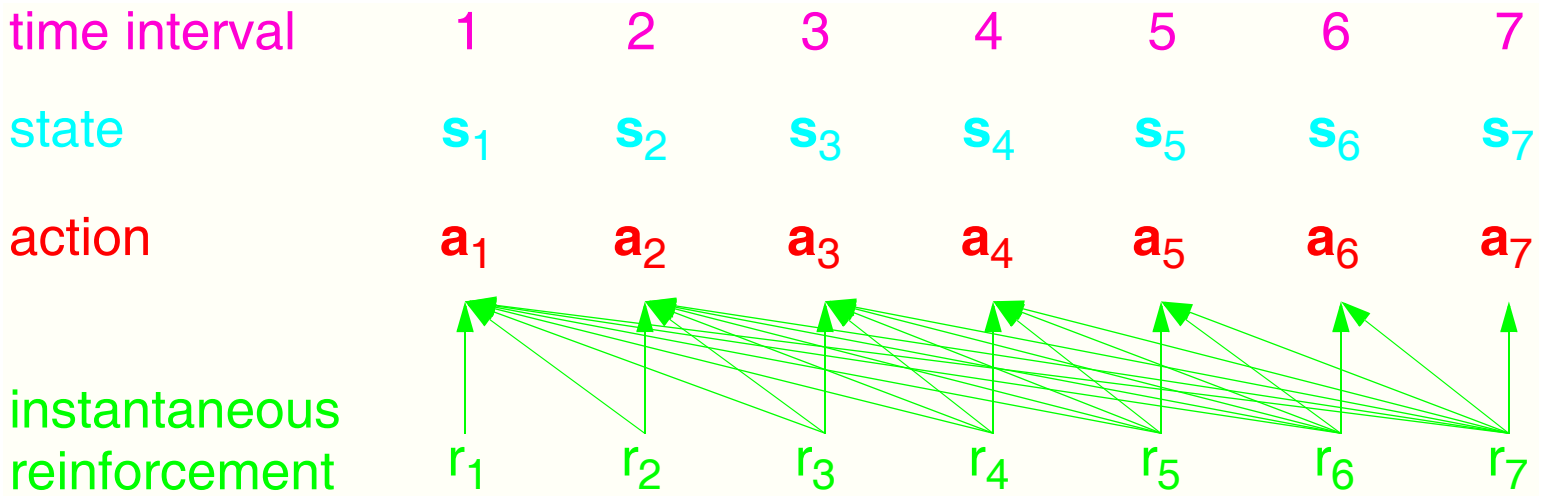
reward(action | state)

This is also notated as “Q(state, action)”

**Given the expected reward, agent can choose best action:**

if Q(hungry, Turley’s) > Q(hungry, lie on bed) then go to Turley’s
else lie on bed

# Reinforcement Learning in the Real World




## Issues

Delayed reinforcement (e.g., car accident due to worn tires)

Occasional reinforcement (e.g., chess playing)

Short term versus long term rewards (e.g., skipping class)

Exploration versus exploitation (e.g., trying new restaurants)

Partially observable state (e.g., viral infection)

Multiple agents (e.g., multiple elevators)

# Elevator Control

In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, eds., *Advances in Neural Information Processing Systems 8*. MIT Press, Cambridge MA, 1996.

## Improving Elevator Performance Using Reinforcement Learning

**Robert H. Crites**
Computer Science Department
University of Massachusetts
Amherst, MA 01003-4610
crites@cs.umass.edu

**Andrew G. Barto**
Computer Science Department
University of Massachusetts
Amherst, MA 01003-4610
barto@cs.umass.edu

### Abstract

This paper describes the application of reinforcement learning (RL) to the difficult real world problem of elevator dispatching. The elevator domain poses a combination of challenges not seen in most RL research to date. Elevator systems operate in continuous state spaces and in continuous time as discrete event dynamic systems. Their states are not fully observable and they are nonstationary due to changing passenger arrival rates. In addition, we use a team of RL agents, each of which is responsible for controlling one elevator car. The team receives a global reinforcement signal which appears noisy to each agent due to the effects of the actions of the other agents, the random nature of the arrivals and the incomplete observation of the state. In spite of these complications, we show results that in simulation surpass the best of the heuristic elevator control algorithms of which we are aware. These results demonstrate the power of RL on a very large scale stochastic dynamic optimization problem of practical utility.

# Elevator Control

| Algorithm | AvgWait | SquaredWait | SystemTime | Percent>60 secs |
|---|---|---|---|---|
| SECTOR | 27.3 | 1252 | 54.8 | 9.24 |
| DLB | 21.7 | 826 | 54.4 | 4.74 |
| BASIC HUFF | 22.0 | 756 | 51.1 | 3.46 |
| LQF | 21.9 | 732 | 50.7 | 2.87 |
| HUFF | 19.6 | 608 | 50.5 | 1.99 |
| ESA | 18.0 | 524 | 50.0 | 1.56 |
| FIM | 17.9 | 476 | 48.9 | 0.50 |
| RLp | 16.9 | 476 | 42.7 | 1.53 |
| RLd | 16.9 | 468 | 42.7 | 1.40 |

Table 3: Results for Down–Peak Profile with Up and Down Traffic

Table 4 shows the results for the down–peak traffic profile with up and down traffic, including an average of 4 up passengers per minute at the lobby. This time there is twice as much up traffic, and the RL agents generalize extremely well to this new situation.

| Algorithm | AvgWait | SquaredWait | SystemTime | Percent>60 secs |
|---|---|---|---|---|
| SECTOR | 30.3 | 1643 | 59.5 | 13.50 |
| HUFF | 22.8 | 884 | 55.3 | 5.10 |
| DLB | 22.6 | 880 | 55.8 | 5.18 |
| LQF | 23.5 | 877 | 53.5 | 4.92 |
| BASIC HUFF | 23.2 | 875 | 54.7 | 4.94 |
| FIM | 20.8 | 685 | 53.4 | 3.10 |
| ESA | 20.1 | 667 | 52.3 | 3.12 |
| RLd | 18.8 | 593 | 45.4 | 2.40 |
| RLp | 18.6 | 585 | 45.7 | 2.49 |

Table 4: Results for Down–Peak Profile with Twice as Much Up Traffic

# Q learning

(Watkins, 1989; Watkins & Dayan, 1992)

**Q(x,u): If action u is taken in state x, what is the minimum cost we can expect to obtain?**

**Policy based on Q values:**

exploration rate

$$\pi(\mathbf{x}_t) = \begin{cases} \text{argmin}_{\mathbf{u}} Q(\mathbf{x}_t, \mathbf{u}_t) & \text{with probability } (1-\theta) \\ \text{random} & \text{with probability } \theta \end{cases}$$

**Incremental update rule for Q values:**

$$Q(\mathbf{x}_t, \mathbf{u}_t) \leftarrow (1-\alpha) Q(\mathbf{x}_t, \mathbf{u}_t) + \alpha \max_{\hat{\mathbf{u}}} [c_t + \lambda Q(\mathbf{x}_{t+1}, \hat{\mathbf{u}})]$$

learning rate

discount factor

**Given fully observable state, infinite exploration, etc., guaranteed to converge on optimal policy.**

# The Adaptive House


Michael Mozer[+*]
Robert Dodier[#]
Debra Miller[*]
Marc Anderson[*]

Josh Anderson[☆]
Dan Bertini[#]
Matt Bronder[*]
Michael Colagrosso[*]
Robert Cruickshank[#]
Brian Daugherty[*]
Mark Fontenot[▲]
Okechukwu Ikeako[☆]
Paul Kooros[☆]

Diane Lukianow[☆]
Tom Moyer[■]
Charles Myers[☆]
Tom Pennell[*]
James Ries[☆]
Erik Skorpen[☆]
Joel Sloss[☆]
Lucky Vidmar[*]
Matthew Weeks[☆]

## University of Colorado

[*]Department of Computer Science
[+]Institute of Cognitive Science
[#]Department of Civil, Environmental, and Architectural Engineering
[☆]Department of Electrical and Computer Engineering
[■]Department of Mechanical Engineering
[▲]Department of Aerospace Engineering


http://www.cs.colorado.edu/~mozer/adaptive-house

# The adaptive house

**Not a programmable house, but a house that *programs itself*.**

**House *adapts* to the lifestyle of the inhabitants.**

House monitors environmental state and senses actions of inhabitant.

House learns inhabitants' schedules, preferences, and occupancy patterns.

House uses this information to achieve two objectives:
(1) anticipate inhabitant needs
(2) conserve energy

**Domain: home comfort systems**

- air heating
- lighting
- water heating
- ventilation

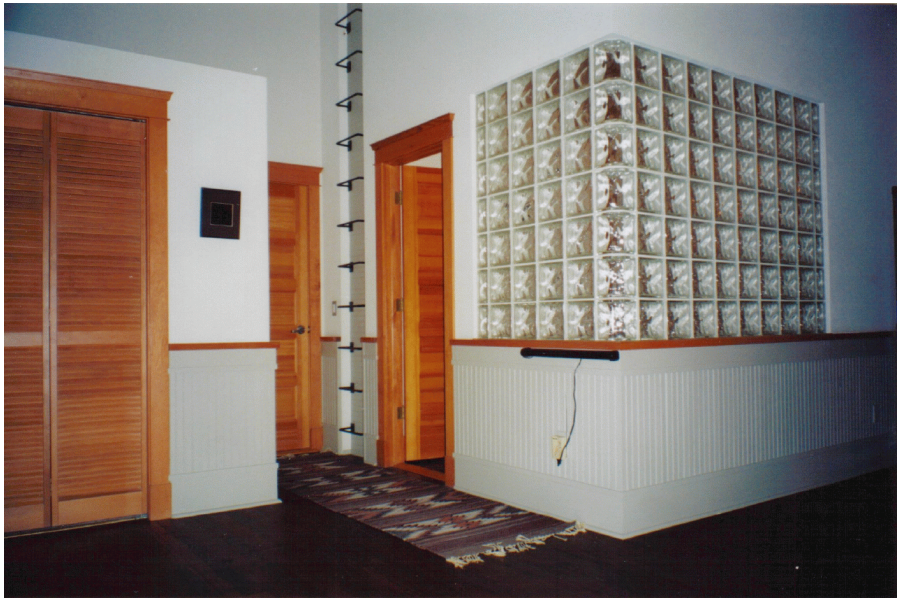
# The adaptive house

## Residence in Marshall, Colorado, outside of Boulder

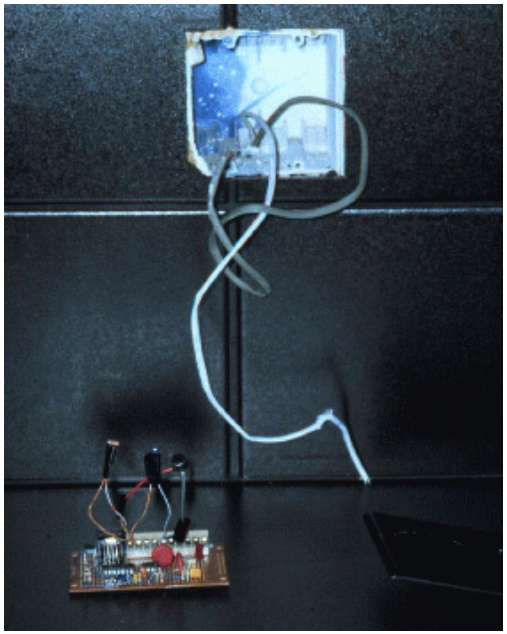
# Some of the gang

# Great room

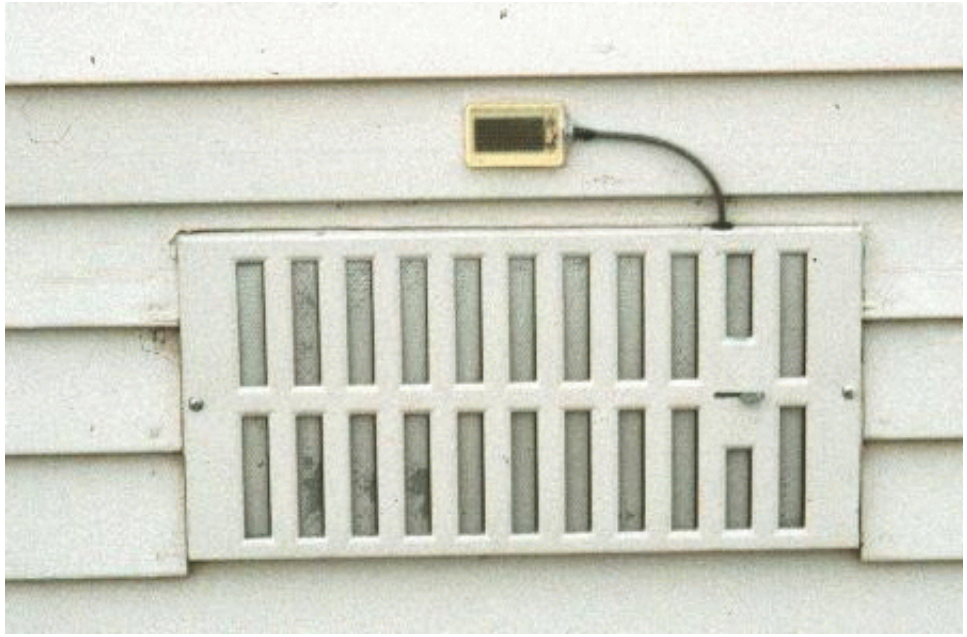
# Bedrooms and bathrooms



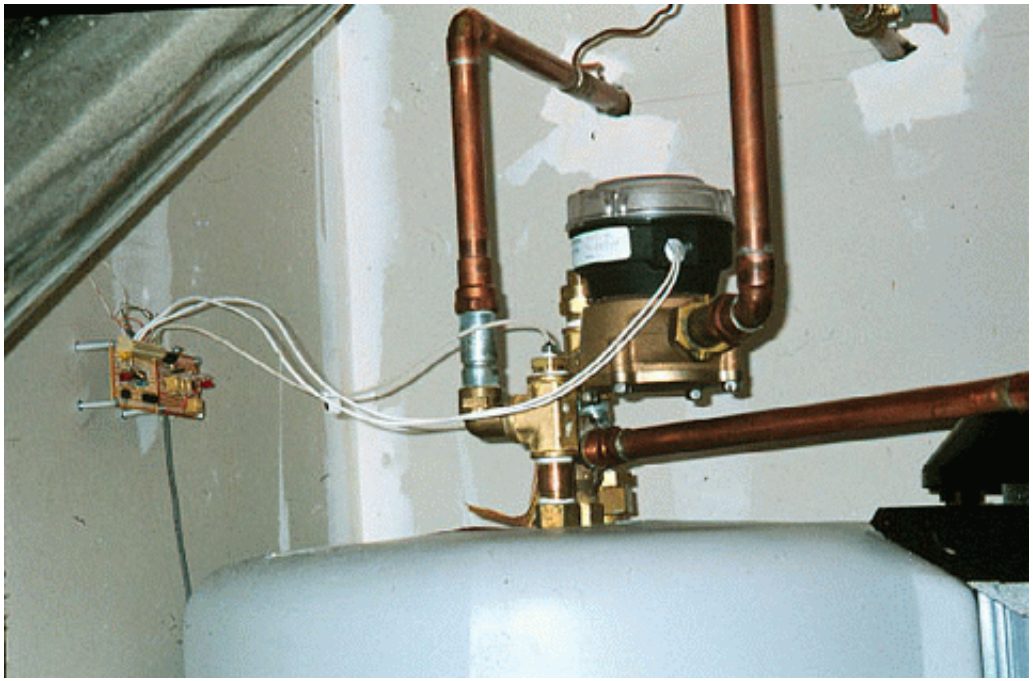
# Sensors

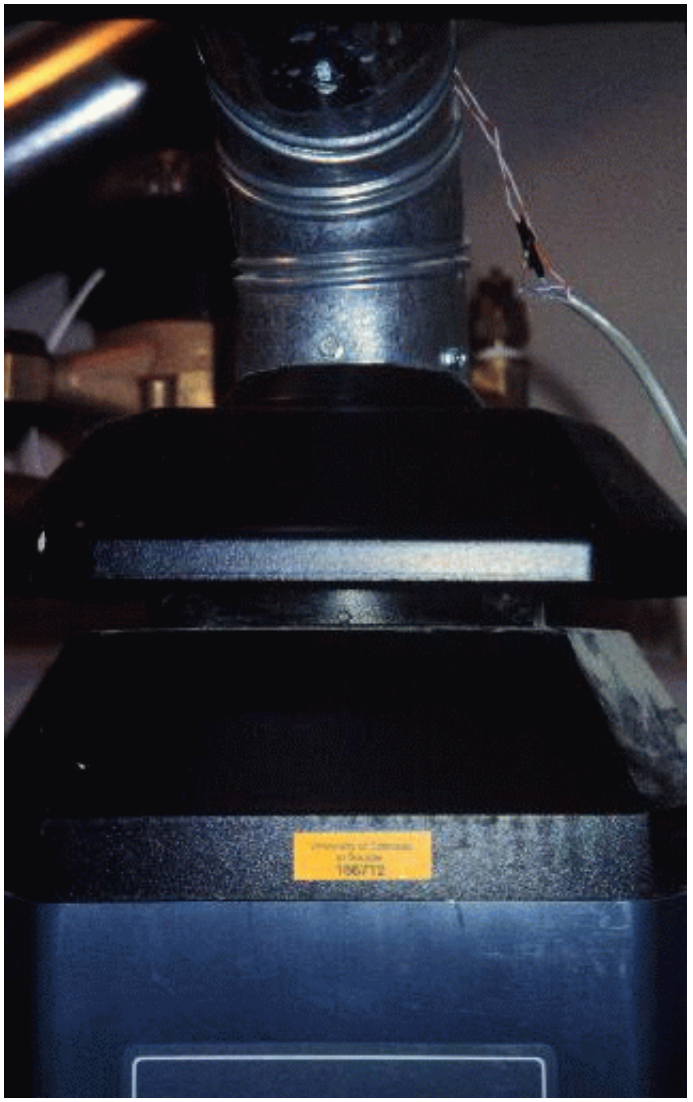


# Sensors

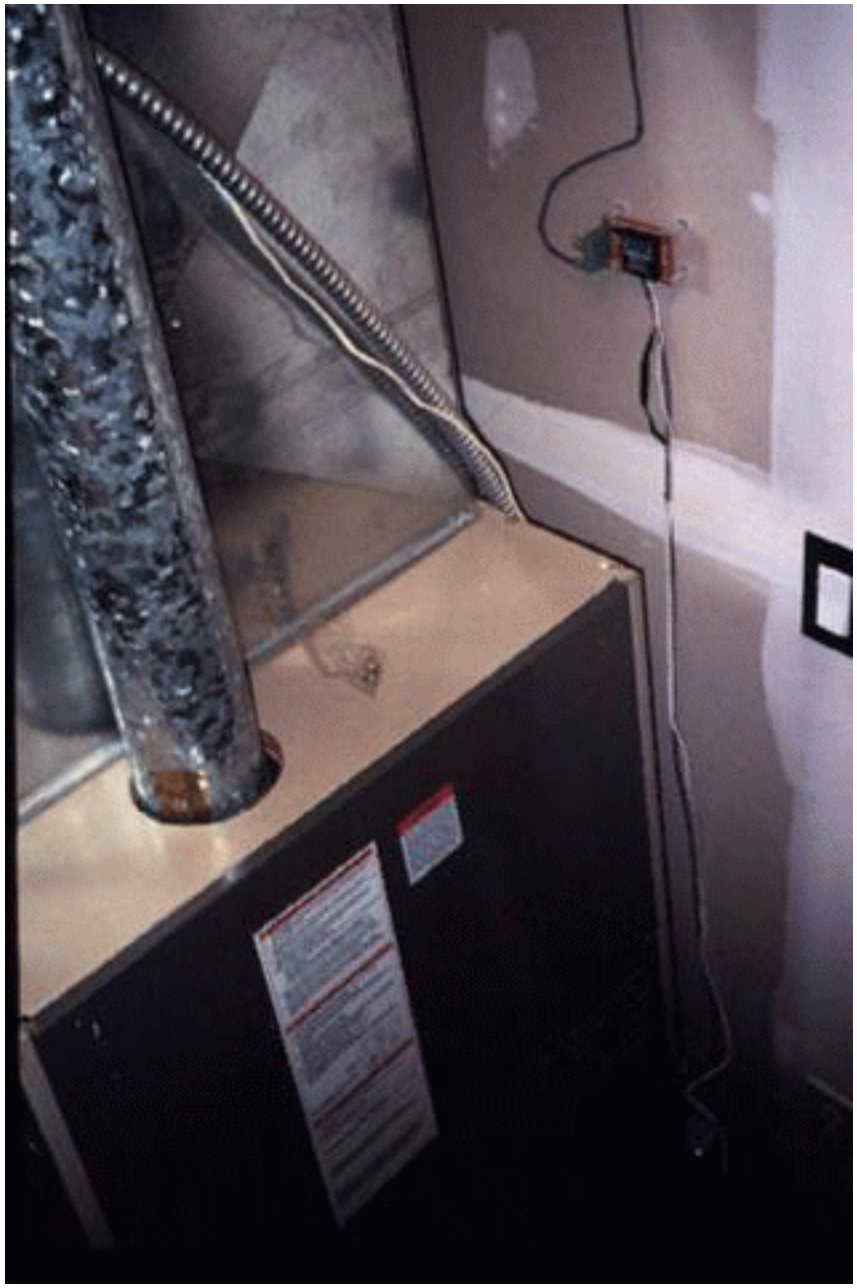


# Water heater

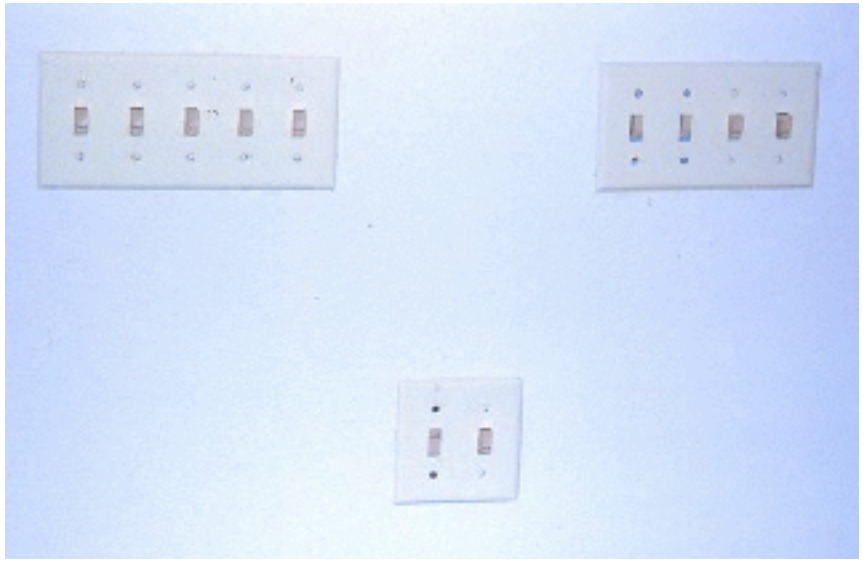




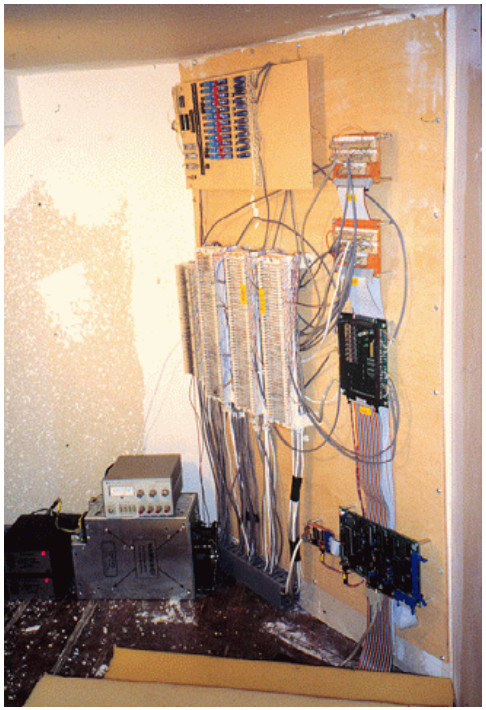
# Furnace



# Controls

# Computers



# Training signals

## Actions performed by inhabitant specify *setpoints*

- ➜ anticipation of inhabitant desires

## Gas and electricity costs

- ➜ energy conservation

# An reinforcement learning framework

**Each constraint has an associated cost:**

discomfort cost if inhabitant preferences are neglected

energy cost depends on device and intensity setting

**The optimal control policy minimizes**

$$J(t_0) = \mathrm{E}\left[\lim_{\kappa\to\infty} \frac{1}{\kappa} \sum_{t=t_0+1}^{t_0+\kappa} d(\mathbf{x}_t) + e(\mathbf{u}_t)\right]$$

**where $t$ = index over nonoverlapping time intervals**
**$t_0$ = current time interval**
**$\mathbf{u}_t$ = control decision for interval $t$**
**$\mathbf{x}_t$ = environmental state during interval $t$**

# ACHE
# (Adaptive Control of Home Environments)

## Separate control system for each task

air temperature regulation

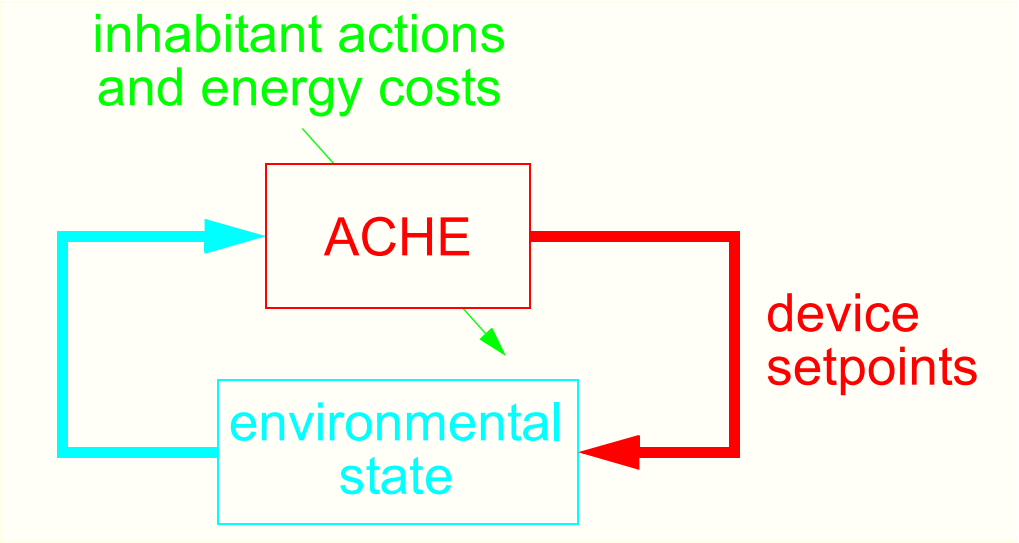
- furnace
- space heaters
- fans
- dampers
- blinds

lighting regulation

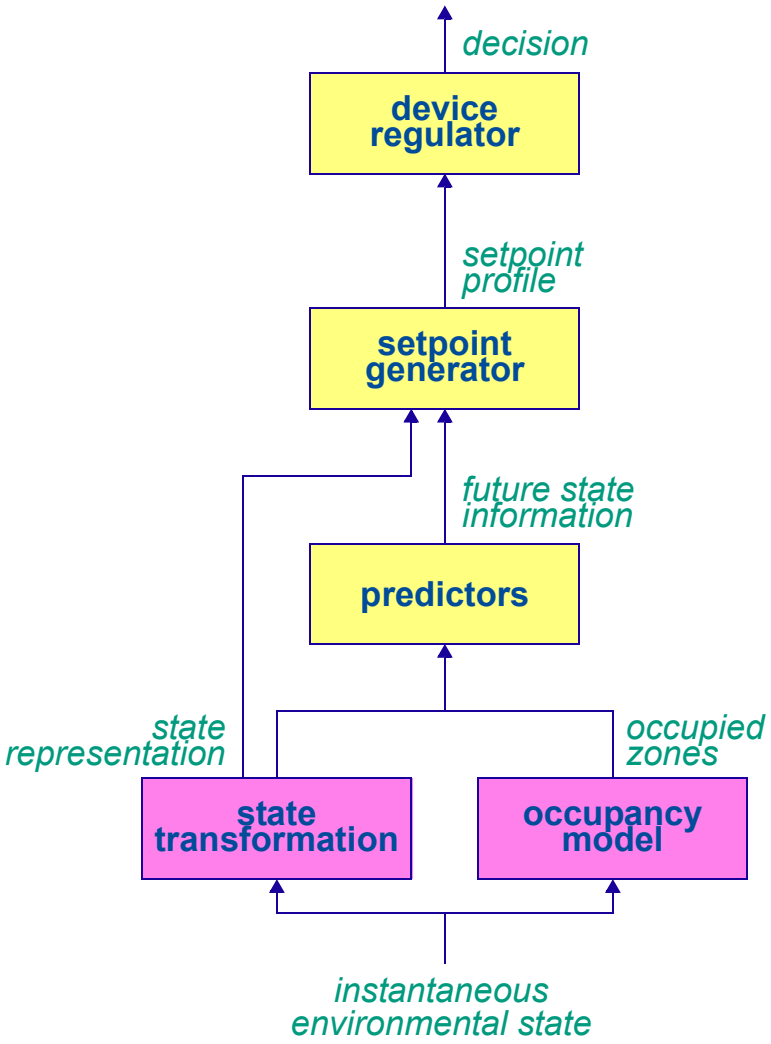
- wall sconces
- overhead lights

water temperature regulation

- hot water heater



# General architecture of ACHE



# Lighting control

## What makes lighting control a challenge?

Twenty-two banks of lights, each with 16 intensity levels; seven banks of lights in great room alone

Motion-triggered lighting does not work

Lighting moods

Two constraints must be satisfied simultaneously
- maintaining lighting according to inhabitant preferences
- conserving energy

Range of time scales involved

Sluggishness of system

# Resolving the sluggishness dilemma

## *Anticipator*: Neural network that predicts which zone(s) will become occupied in the next two seconds

## Input

| | |
|---|---|
| 1, 3, and 6 second average of motion signals | (36) |
| instantaneous and 2 second average of door status | (20) |
| instantaneous, 1 second, and 3 second average of sound level | (33) |
| current zone occupancy status and durations | (16) |
| time of day | (2) |

## Output

| | |
|---|---|
| p(zone *i* becomes occupied in next 2 seconds \| currently unoccupied) | (8) |

## Runs every 250 ms

# Training anticipator

## Occupancy model provides training signal

## Two types of errors

miss

state(t – 2000 ms)
state(t – 1750 ms)
...
state(t – 250 ms)

→ zone *i* becomes occupied

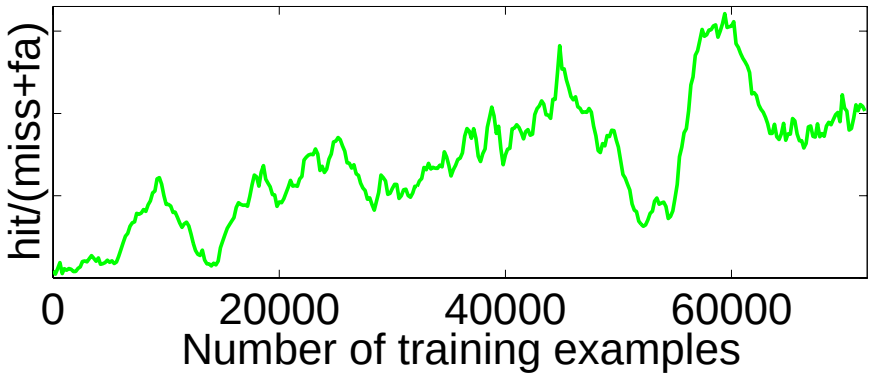
false alarm

state(t)

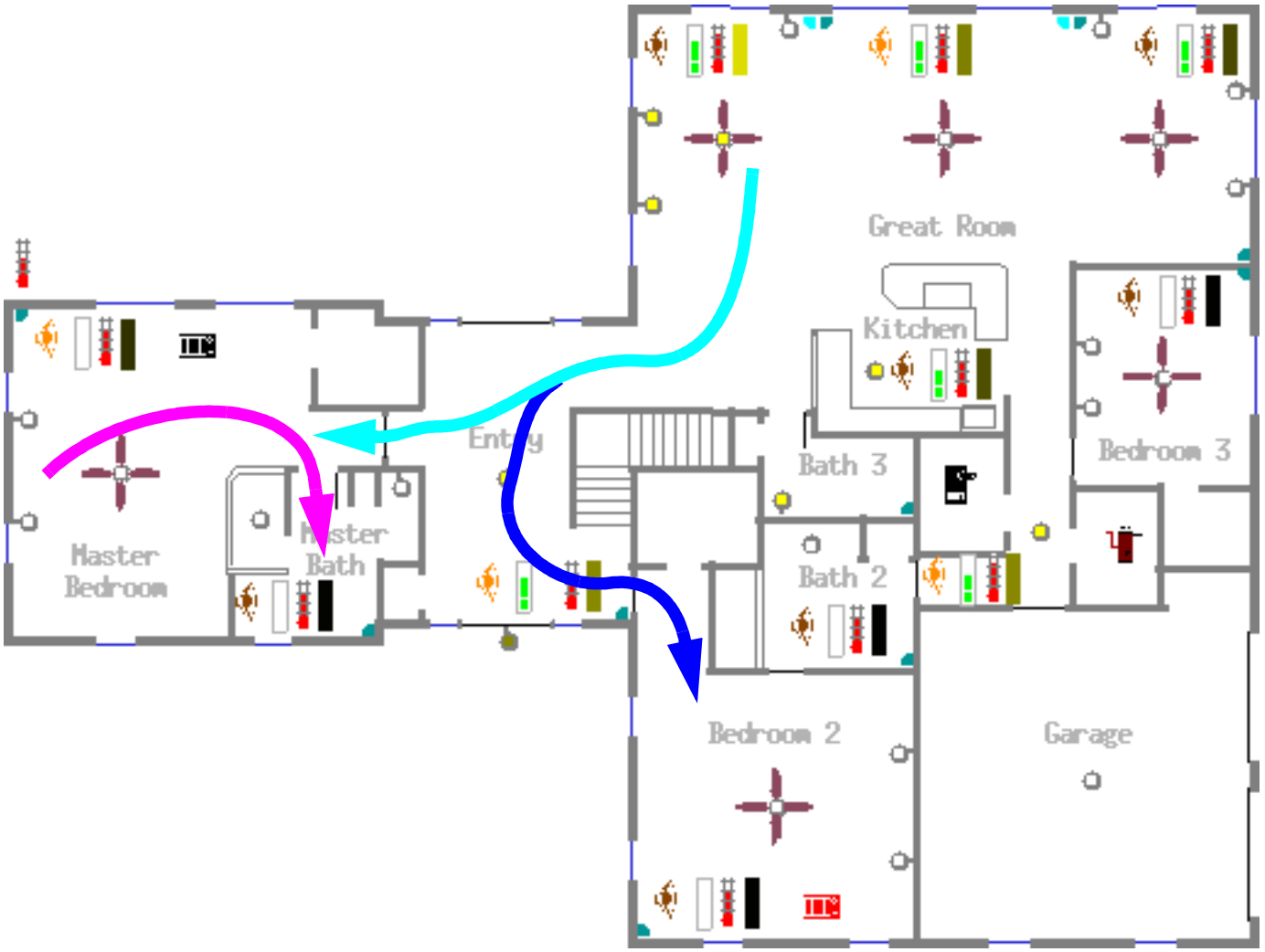
→ zone *i* vacant

## Training procedure

Given partially trained net, collect misses and false alarms.
Retrain net when 200 additional examples collected.
TD algorithm for misses



# Examples of anticipator performance



# Lighting controller costs

## Energy cost

7.2 cents per kW-hr

## Discomfort cost

1 cent per device whose level is manually adjusted

## Anticipator miss cost

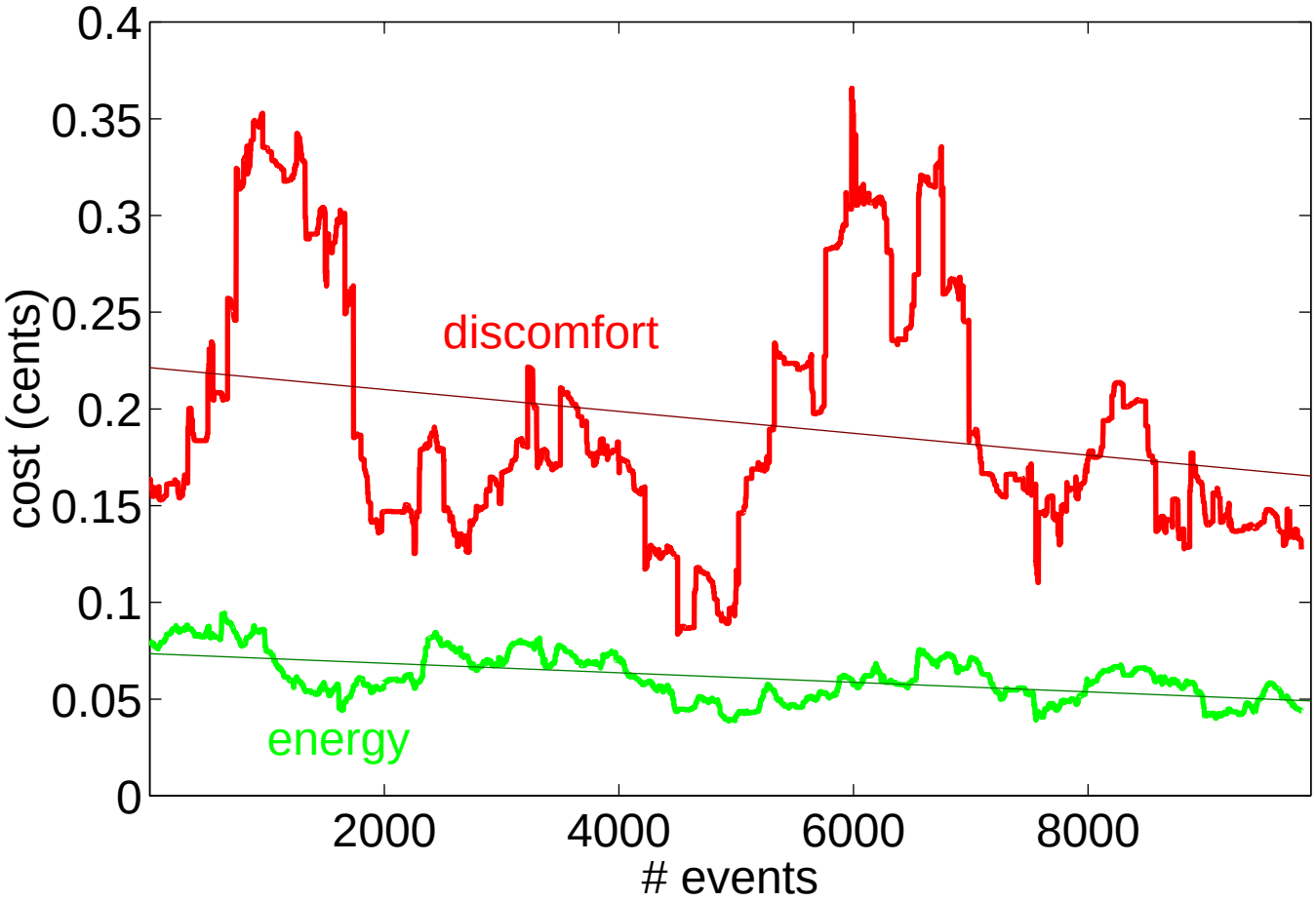
.1 cent per device that was off and should have been on

## Anticipator false alarm cost

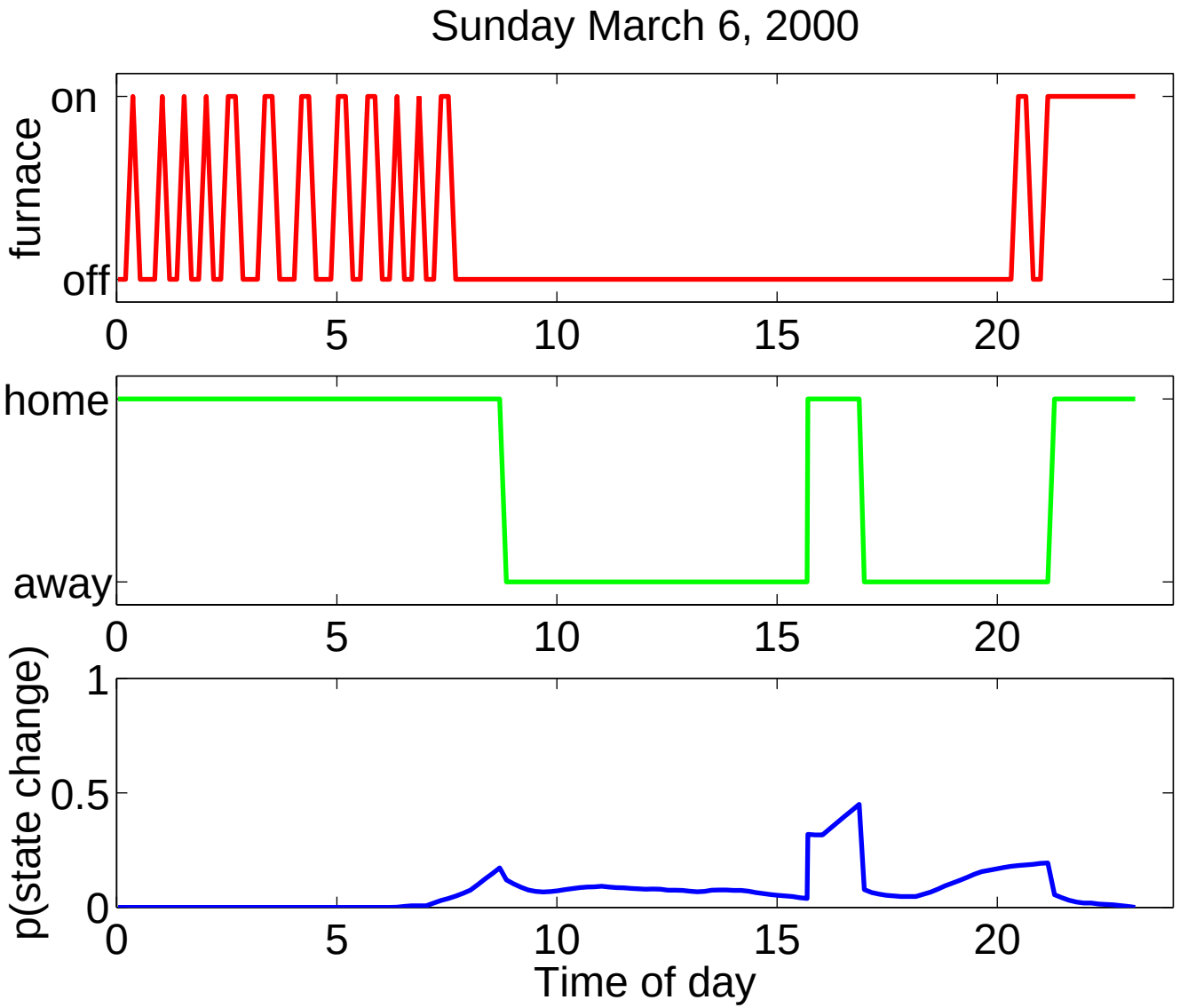
.1 cent per device that was turned on

# Results

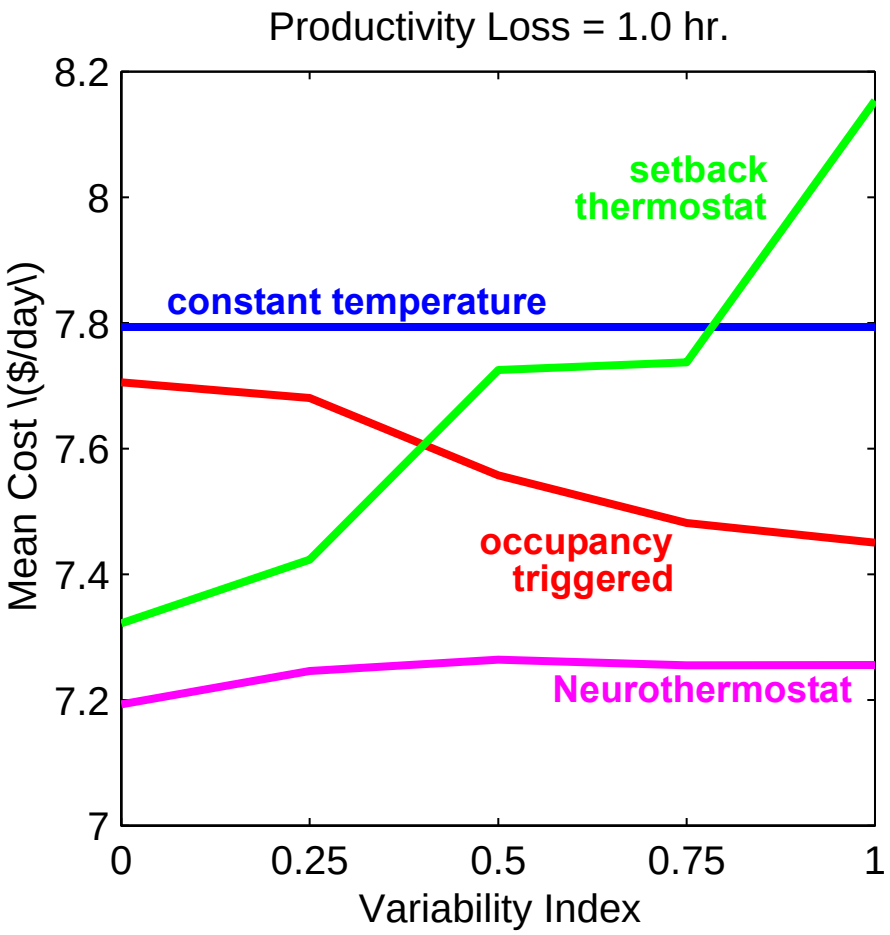
- about three months of data collection
- events logged only from 19:00 – 06:59

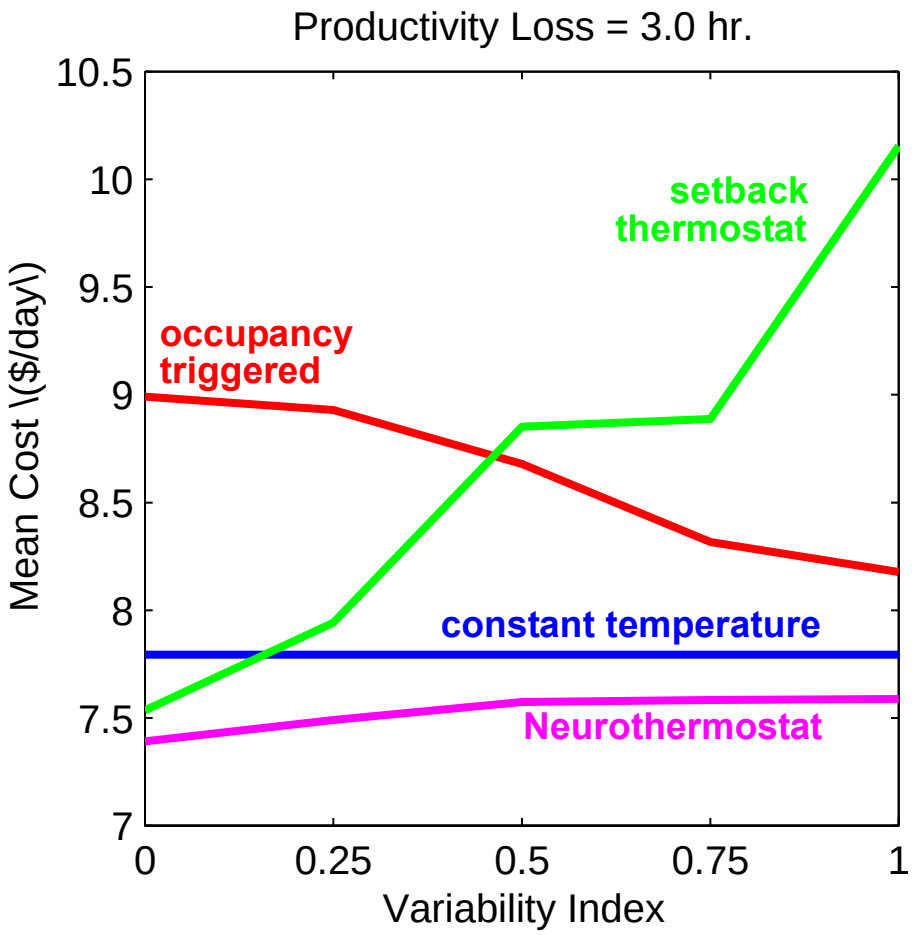


# Air temperature control



# Comparison of control policies using artificial occupancy data



# Comparison of control policies using real occupancy data

| | Mean Daily Cost | |
|---|---|---|
| | productivity loss | |
| | $\rho = 1$ | $\rho = 3$ |
| Neurothermostat | $6.77 | $7.05 |
| constant temperature | $7.85 | $7.85 |
| occupancy triggered | $7.49 | $8.66 |
| setback thermostat | $8.12 | $9.74 |