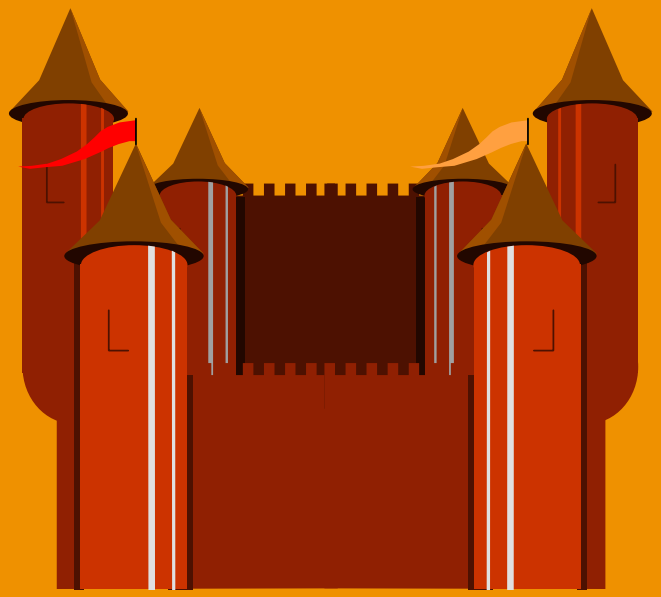
Minisymposium:
A Unified Framework for Optimization Under Uncertainty

Informs, Seattle

October 20, 2019

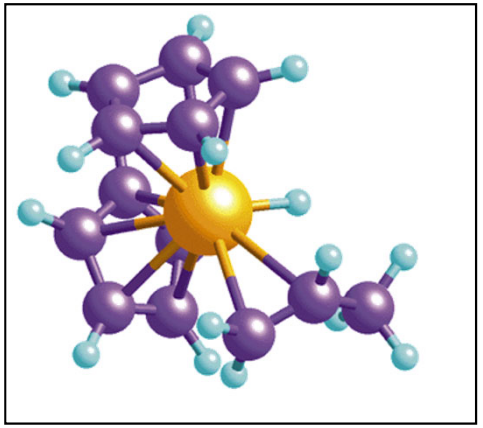


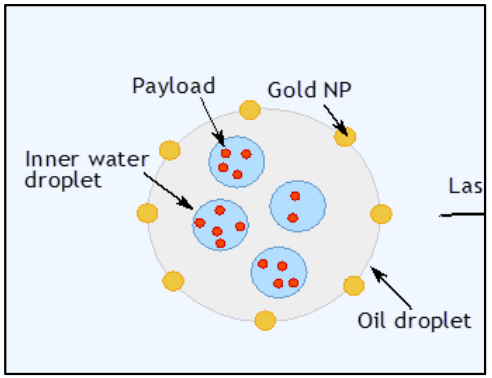
Warren B. Powell

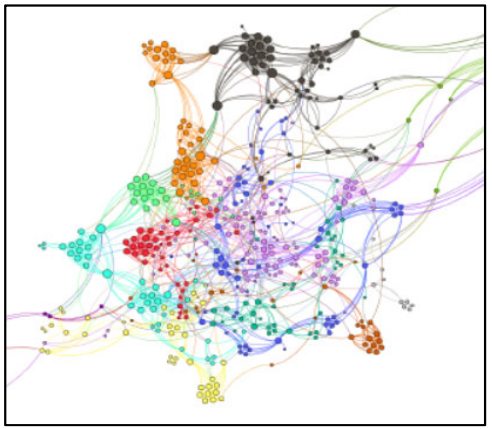
Princeton University
Department of Operations Research and Financial Engineering

Health applications

- Health sciences
  - » Sequential design of experiments for drug discovery
  - » Drug delivery – Optimizing the design of protective membranes to control drug release
  - » Medical decision making – Optimal learning for medical treatments.

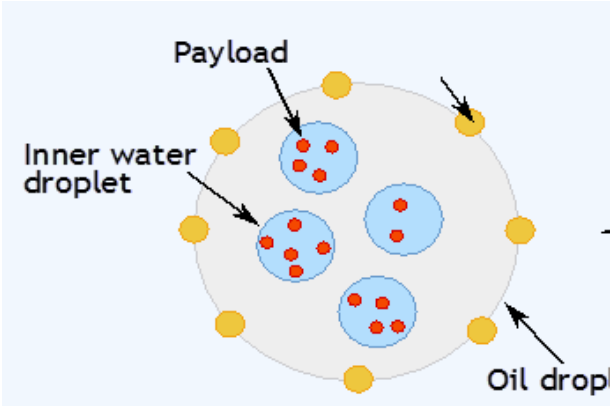




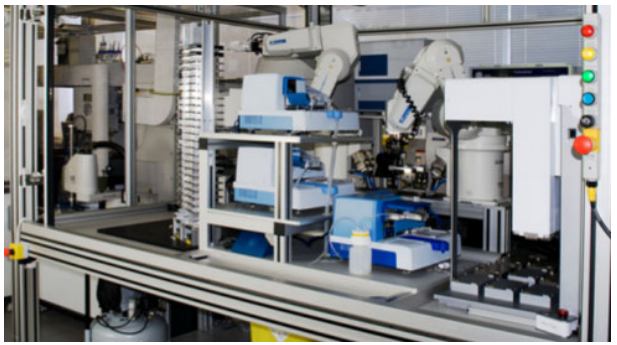


Laboratory sciences

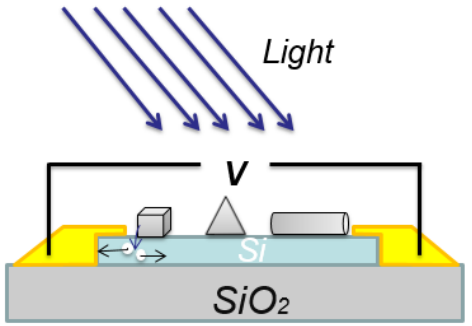
- Materials science
  - » Optimizing payloads: reactive species, biomolecules, fluorescent markers, …



  - » Controllers for robotic scientist for materials science experiments

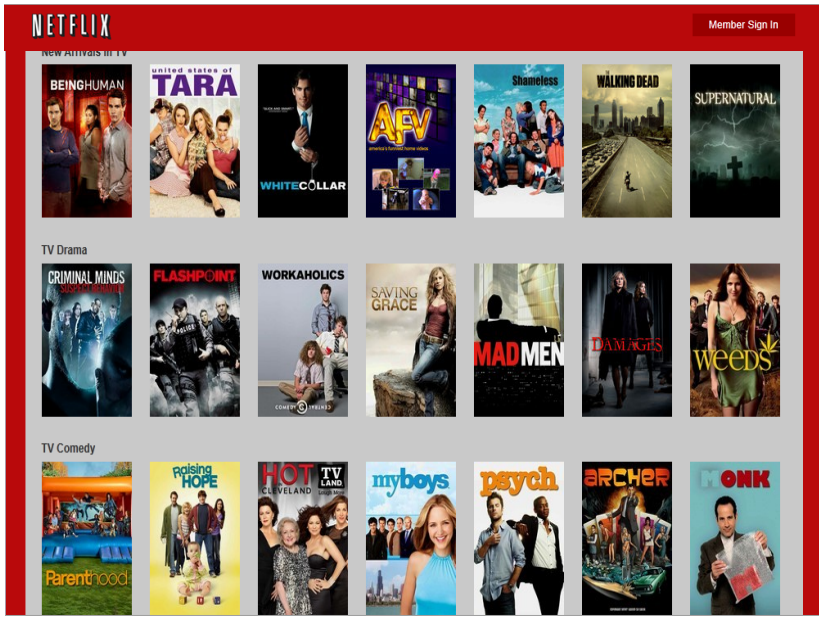


  - » Optimizing nanoparticles to maximize photoconductivity

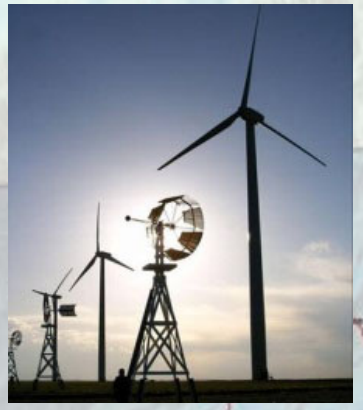


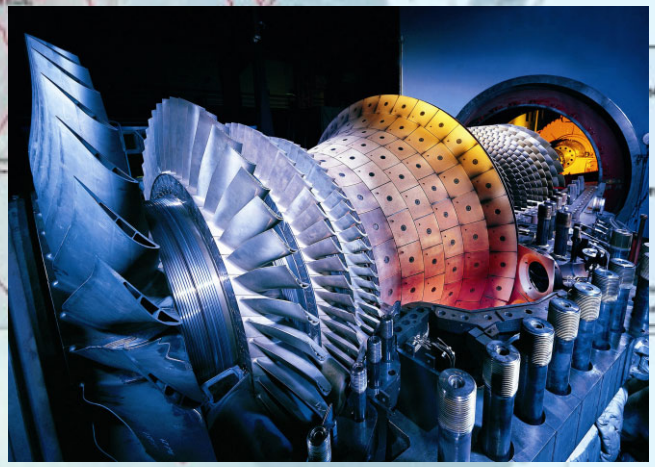
E-commerce

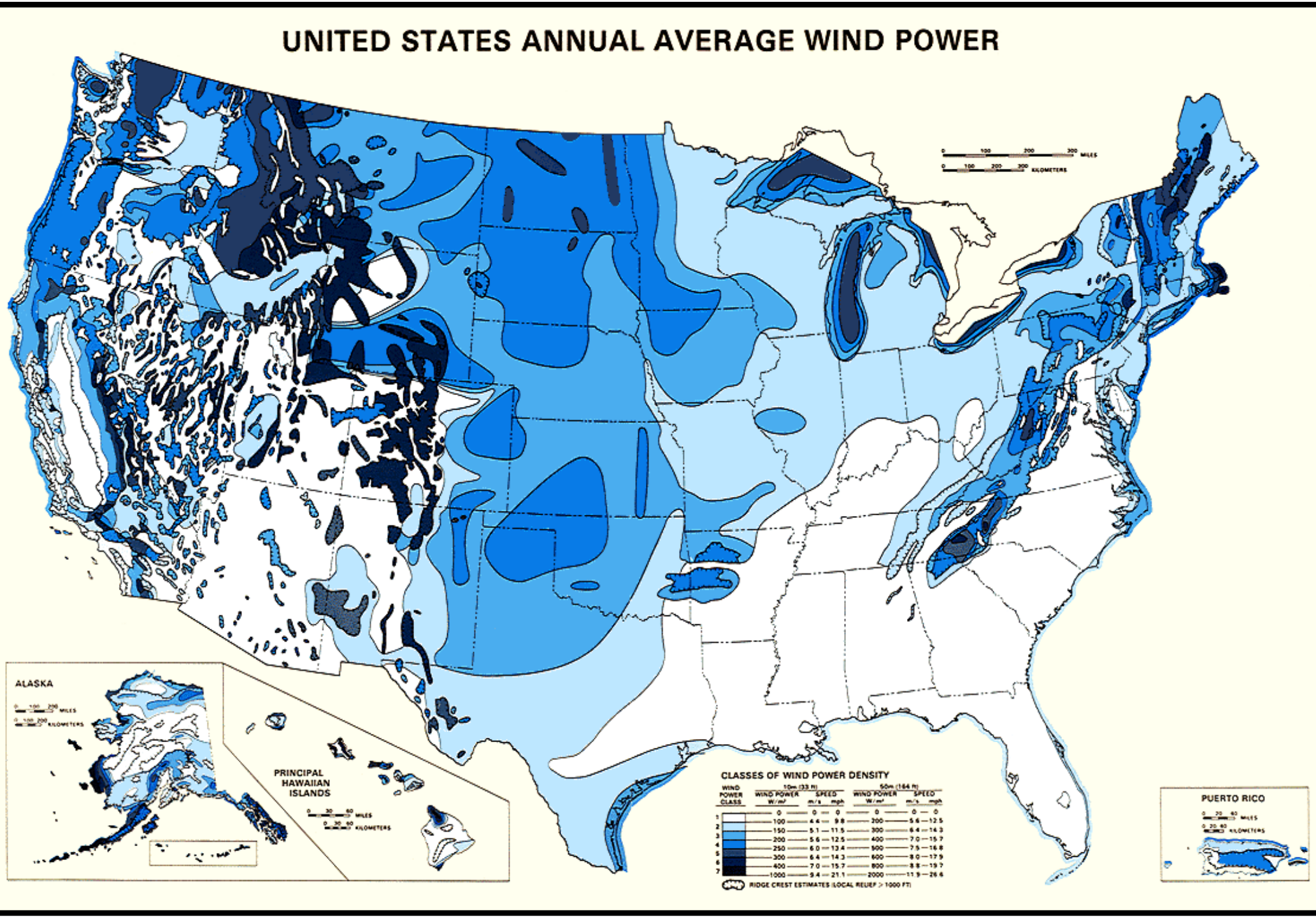
- Revenue management
  - » Optimizing prices to maximize total revenue for a particulate night in a hotel.
- Ad-click optimization
  - » How much to bid for ads on the internet.
- Personalized offer optimization
  - » Designing offers for individual customers.



An energy generation portfolio

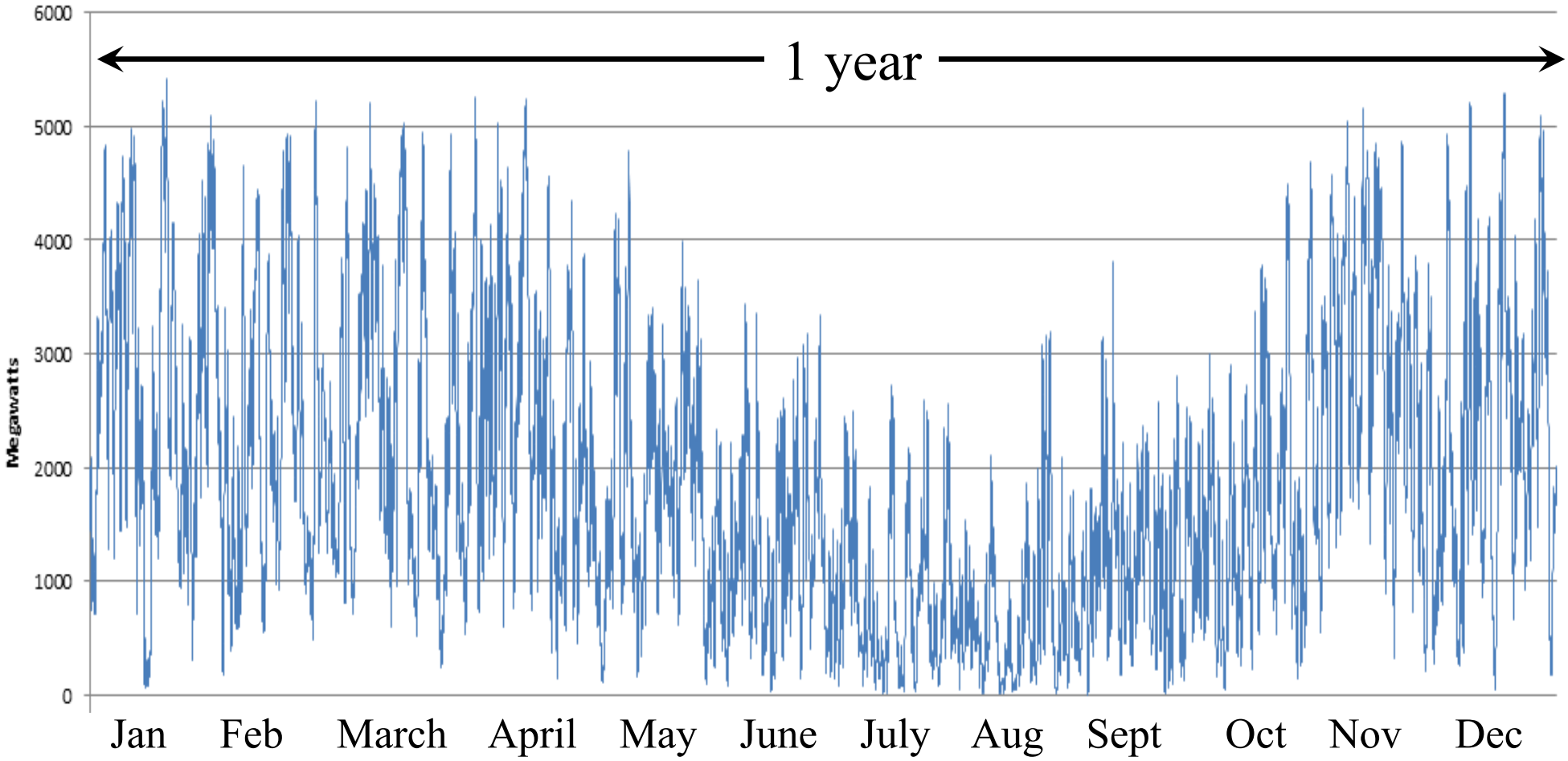




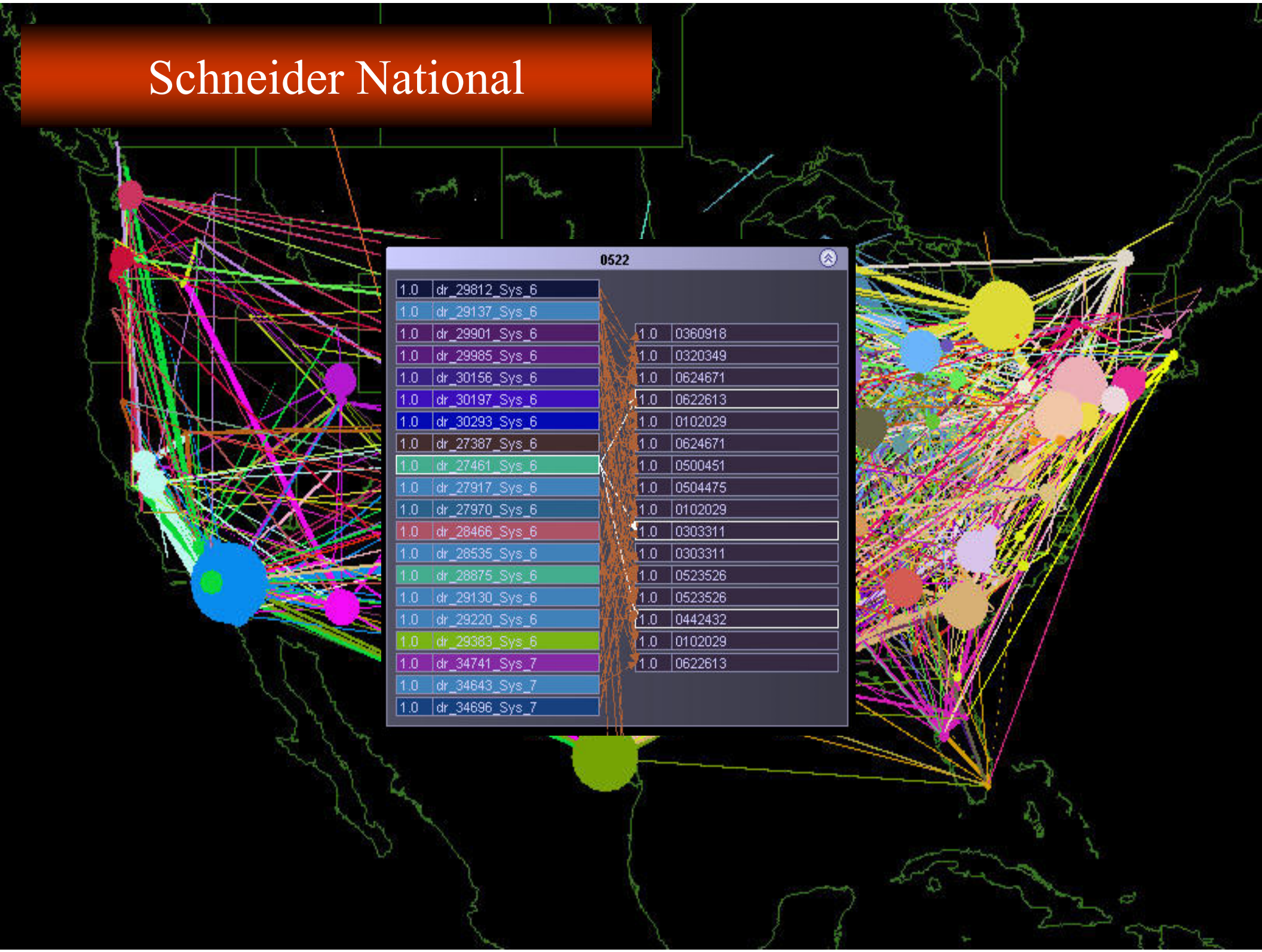


Energy from wind

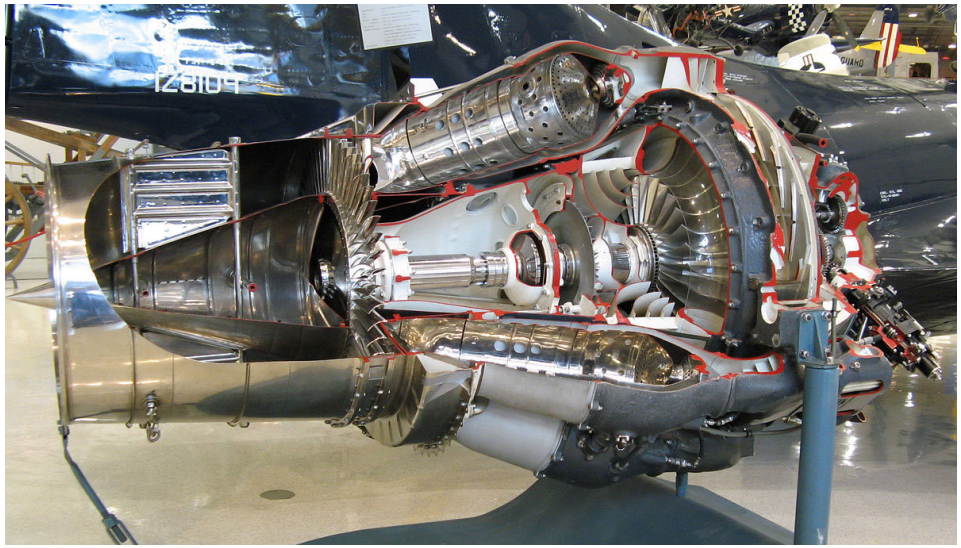
- Wind power from all PJM wind farms



Schneider National

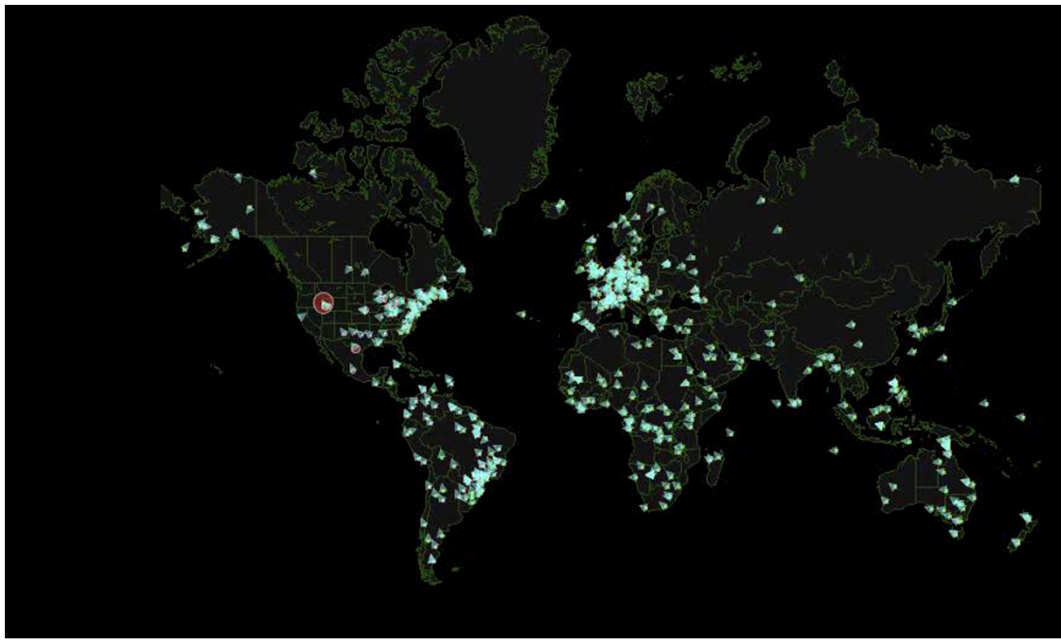


Multiagent supply chain management

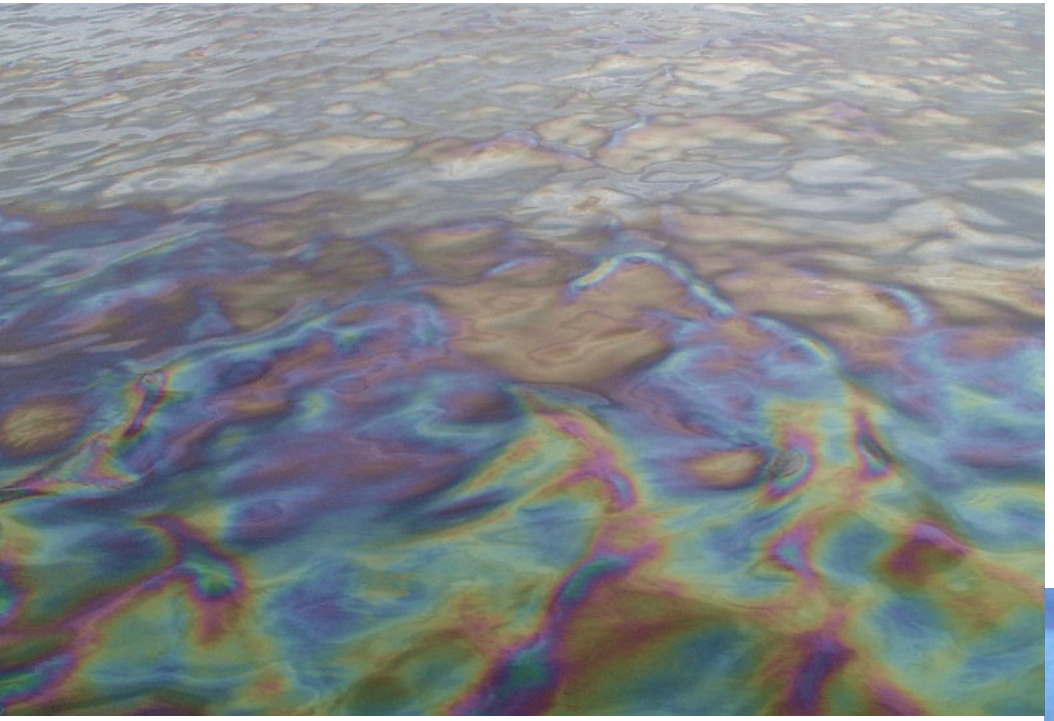


- Pratt&Whitney jet engines
  - » Over 1,000 parts
  - » Median lead time for a part is 120 days. Some lead times are over 300 days.
  - » Parts often require reworking.

- Managing the supply chain
  - » Challenge is determining when to order parts given the long lead times, and production uncertainties.
  - » Suppliers work for multiple customers.



Oil spills



- Sensing
  - » Drones fly over the ocean to detect the presence of oil.
- Communication
  - » Drones coordinate by sharing information.

- Mitigation
  - » Information from drones can be used to guide surface vehicles performing cleanup.

Emergency storm response

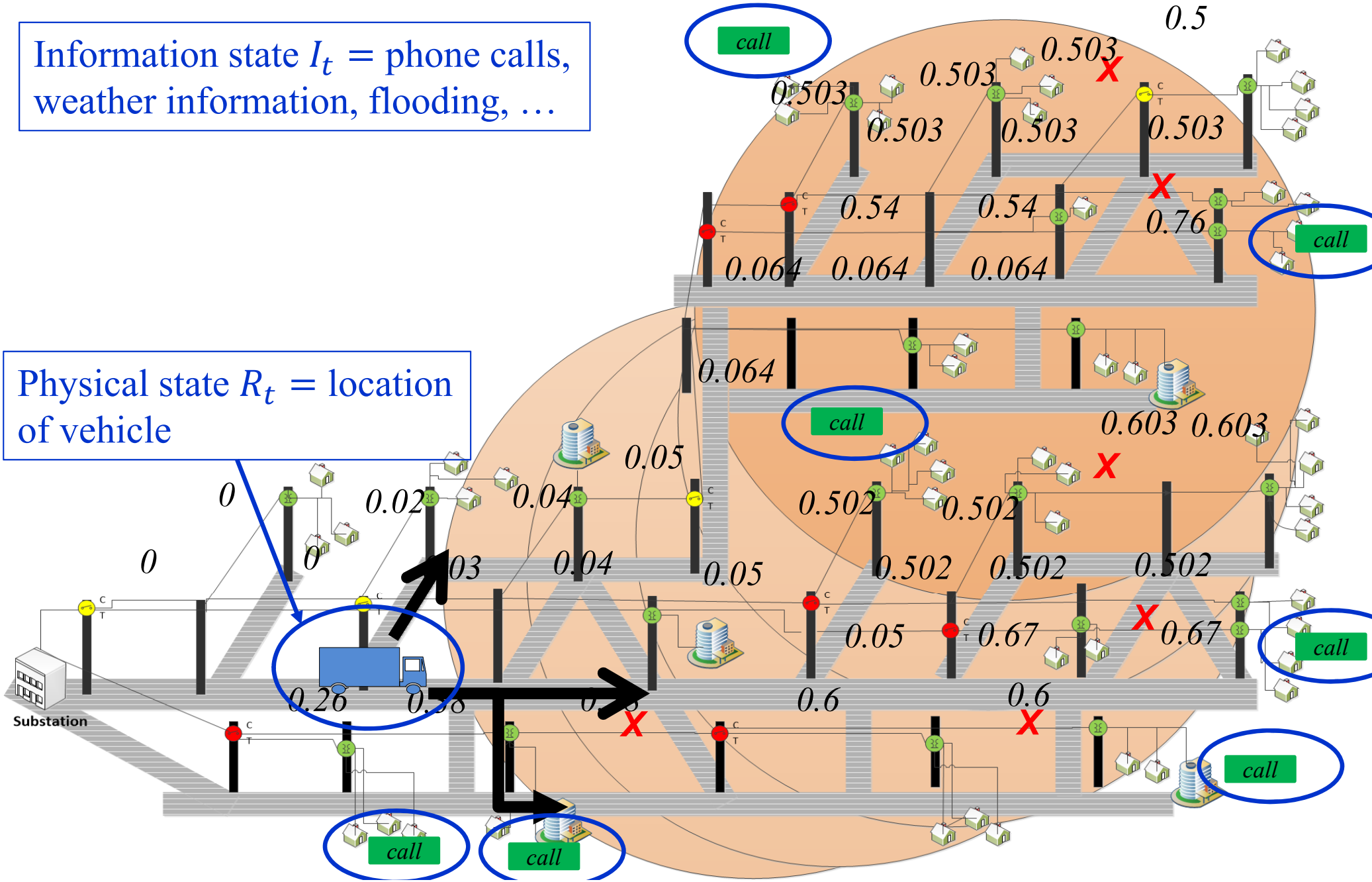
- Hurricane Sandy
  - » Once in 100 years?
  - » Rare convergence of events
  - » But, meteorologists did an amazing job of forecasting the storm.

- The power grid
  - » Loss of power creates cascading failures (lack of fuel, inability to pump water)
  - » How to plan?
  - » How to react?

Emergency storm response

Information state I_t = phone calls, weather information, flooding, …

Physical state R_t = location of vehicle

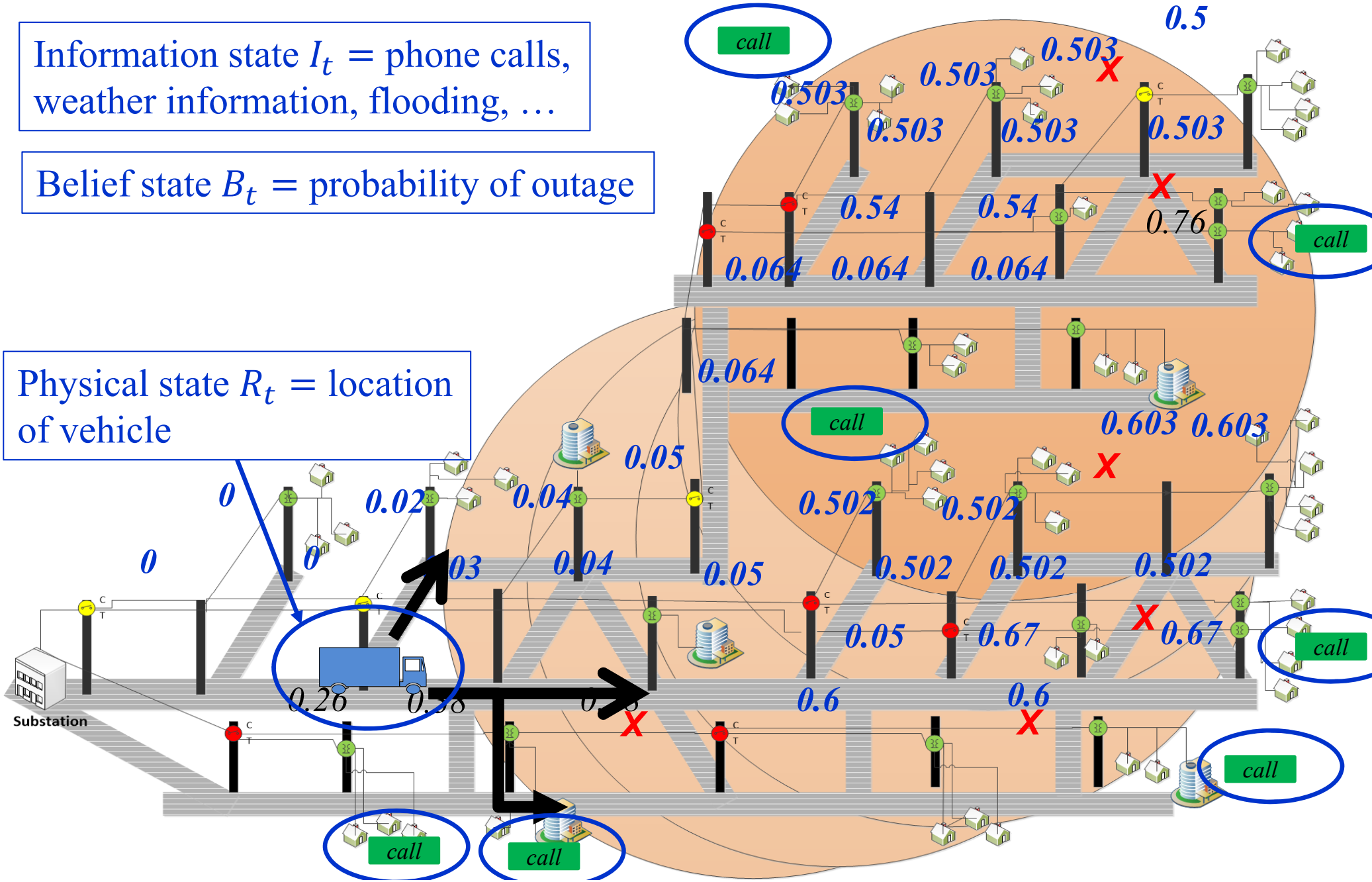


Emergency storm response

Information state I_t = phone calls, weather information, flooding, …

Belief state B_t = probability of outage

Physical state R_t = location of vehicle

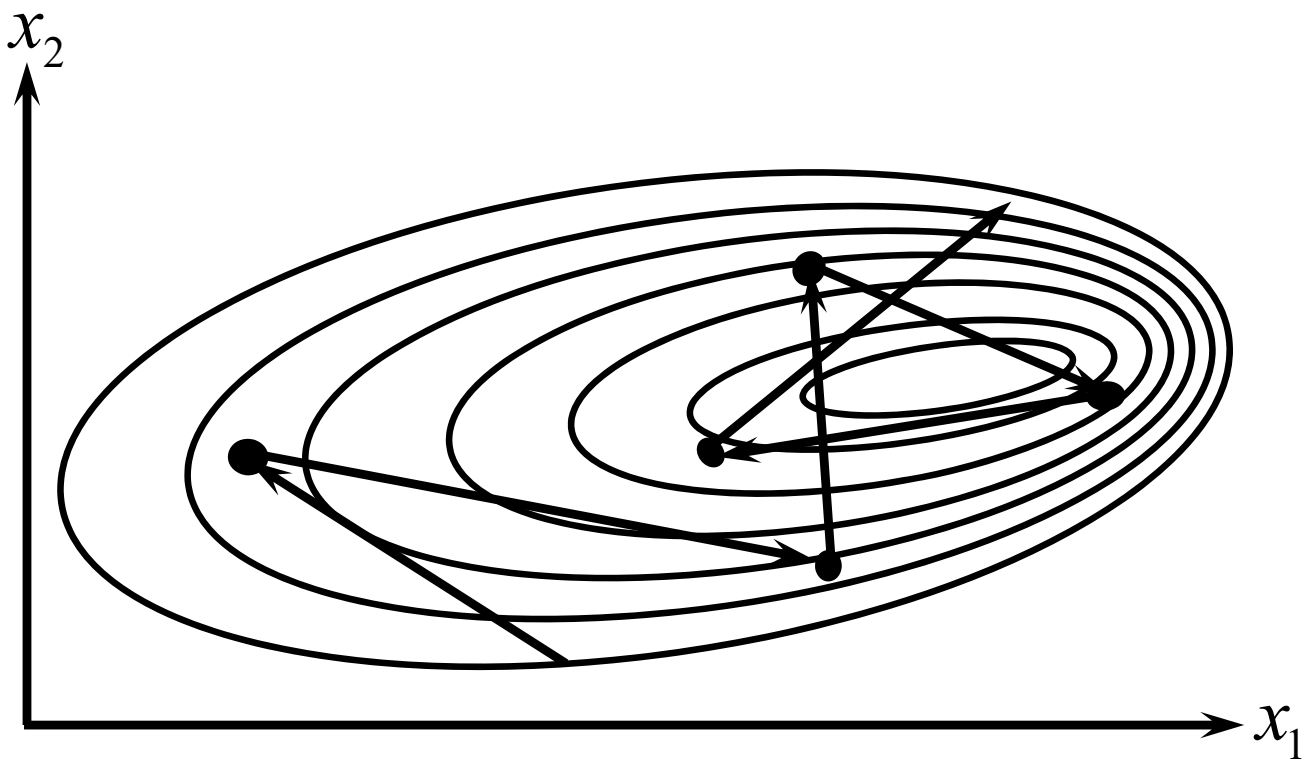


Derivative-based stochastic optimization

- Stochastic gradient algorithms:

$$x^{n+1} = x^n + \alpha_n \nabla F(x^n, W^{n+1})$$

are a form of sequential decision problem.



14

Sequential decision problems

- All of these applications are sequential decision problems:
  - » States, decisions, new information …

$$S_0, x_0, W_1, \ldots, S_t, x_t, W_{t+1}, \ldots$$

What we know (or believe)

What we control

What we observe (or learn)

  - » As we progress, we receive a "contribution" (cost, reward, …) $C(S_t, x_t)$ (or $C(S_t, x_t, W_{t+1})$).
  - » Decisions are made with a "policy" (or rule, or algorithm) $x_t = X^{\pi}(S_t)$.
  - » The challenge is to find the best policy that optimizes the contributions.

Sequential decision problems

- A sequential decision problems

  » We often need to write this using a counter n:

  $$S^0, x^0, W^1, \ldots, S^n, x^n, W^{n+1}, \ldots$$

  n might index experiments, arrivals, or iterations of an algorithm.

  » Sometimes we need to index both iterations and time

  $$S_0^0, x_0^0, W_1^0, \ldots, S_t^0, x_t^0, W_{t+1}^0, \ldots, S_t^n, x_t^n, W_{t+1}^n$$

  t might index hour of week, while *n* indexes the week.

Sequential decision problems

- Major problem domains
  - » “Reinforcement learning” - Discrete
  - » Optimal control – Continuous
  - » Dynamic resource allocation
  - » Stochastic search (algorithms)
  - » Multiarmed bandit problems (“active learning”)
  - » Games (two agent)
  - » Multiagent systems

Outline

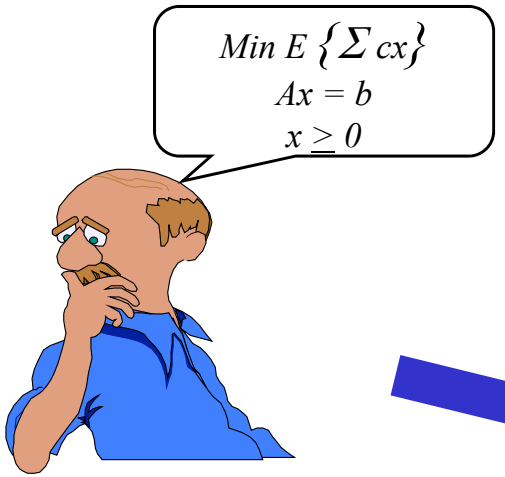
- Elements of a dynamic model
- An energy storage illustration
- Modeling uncertainty
- Designing policies
- Educational materials

Outline

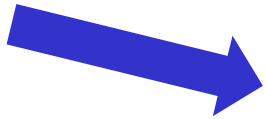
- Elements of a dynamic model
- An energy storage illustration
- Modeling uncertainty
- Designing policies
- Educational materials

Elements of a dynamic model

- Before we can *solve* complex problems, we have to know how to *think* about them.



Mathematician



Software

- The biggest challenge when making decisions under uncertainty is *modeling*.

Elements of a dynamic model

- For deterministic problems, we speak the language of mathematical programming
  - » Linear programming:

$$\min_x cx$$
$$Ax = b$$
$$x \geq 0$$

  - » For time-staged problems

$$\min_{x_0,\ldots,x_T} \sum_{t=0}^{T} c_t x_t$$
$$A_t x_t - B_{t-1} x_{t-1} = b_t$$
$$D_t x_t \leq u_t$$
$$x_t \geq 0$$

Arguably Dantzig's biggest contribution, more so than the simplex algorithm, was his articulation of optimization problems in a standard format, which has given algorithmic researchers a common language.

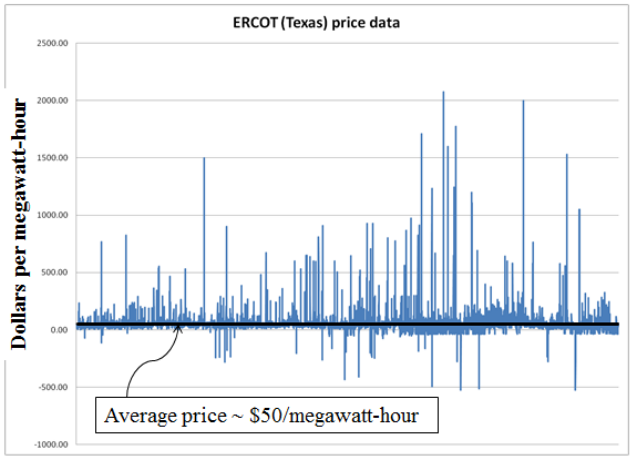

John R. Birge
François Louveaux
Introduction to Stochastic Programming
Second Edition
Michael C. Fu Editor
Handbook of Simulation Optimization
Princeton Series in APPLIED MATHEMATICS
Robust Optimization
Introduction to Decision Analysis
A Practitioner's Guide to Improving Decision Quality
SECOND EDITION
Approximate Dynamic Programming
Solving the Curses of Dimensionality
Warren B. Powell
Optimal Learning
MULTI-ARMED BANDIT ALLOCATION INDICES
SECOND EDITION
John Gittins, Kevin
and Richard W
VOLUME 2 • 4th EDITION
Dynamic Programming and Optimal Control
APPROXIMATE DYNAMIC PROGRAMMING
Dimitri P. Bertsekas
WILEY
Model Predictive Control
OPTIMAL CONTROL
Frank L. Lewis
MORGAN & CLAYPOOL PUBLISHERS
Active Learning
Burr Settles
INTRODUCTION TO STOCHASTIC SEARCH AND OPTIMIZATION
Estimation, Simulation, and Control
JAMES C. SPALL
Athena Scientific
WILEY
Markov Decision Processes
Discrete Stochastic Dynamic Programming
MARTIN L. PUTERMAN
Online Computation and Competitive Analysis
Allan Borodin
Ran El-Yaniv
STOCHASTIC SIMULATION OPTIMIZATION
An Optimal Computing Budget Allocation
Chun-Hung Chen • Loo Hay Lee
43
Jiongmin Yong
Xun Yu Zhou
Stochastic Controls
Hamiltonian Systems and HJB Equations
Reinforcement Learning
An Introduction
second edition
Richard S. Sutton and Andrew G. Barto

Elements of a dynamic model

- All sequential decision problems can be modeled using five core components:
  - » State variables $S_t = (R_t, I_t, B_t)$
    - Physical state R_t, other information I_t, belief state B_t.
  - » Decision variables (x_t, a_t, u_t)
    - Made with *policy* $X^\pi(S_t|\theta)$ (or $A^\pi(S_t)$ or $U^\pi(S_t)$)
  - » Exogenous information W_{t+1}
    - What do we learn for the first time between t and $t+1$?
  - » Transition function $S_{t+1} = S^M(S_t, x_t, W_{t+1})$
    - How do the state variables evolve over time?
  - » Objective function
    - $\max_\pi \mathbb{E}_{S_0} \mathbb{E}_{W_1,\ldots,W_T|S_0} \sum_{t=0}^{T} C(S_t, X^\pi(S_t))$

Energy storage example

- Battery arbitrage – When to charge, when to discharge, given stochastic prices



Elements of a dynamic model

- The state variable:

Controls community

x_t = "Information state"

Operations research/MDP/Computer science

$S_t = (R_t, I_t, B_t)$ = System state, where:

R_t = Resource state (physical state)

Location/status of truck/train/plane

Inventories (product, fleets, supplies)

I_t = Other (deterministic) information

Prices

Weather

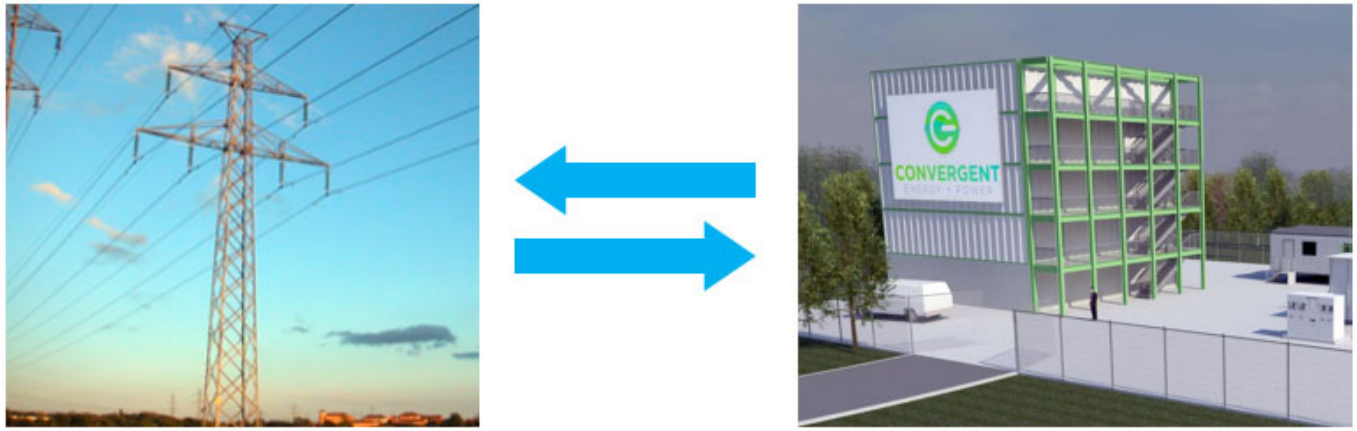
B_t = Belief state ("state of knowledge")

Belief about market response to prices

Belief about the status of equipment

Energy storage example

- State variables S_t



  - » R_t =Energy stored in the battery
  - » D_t =Current demand
  - » p_t =Current grid price
  - » State variable $S_t = (R_t, D_t, p_t)$

Elements of a dynamic model

- Decisions:

Markov decision processes/Computer science

a_t = Discrete action

Control theory

u_t = Low-dimensional continuous vector

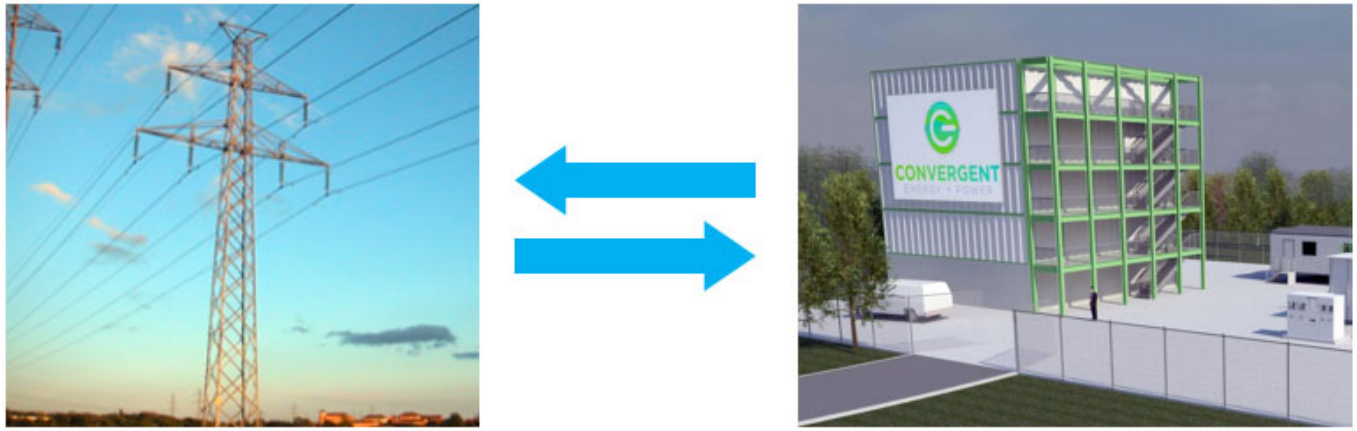
Operations research

x_t = Usually a discrete or continuous but high-dimensional vector of decisions.

At this point, we do not specify *how* to make a decision. Instead, we define the function $X^\pi(S_t)$ (or $A^\pi(S_t)$ or $U^\pi(S_t)$), where π specifies the type of policy. "π" carries information about the type of function f, and any tunable parameters $\theta \in \Theta^f$.

Energy storage example

- Decision variables



» x_t =How much to sell ($x_t > 0$) or buy ($x_t < 0$) to/from the grid.

» Constraints:

- $x_t \leq R_t$
- $-x_t \leq R^{max} - R_t$

» Policy: $x_t = X^{\pi}(S_t)$

Elements of a dynamic model

- Styles of decisions
  - » Binary

    $x \in X = \{0,1\}$
  - » Finite

    $x \in X = \{1,2,...,M\}$
  - » Continuous scalar

    $x \in X = [a,b]$
  - » Continuous vector

    $x = (x_1,...,x_K), \quad x_k \in \mathbb{R}$
  - » Discrete vector

    $x = (x_1,...,x_K), \quad x_k \in \mathbb{Z}$
  - » Categorical

    $x = (a_1,...,a_I), \quad a_i$ is a category (e.g. red/green/blue)

Elements of a dynamic model

- Exogenous information:

W_t = New information that first became known at time t

$= \left(\hat{R}_t, \hat{D}_t, \hat{p}_t, \hat{E}_t\right)$

$\hat{R}_t$ = Equipment failures, delays, new arrivals

New drivers being hired to the network

$\hat{D}_t$ = New customer demands

$\hat{p}_t$ = Changes in prices

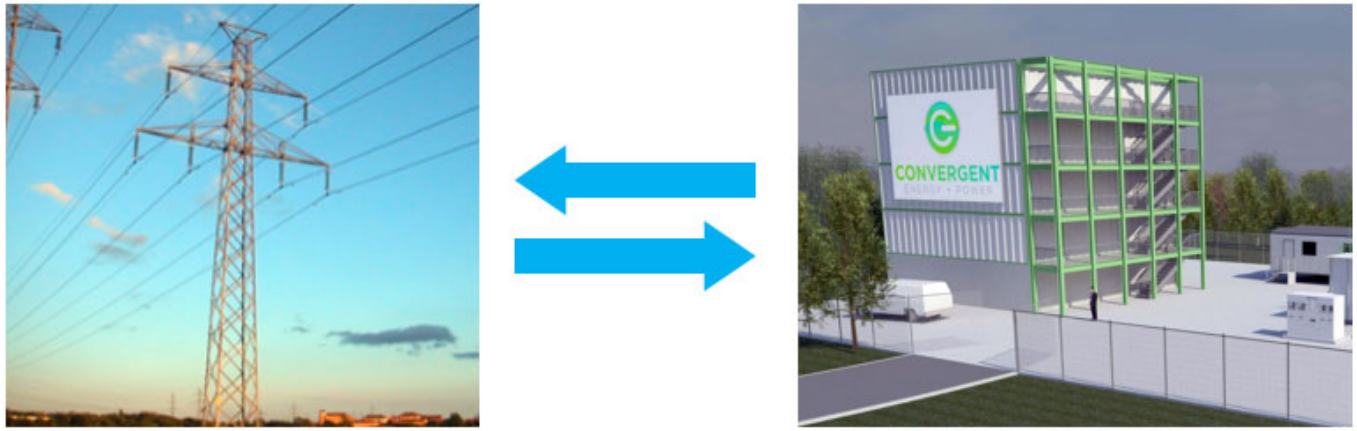
$\hat{E}_t$ = Information about the environment (temperature, ...)

Note: Any variable indexed by t is known at time t. This convention, which is not standard in control theory, dramatically simplifies the modeling of information.

W_{t+1} may be a function of the state S_t and/or the action x_t, so we may write it as $W_{t+1}(S_t, x_t)$.

Energy storage example

- Exogenous information



» $W_{t+1} =$

- $\hat{p}_{t+1}$ =Change in the electricity price p_t between t and $t+1$.
- $\widehat{D}_{t+1}$ =Demand for energy between time t and $t+1$

Elements of a dynamic model

- The transition function

$$S_{t+1} = S^M(S_t,\ x_t,\ W_{t+1})$$

$$R_{t+1} = R_t + x_t + \hat{R}_{t+1} \qquad \text{Inventories}$$

$$p_{t+1} = p_t \quad + \quad \hat{p}_{t+1} \qquad \text{Spot prices}$$

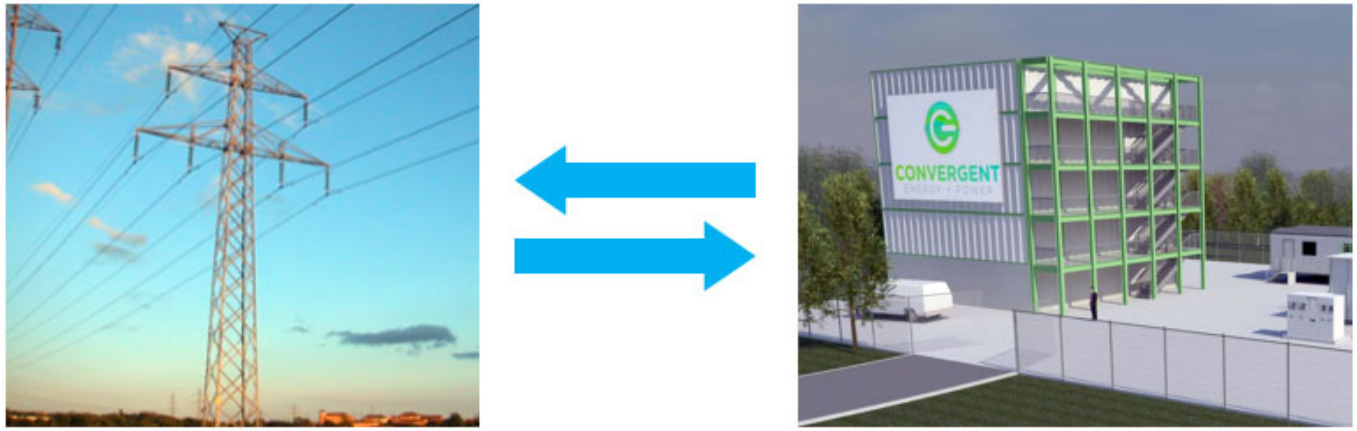
$$D_{t+1} = \hat{D}_{t+1} \qquad \text{Market demands}$$

Also known as the:

"System model"	"Transfer function"
"State transition model"	"Transformation function"
"Plant model"	"Law of motion"
"Plant equation"	"Model"
"State equation"	"transition function"

Energy storage example

- Transition function



» $R_{t+1} = R_t + \eta x_t$

» $p_{t+1} = p_t + \hat{p}_{t+1}$

» $D_{t+1} = \widehat{D}_{t+1}$

For many problems, the transition function can be very complex ("500 lines of Matlab code").

Elements of a dynamic model

- Objective functions
  - » Cumulative reward ("online learning")

$$\max_{\pi} \mathbb{E}\left\{\sum_{t=0}^{T} C_t\left(S_t, X^{\pi}(S_t), W_{t+1}\right) \mid S_0\right\}$$

    - Policies have to work well *over time.*
  - » Final reward ("offline learning")

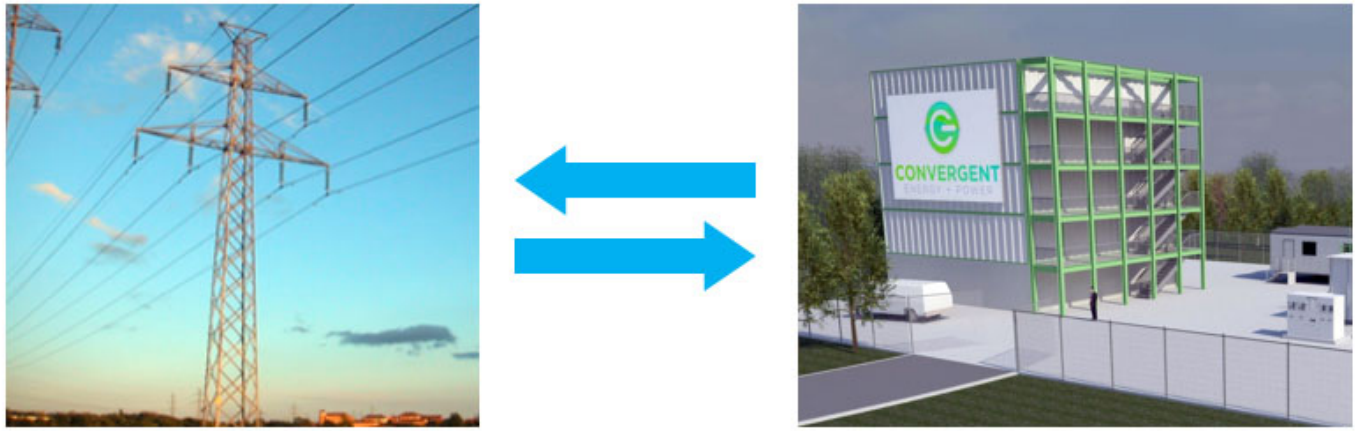
$$\max_{\pi} \mathbb{E}\left\{F(x^{\pi,N}, \hat{W}) \mid S_0\right\}$$

    - We only care about how well the final decision $x^{\pi,N}$ works.
  - » Risk

$$\max_{\pi} \rho\left\{C(S_0, X_0^{\pi}(S_0)), C(S_1, X_1^{\pi}(S_1)), \ldots, C(S_T, X_T^{\pi}(S_T)) \mid S_0\right\}$$

Energy storage example

- Objective function



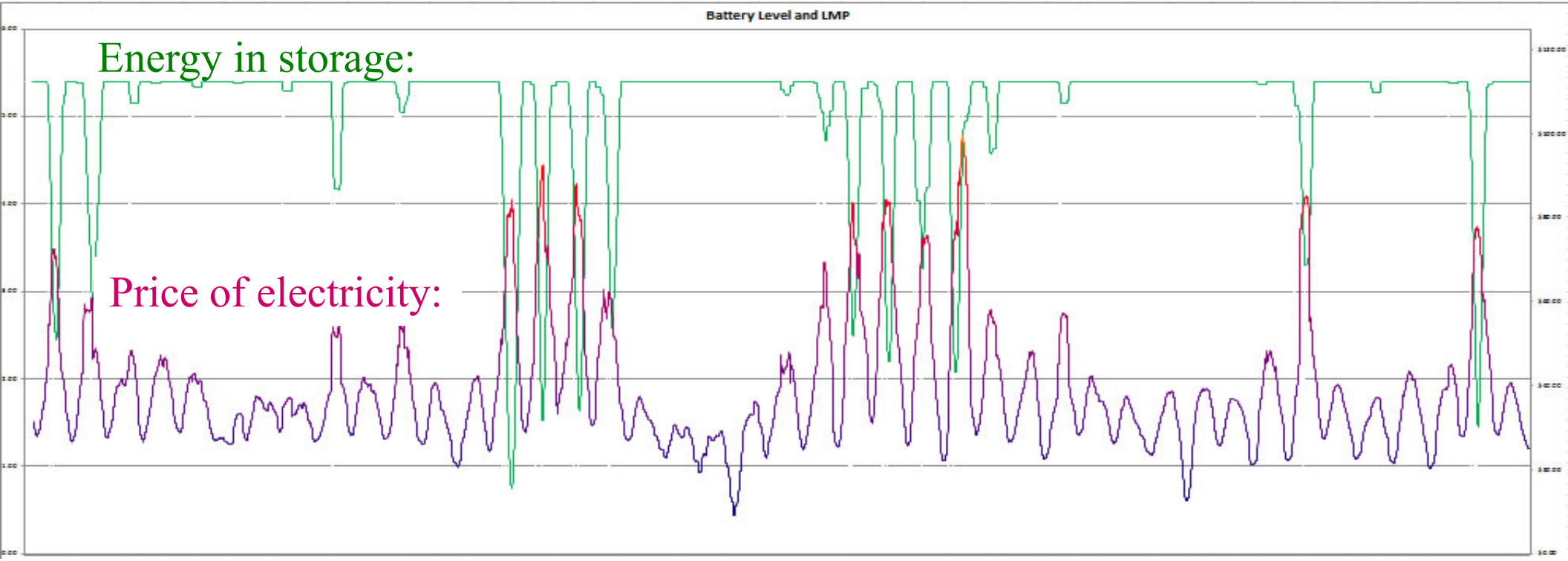
» $C(S_t, x_t) = p_t x_t$

» $\max_{\pi} \mathbb{E}\{\sum_{t=0}^{T} C(S_t, X^{\pi}(S_t)) \,|S_0\}$

Energy arbitrage

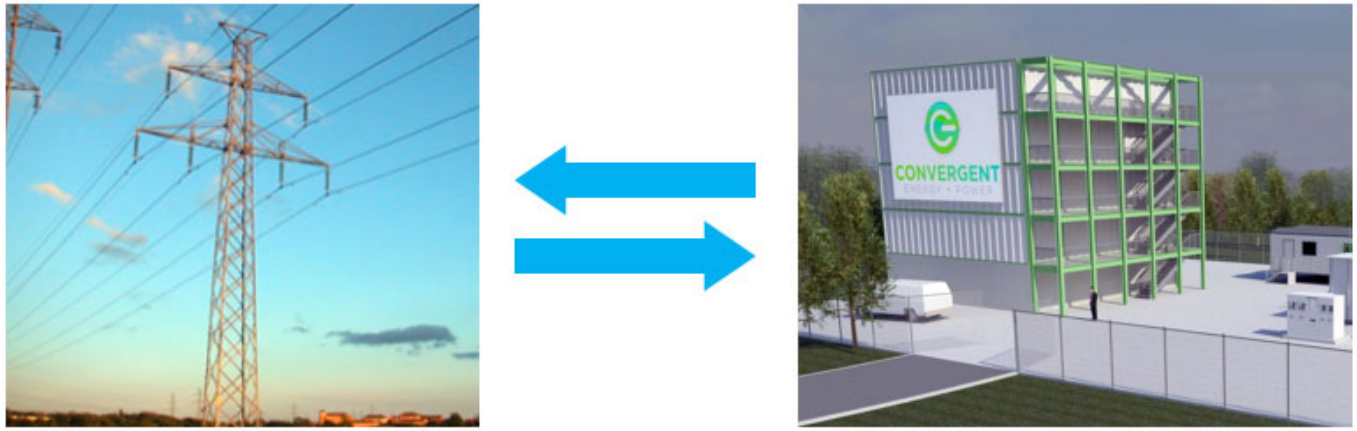
- Our policy function might be the parametric model (this is nonlinear in the parameters):

$$X^{\pi}(S_t \mid \rho) = \begin{cases} +1 & \text{if } p_t < \rho^{\text{charge}} \\ 0 & \text{if } \rho^{\text{charge}} < p_t < \rho^{\text{discharge}} \\ -1 & \text{if } p_t > \rho^{\text{charge}} \end{cases}$$



Energy arbitrage

- Objective function



» Now the policy search is to find the best parameters ρ:

$$\max_{\rho} \mathbb{E}\left\{\sum_{t=0}^{T} C(S_t, X^{\pi}(S_t|\rho))\,|S_0\right\}$$

Elements of a dynamic model

- The complete model:
  - » Objective function
    - Cumulative reward ("online learning")

$$\max_\pi \mathbb{E}\left\{\sum_{t=0}^{T} C_t\left(S_t, X_t^\pi(S_t), W_{t+1}\right) \mid S_0\right\}$$

    - Final reward ("offline learning")

$$\max_\pi \mathbb{E}\left\{F(x^{\pi,N}, \hat{W}) \mid S_0\right\}$$

    - Risk:

$$\max_\pi \rho\left\{C(S_0, X_0^\pi(S_0)), C(S_1, X_1^\pi(S_1)), \ldots, C(S_T, X_T^\pi(S_T)) \mid S_0\right\}$$

  - » Transition function:

$$S_{t+1} = S^M\left(S_t, x_t, W_{t+1}\right)$$

  - » Exogenous information:

$$\left(S_0, W_1, W_2, \ldots, W_T\right)$$

Elements of a dynamic model

- Deterministic
  - » Objective function

$$\min_{x_0,\ldots,x_T} \sum_{t=0}^{T} c_t x_t$$

  - » Decision variables:

$$(x_0,\ldots,x_T)$$

  - » Constraints:
    - at time t

$$\left.\begin{aligned} A_t x_t &= R_t \\ x_t &\geq 0 \end{aligned}\right\} \mathcal{X}_t$$

    - Transition function

$$R_{t+1} = b_{t+1} + B_t x_t$$

- Stochastic
  - » Objective function

$$\max_\pi E\left\{\sum_{t=0}^{T} C_t\left(S_t, X_t^\pi(S_t), W_{t+1}\right) \mid S_0\right\}$$

  - » Policy

$$X^\pi : S \mapsto \mathcal{X}$$

  - » Constraints at time t

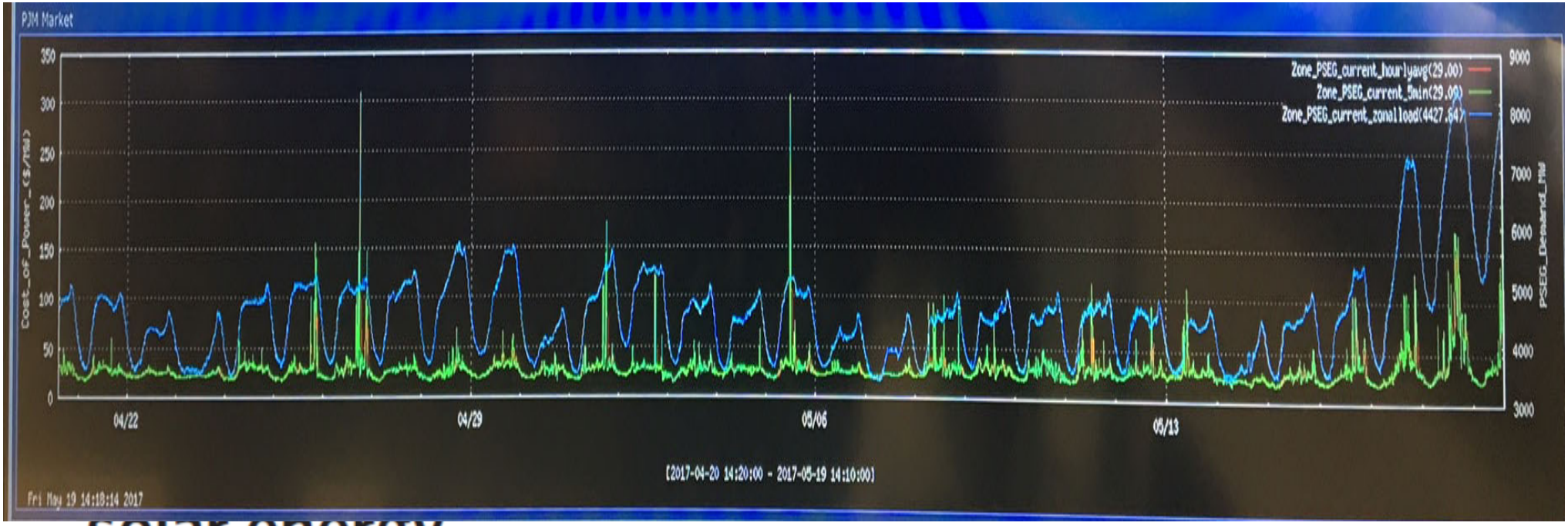
$$x_t = X_t^\pi(S_t) \in \mathcal{X}_t$$

  - » Transition function

$$S_{t+1} = S^M\left(S_t, x_t, W_{t+1}\right)$$

  - » Exogenous information

$$(S_0, W_1, W_2, \ldots, W_T)$$



solar energy

$$R_{\text{total}} = \max\left(\sum_{t=0}^{N} P(t)\left(x_{\text{generation}}(t) + x_{\text{discharge}}(t) - x_{\text{charge}}(t)/\eta\right)\right)$$

subject to:

Stochastic prices

Charge/discharge decisions

$$0 \le x_{\text{discharge}} \le \dot{E}_{\max}$$

$$0 \le x_{\text{charge}} \le \min(\eta x_{\text{generation}}(t), \eta \dot{E}_{\max})$$

$$0 \le \sum_{t=0}^{N} \left(x_{\text{charge}}(t) - x_{\text{discharge}}(t)\right) \le h\dot{E}_{\max} \qquad (2)$$

From deterministic to stochastic

- Imagine that you would like to solve the time-dependent linear program:

$$\max_{x_0,\ldots,x_T} \sum_{t=0}^{T} c_t x_t$$

» subject to

$$A_0 x_0 = b_0$$

$$A_t x_t - B_{t-1} x_{t-1} = b_t, \quad t \geq 1.$$

$$R_{\text{total}} = \max \left(\sum_{t=0}^{N} P(t)\left(x_{\text{generation}}(t) + x_{\text{discharge}}(t) - x_{\text{charge}}(t)/\eta\right) \right)$$

- We can convert this to a proper stochastic model by replacing x_t with $X_t^\pi(S_t)$ and taking an expectation:

$$\max_\pi \mathbb{E} \sum_{t=0}^{T} c_t X_t^\pi(S_t) \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=0}^{T} c_t X_t^\pi(S_t^n(\omega^n))$$

The policy $X_t^\pi(S_t)$ has to satisfy $A_t x_t = R_t$ with transition function:

$$S_{t+1} = S^M\left(S_t, x_t, W_{t+1}\right)$$

Outline

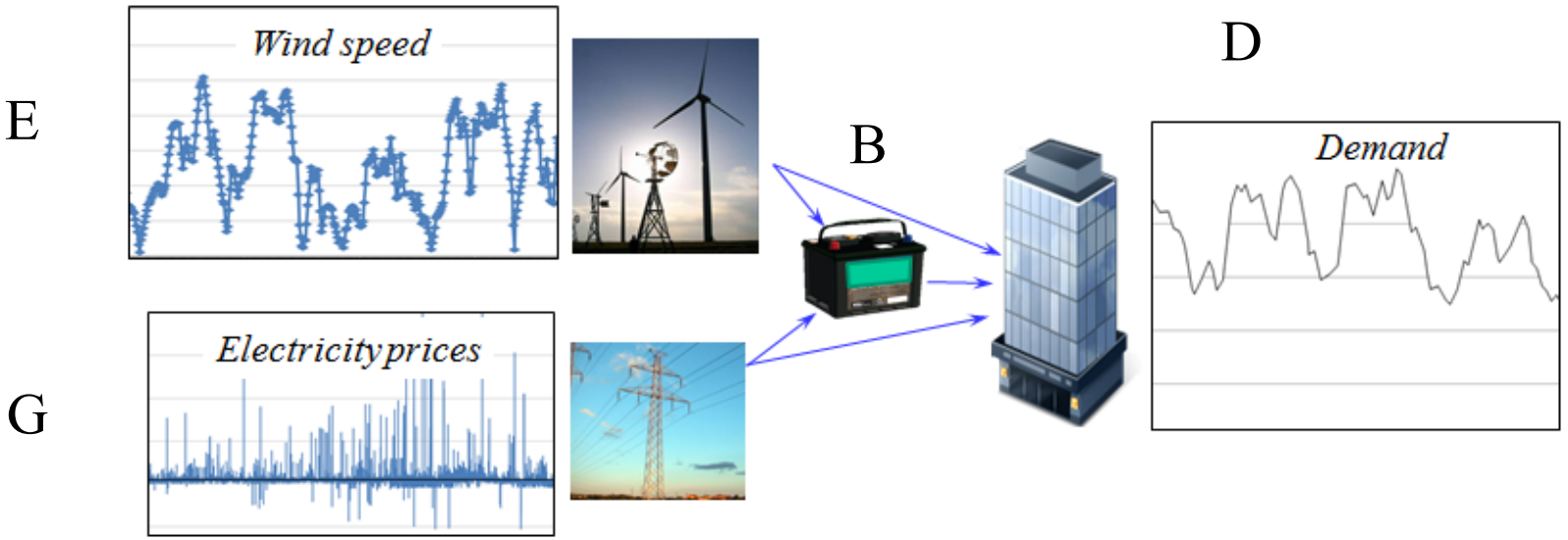
- Elements of a dynamic model
- An energy storage illustration
- Modeling uncertainty
- Designing policies
- Educational materials

An energy storage example

A more complex model

An energy storage problem

- State variables

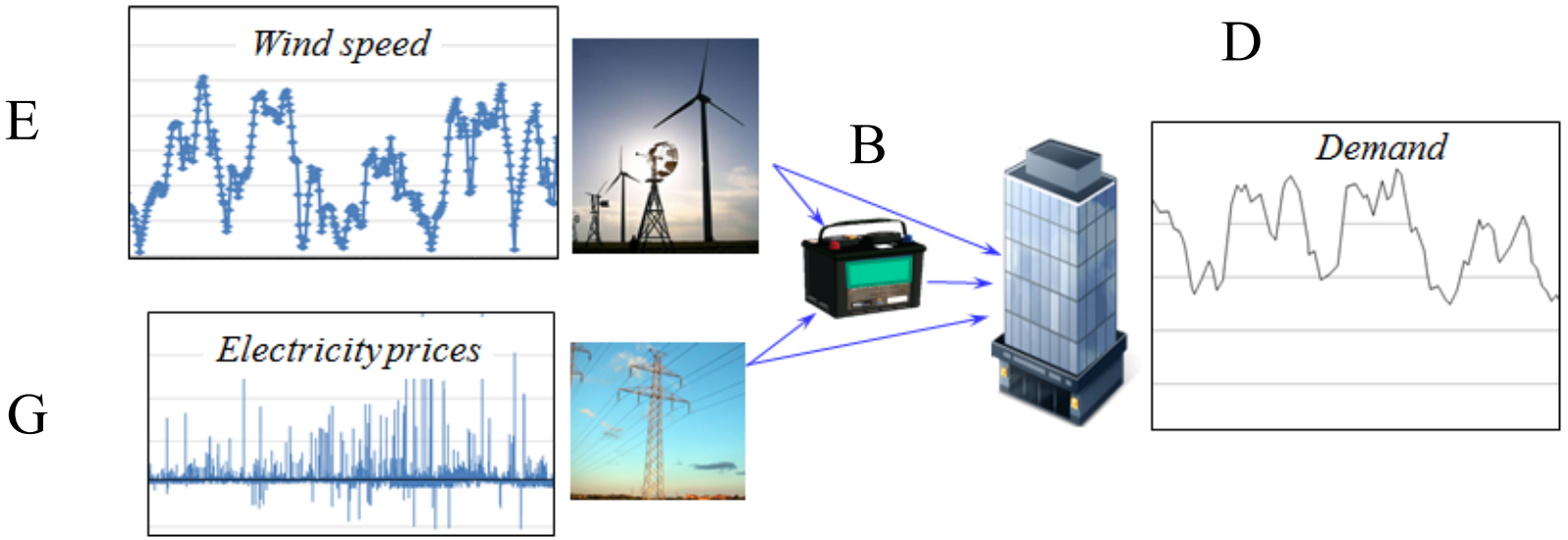


» We will present the full model, accumulating the information we need in the state variable.

» We will highlight information we need as we proceed. This information will make up our state variable.

An energy storage problem

- Decision variables



$$x_t = \left(x_t^{EL}, x_t^{EB}, x_t^{GL}, x_t^{GB}, x_t^{BL},\right)$$

» Constraints;

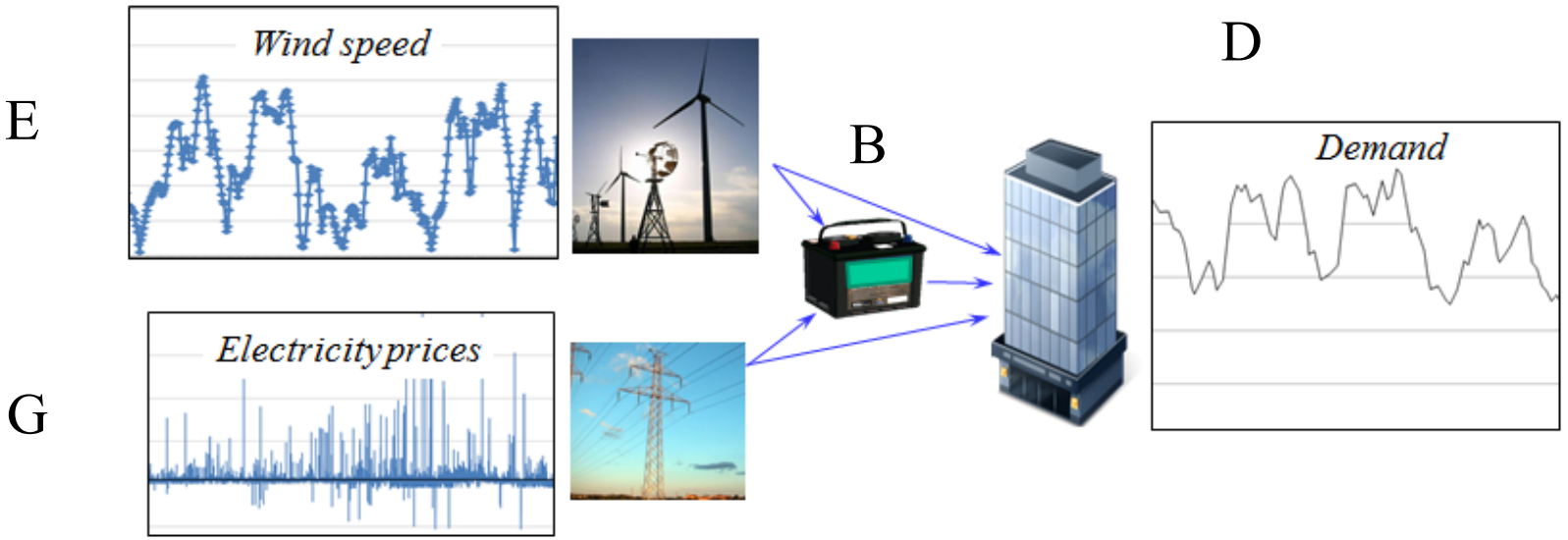
$$x_t^{EL} + x_t^{EB} \leq E_t$$

$$x_t^{GL} + x_t^{EL} + x_t^{BL} = D_t$$

$$x_t^{BL} \leq R_t$$

An energy storage problem

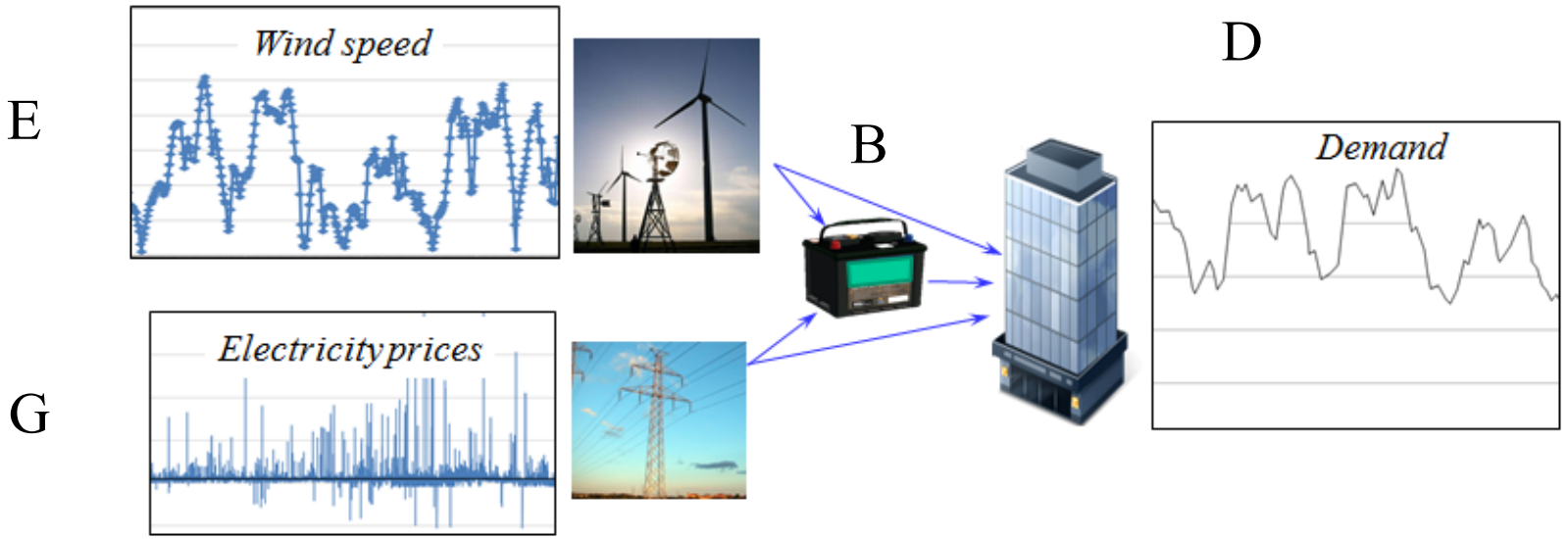
- Exogenous information



$$W_t = \begin{cases} \hat{E}_t = \text{Change in energy from wind between } t-1 \text{ and } t \\ \hat{D}_t = \text{Observed demand at time } t \\ \hat{p}_t = \text{Change in price between } t-1 \text{ and } t \end{cases}$$

An energy storage problem

- Transition function



$$R_{t+1}^{battery} = R_t^{battery} + x_t$$

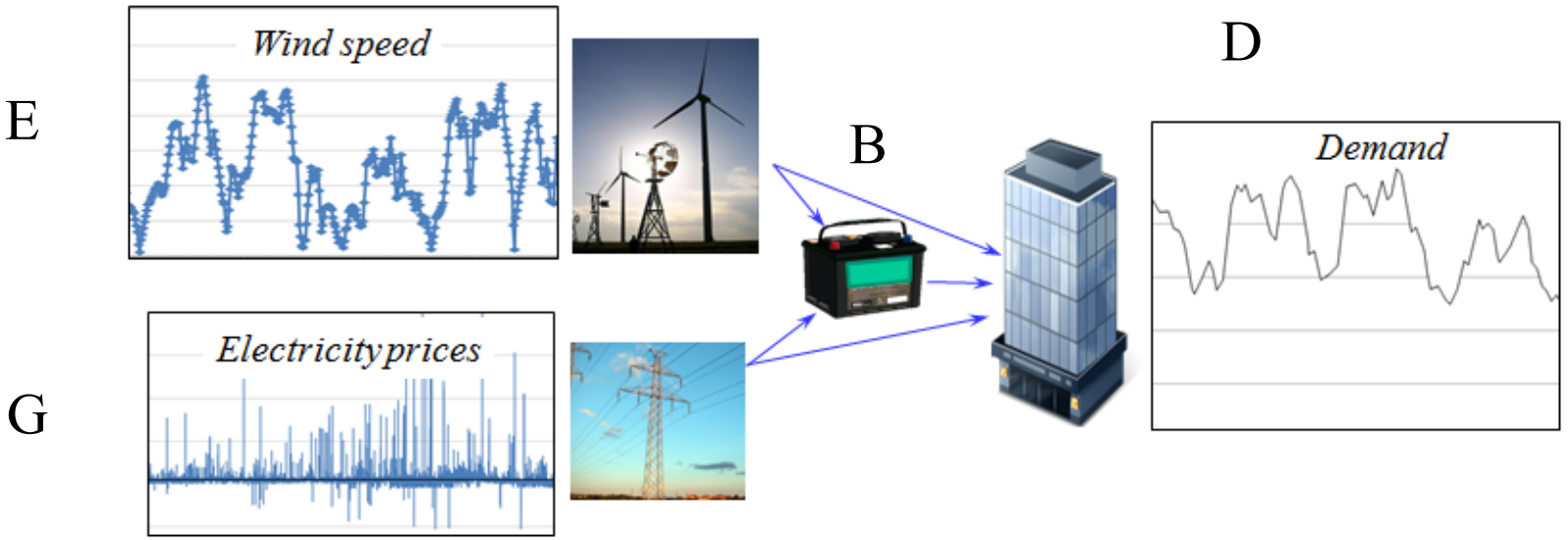
$$E_{t+1} = E_t + \hat{E}_{t+1}$$

$$p_{t+1} = p_t + \hat{p}_{t+1}$$

$$D_{t+1} = \hat{D}_{t+1}$$

An energy storage problem

- Objective function



$$C(S_t, x_t) = p_t \left(x_t^{GB} + x_t^{GL} \right)$$

$$\min_\pi \mathbb{E} \left\{ \sum_{t=0}^{T} C_t(S_t, X_t^\pi(S_t)) \mid S_0 \right\}$$

An energy storage problem

- State variables

$$S_t = (R_t, D_t, E_t, p_t)$$

  - » Cost function

    p_t = Price of electricity

  - » Decision function

    Constraints:

$$x_t^{EL} + x_t^{EB} \le E_t$$

$$x_t^{GL} + x_t^{EL} + x_t^{BL} = D_t$$

$$x_t^{BL} \le R_t$$

  - » Transition function

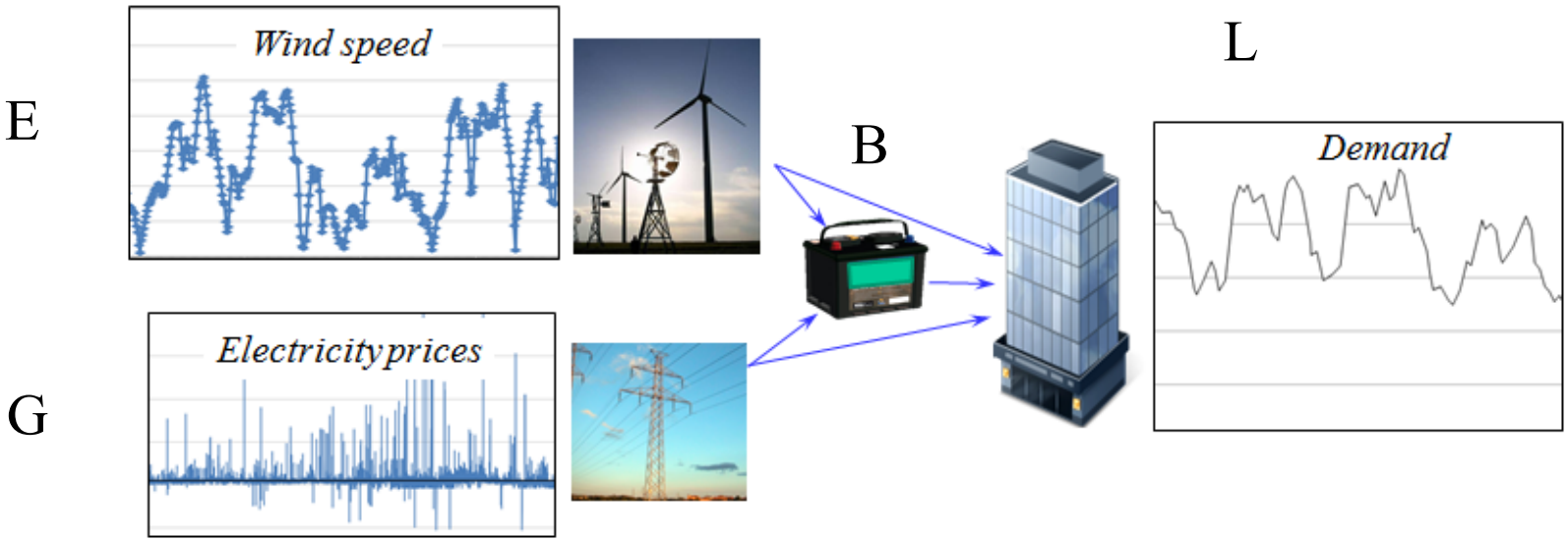
$$p_{t+1} = p_t + \hat{p}_{t+1}$$

An energy storage example

A time series price model

An energy storage problem

- Transition function



» ARIMA price model:

$$p_{t+1} = \theta_0 p_t + \theta_1 p_{t-1} + \theta_2 p_{t-2} + \varepsilon^p_{t+1}$$

An energy storage problem

- State variables

$$S_t = \left(R_t, D_t, E_t, (p_t, p_{t-1}, p_{t-2})\right)$$

  - » Cost function

    p_t = Price of electricity

  - » Decision function

    Constraints:

$$x_t^{EL} + x_t^{EB} \le E_t$$

$$x_t^{GL} + x_t^{EL} + x_t^{BL} = D_t$$

$$x_t^{BL} \le R_t$$

  - » Transition function

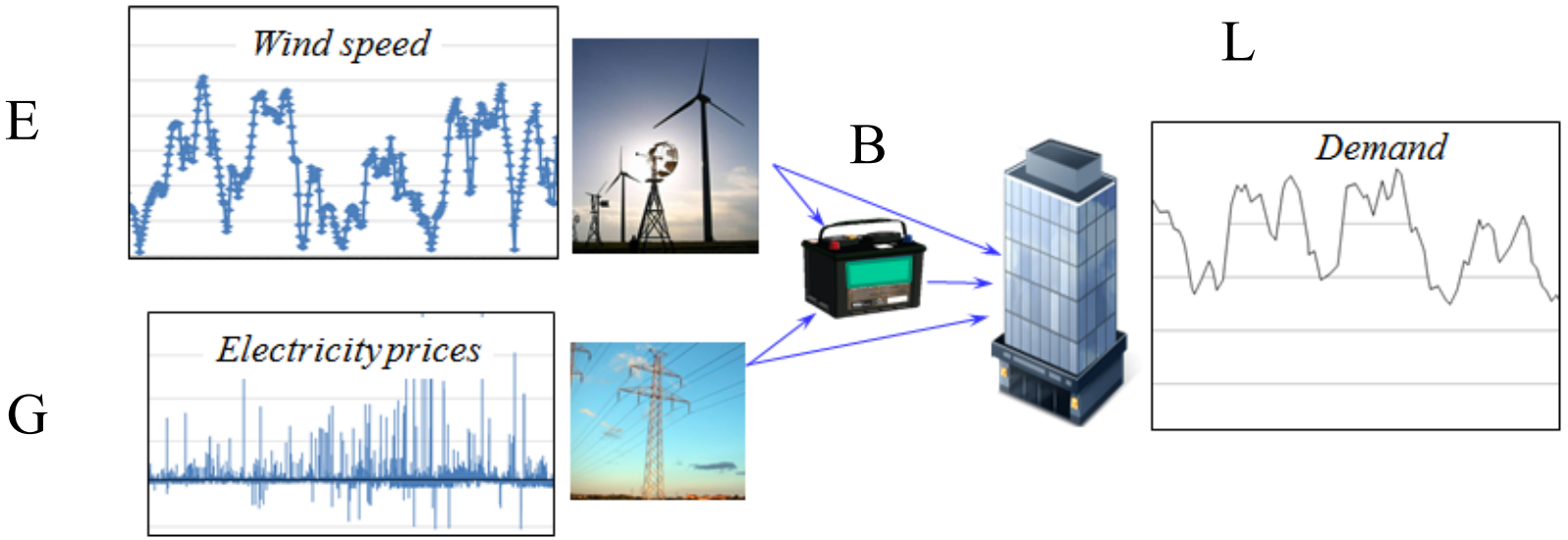
$$p_{t+1} = \theta_0 p_t + \theta_1 p_{t-1} + \theta_2 p_{t-2} + \varepsilon_{t+1}^p$$

An energy storage example

(Passive) learning the price process

An energy storage problem

- Transition function



» ARIMA price model with learning:

$$p_{t+1} = \bar{\theta}_{t0} p_t + \bar{\theta}_{t1} p_{t-1} + \bar{\theta}_{t2} p_{t-2} + \varepsilon^p_{t+1}$$

Need to learn $\bar{\theta}_{ti}$

Learning in stochastic optimization

- Updating the demand parameter
  - » Let p_{t+1} be the new price and let

$$\bar{F}_t^{price}(\bar{p}_t \mid \bar{\theta}_t) = (\bar{\theta}_t)^T \bar{p}_t = \bar{\theta}_{t0} p_t + \bar{\theta}_{t1} p_{t-1} + \bar{\theta}_{t2} p_{t-2}$$

  - » We update our estimate $\bar{\theta}_t$ using our recursive least squares equations:

$$\bar{\theta}_{t+1} = \bar{\theta}_t + \frac{1}{\gamma_t} M_t \bar{p}_t \varepsilon_{t+1}$$

$$\varepsilon_{t+1} = \bar{F}_t^{price}(\bar{p}_t \mid \bar{\theta}_t) - p_{t+1}$$

$$M_{t+1} = M_t - \frac{1}{\gamma_t} M_t \bar{p}_t (\bar{p}_t)^T M_t$$

$$\gamma_{t+1} = 1 - (\bar{p}_t)^T M_t \bar{p}_t$$

An energy storage problem

- State variables

  » Cost function

  p_t = Price of electricity

  » Decision function

  Constraints:

$$x_t^{EL} + x_t^{EB} \le E_t$$

$$x_t^{GL} + x_t^{EL} + x_t^{BL} = D_t$$

$$x_t^{BL} \le R_t$$

  » Transition function

$$p_{t+1} = \theta_0 p_t + \theta_1 p_{t-1} + \theta_2 p_{t-2} + \varepsilon_{t+1}^p$$

$$\bar{\theta}_{t+1} = \bar{\theta}_t + \frac{1}{\gamma_t} M_t \bar{p}_t \varepsilon_{t+1}$$

$$S_t = \left((R_t, D_t, E_t), (p_t, p_{t-1}, p_{t-2}), (\bar{\theta}_t, M_t) \right)$$

Physical state variables

Other information I_t

Belief state B_t

An energy storage problem

- Types of learning:
  - » No learning ($\theta's$ are known)

$$p_{t+1} = \theta_0 p_t + \theta_1 p_{t-1} + \theta_2 p_{t-2} + \varepsilon^p_{t+1}$$

  - » Passive learning (learn θs from price data)

$$p_{t+1} = \bar{\theta}_{t0} p_t + \bar{\theta}_{t1} p_{t-1} + \bar{\theta}_{t2} p_{t-2} + \varepsilon^p_{t+1}$$

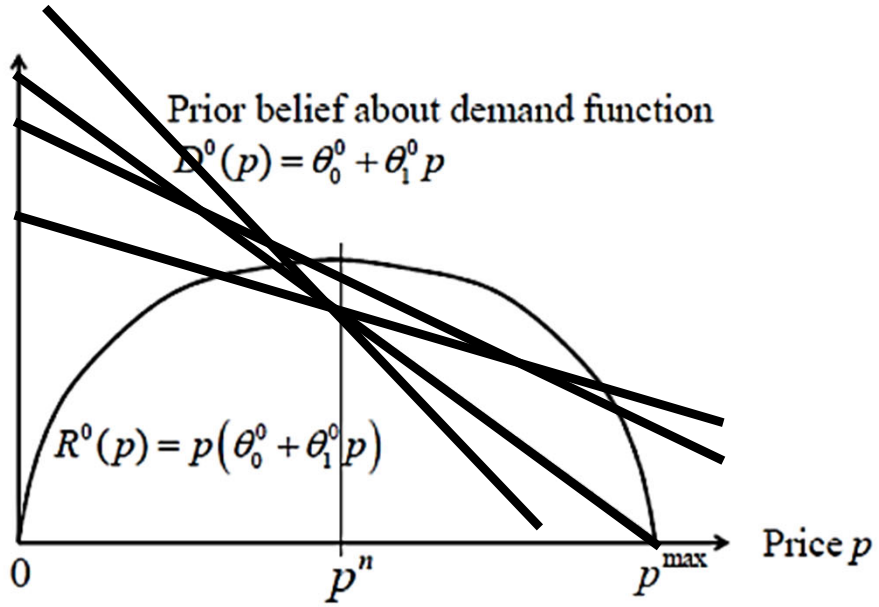
  - » Active learning ("bandit problems")

$$p_{t+1} = \bar{\theta}_{t0} p_t + \bar{\theta}_{t1} p_{t-1} + \bar{\theta}_{t2} p_{t-2} + \bar{\theta}_{t3} x^{GB}_t + \varepsilon^p_{t+1}$$

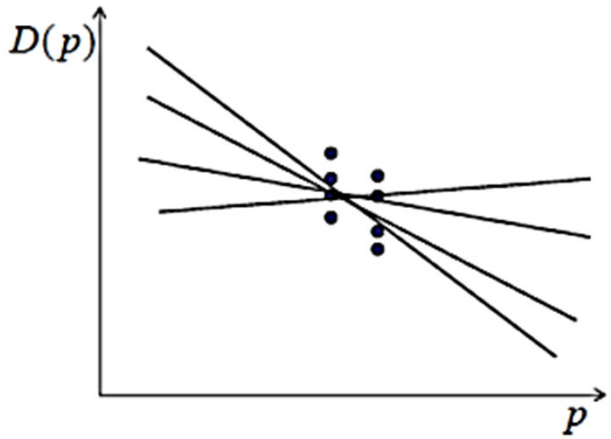
Modeling the active learning is the same as passive, but the choice of policy will change.

An energy storage problem

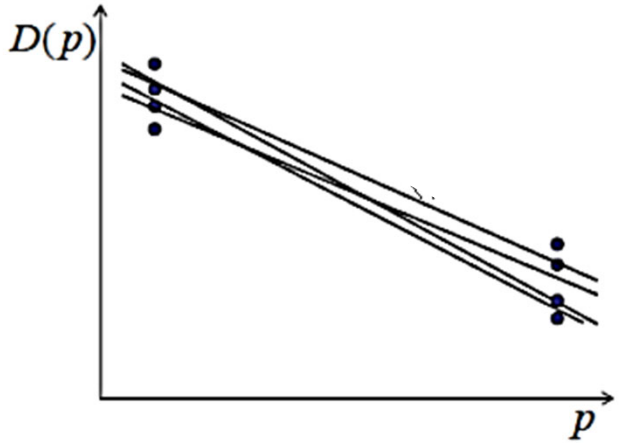
- Learning a demand response function



Sampling to maximize revenue:



Sampling to learn demand response:

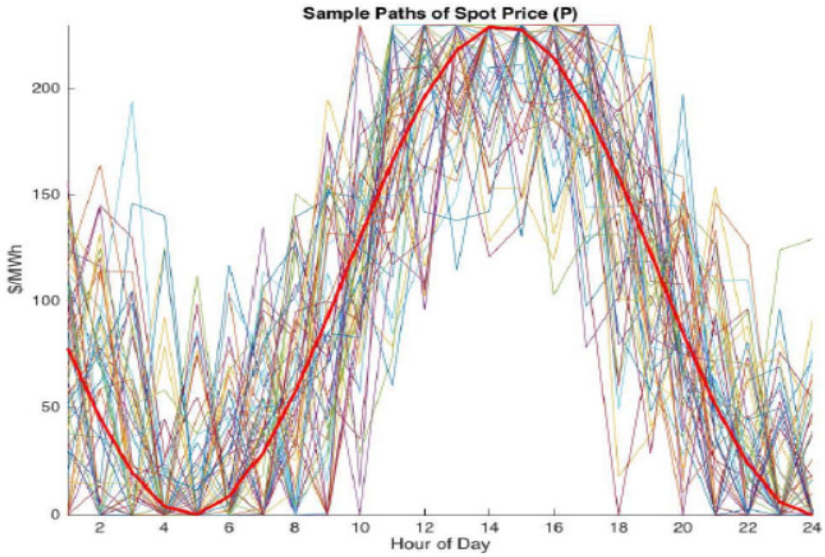


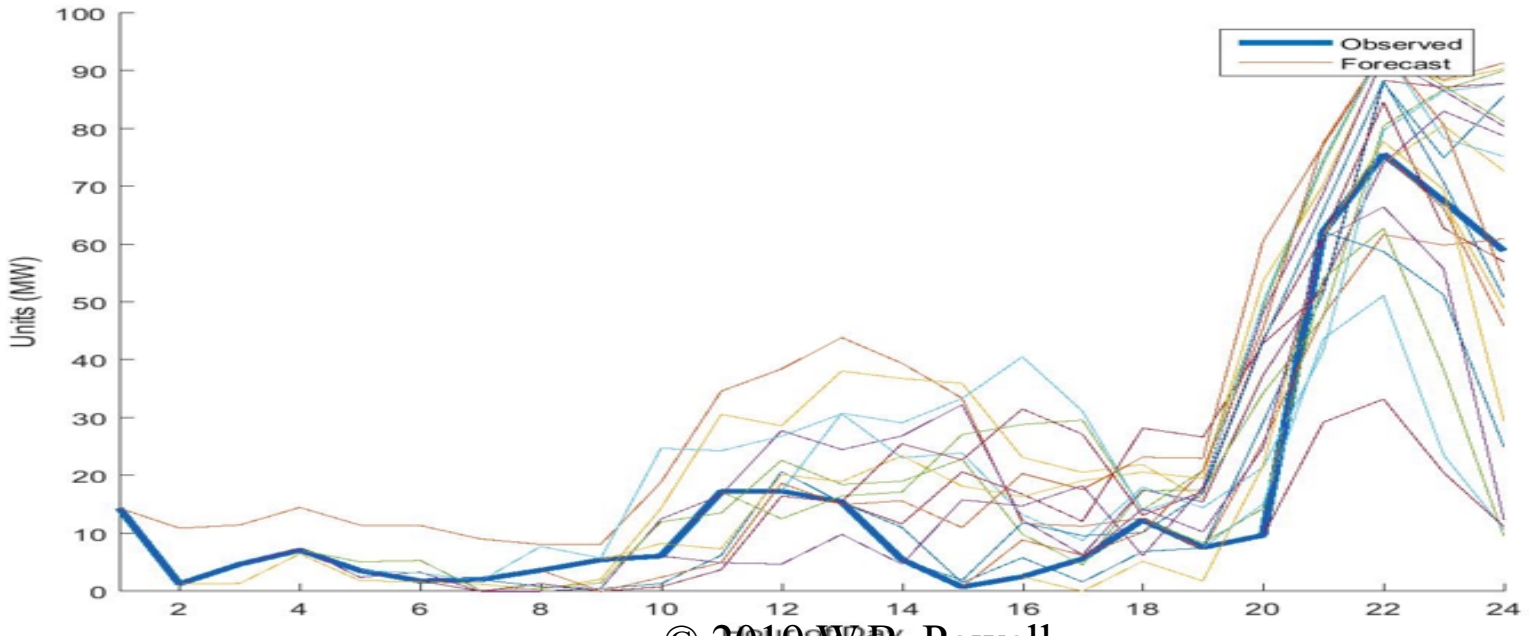
An energy storage example

Planning with rolling forecasts

Parametric cost function approximation

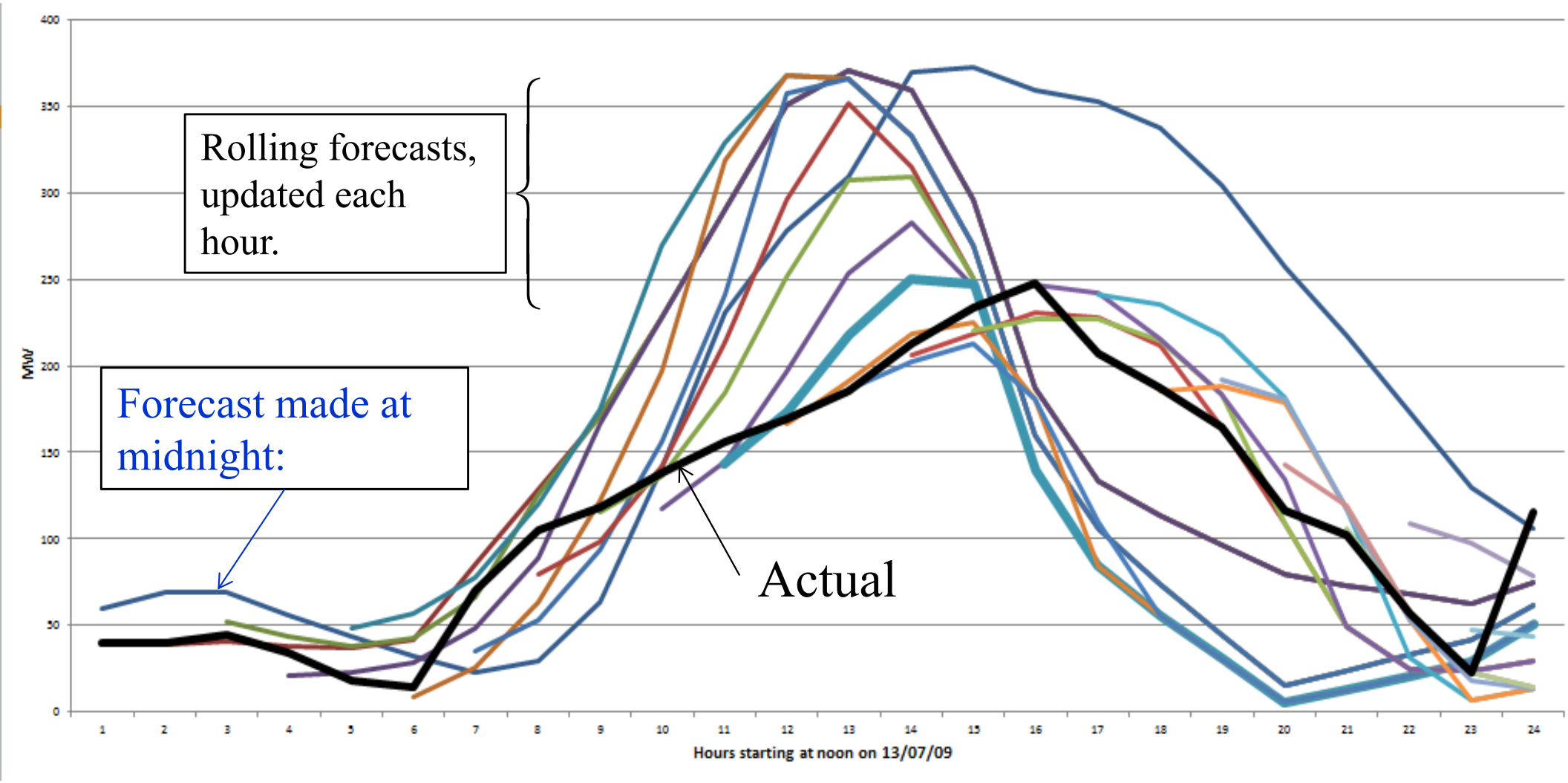
- An energy storage problem:





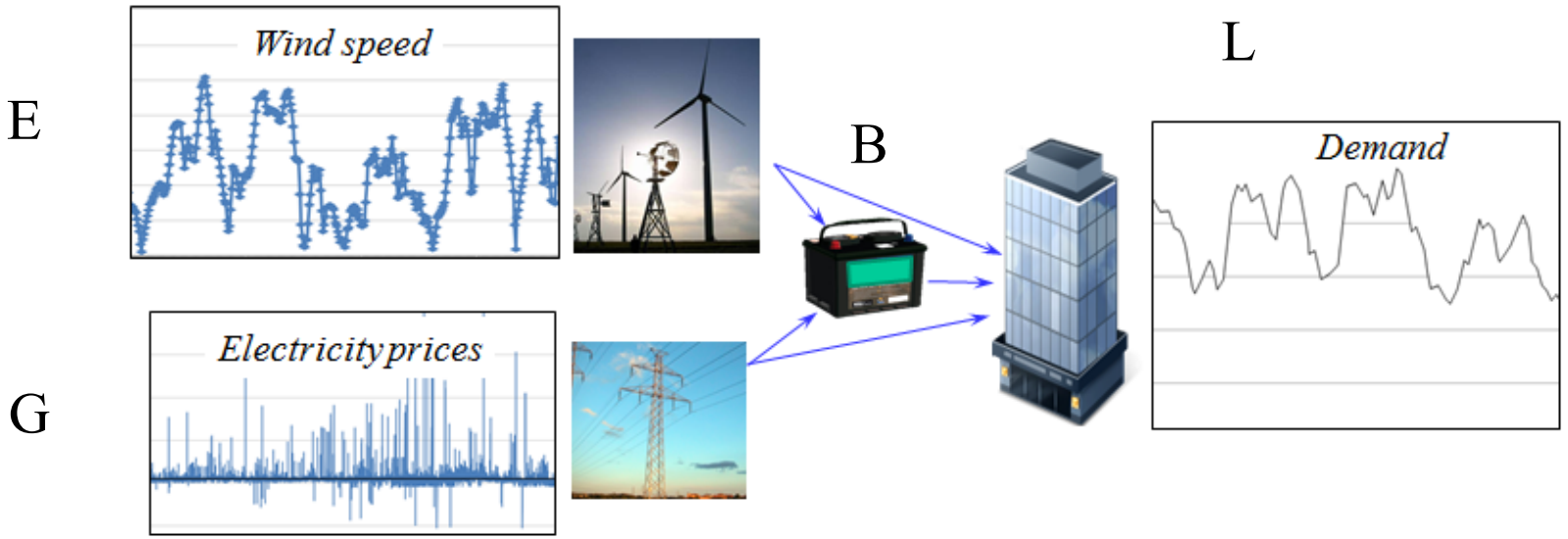
Parametric cost function approximation

- Forecasts evolve over time as new information arrives:



An energy storage problem

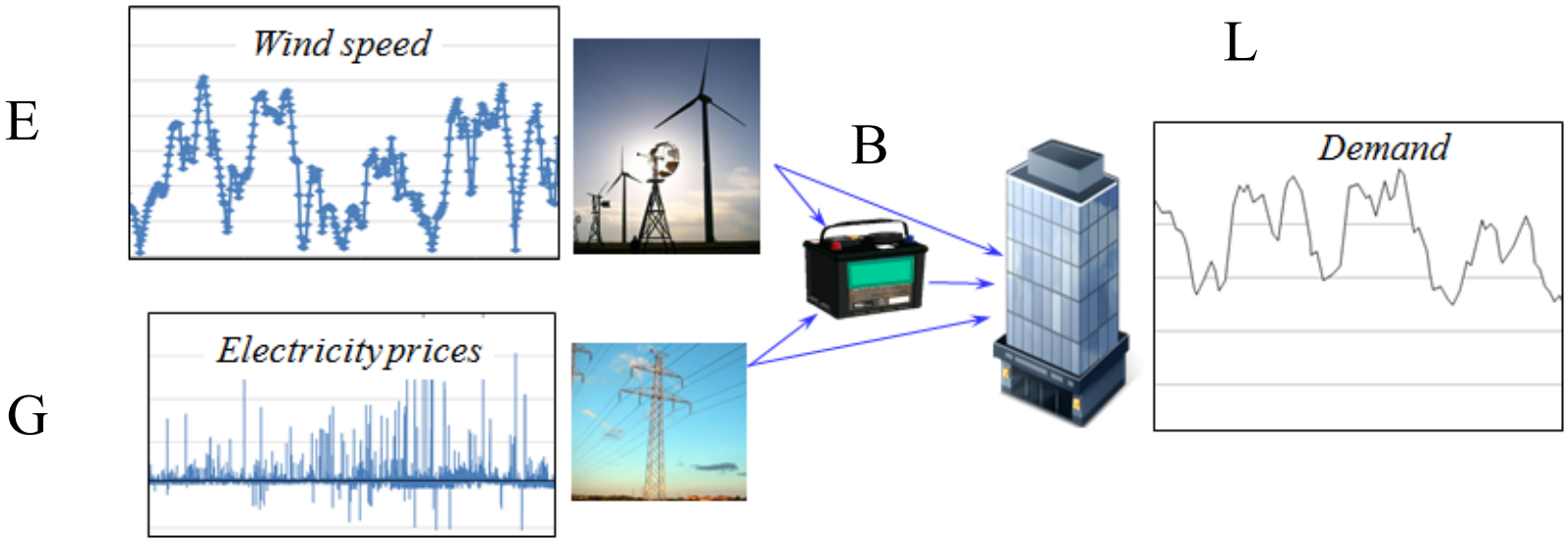
- Exogenous information



$$W_t = \begin{cases} \varepsilon^f_{t,t'} = \text{Difference between forecasted energy } E_{t'} \text{ at time} \\ \qquad t' \text{ made at } t-1 \text{ and } t \end{cases}$$

An energy storage problem

- Transition function



$$E_{t+1} = f^E_{t,t+1} + \varepsilon^E_{t+1,t+1} \qquad \varepsilon^E_{t+1,t+1} \sim N(0,\sigma^2_\varepsilon)$$

$$f^E_{t+1,t'} = f^E_{t,t'} + \varepsilon^E_{t+1,t'},\ t' > t+1 \qquad \varepsilon^E_{t+1,t'} \sim N(0,(t'-(t+1))\sigma^2_\varepsilon)$$

An energy storage problem

- State variables

$$S_t = \left(R_t, D_t, E_t, (p_t, p_{t-1}, p_{t-2}), (\bar{\theta}_t, M_t), f_t^E\right)$$

  - » Cost function

    p_t = Price of electricity

  - » Decision function

    Constraints:

$$x_t^{EL} + x_t^{EB} \le E_t$$

$$x_t^{GL} + x_t^{EL} + x_t^{BL} = D_t$$

$$x_t^{BL} \le R_t$$

  - » Transition function

$$p_{t+1} = \theta_0 p_t + \theta_1 p_{t-1} + \theta_2 p_{t-2} + \varepsilon_{t+1}^p$$

$$\bar{\theta}_{t+1} = \bar{\theta}_t + \frac{1}{\gamma_t} M_t \bar{p}_t \varepsilon_{t+1} \qquad f_{t+1,t'}^E = f_{t,t'}^E + \varepsilon_{t+1,t'}^E$$

Outline

- Elements of a dynamic model
- An energy storage illustration
- Modeling uncertainty
- Designing policies
- Educational materials

Modeling uncertainty

- Classes of uncertainty
  - » Observational uncertainty
  - » Prognostic uncertainty (forecasting)
  - » Experimental noise/variability
  - » Transitional uncertainty
  - » Inferential uncertainty
  - » Model uncertainty
  - » Systematic exogenous uncertainty
  - » Control/implementation uncertainty
  - » Algorithmic noise
  - » Goal uncertainty

Modeling uncertainty in the context of stochastic optimization is a relatively untapped area of research.

Outline

- Elements of a dynamic model
- An energy storage illustration
- Modeling uncertainty
- Designing policies
- Educational materials

Designing policies

- We have to start by describing what we mean by a policy.
  - » Definition:

    A policy is a mapping from a state to an action.

    ... any mapping.

- How do we search over an arbitrary space of policies?

Designing policies

- "Policies" and the English language

Behavior	Formula	Principle
Belief	Habit	Procedure
Bias	Laws/bylaws	Process
Commandment	Manner	Protocols
Conduct	Method	Recipe
Control law	Mode	Ritual
Convention	Mores	Rule
Culture	Patterns	Style
Customs	Plans	Technique
Dogma	Policies	Tenet
Etiquette	Practice	Tradition
Fashion	Prejudice	Way of life

Designing policies

- Two fundamental strategies:

Policy search – Search over a class of functions for making decisions to optimize some metric.

$$\max_{\pi=(f\in F,\theta^f\in\Theta^f)} E\left\{\sum_{t=0}^{T} C\left(S_t, X_t^{\pi}(S_t\,|\,\theta)\right)|\, S_0\right\}$$

Lookahead approximations – Approximate the impact of a decision now on the future.

$$X_t^*(S_t) = \arg\max_{x_t}\left(C(S_t,x_t) + \mathbb{E}\left\{\max_{\pi\in\Pi}\left\{\mathbb{E}\sum_{t'=t+1}^{T} C(S_{t'}, X_{t'}^{\pi}(S_{t'}))\,|\,S_{t+1}\right\}|\,S_t,x_t\right\}\right)$$

Designing policies

- **Policy search**:

1) Policy function approximations (PFAs) $x_t = X^{PFA}(S_t \mid \theta)$
   - Lookup tables
     - "when in this state, take this action"
   - Parametric functions
     - Order-up-to policies: if inventory is less than s, order up to S.
     - Affine policies - $x_t = X^{PFA}(S_t \mid \theta) = \sum_{f \in F} \theta_f \phi_f(S_t)$
     - Shallow neural networks
   - Locally/semi/non parametric
     - Kernel regression
     - Deep neural networks

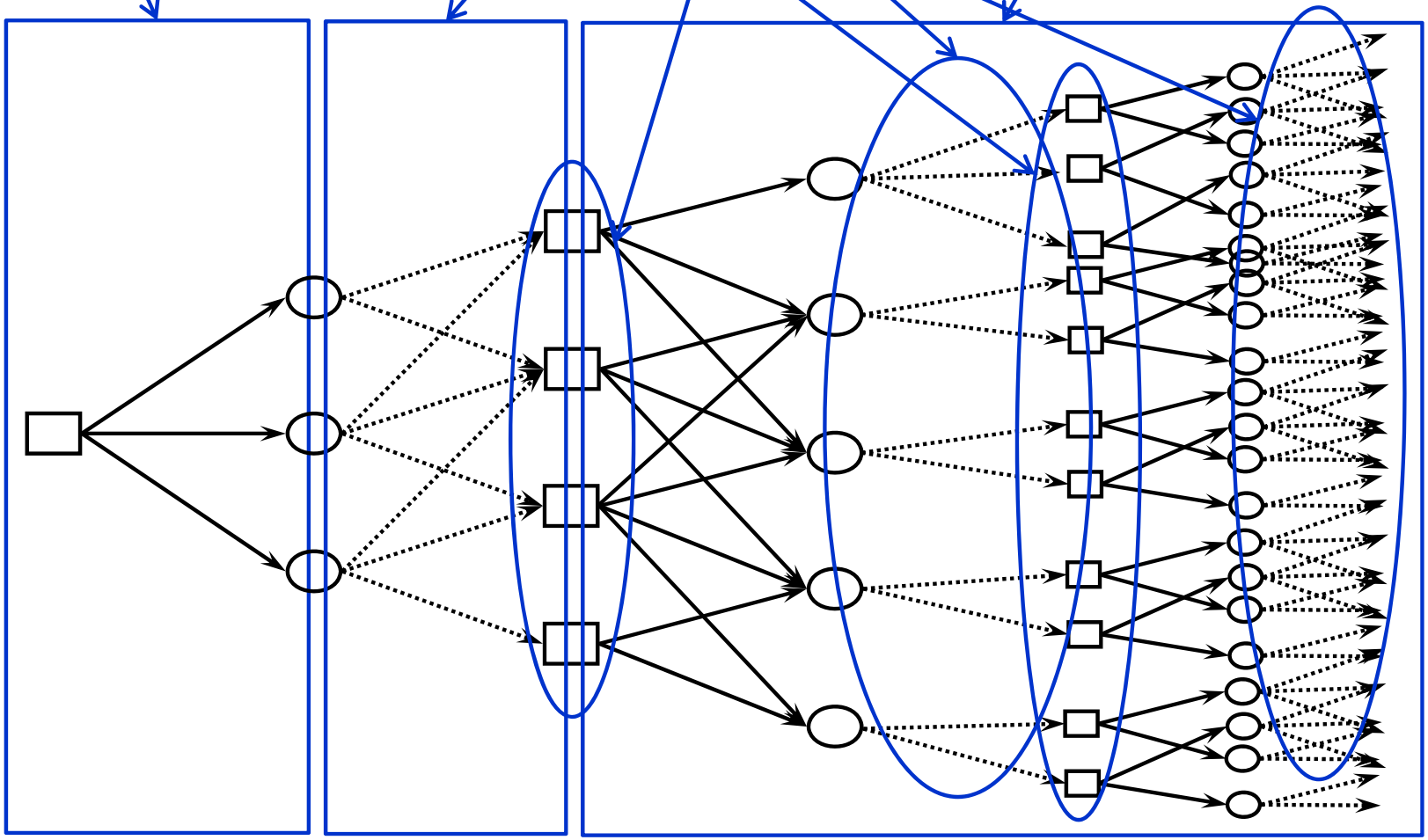
2) Cost function approximations (CFAs)
   - Optimizing a deterministic model modified to handle uncertainty (buffer stocks, schedule slack)

$$X^{CFA}(S_t \mid \theta) = \arg\max_{x_t \in \bar{X}_t^{\pi}(\theta)} \bar{C}^{\pi}(S_t, x_t \mid \theta)$$

Designing policies

- The ultimate lookahead policy is optimal

$$X_t^*(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \mathbb{E}\left\{ \max_{\pi \in \Pi} \left\{ \mathbb{E} \sum_{t'=t+1}^{T} C(S_{t'}, X_{t'}^{\pi}(S_{t'})) \mid S_{t+1} \right\} \mid S_t, x_t \right\} \right)$$



Designing policies

- **Lookahead approximations** – Approximate the impact of a decision now on the future:
  - » An optimal policy (based on looking ahead):

$$X_t^*(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \mathbb{E}\left\{ \max_{\pi \in \Pi} \left\{ \mathbb{E} \sum_{t'=t+1}^{T} C(S_{t'}, X_{t'}^{\pi}(S_{t'})) \mid S_{t+1} \right\} \mid S_t, x_t \right\} \right)$$

3) Approximating the value of being in a downstream state using machine learning ("value function approximations")

$$X_t^*(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \mathbb{E}\left\{ V_{t+1}(S_{t+1}) \mid S_t, x_t \right\} \right)$$

$$X_t^{VFA}(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \mathbb{E}\left\{ \bar{V}_{t+1}(S_{t+1}) \mid S_t, x_t \right\} \right)$$

$$= \arg\max_{x_t} \left(C(S_t, x_t) + \bar{V}_t^x(S_t^x) \right)$$

$$= \arg\max_{x_t} \bar{Q}_t(S_t, x_t) \quad \text{("Q-learning")}$$

Designing policies

- **Lookahead approximations** – Approximate the impact of a decision now on the future:
  - » An optimal policy (based on looking ahead):

$$X_t^*(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \mathbb{E}\left\{ \max_{\pi \in \Pi} \left\{ \mathbb{E} \sum_{t'=t+1}^{T} C(S_{t'}, X_{t'}^{\pi}(S_{t'})) \mid S_{t+1} \right\} \mid S_t, x_t \right\} \right)$$

3) Approximating the value of being in a downstream state using machine learning ("value function approximations")

$$X_t^*(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \mathbb{E}\left\{ V_{t+1}(S_{t+1}) \mid S_t, x_t \right\} \right)$$

$$X_t^{VFA}(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \mathbb{E}\left\{ \bar{V}_{t+1}(S_{t+1}) \mid S_t, x_t \right\} \right)$$

$$= \arg\max_{x_t} \left(C(S_t, x_t) + \bar{V}_t^x(S_t^x) \right)$$

$$= \arg\max_{x_t} \bar{Q}_t(S_t, x_t) \quad \text{("Q-learning")}$$

Designing policies

- **Lookahead approximations** – Approximate the impact of a decision now on the future:
  - » An optimal policy (based on looking ahead):

$$X_t^*(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \mathbb{E}\left\{ \max_{\pi \in \Pi} \left\{ \mathbb{E} \sum_{t'=t+1}^{T} C(S_{t'}, X_{t'}^{\pi}(S_{t'})) \mid S_{t+1} \right\} \mid S_t, x_t \right\} \right)$$

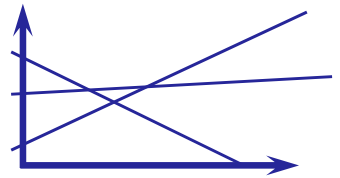
3) Approximating the value of being in a downstream state using machine learning ("value function approximations")

$$X_t^*(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \mathbb{E}\left\{ V_{t+1}(S_{t+1}) \mid S_t, x_t \right\} \right)$$

$$X_t^{VFA}(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \mathbb{E}\left\{ \bar{V}_{t+1}(S_{t+1}) \mid S_t, x_t \right\} \right)$$

$$= \arg\max_{x_t} \left(C(S_t, x_t) + \bar{V}_t^x(S_t^x) \right)$$

$$= \arg\max_{x_t} \bar{Q}_t(S_t, x_t) \quad \text{("Q-learning")}$$



Designing policies

- The ultimate lookahead policy is optimal

$$X_t^*(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \mathbb{E}\left\{ \max_{\pi \in \Pi} \left\{ \mathbb{E} \sum_{t'=t+1}^{T} C(S_{t'}, X_{t'}^{\pi}(S_{t'})) \mid S_{t+1} \right\} \mid S_t, x_t \right\} \right)$$

Maximization that we cannot compute

Expectations that we cannot compute

Designing policies

- The ultimate lookahead policy is optimal

$$X_t^*(S_t) = \arg\max_{x_t}\left(C(S_t,x_t) + \mathbb{E}\left\{ \max_{\pi\in\Pi}\left\{ \mathbb{E}\sum_{t'=t+1}^{T} C(S_{t'}, X_{t'}^{\pi}(S_{t'})) \mid S_{t+1}\right\} \mid S_t, x_t\right\}\right)$$

4) Instead, we have to create an approximation called the *lookahead model:*

$$\left(S_t, x_t, \tilde{W}_{t,t+1}, \tilde{S}_{t,t+1}, \tilde{x}_{t,t+1}, \tilde{W}_{t,t+2}, \ldots., \tilde{S}_{t,t'}, \tilde{x}_{t,t'}, \tilde{W}_{t,t'+1}, \ldots\right)$$

» *A direct lookahead policy* (DLA) works by solving the *lookahead model*:

$$X_t^*(S_t) = \arg\max_{x_t}\left(C(S_t,x_t) + \tilde{\mathbb{E}}\left\{ \max_{\tilde{\pi}\in\tilde{\Pi}}\left\{ \tilde{\mathbb{E}}\sum_{t'=t+1}^{t+H} C(\tilde{S}_{tt'}, \tilde{X}_{tt'}^{\pi}(\tilde{S}_{tt'})) \mid \tilde{S}_{t,t+1}\right\} \mid S_t, x_t\right\}\right)$$

Designing policies

Policy search

1) **Policy function approximations (PFAs)**
   - » Lookup tables, rules, parametric/nonparametric functions
2) **Cost function approximation (CFAs)**
   - » $X^{CFA}(S_t \mid \theta) = \arg\max_{x_t \in \bar{X}_t^{\pi}(\theta)} \bar{C}^{\pi}(S_t, x_t \mid \theta)$

Lookahead approximations

3) **Policies based on value function approximations (VFAs)**
   - » $X_t^{VFA}(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \bar{V}_t^{x} \left(S_t^{x}(S_t, x_t) \right) \right)$
4) **Direct lookahead policies (DLAs)**
   - » *Deterministic lookahead/rolling horizon proc./model predictive control*

$$X_t^{LA-D}(S_t) = \arg\max_{\tilde{x}_{tt}, \ldots, \tilde{x}_{t,t+H}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}, \tilde{x}_{tt'})$$

   - » *Chance constrained programming*

$$P[A_t x_t \le f(W)] \le 1 - \delta$$

   - » *Stochastic lookahead /stochastic prog/Monte Carlo tree search*

$$X_t^{LA-S}(S_t) = \arg\max_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{\tilde{\omega} \in \tilde{\Omega}_t} p(\tilde{\omega}) \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}(\tilde{\omega}), \tilde{x}_{tt'}(\tilde{\omega}))$$

   - » *"Robust optimization"*

$$X_t^{LA-RO}(S_t) = \arg\max_{\tilde{x}_{tt}, \ldots, \tilde{x}_{t,t+H}} \min_{w \in W_t(\theta)} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}(w), \tilde{x}_{tt'}(w))$$

Designing policies

Function approx.

1) **Policy function approximations (PFAs)**
   - » Lookup tables, rules, parametric/nonparametric functions
2) **Cost function approximation (CFAs)**
   - » $X^{CFA}(S_t \mid \theta) = \arg\max_{x_t \in \bar{X}_t^{\pi}(\theta)} \bar{C}^{\pi}(S_t, x_t \mid \theta)$
3) **Policies based on value function approximations (VFAs)**
   - » $X_t^{VFA}(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \bar{V}_t^{x} \left(S_t^{x}(S_t, x_t) \right) \right)$
4) **Direct lookahead policies (DLAs)**
   - » *Deterministic lookahead/rolling horizon proc./model predictive control*

$$X_t^{LA-D}(S_t) = \arg\max_{\tilde{x}_{tt}, \ldots, \tilde{x}_{t,t+H}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}, \tilde{x}_{tt'})$$

   - » *Chance constrained programming*

$$P[A_t x_t \le f(W)] \le 1 - \delta$$

   - » *Stochastic lookahead /stochastic prog/Monte Carlo tree search*

$$X_t^{LA-S}(S_t) = \arg\max_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{\tilde{\omega} \in \tilde{\Omega}_t} p(\tilde{\omega}) \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}(\tilde{\omega}), \tilde{x}_{tt'}(\tilde{\omega}))$$

   - » *"Robust optimization"*

$$X_t^{LA-RO}(S_t) = \arg\max_{\tilde{x}_{tt}, \ldots, \tilde{x}_{t,t+H}} \min_{w \in W_t(\theta)} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}(w), \tilde{x}_{tt'}(w))$$

Designing policies

1) **Policy function approximations (PFAs)**

» Lookup tables, rules, parametric/nonparametric functions

2) **Cost function approximation (CFAs)**

» $X^{CFA}(S_t \mid \theta) = \arg\max_{x_t \in \bar{X}_t^{\pi}(\theta)} \bar{C}^{\pi}(S_t, x_t \mid \theta)$

3) **Policies based on value function approximations (VFAs)**

» $X_t^{VFA}(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \bar{V}_t^{x} \left(S_t^{x}(S_t, x_t) \right) \right)$

4) **Direct lookahead policies (DLAs)**

» *Deterministic lookahead/rolling horizon proc./model predictive control*

$$X_t^{LA-D}(S_t) = \arg\max_{\tilde{x}_{tt}, \ldots, \tilde{x}_{t,t+H}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}, \tilde{x}_{tt'})$$

» *Chance constrained programming*

$$P[A_t x_t \le f(W)] \le 1 - \delta$$

» *Stochastic lookahead /stochastic prog/Monte Carlo tree search*

$$X_t^{LA-S}(S_t) = \arg\max_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{\tilde{\omega} \in \tilde{\Omega}_t} p(\tilde{\omega}) \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}(\tilde{\omega}), \tilde{x}_{tt'}(\tilde{\omega}))$$

» *"Robust optimization"*

$$X_t^{LA-RO}(S_t) = \arg\max_{\tilde{x}_{tt}, \ldots, \tilde{x}_{t,t+H}} \min_{w \in W_t(\theta)} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}(w), \tilde{x}_{tt'}(w))$$

Imbedded optimization

The Universal Framework for Sequential Decisions

Warren B. Powell, Princeton University

$$\max_{\pi} \mathbb{E}\left\{\sum_{t=0}^{T} C\big(S_t, X^{\pi}(S_t)\big)|S_0\right\}$$

$$where \quad S_{t+1} = S^M(S_t, X^{\pi}(S_t), \mathrm{W}_{t+1})$$

$$and\ given\ (S_0, W_1, W_2, \ldots, W_t, \ldots)$$

Policy

Policy Search

Policy Function Approximation(PFA)

$$X^{PFA}(S_t|\theta) = \begin{cases} If\ this\ then\ do \\ \sum_{f\in F} \theta_f \phi_f(S_t) \\ Neural\ network \end{cases}$$

Cost Function Approximation (CFA)

$$X^{CFA}(S_t|\theta) = \begin{cases} argmax_x c_t x_t + \sum_f \theta_f \phi_f(S_t) \\ argmax_x(\mu_{tx} + \theta^{IE}\bar{\sigma}_{tx}) \end{cases}$$

Lookahead Approximations

Value Function Approximation (VFA)

$$\begin{aligned} X^{VFA}(S_t|\theta) &= argmax_x(C(S_t,x) + \mathbb{E}\{V_{t+1}(S_{t+1})|S_t, x_t\}) \\ &= argmax_x(C(S_t,x) + \bar{V}_t^x(S_t^x) \\ &= argmax\ Q(S_t,x) \end{aligned}$$

Direct Lookahead (DLA)

$$X^{DLA}(S_t|\theta) = argmax_x\left(c_t x_t + \sum_{t'=t+1}^{t+H} \tilde{c}_{tt'}\tilde{x}_{tt'}\right)$$

The four classes of policies (PFAs, CFAs, VFAs and DLAs) are *universal*. Any sequential decision problem will use one of these four classes (or a hybrid), including whatever you might be doing now.

The optimal policy (if we could solve it) is given by

$$X^*(S_t) = argmax_x\left(C(S_t,x) + \mathbb{E}\left\{\max_{\pi}\mathbb{E}\left\{\sum_{t'=t+1}^{t+H} C(S_{t'}, X^{\pi}(S_{t'}))\,|S_{t+1}\right\}|S_t, x_t\right\}\right)$$

Available at ***jungle.princeton.edu***

Designing policies

- Notes:
  - » Policies in the "policy search class" are simpler, and as a result this is what you are most likely going to see (and use) in practice.
  - » "The price of simplicity is tunable parameters"
  - » *Tuning is hard!*
    - ... but you do tuning offline.

Learning problems

- Classes of machine learning problems in stochastic optimization

  1) Approximating the objective

  $$\bar{F}(x|\theta) \approx \mathbb{E}F(x, W).$$

  2) Designing a policy $X^{\pi}(S|\theta)$.

  3) A value function approximation

  $$\bar{V}_t(S_t|\theta) \approx V_t(S_t).$$

  4) Designing a cost function approximation:
     - The objective function $\bar{C}^{\pi}(S_t, x_t|\theta)$.
     - The constraints $X^{\pi}(S_t|\theta)$

  5) Approximating the transition function

  $$\bar{S}^M(S_t, x_t, W_{t+1}|\theta) \approx S^M(S_t, x_t, W_{t+1})$$

Outline

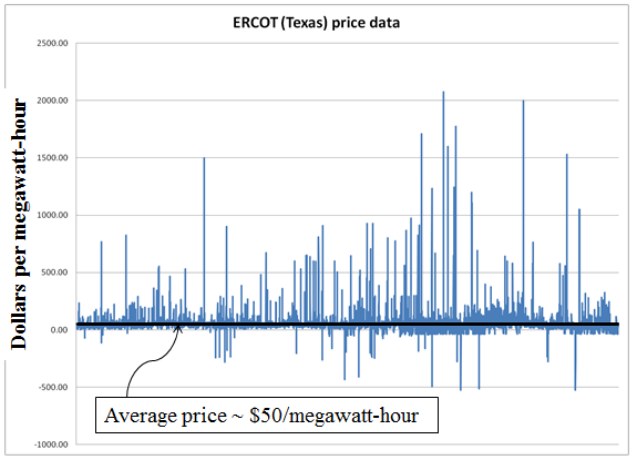
- The four classes of policies
  - » Policy function approximations (PFAs)
  - » Cost function approximations (CFAs)
  - » Value function approximations (VFAs)
  - » Direct lookahead policies (DLAs)
  - » Hybrid direct lookahead/CFA
  - » Any of the four classes may work best!

Outline

- The four classes of policies
  - » Policy function approximations (PFAs)
  - » Cost function approximations (CFAs)
  - » Value function approximations (VFAs)
  - » Direct lookahead policies (DLAs)
  - » Hybrid direct lookahead/CFA
  - » Any of the four classes may work best!

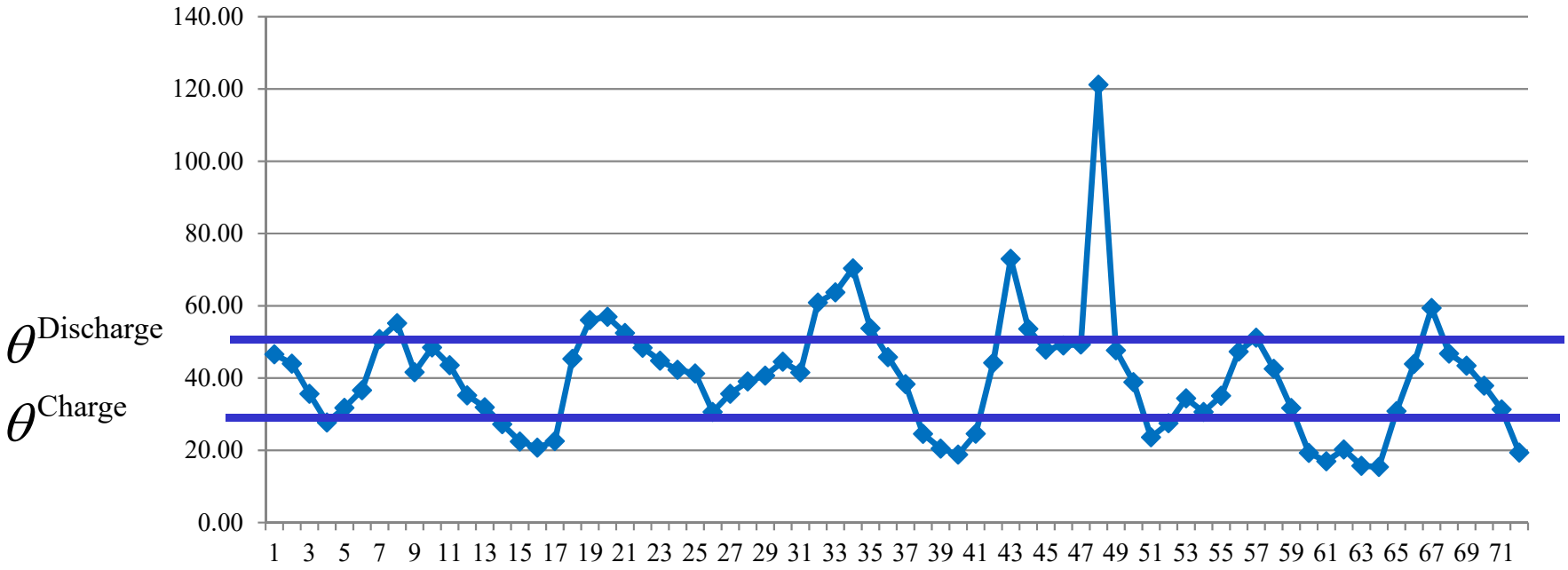
Policy function approximations

- Battery arbitrage – When to charge, when to discharge, given volatile LMPs



Policy function approximations

- Grid operators require that batteries bid charge and discharge prices, an hour in advance.

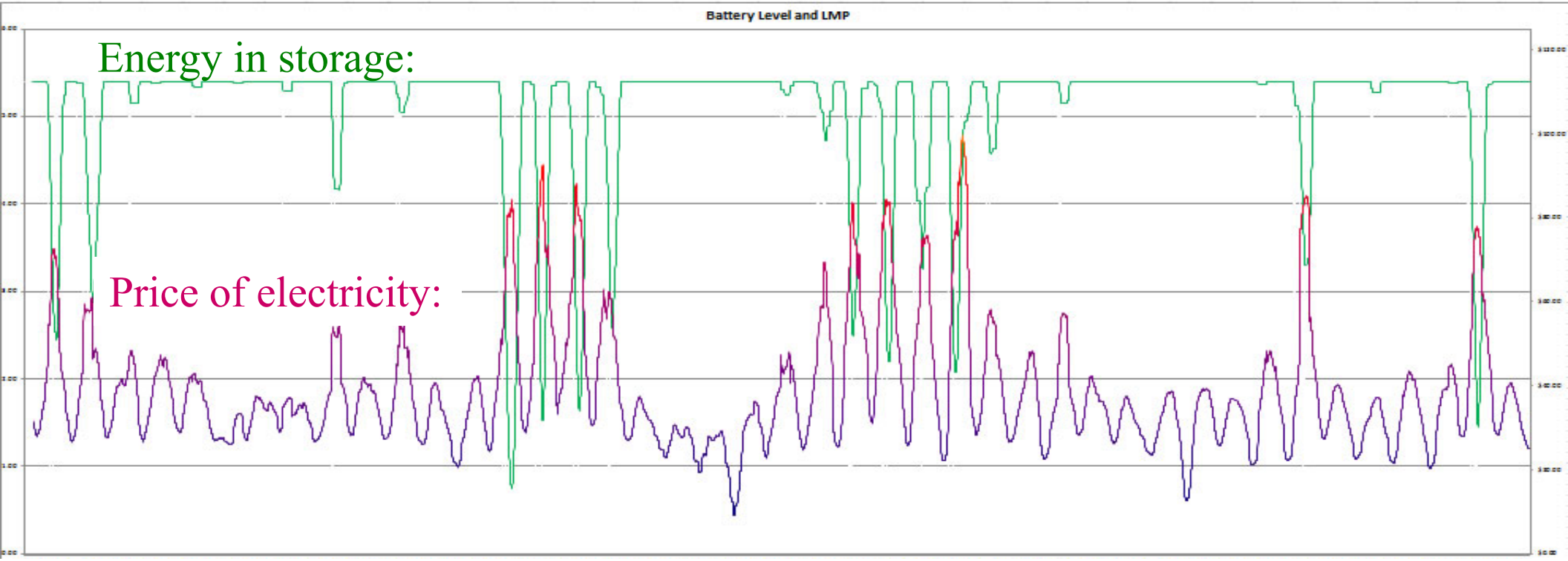


- We have to search for the best values for the policy parameters θ^{Charge} and $\theta^{\text{Discharge}}$.

Policy function approximations

- Our policy function might be the parametric model (this is nonlinear in the parameters):

$$X^{\pi}(S_t \mid \theta) = \begin{cases} +1 & \text{if } p_t < \theta^{\text{charge}} \\ 0 & \text{if } \theta^{\text{charge}} < p_t < \theta^{\text{discharge}} \\ -1 & \text{if } p_t > \theta^{\text{charge}} \end{cases}$$

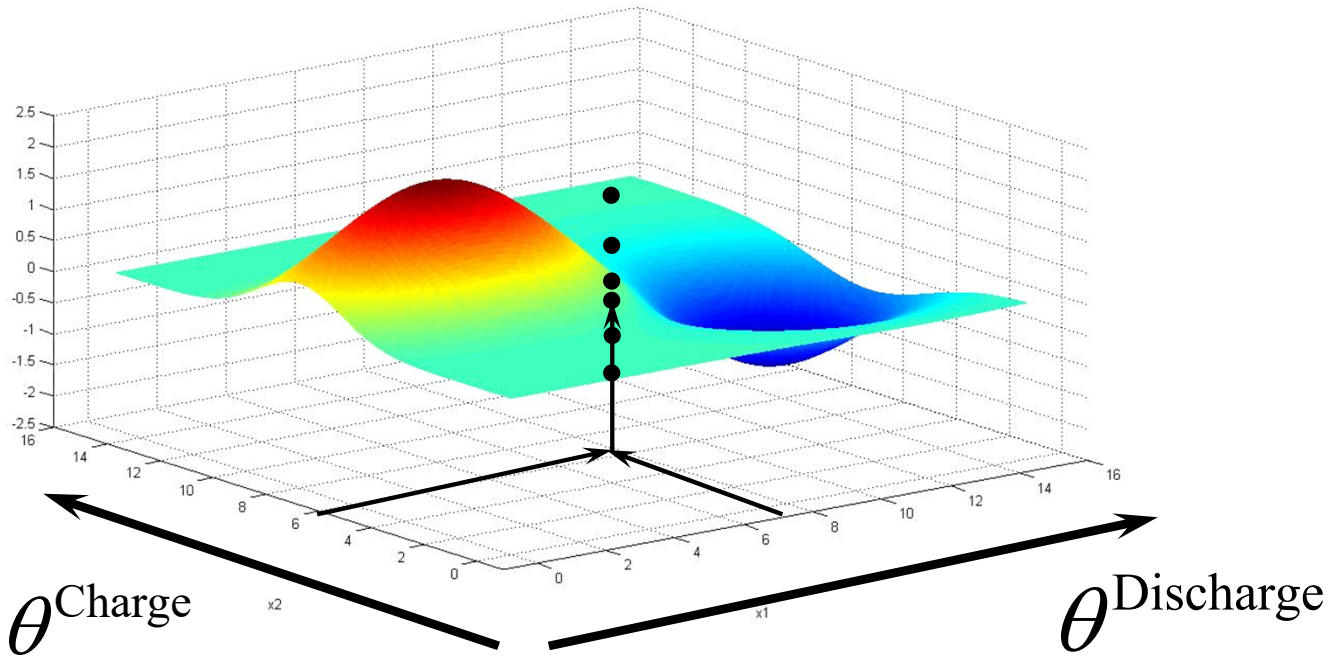


Policy function approximations

- Finding the best policy
  - » We need to maximize

$$\max_\theta F(\theta) = \mathbb{E}\sum_{t=0}^{T} \gamma^t C\left(S_t, X_t^\pi(S_t \mid \theta)\right)$$

  - » We cannot compute the expectation, so we run simulations:



Policy function approximations

- How do we search for the best θ?
  - » Derivative-based
    - Use classical stochastic gradient methods:

$$x^{n+1} = x^n + \alpha_n \nabla_x F(x^n, W^{n+1})$$

    - The gradient $\nabla_x(Fx^n, W^{n+1})$ may be computed analytically, but more often it is computed numerically.
  - » Derivative-free
    - Build a belief model $\bar{F}(x) \approx \mathbb{E}F(x, W)$ that approximates our function.
  - » Both of these approaches are sequential decision problems!

Policy function approximations

- Derivative-based stochastic search
  - » Basic problem:

$$\max_x \mathbb{E}\{F(x,W) \mid S_0\}$$

  - » Stochastic gradient algorithm

$$x^{n+1} = x^n + \alpha_n \nabla_x F(x^n, W^{n+1})$$

  - » Let π be our "algorithm" and let $x^{\pi,N}$ be the solution that "algorithm" π produces after N iterations.
  - » Now we would state the optimization problem as

$$\max_\pi \mathbb{E} F(x^{\pi,N}, W)$$

    where π is our stochastic gradient algorithm.
  - » In other words, we want the *optimal algorithm*.

Policy function approximations

- Derivative-based stochastic search:
  - » Assume that our stepsize rule

$$\alpha_n = \frac{\theta}{\theta + N^n}$$

where N^n = number of times the solution has not improved:

$$N^{n+1} = \begin{cases} N^n + 1 & \text{If } \nabla_x F(x^{n-1}, W^n) \nabla_x F(x^n, W^{n+1}) < 0 \\ N^n & \text{Otherwise} \end{cases}$$

  - » The stepsize α_n is the "decision." The stepsize rule is the "policy" which is a form of policy function approximation.

Derivative-based, finite horizon

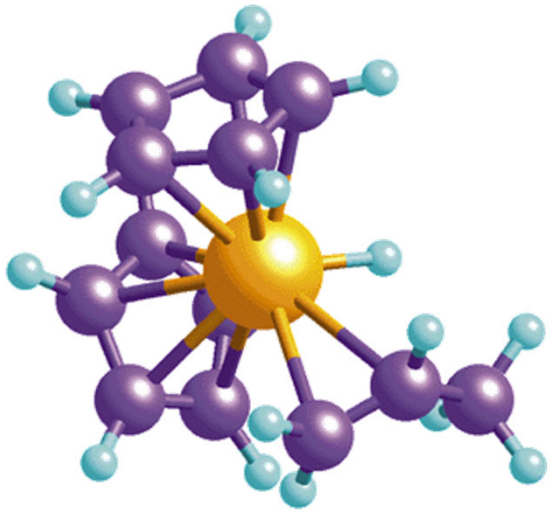
- Derivative-based stochastic search – finite horizon
  - » State variables
$$S^n = (x^n, N^n)$$
  - » Decision variable: α_n made using policy
$$\alpha^\pi(S^n|\theta) = \frac{\theta}{\theta + N^n}$$
  - » Exogenous information: W^{n+1}
  - » Transition function $S^{n+1} = S^M(S^n, \alpha_n, W^{n+1})$
$$x^{n+1} = x^n + \alpha_n \nabla_x F(x^n, W^{n+1})$$
$$N^{n+1} = N^n + \begin{cases} 1 & \text{If } \nabla_x F\left(x^n, W^{n+1}\right)\nabla_x F\left(x^{n-1}, W^n\right) < 0 \\ 0 & \text{Otherwise} \end{cases}$$
  - » Our objective is to find the best stepsize policy that solves
$$\max_\theta \mathbb{E} F(x^{\pi,N}, \hat{W})$$

Outline

- The four classes of policies
  - » Policy function approximations (PFAs)
  - » Cost function approximations (CFAs)
  - » Value function approximations (VFAs)
  - » Direct lookahead policies (DLAs)
  - » Hybrid direct lookahead/CFA
  - » Any of the four classes may work best!

Cost function approximations

- Materials science
  - » We need to find the best of seven catalysts to maximize the strength of materials.

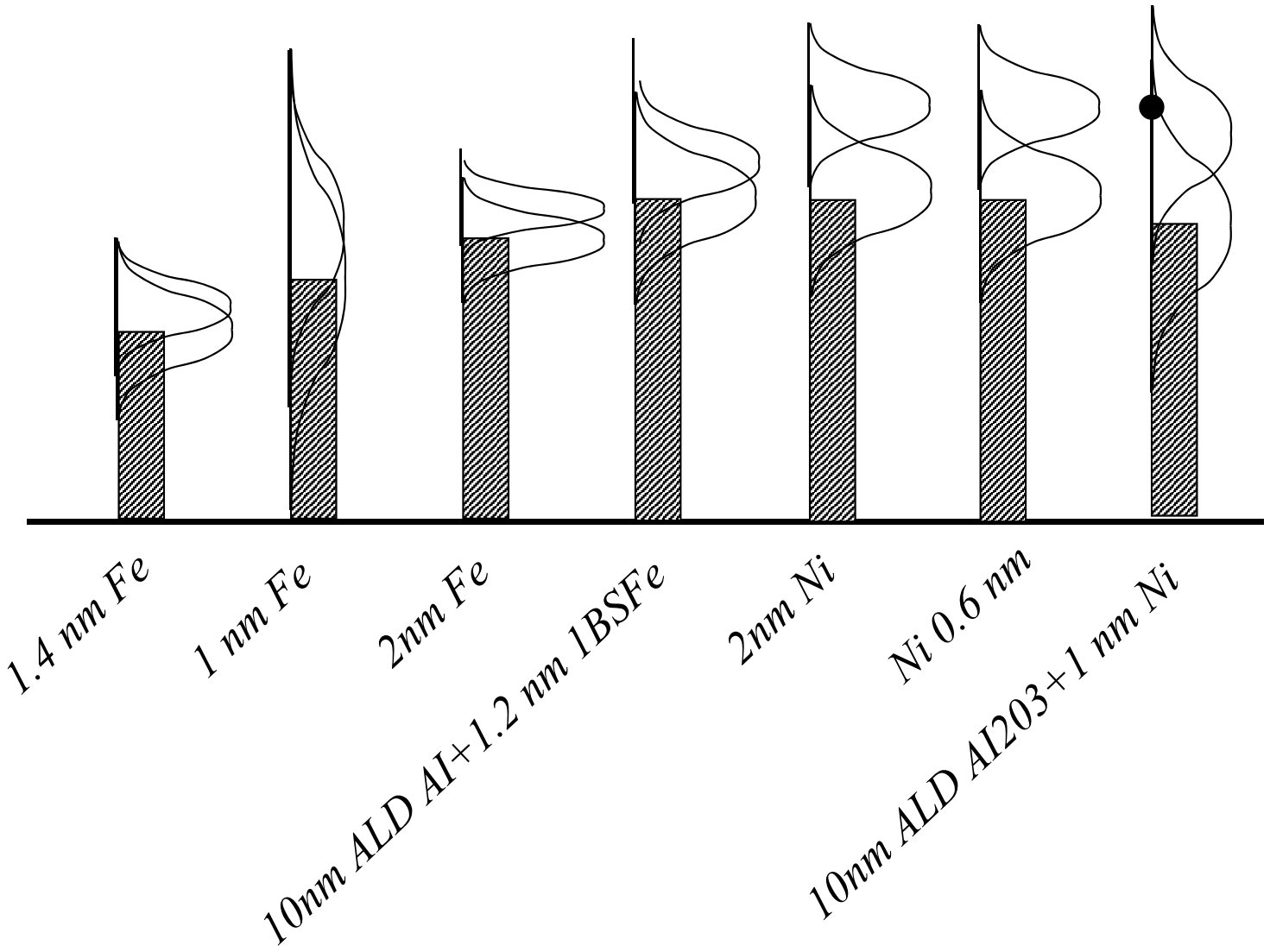


$Cov(\mu_x, \mu_{x'})$

	1.4 nm Fe	1 nm Fe	2nm Fe	10nm ALD Al2O3+1.2 nm IBS Fe	2 nm Ni	Ni 0.6 nm	10nm ALD Al2O3+1 nm Ni
1.4 nm Fe	1	0.7	0.7	0.6	0.4	0.4	0.2
1 nm Fe	0.7	1	0.7	0.6	0.4	0.4	0.2
2nm Fe	0.7	0.7	1	0.6	0.4	0.4	0.2
10nm ALD Al2O3+1.2 nm IBS Fe	0.6	0.6	0.6	1	1	0.3	0
2 nm Ni	0.4	0.4	0.4	1	1	0.7	0.6
Ni 0.6 nm	0.4	0.4	0.4	0.3	0.7	1	0.6
10nm ALD Al2O3+1 nm Ni	0.2	0.2	0.2	0	0.6	0.6	1

Cost function approximations

- Correlated beliefs: Testing one material teaches us about other materials

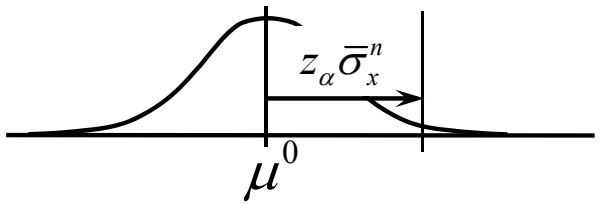


Cost function approximations

- Cost function approximations (CFA)
  - » Upper confidence bounding

$$X^{UCB}(S^n \mid \theta^{UCB}) = \arg\max_x \left(\bar{\mu}_x^n + \theta^{UCB} \sqrt{\frac{\log n}{N_x^n}} \right)$$

  - » Interval estimation



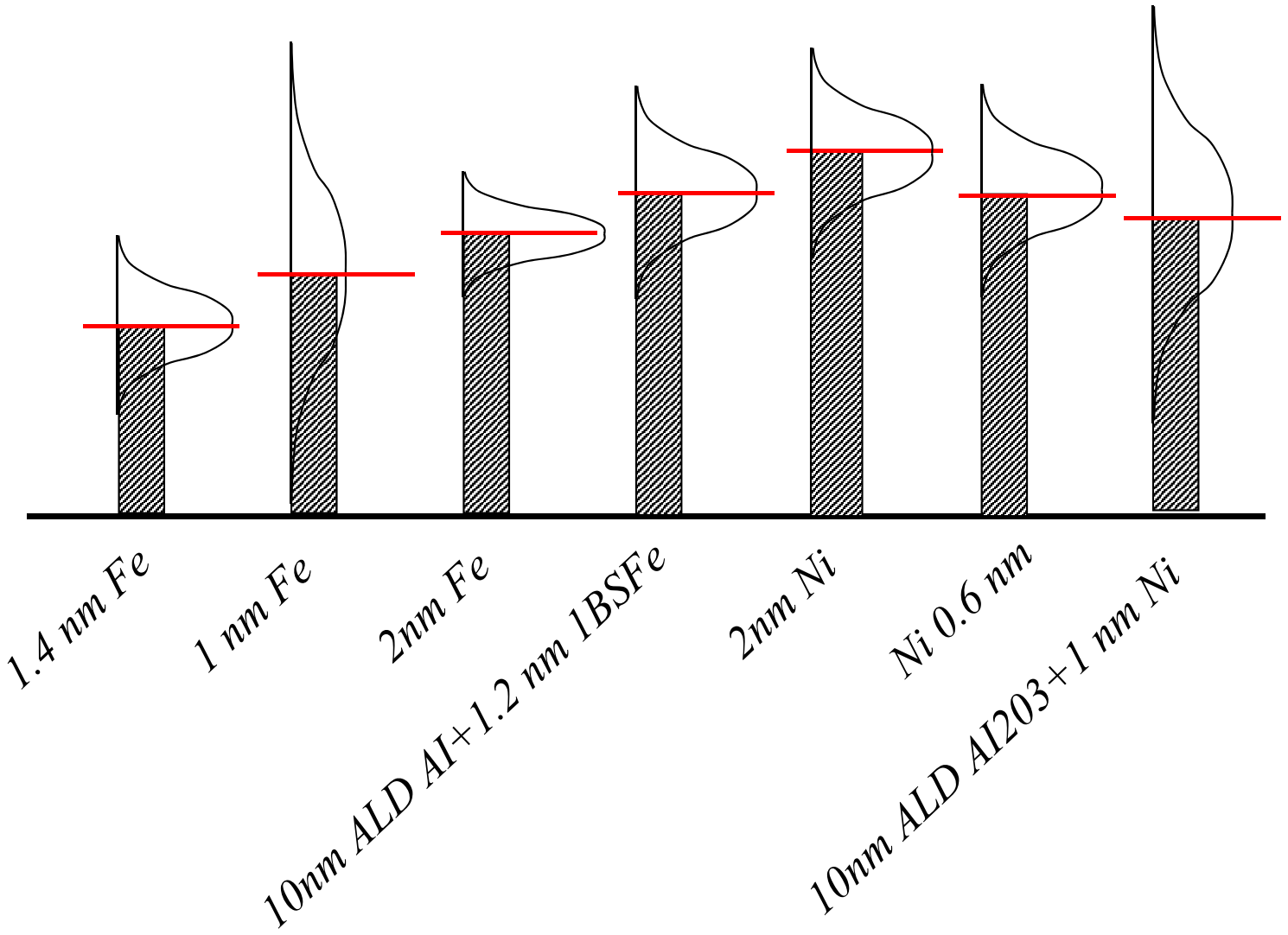
$$X^{IE}(S^n \mid \theta^{IE}) = \arg\max_x \left(\bar{\mu}_x^n + \theta^{IE} \bar{\sigma}_x^n \right)$$

  - » Thompson sampling

$$x^n = argmax_x \hat{\mu}_x^n \qquad \hat{\mu}_x^n \sim N(\bar{\mu}_x^n, \theta^{TS} \bar{\sigma}_x^{2,n})$$

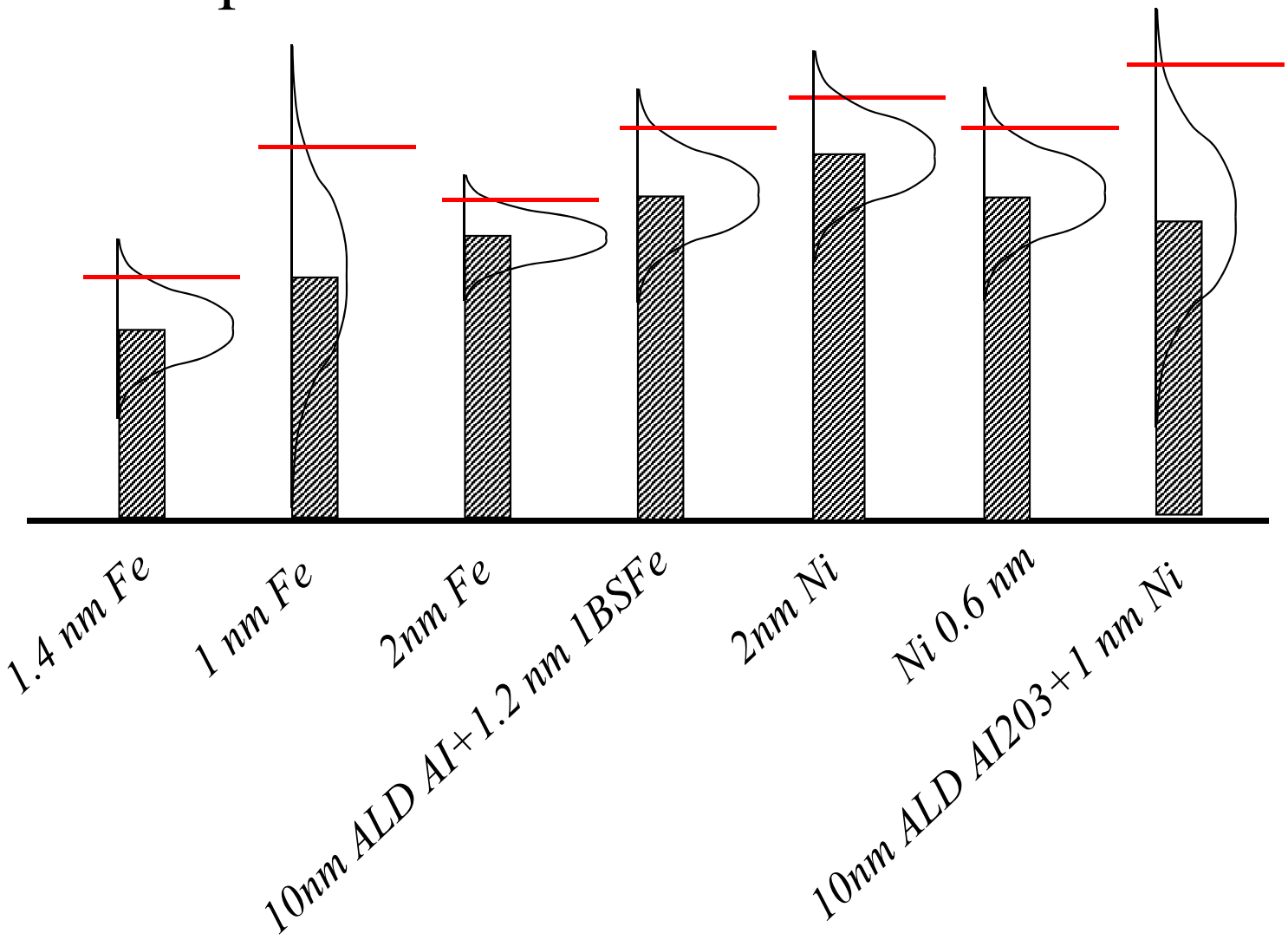
Cost function approximations

- Picking $\theta^{IE} = 0$ means we are evaluating each choice at the mean.



Cost function approximations

- Picking $\theta^{IE} = 2$ means we are evaluating each choice at the 95th percentile.



Cost function approximations

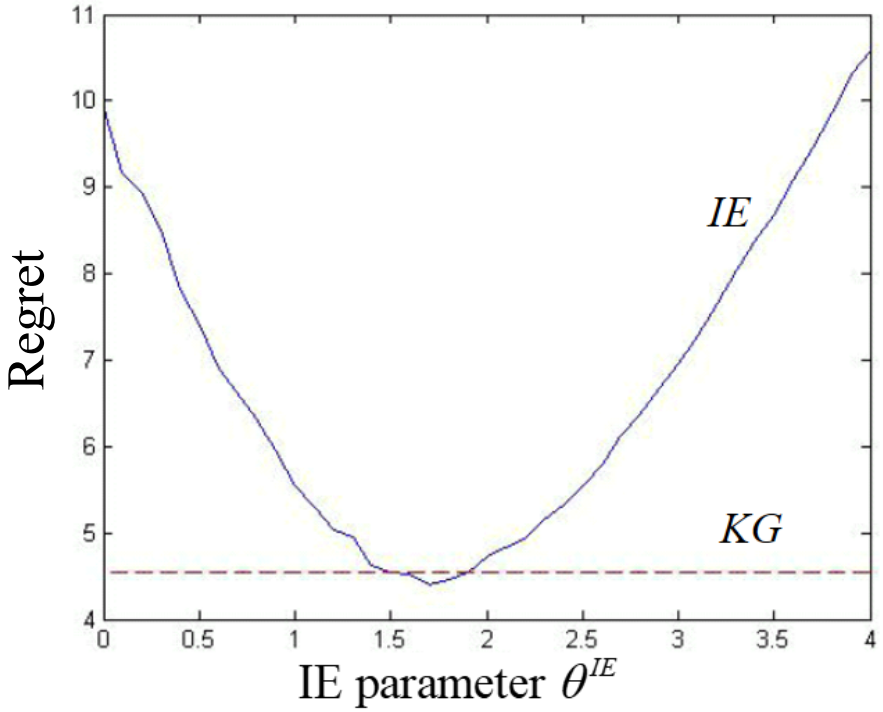
- Optimizing the policy
  - » We optimize θ^{IE} to maximize:

$$\max_{\theta^{IE}} F(\theta^{IE}) = \mathbb{E}F\left(x^{\pi,N}, W\right)$$

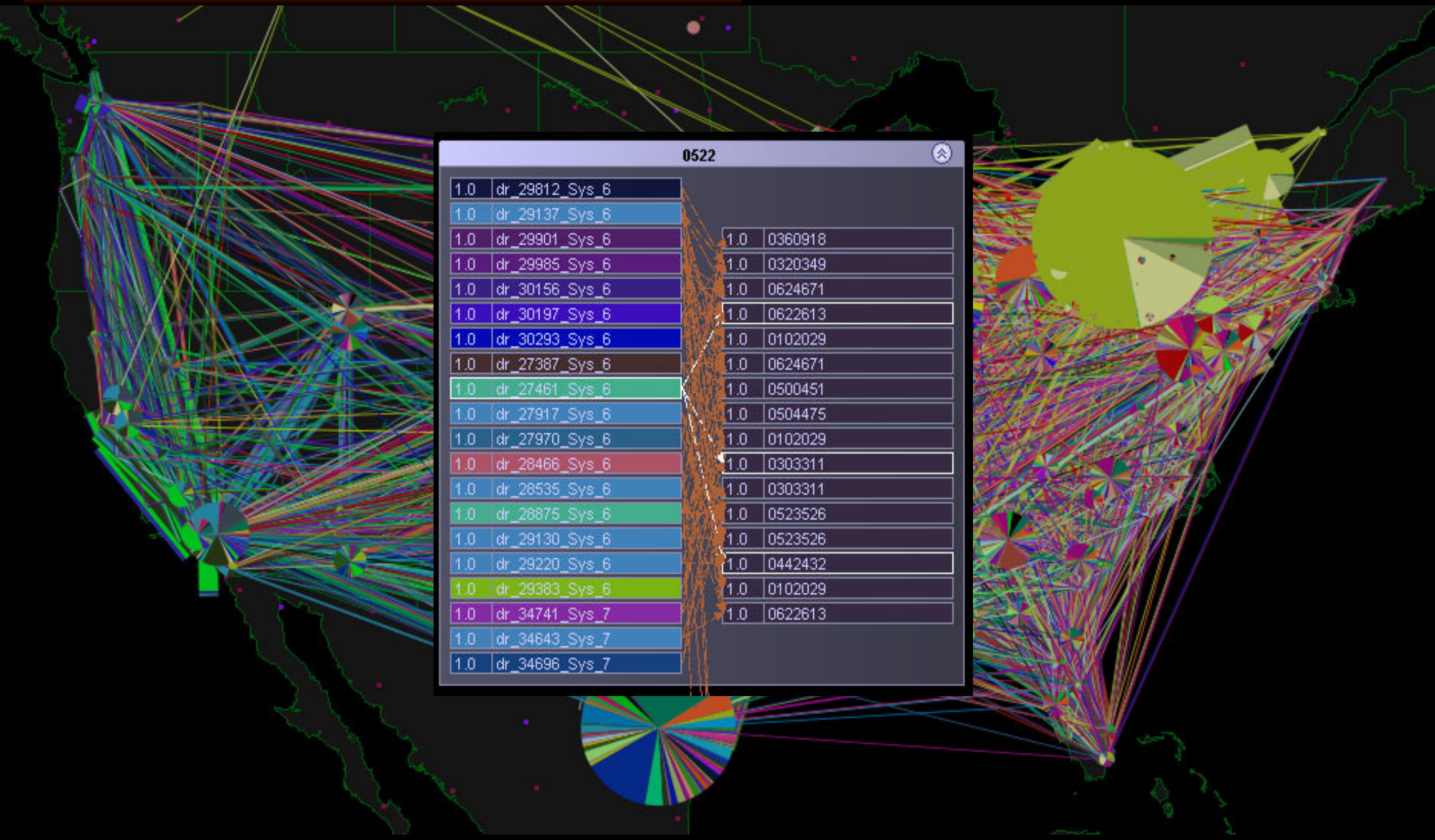
where

$$x^n = X^{IE}(S^n \mid \theta^{IE}) = \arg\max_x \left(\bar{\mu}_x^n + \theta^{IE}\bar{\sigma}_x^n\right) \qquad S^n = (\bar{\mu}_x^n, \bar{\sigma}_x^n)$$

- Notes:
  - » This can handle any belief model, including correlated beliefs, nonlinear belief models.
  - » All we require is that we be able to simulate a policy.

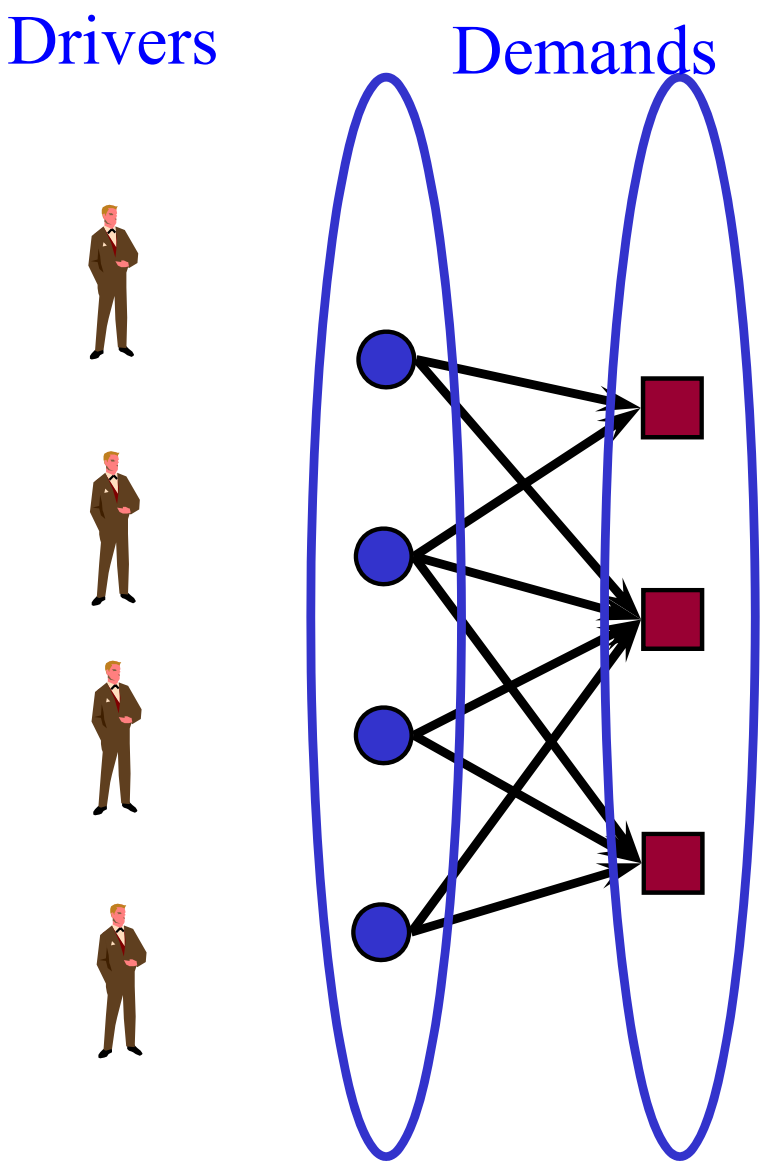


Schneider National

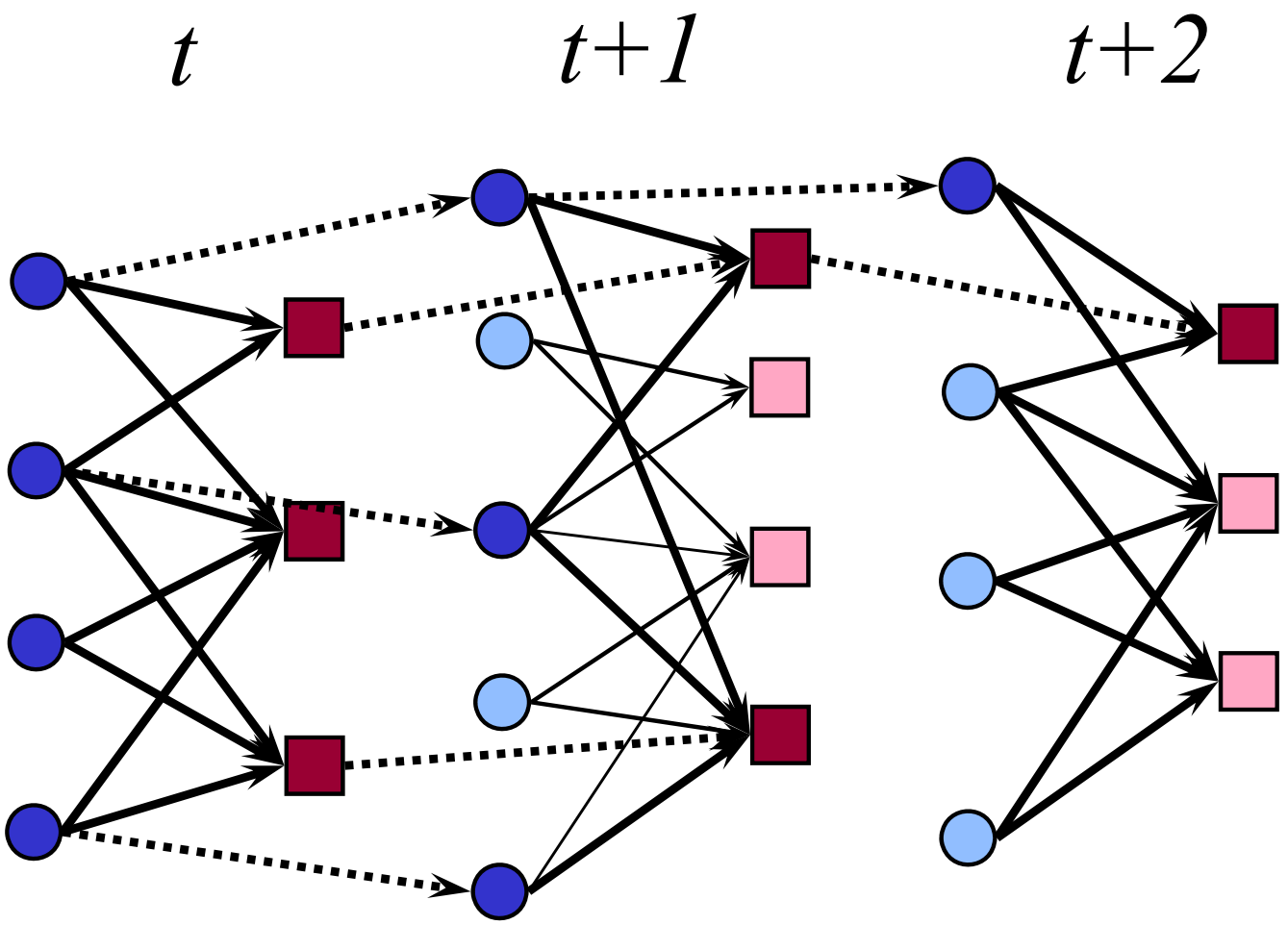



SCHNEIDER
NATIONAL
Week 45
1 Way Rank
Dedic Rank
TrkRl Rank
Brok Rank
Miles
18
4
1
3
Working units
18
1
By industry
Auto Asmby
Auto Parts
Retail
Paper
Miles
8
27
1
9
33
Nov YOY Growth
1 Way
Dedic
TrkRl
-5%
-5%
29%
SCHNEIDER

Cost function approximations



Cost function approximations



The assignment of drivers to loads evolves over time, with new loads being called in, along with updates to the status of a driver.

Cost function approximations

- A purely myopic policy would solve this problem using

$$\min_x \sum_d \sum_l c_{tdl} x_{tdl}$$

where

$$x_{tdl} = \begin{cases} 1 & \text{If we assign driver } d \text{ to load } l \\ 0 & \text{Otherwise} \end{cases}$$

$$c_{tdl} = \text{Cost of assigning driver } d \text{ to load } l \text{ at time } t$$

What if a load it not assigned to any driver, and has been delayed for a while? This model ignores the fact that we eventually have to assign someone to the load.

Cost function approximations

- We can minimize delayed loads by solving a modified objective function:

$$\min_x \sum_d \sum_l \left(c_{tdl} - \theta \tau_{tl} \right) x_{tdl}$$

where

τ_{tl} = How long load l has been delayed by time t

θ = Bonus for moving a delayed load

We refer to our modified objective function as a *cost function approximation*.

Cost function approximations

- We now have to tune our policy, which we define as:

$$X^{\pi}(S_t \mid \theta) = \arg\min_x \underbrace{\sum_d \sum_l \left(c_{tdl} - \theta\tau_{tl}\right) x_{tdl}}_{\bar{C}^{\pi}(S_t, x_t \mid \theta)}$$

We can now optimize θ, another form of *policy search,* by solving

$$\min_{\theta} F^{\pi}(\theta) = \mathbb{E}\sum_{t=0}^{T} C(S_t, X_t^{\pi}(S_t \mid \theta))$$

Cost function approximations

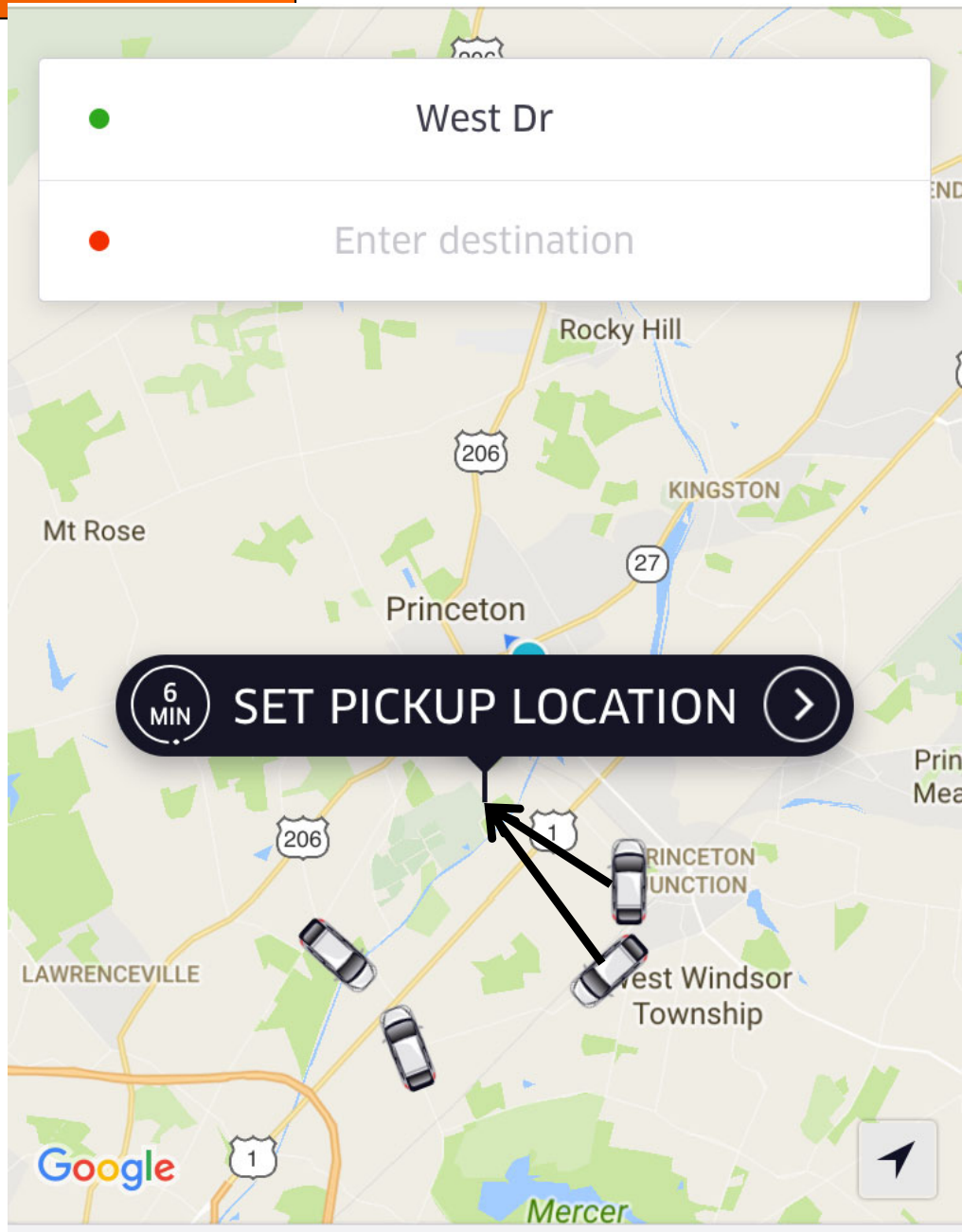
- Other applications
  - » Airlines optimizing schedules with schedule slack to handle weather uncertainty.
  - » Manufacturers using buffer stocks to hedge against production delays and quality problems.
  - » Grid operators scheduling extra generation capacity in case of outages.
  - » Adding time to a trip planned by Google maps to account for uncertain congestion.

Outline

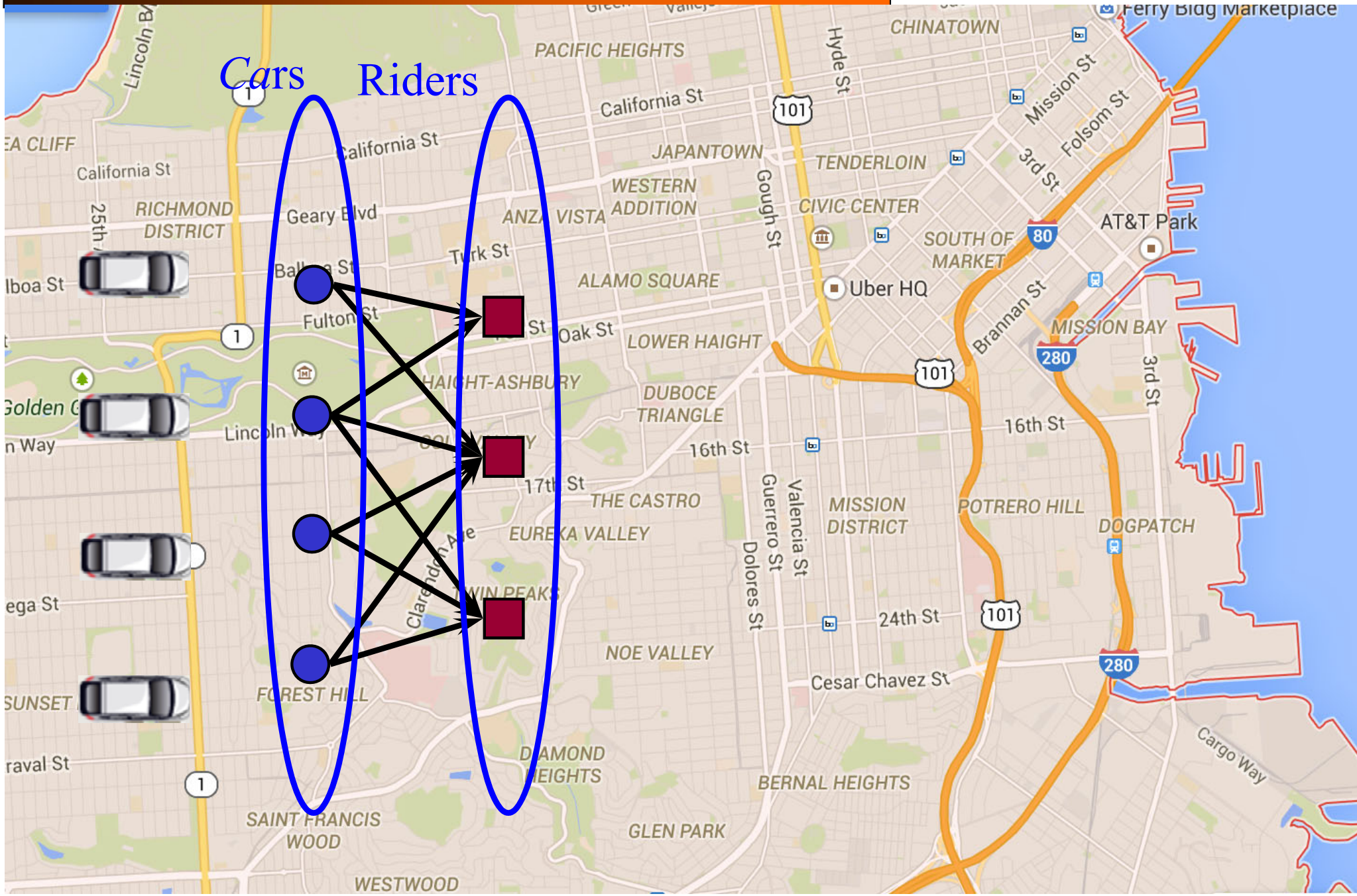
- The four classes of policies
  - » Policy function approximations (PFAs)
  - » Cost function approximations (CFAs)
  - » Value function approximations (VFAs)
  - » Direct lookahead policies (DLAs)
  - » Hybrid direct lookahead/CFA
  - » Any of the four classes may work best!

Driverless EV optimization

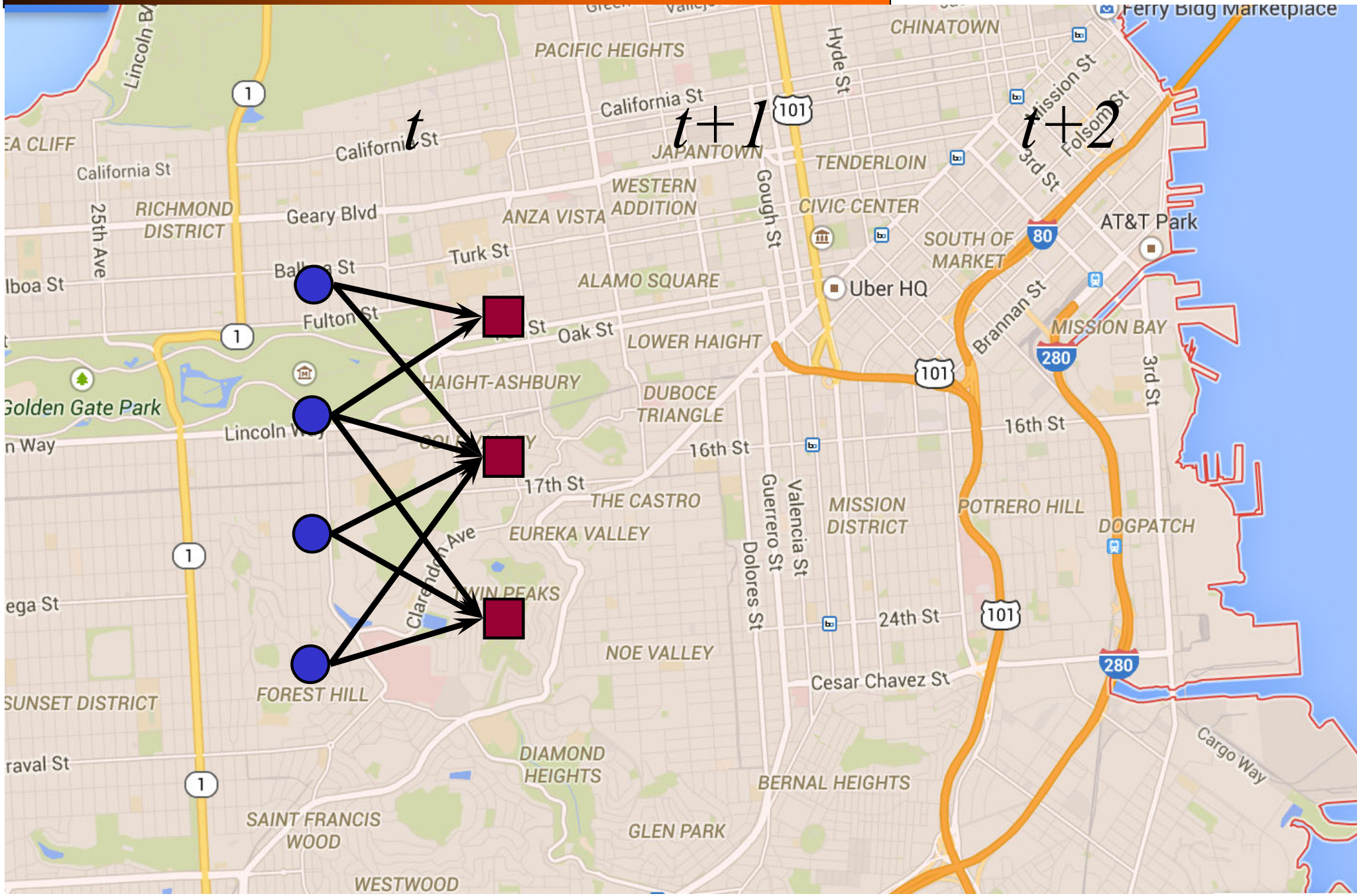
- Current Uber logic:
  - » Show nearest 8 drivers.
  - » Contact closest driver to confirm assignment.
  - » If driver does not confirm, contact second closest driver.
- Limitations:
  - » Ignores potential future opportunities for each driver.



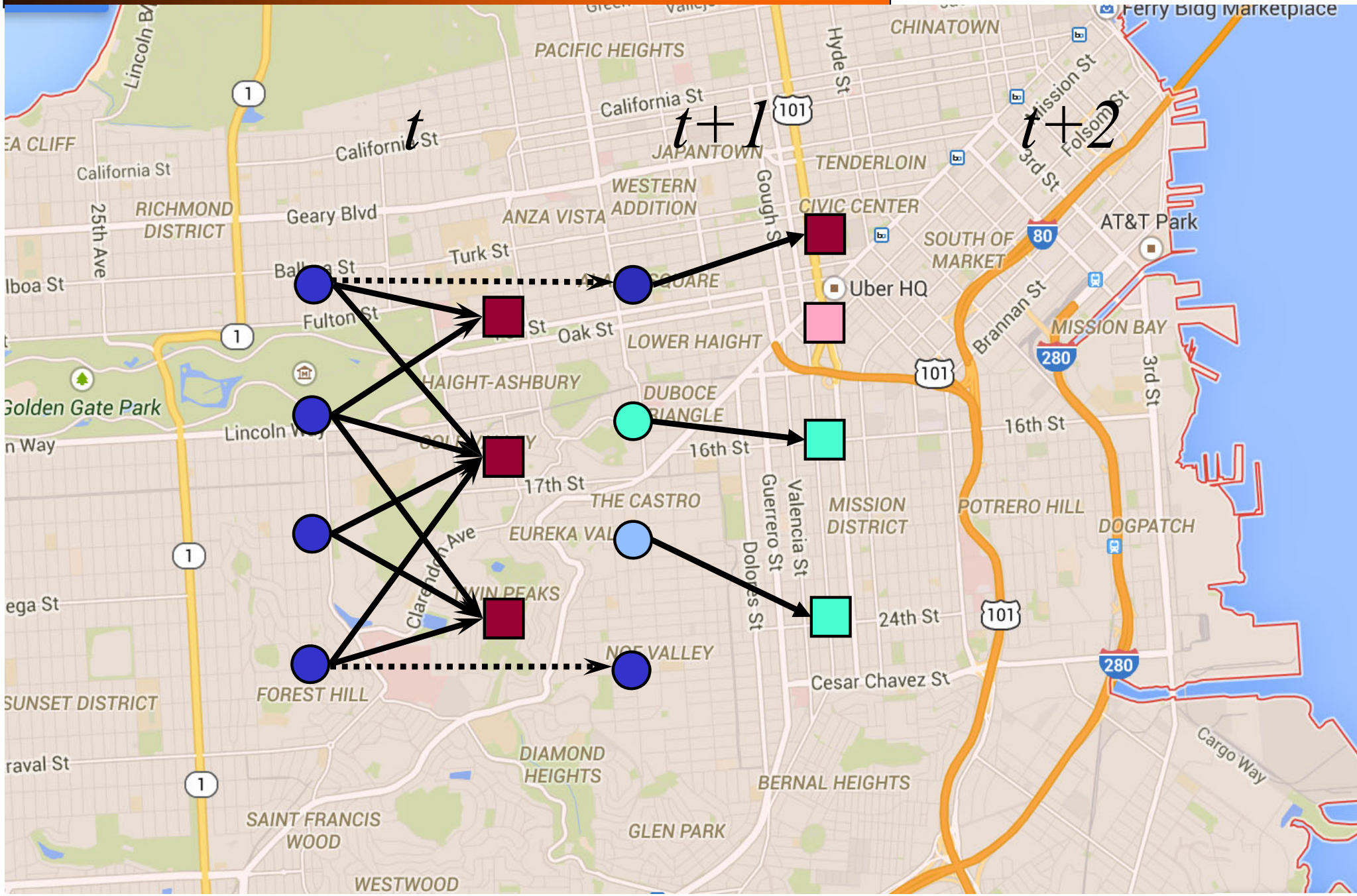
Driverless EV optimization



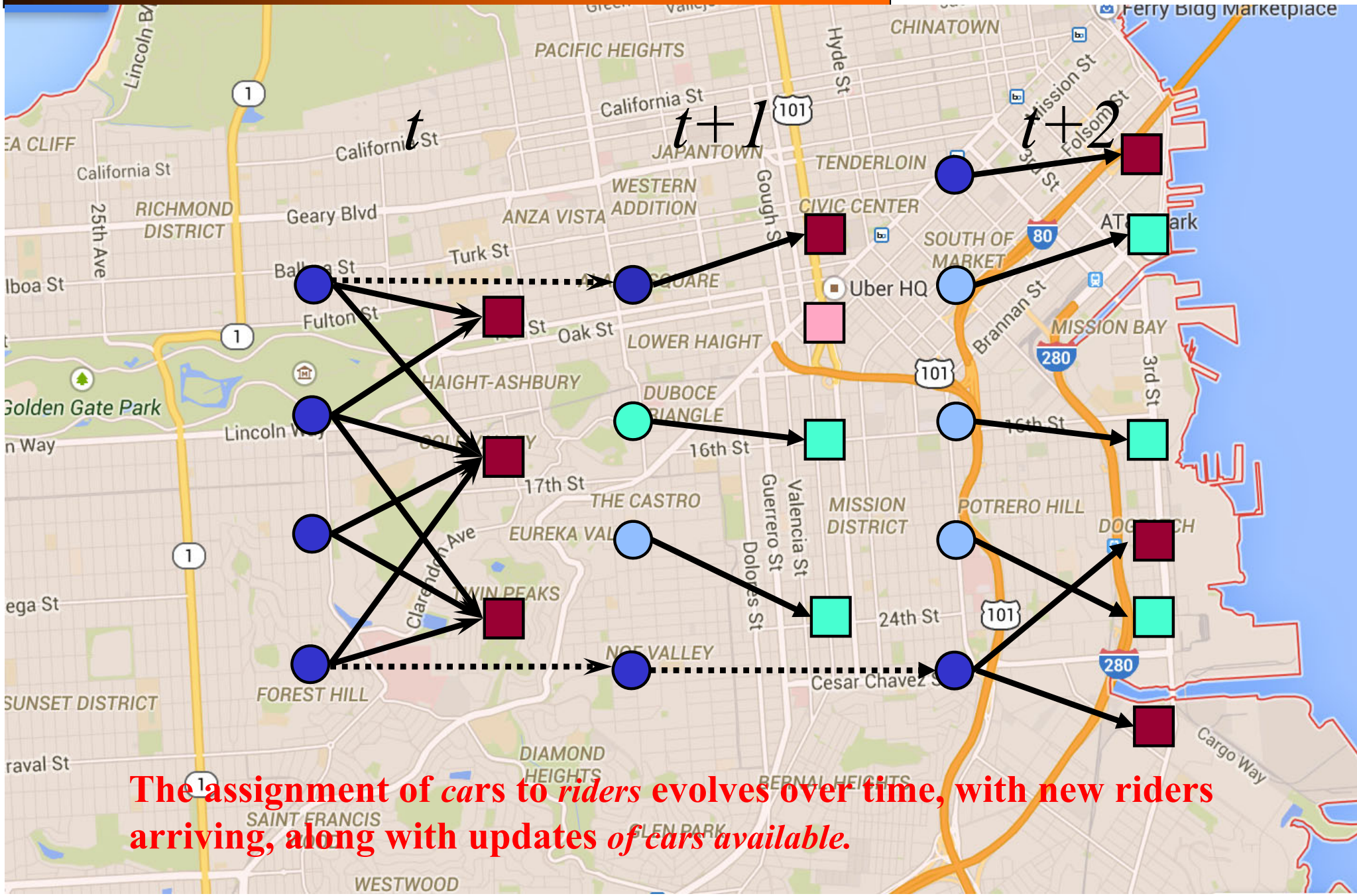
Driverless EV optimization



Driverless EV optimization



Driverless EV optimization



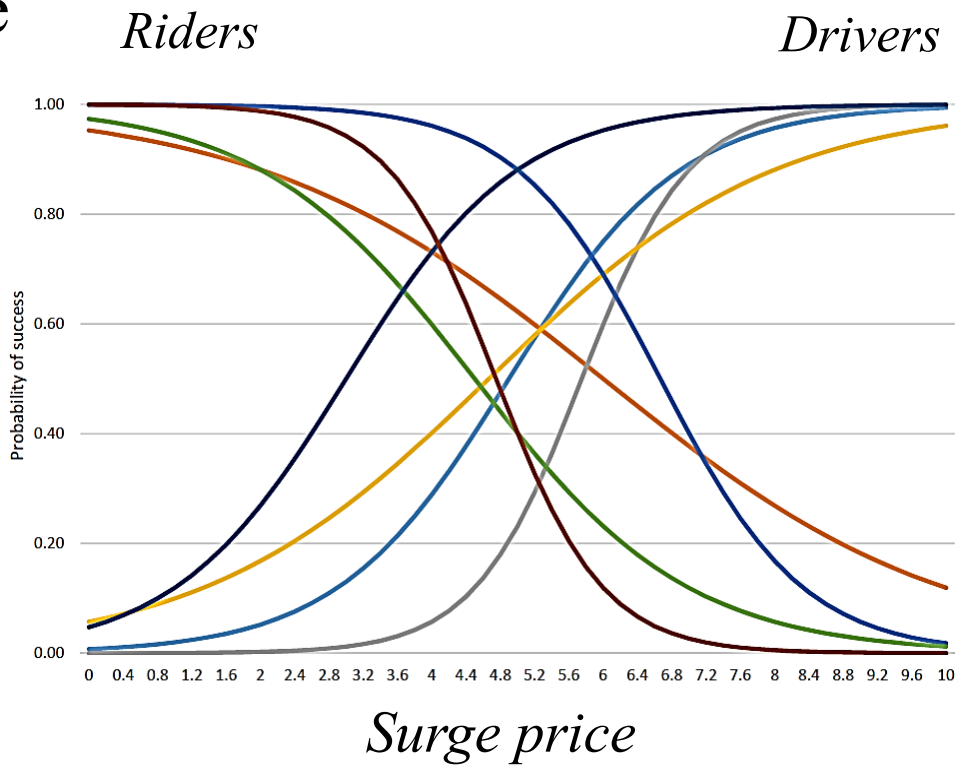
The assignment of *cars* to *riders* evolves over time, with new riders arriving, along with updates *of cars available.*

Driverless EV optimization

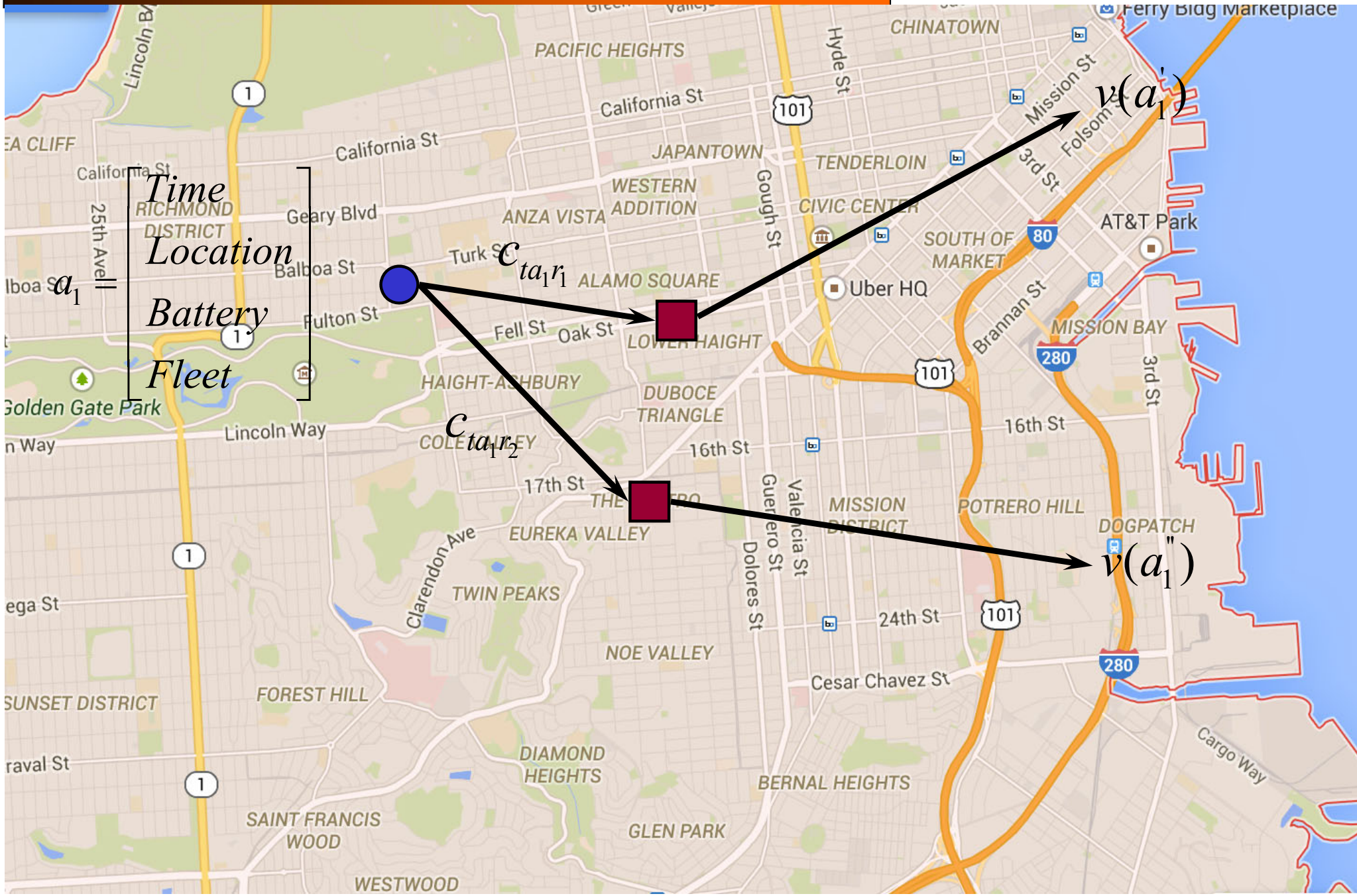
- While we are optimizing the drivers, we also have to learn the response of drivers and riders to price.
  - » We assume a parameterized response curve for buyers, and a separate one for sellers

$$P^Y(p,a \mid \theta) = \frac{e^{\theta_{ij}^0 + \theta_{ij} p + \theta_{ij}^a a}}{1 + e^{\theta_{ij}^0 + \theta_{ij} p + \theta_{ij}^a a}}$$

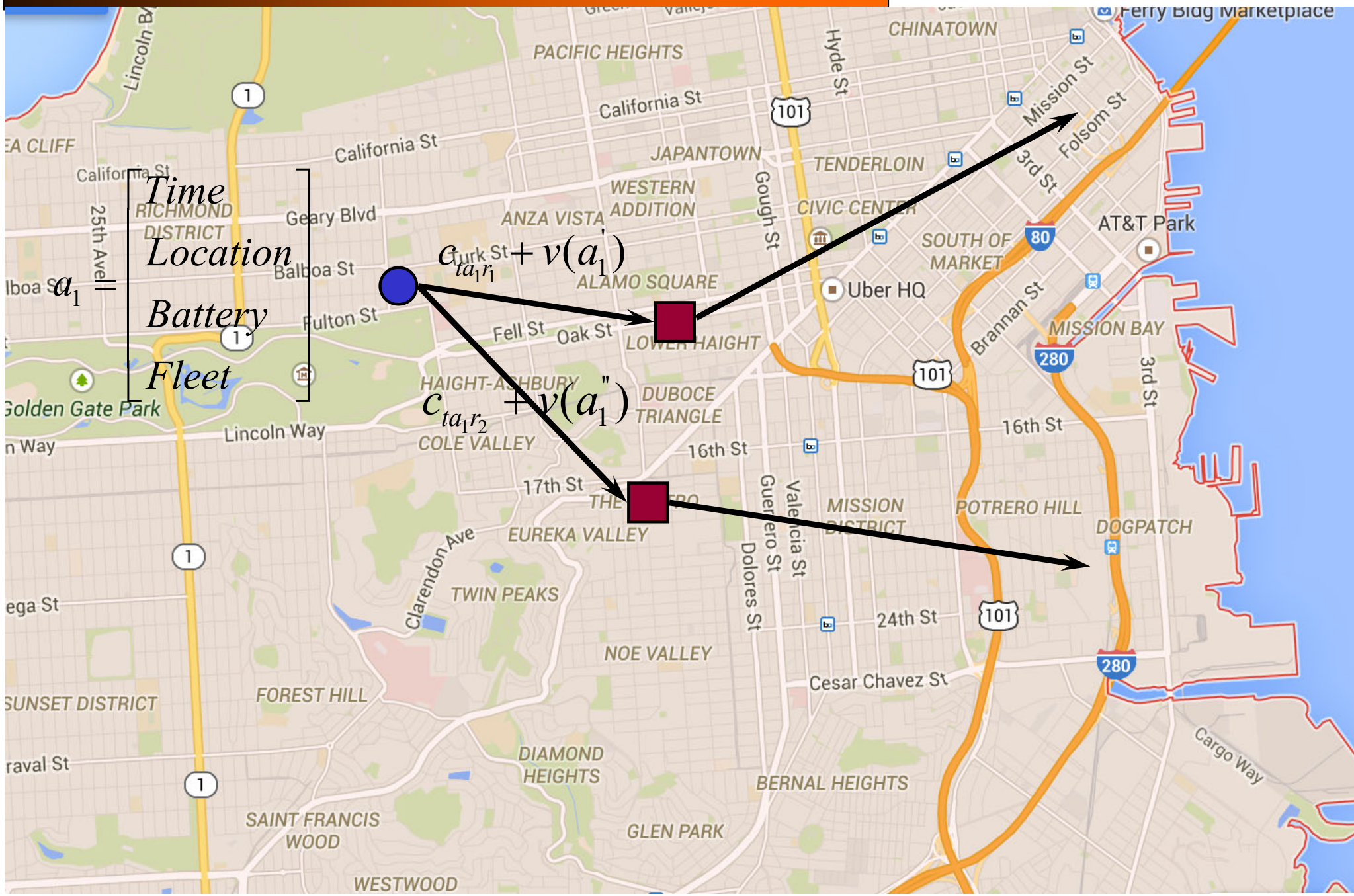
  - » We do not know the right values of the parameters, so we have a *learning problem*.
  - » Our *belief state* is our distribution of belief on θ.



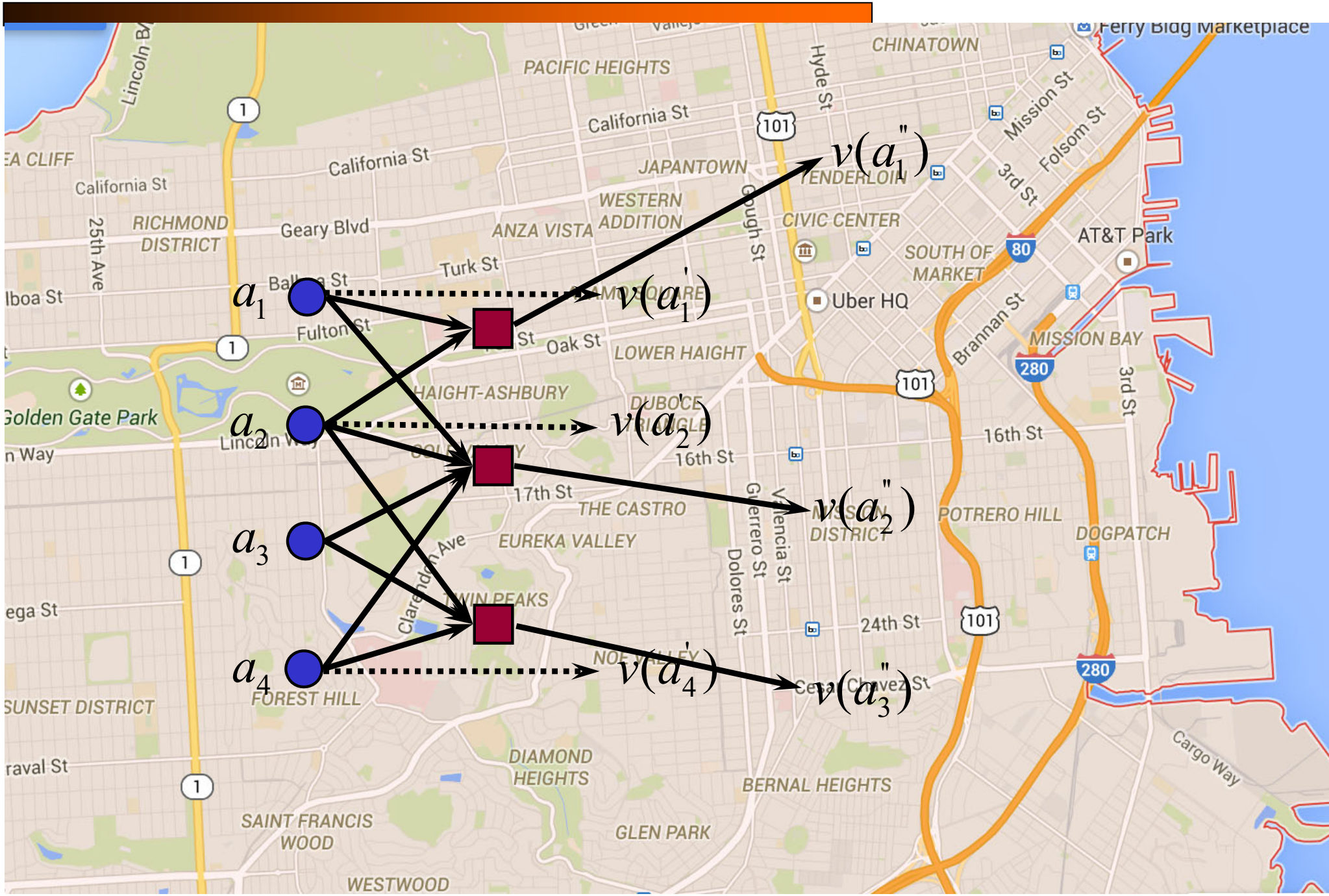
Driverless EV optimization



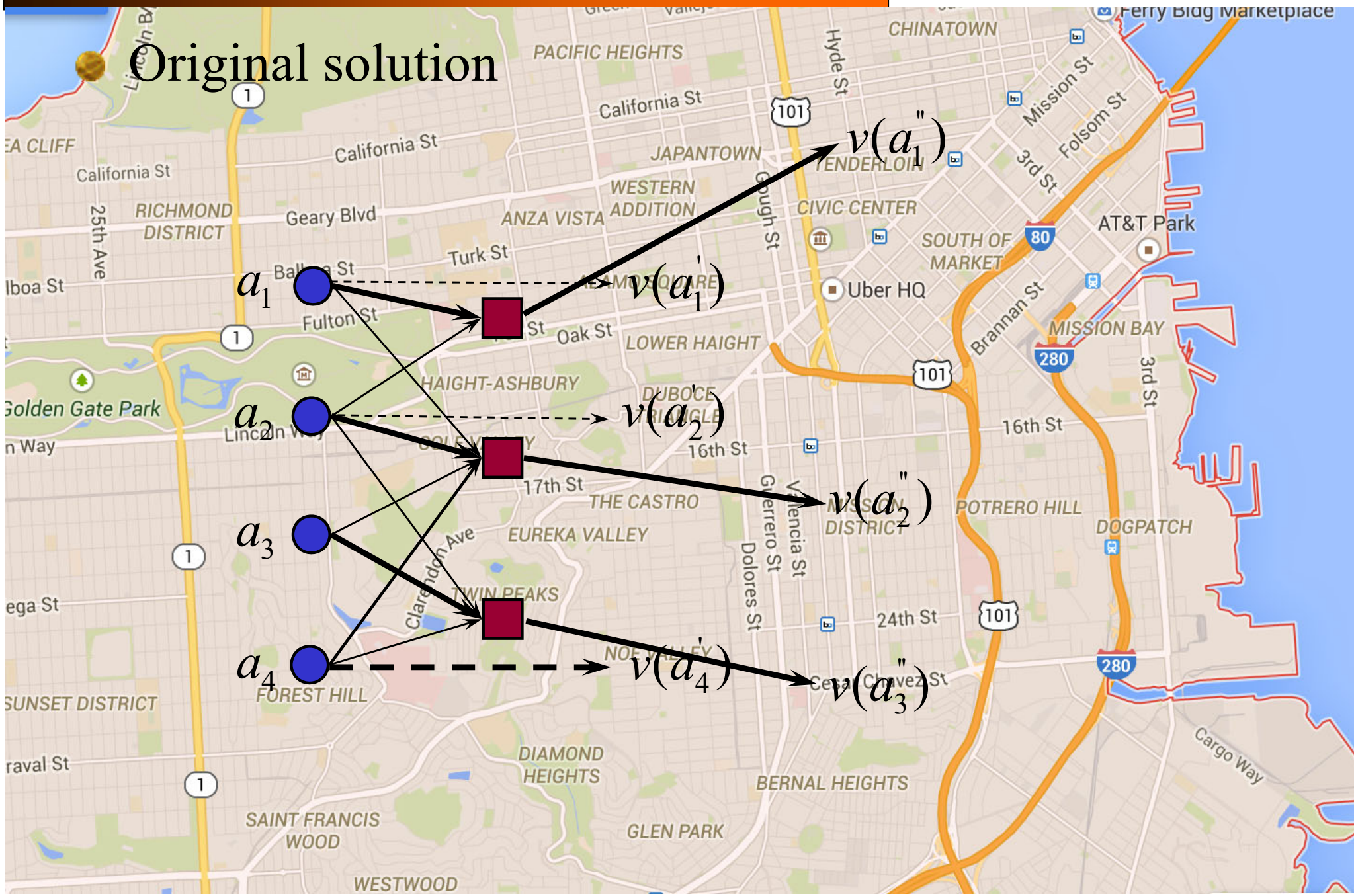
Driverless EV optimization



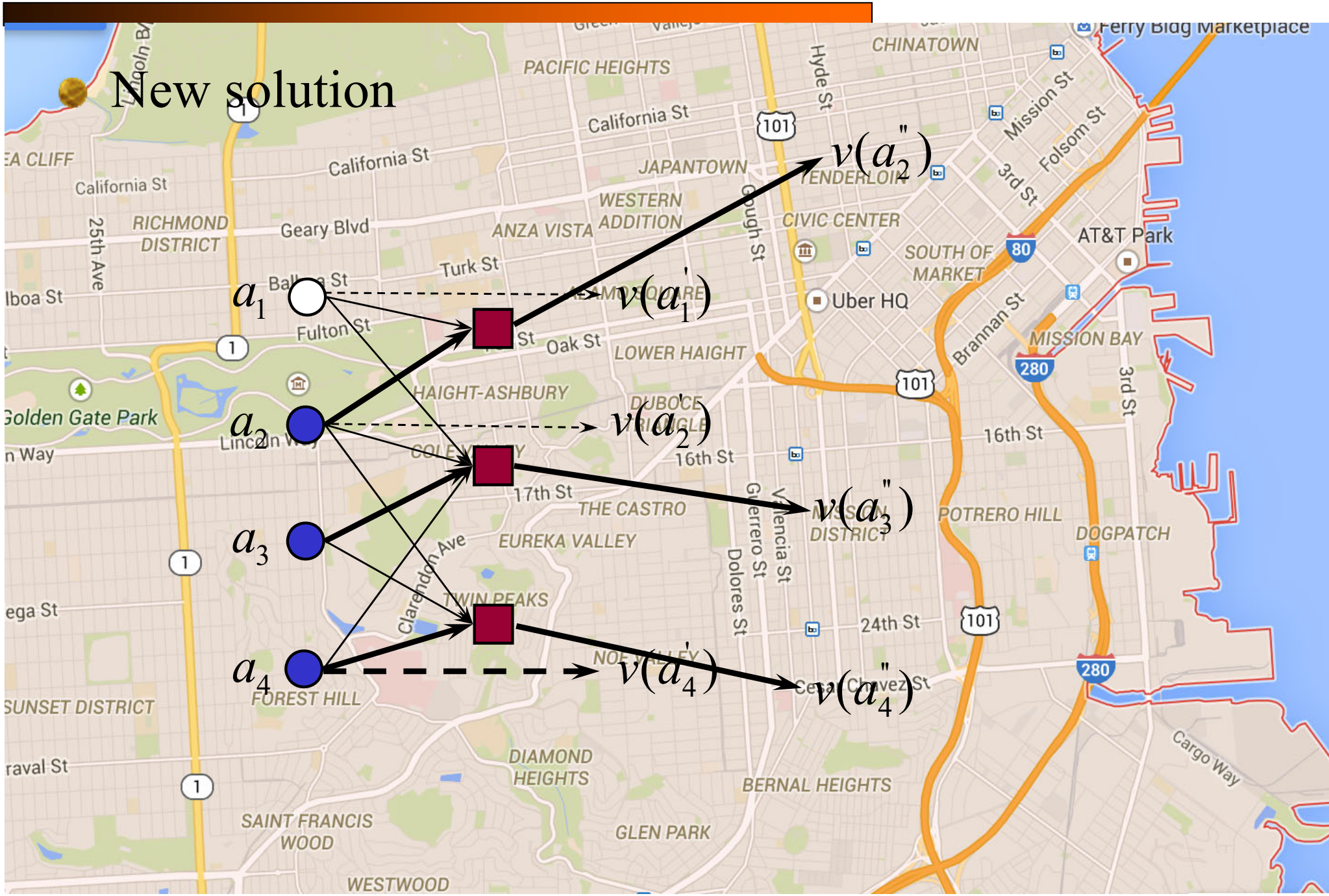
Driverless EV optimization



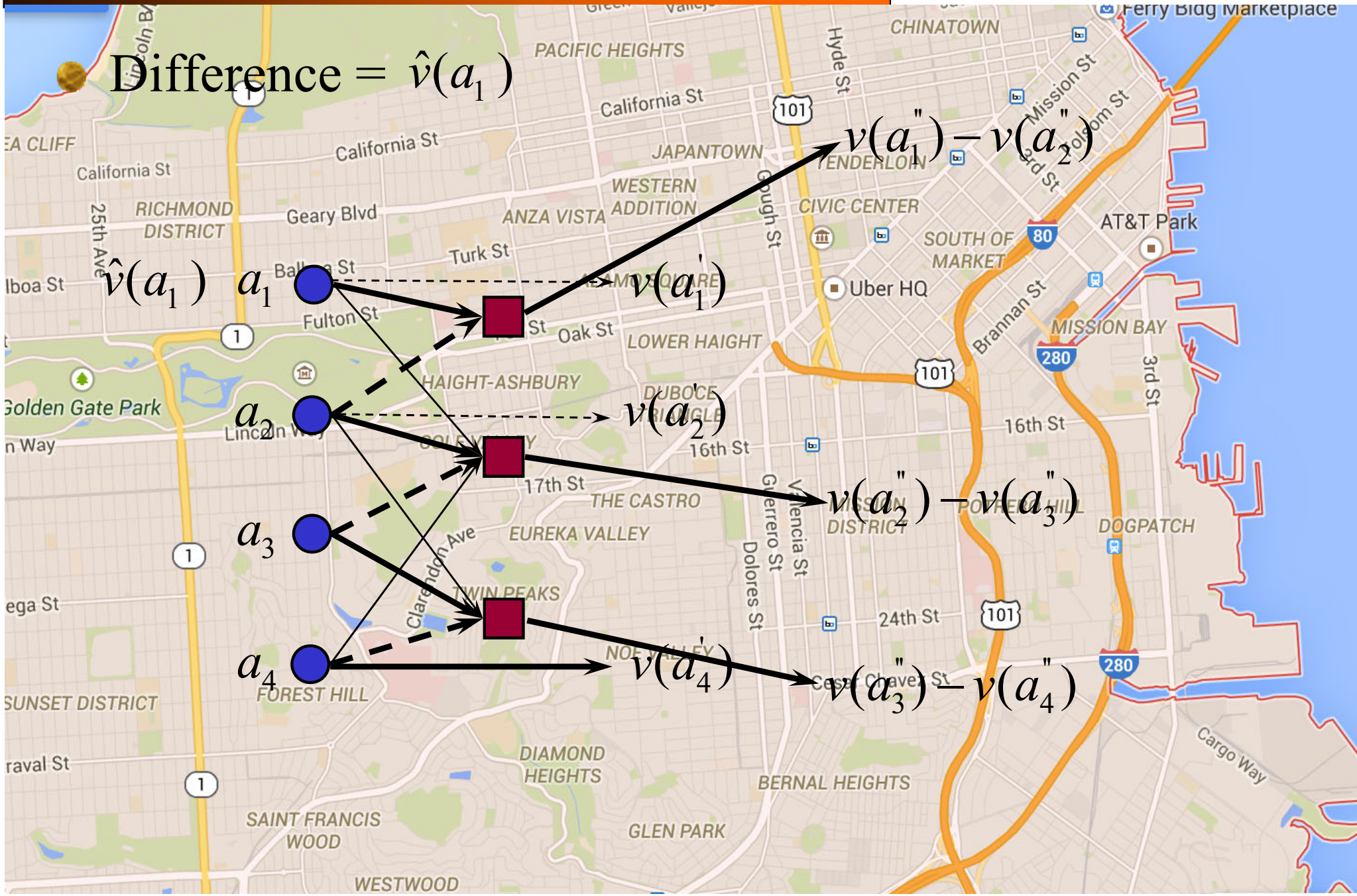
Driverless EV optimization



Driverless EV optimization

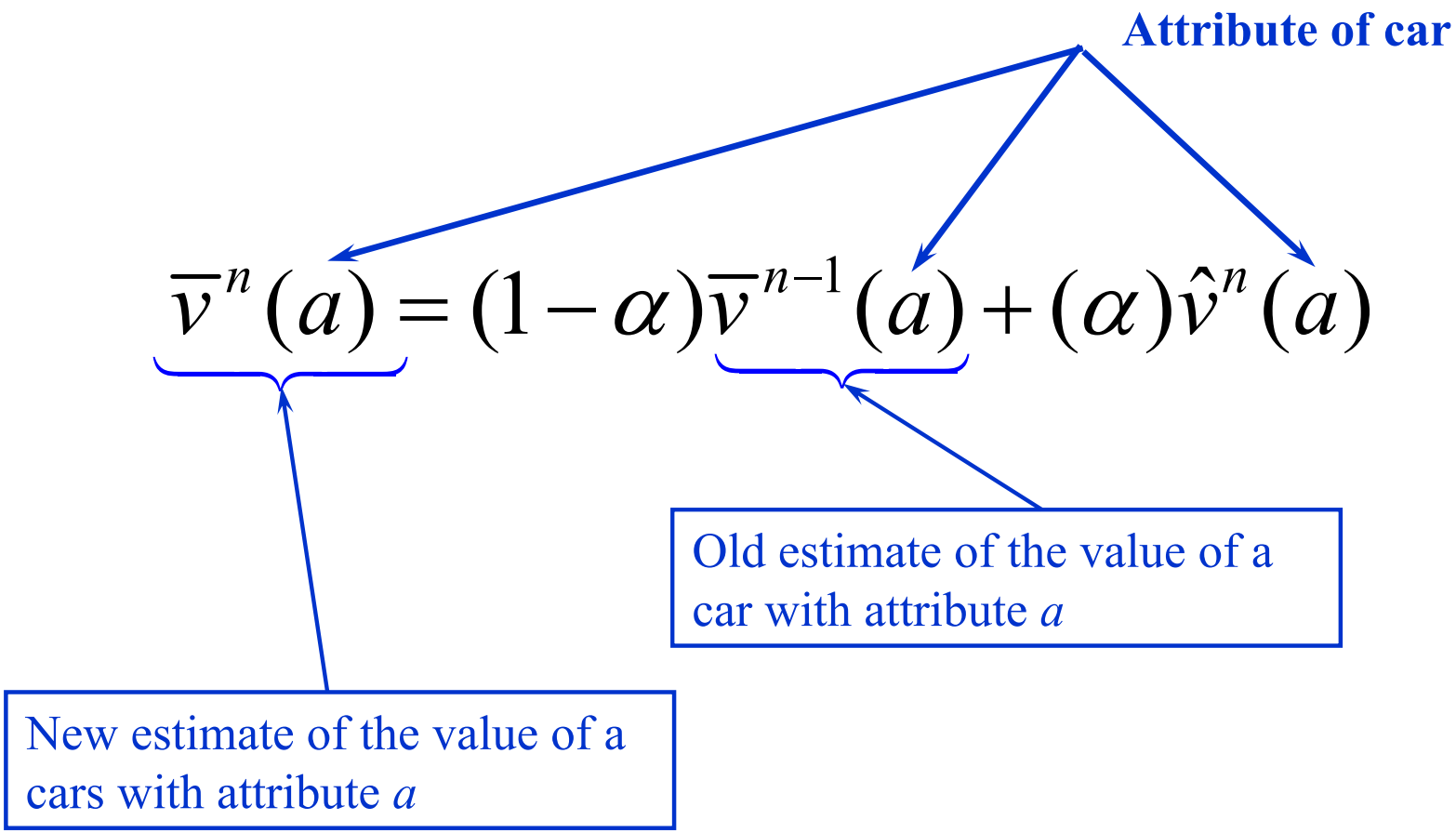


Driverless EV optimization



Driverless EV optimization

- Estimating average values:



Driverless EV optimization

Step 1: Start with a pre-decision state S_t^n

Step 2: Solve the deterministic optimization using an approximate value function:

$$\hat{v}_t^n = \min_x \left(C_t(S_t^n, x_t) + \bar{V}_t^{n-1}(S^{M,x}(S_t^n, x_t)) \right)$$

to obtain x^n.

Deterministic optimization

Step 3: Update the value function approximation

$$\bar{V}_{t-1}^n(S_{t-1}^{x,n}) = (1-\alpha_{n-1})\bar{V}_{t-1}^{n-1}(S_{t-1}^{x,n}) + \alpha_{n-1}\hat{v}_t^n$$

Recursive statistics

Step 4: Obtain Monte Carlo sample of $W_t(\omega^n)$ and compute the next pre-decision state:

$$S_{t+1}^n = S^M(S_t^n, x_t^n, W_{t+1}(\omega^n))$$

Simulation

Step 5: Return to step 1.

Driverless EV optimization

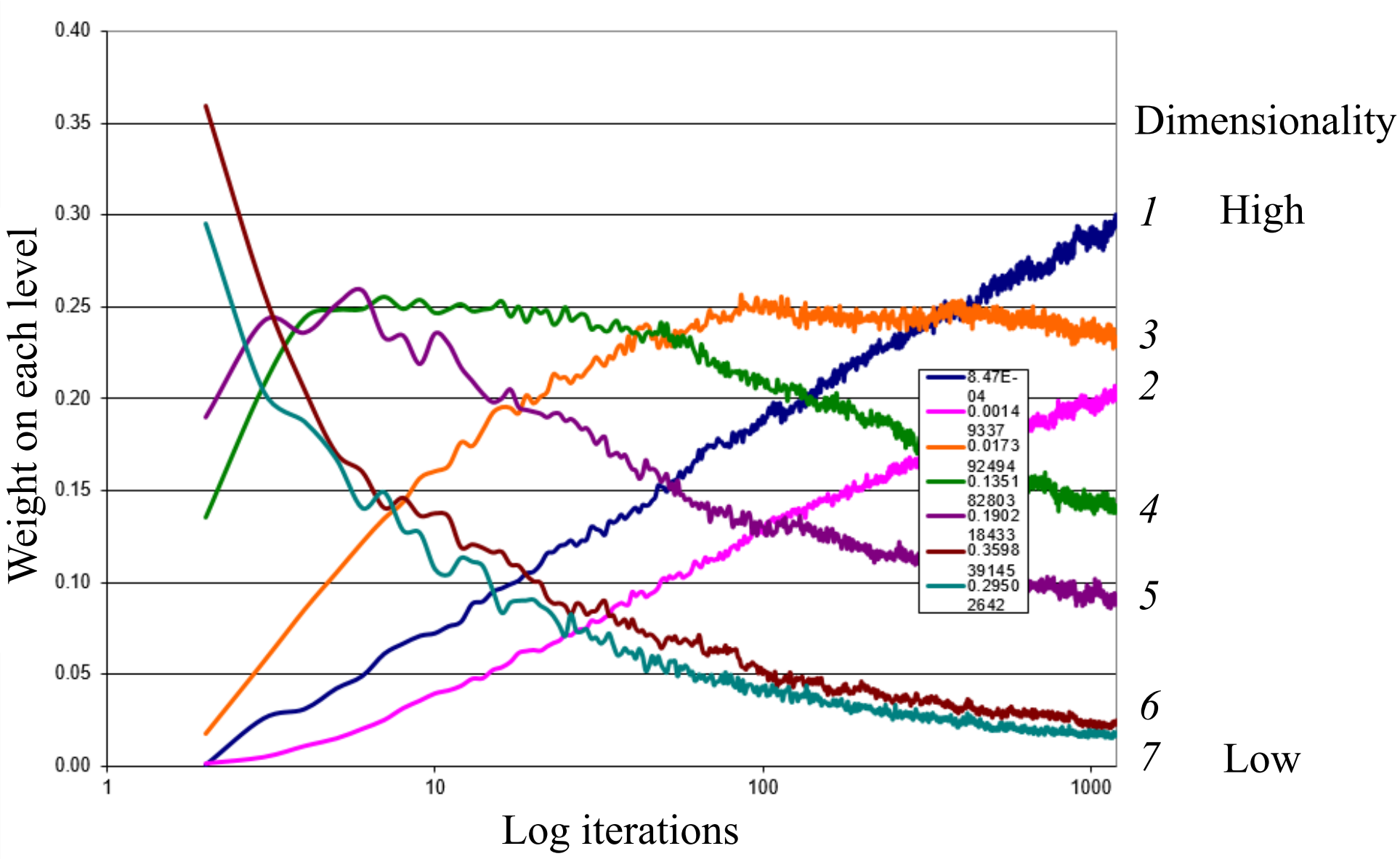
- Estimating average values:
  - » The attributes of a car might include:
    - Location
    - Type of car
    - Charge level
    - Driver attributes?
  - » We estimate values at different levels of aggregation, and then combine them:

$$\bar{v}^n(a) \approx \sum_{g \in G} w_a^{g,n} \bar{v}^{g,n}(a)$$

  - » The weights are proportional to the inverse of the estimate of the variance plus square of the bias.
  - » This is a form of *variable dimensional learning*.

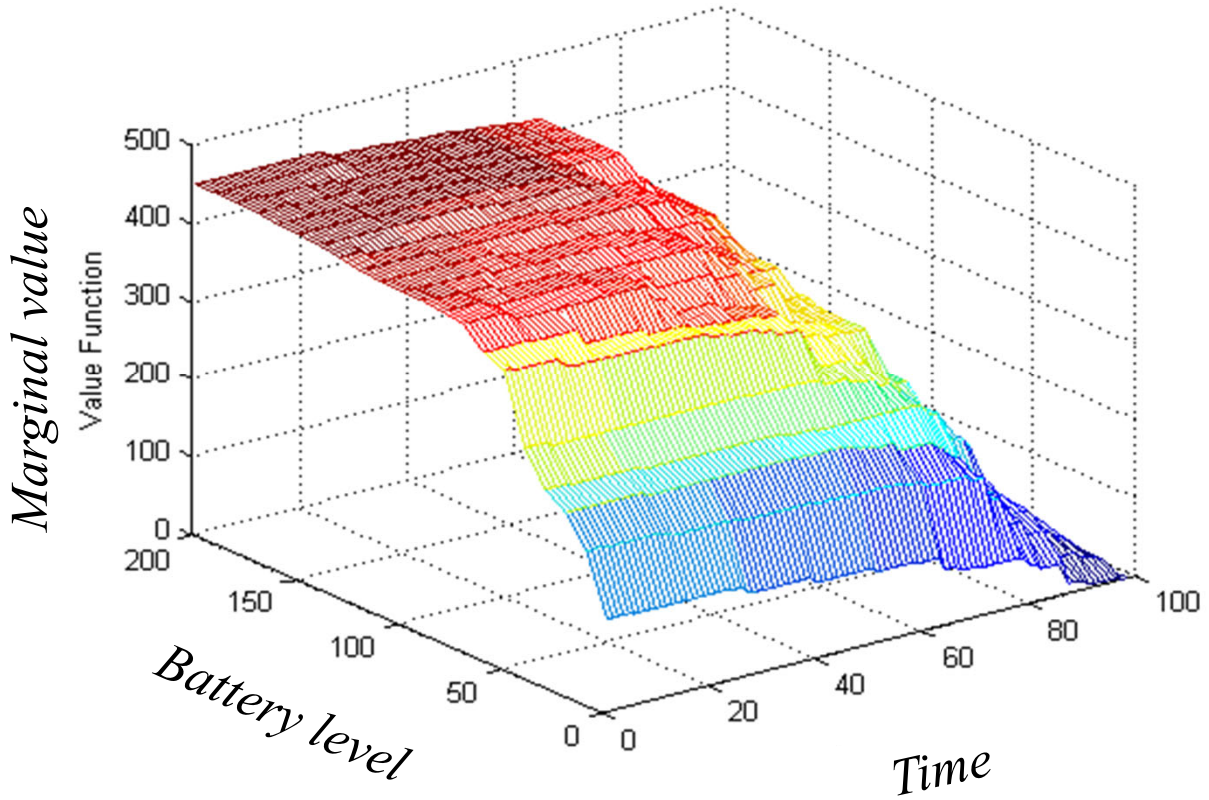
Driverless EV optimization

- Variable-dimensional learning



Driverless EV optimization

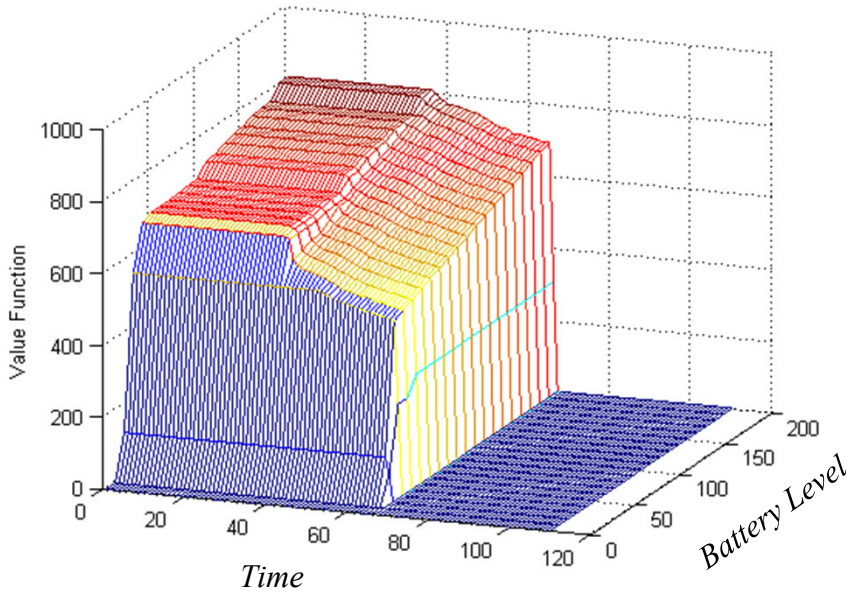
- The value of a vehicle in the future
  - » Value function approximation captures charge level, as well as time and location.
  - » Hierarchical aggregation accelerated the learning process



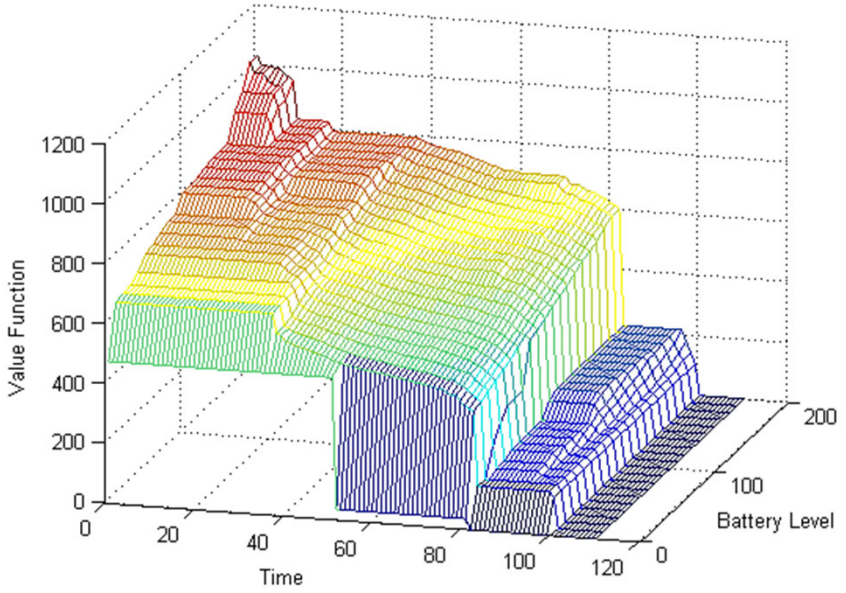
Driverless EV optimization

- Value functions exhibited a variety of shapes
  - » Depends on the patterns of trips out of a zone
  - » This discouraged the use of simple parametric approximations

Value Function for Zone 1

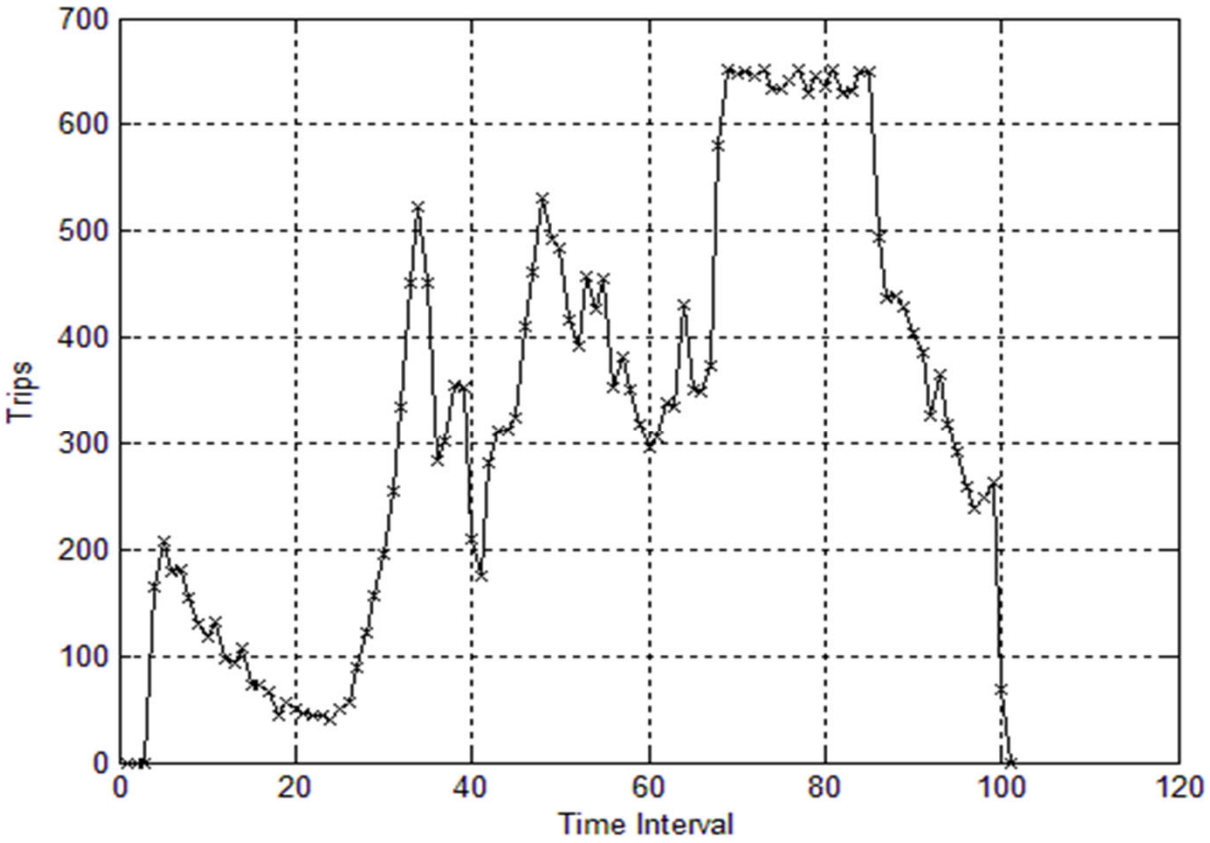


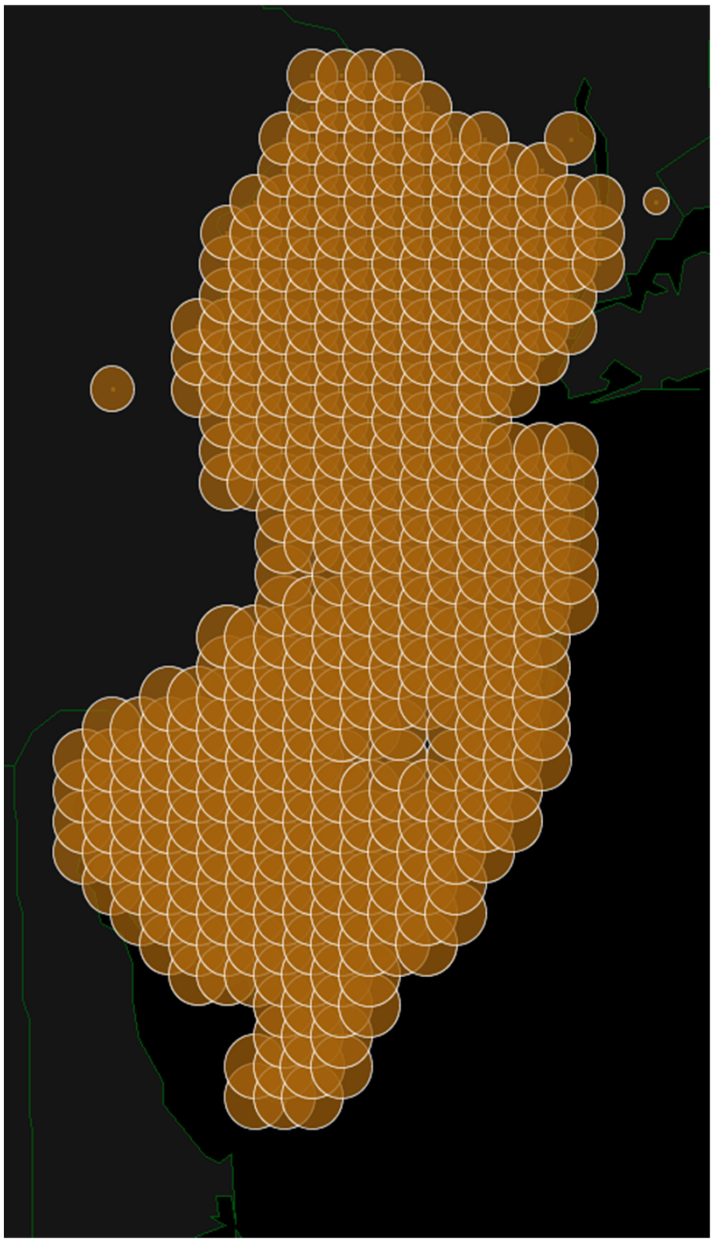
Value Function for Zone 2



Driverless EV optimization

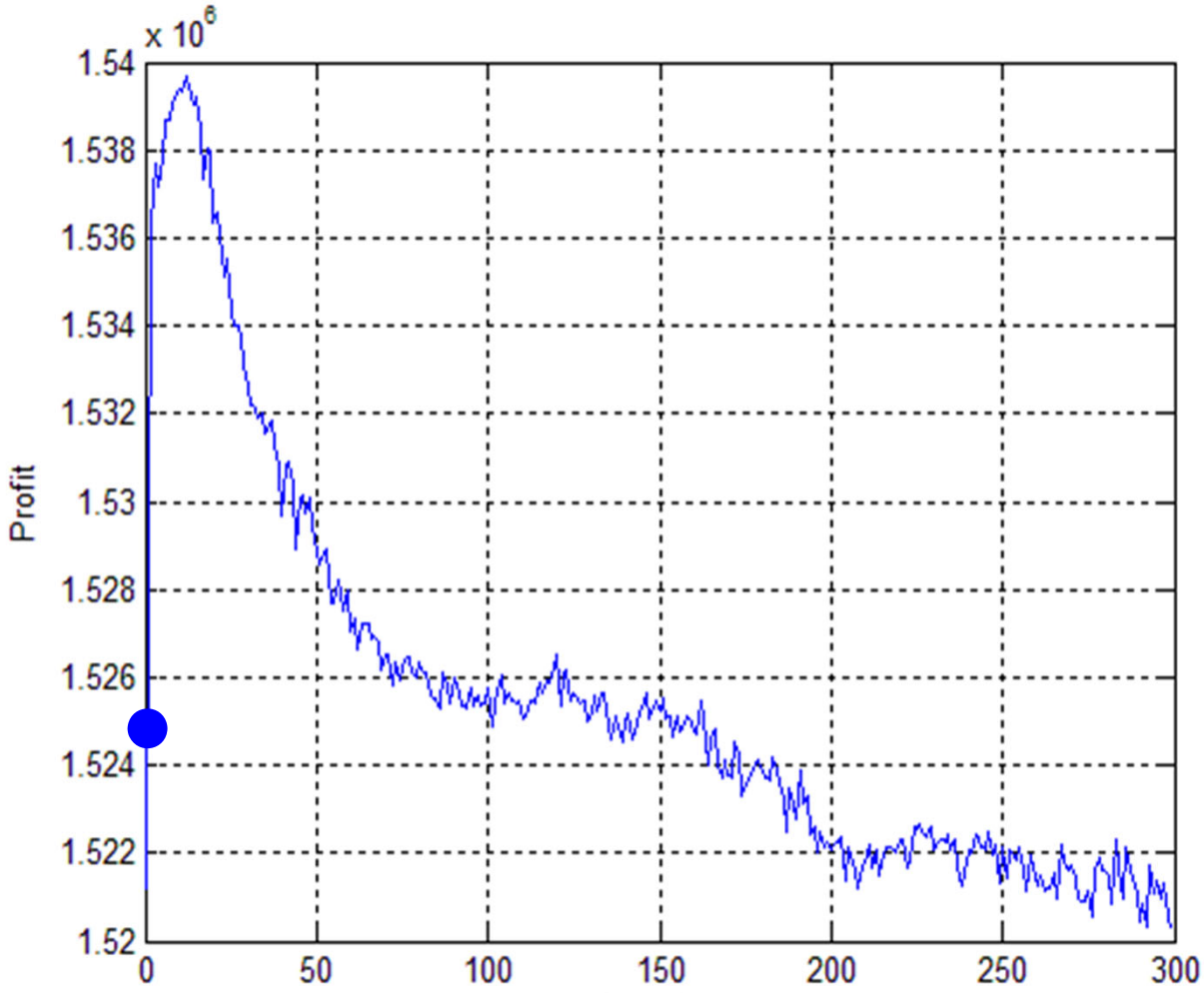
- 22000 zones
- 31560 Trips
- 1000-2000 cars
- Battery capacity: 50-100 KWh





Driverless EV optimization

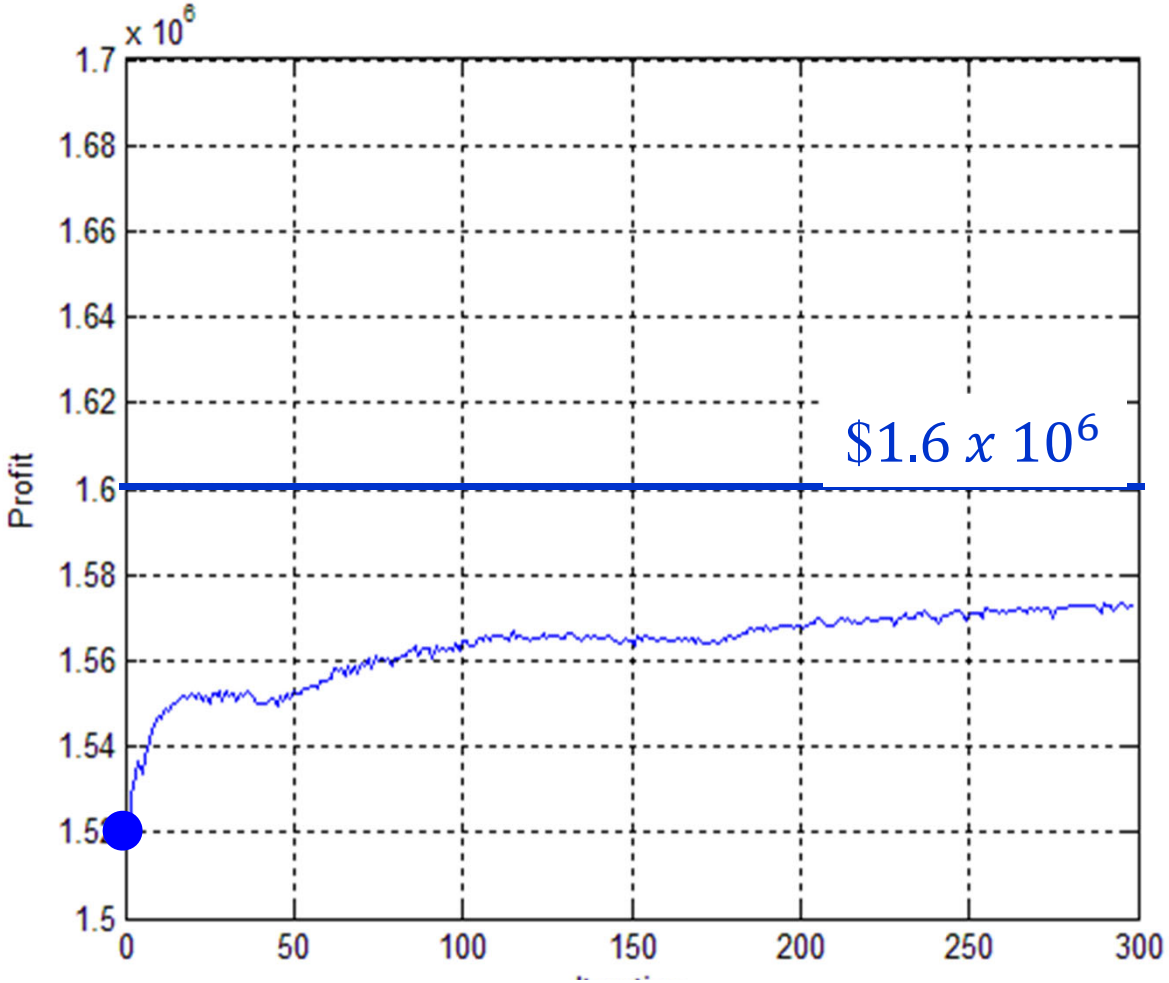
- No aggregation – (!!) Solution gets worse!



Profit versus number of iterations

Driverless EV optimization

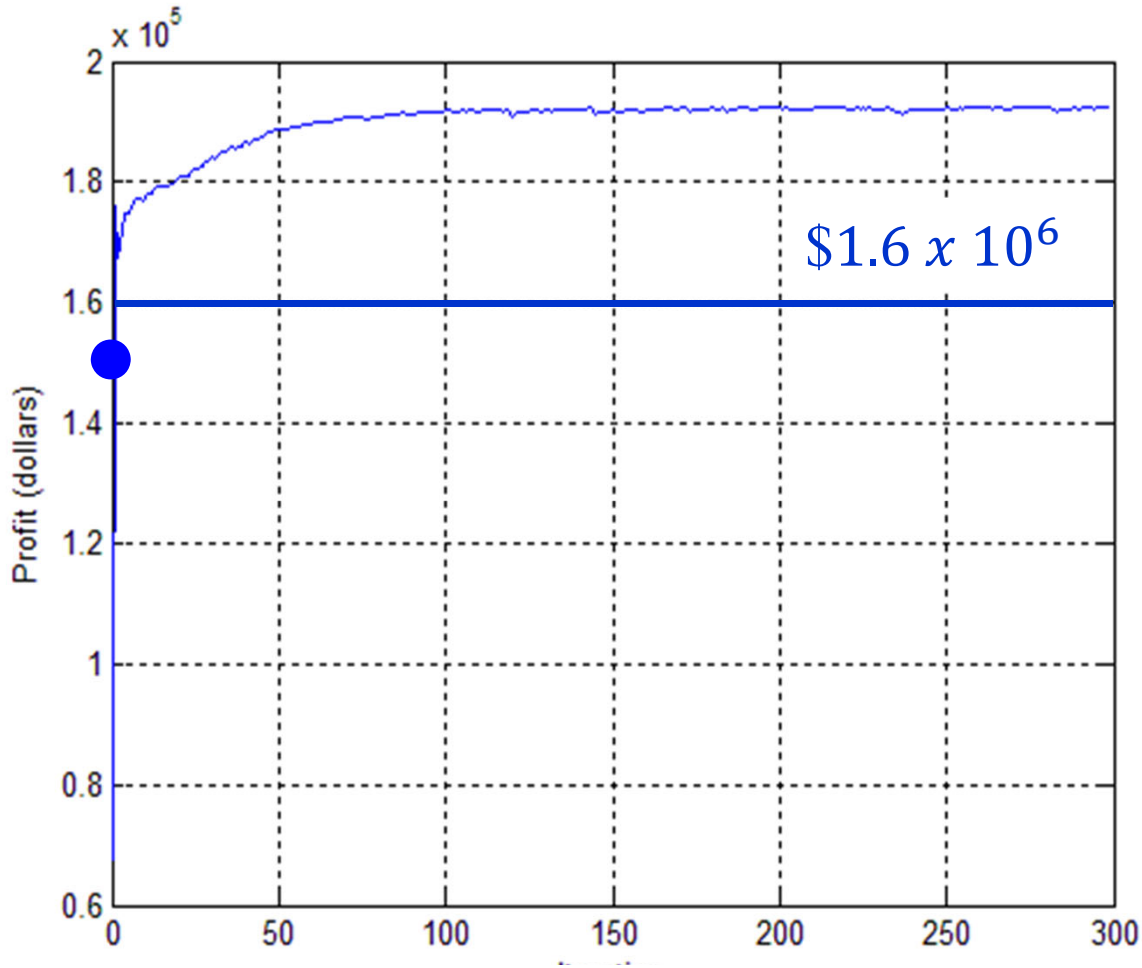
- Three levels of aggregation - Better



Profit versus number of iterations

Driverless EV optimization

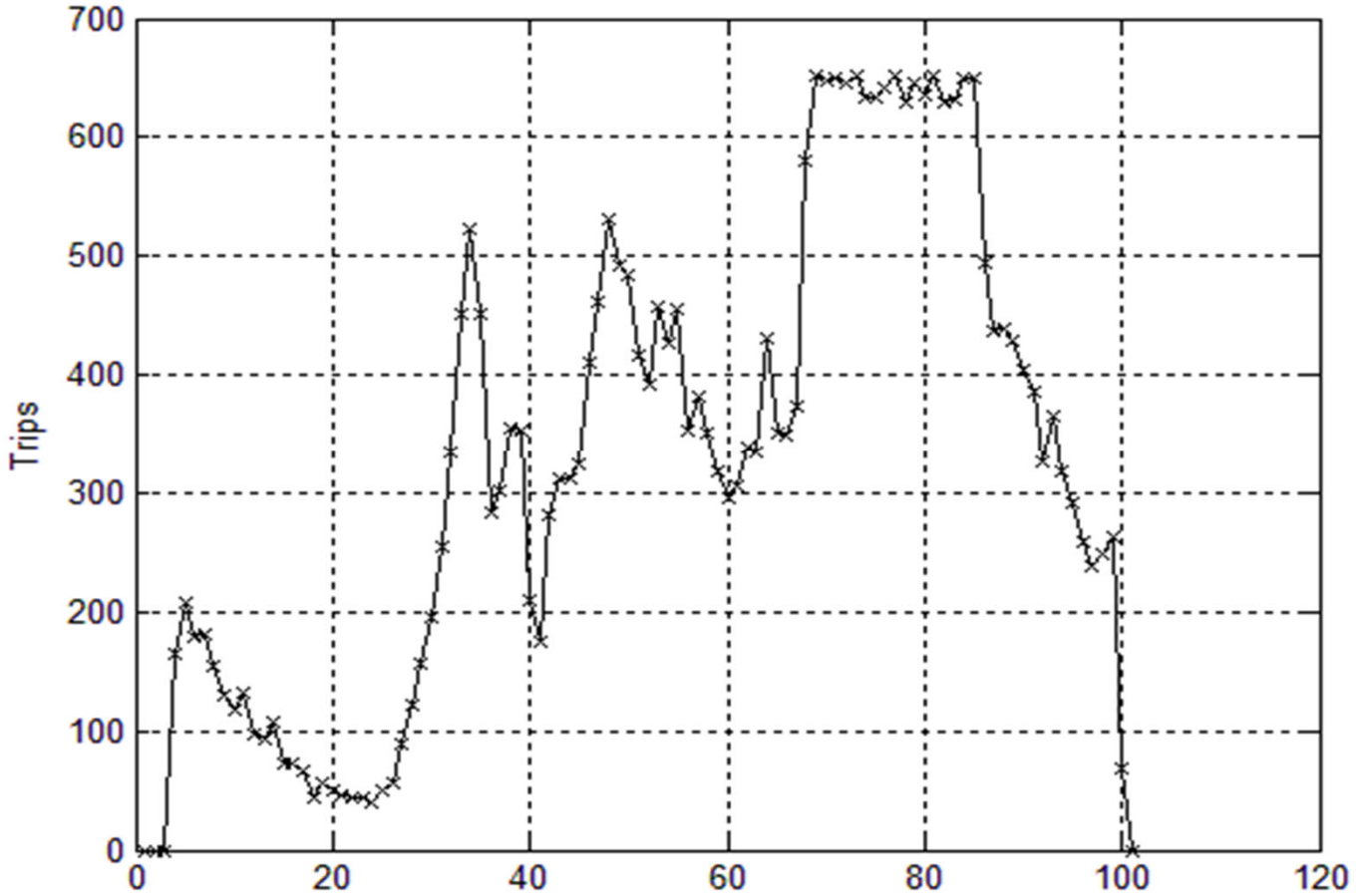
- Five levels of aggregation



Profit versus number of iterations

Driverless EV optimization

- Trip requests over time
  - » Challenge is to recharge during off-peak periods

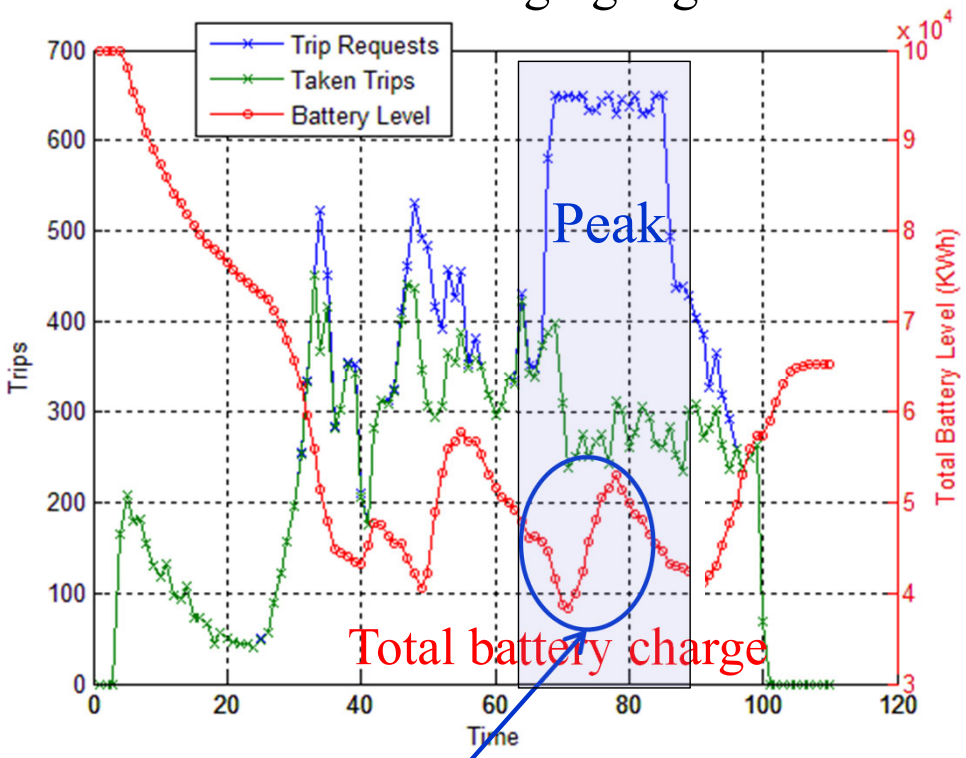


Trip Requests vs Time

Driverless EV optimization

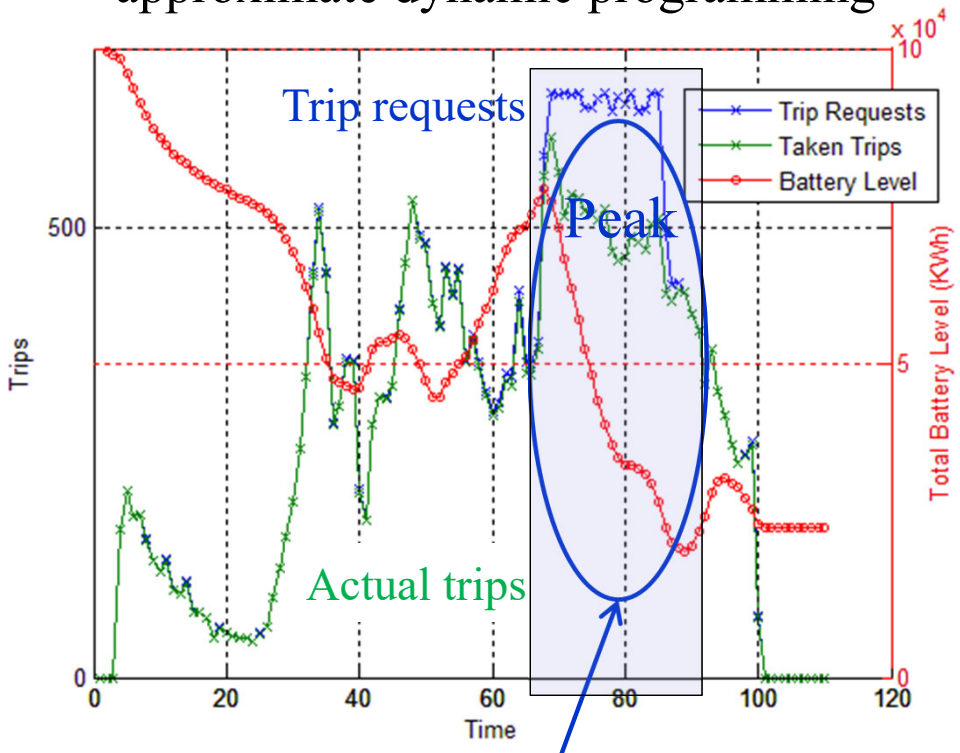
- Heuristic dispatch vs. ADP-based policies
  - » Effect of value function approximations on recharging

Heuristic recharging logic



Recharging during peak period

Recharging controlled by approximate dynamic programming



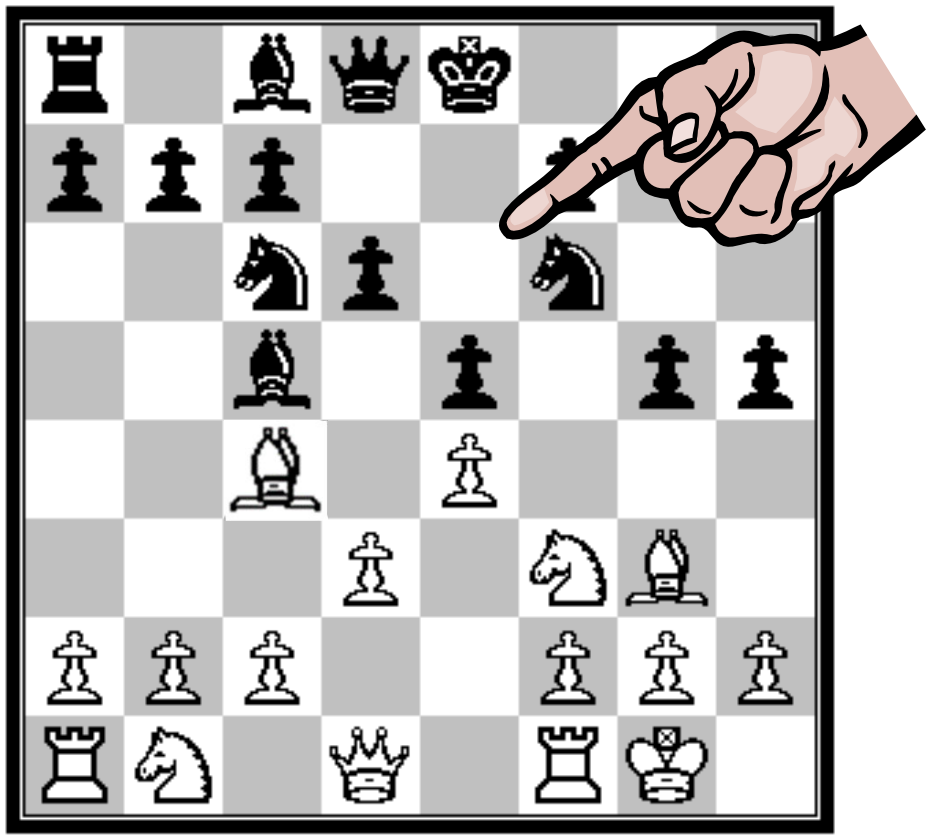
No recharging during peak period

Outline

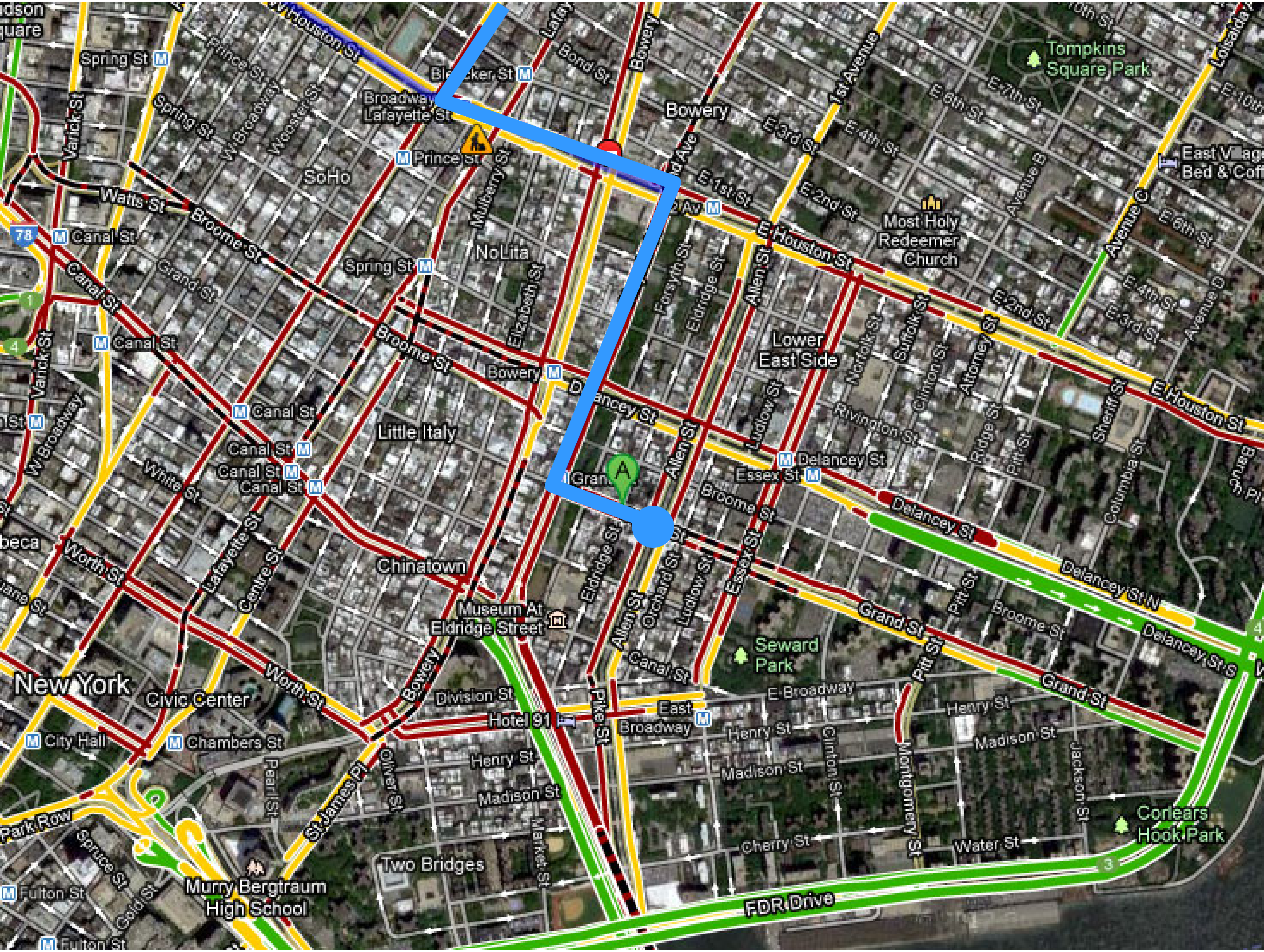
- The four classes of policies
  - » Policy function approximations (PFAs)
  - » Cost function approximations (CFAs)
  - » Value function approximations (VFAs)
  - » Direct lookahead policies (DLAs)
  - » Hybrid direct lookahead/CFA
  - » Any of the four classes may work best!

Lookahead policies

- Planning your next chess move:



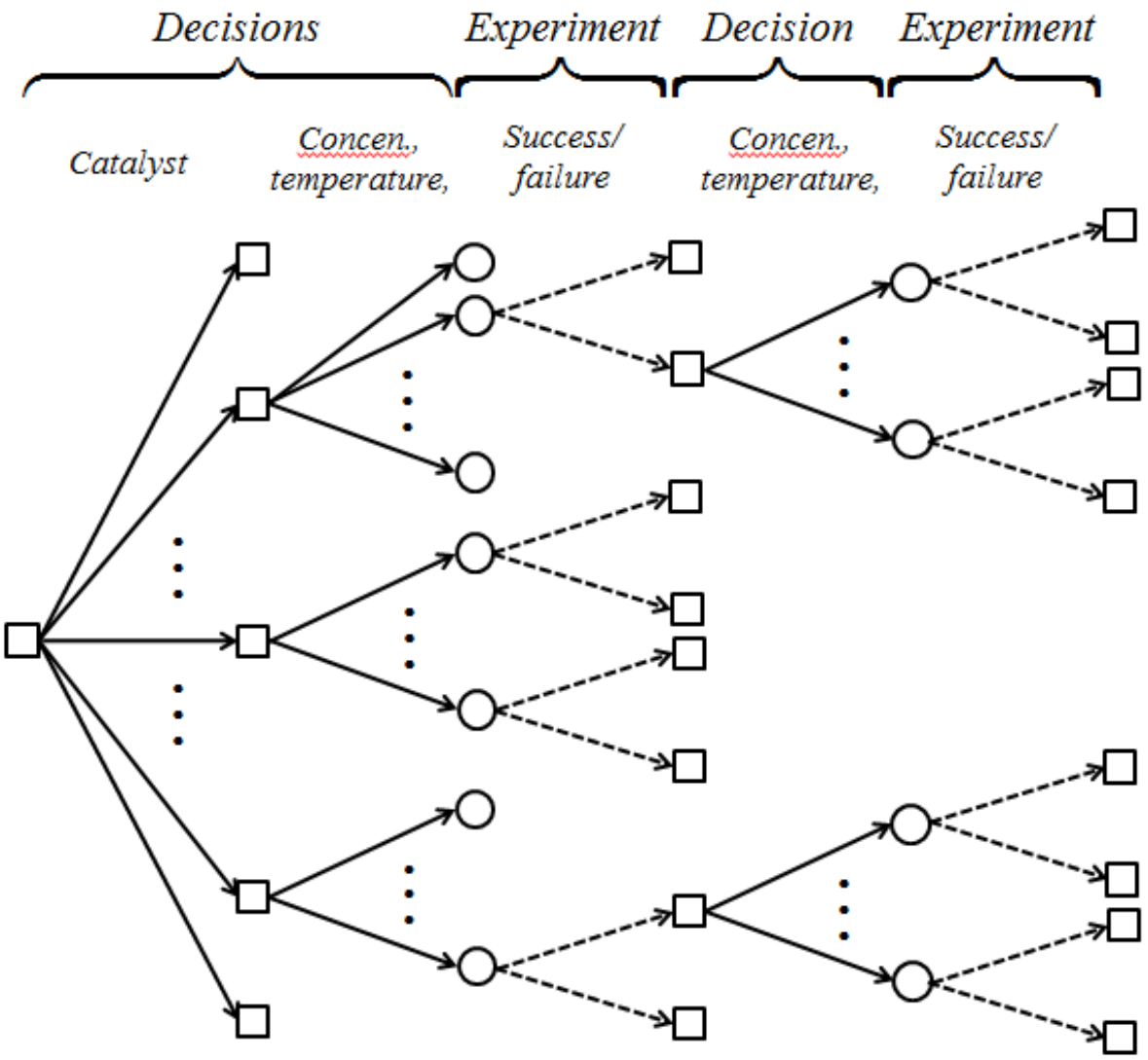
» You put your finger on the piece while you think about moves into the future. This is a lookahead policy, illustrated for a problem with discrete actions.



Spring St
Bleecker St
Broadway Lafayette St
Prince St
Bond St
Bowery
SoHo
NoLita
Spring St
Canal St
Canal St
Canal St
Canal St
Canal St
Little Italy
Bowery
Delancey St
Lower East Side
Houston St
E Houston St
E 1st St
E 2nd St
E 3rd St
E 4th St
Most Holy Redeemer Church
Tompkins Square Park
Essex St
Delancey St
Grand St
Chinatown
Museum At Eldridge Street
Seward Park
New York
Civic Center
City Hall
Chambers St
Division St
Hotel 91
East Broadway
E Broadway
Henry St
Madison St
Cherry St
Water St
Two Bridges
Murry Bergtraum High School
Fulton St
FDR Drive
Corlears Hook Park
Park Row
Worth St
Broome St
Watts St
Varick St
Elizabeth St
Mulberry St
Forsyth St
Eldridge St
Allen St
Orchard St
Ludlow St
Essex St
Norfolk St
Suffolk St
Clinton St
Attorney St
Ridge St
Pitt St
Sheriff St
Columbia St
Avenue B
Avenue C
Avenue D
1st Avenue
Rivington St
Pike St
Market St
Oliver St
St James Pl
Pearl St
Centre St
Lafayette St
White St
Montgomery St
Jackson St
Delancey St N
Delancey St S

Lookahead policies

- Decision trees:

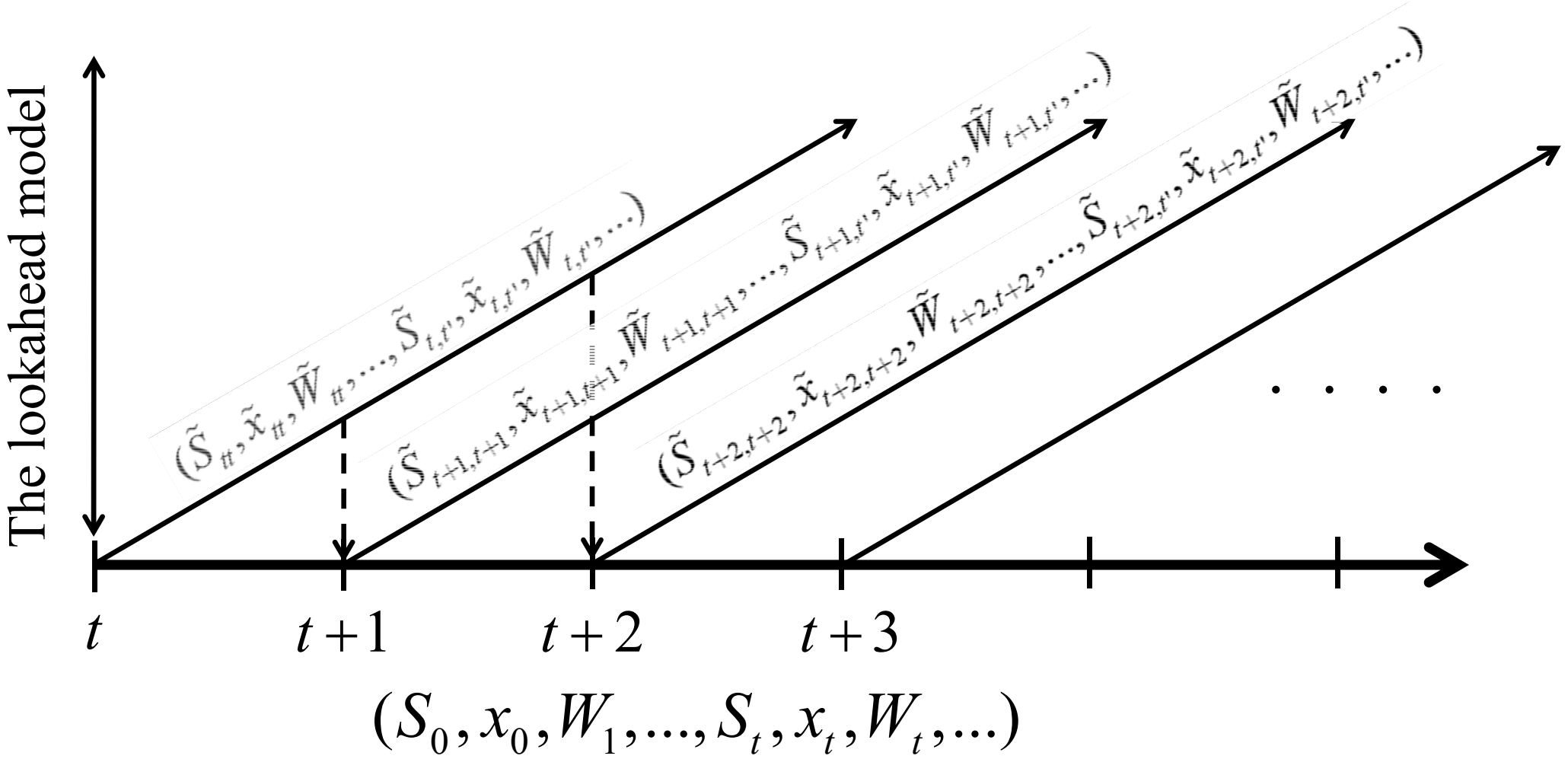


Lookahead policies

- Modeling lookahead policies
  - » Lookahead policies solve a *lookahead model*, which is an approximation of the future.
  - » It is important to understand the difference between the:
    - Base model – this is the model we are trying to solve by finding the best policy. This is usually some form of simulator.
    - The lookahead model, which is our approximation of the future to help us make better decisions now.
  - » The base model is typically a simulator, or it might be the real world.

Lookahead policies

- Lookahead models
  - » Use tilde variables with double time indices.



Lookahead policies

- Lookahead models use five classes of approximations:
  - » Horizon truncation – Replacing a longer horizon problem with a shorter horizon
  - » Stage aggregation – Replacing multistage problems with two-stage approximation.
  - » Outcome aggregation/sampling – Simplifying the exogenous information process
  - » Discretization – Of time, states and decisions
  - » Dimensionality reduction – We may ignore some variables (such as forecasts) in the lookahead model that we capture in the base model (these become *latent* variables in the lookahead model).

Lookahead policies

- The lookahead state variable

$$\tilde{S}_{tt'} = \left(\tilde{R}_{tt'}, \tilde{D}_{tt'}, \tilde{E}_{tt'}, (\tilde{p}_{tt'}, \tilde{p}_{t,t'-1}, \tilde{p}_{t,t'-2}), \underbrace{(\tilde{\theta}_{tt'}, \tilde{M}_{tt'}), \tilde{f}^{D}_{tt'}}_{\text{Latent variables}} \right)$$

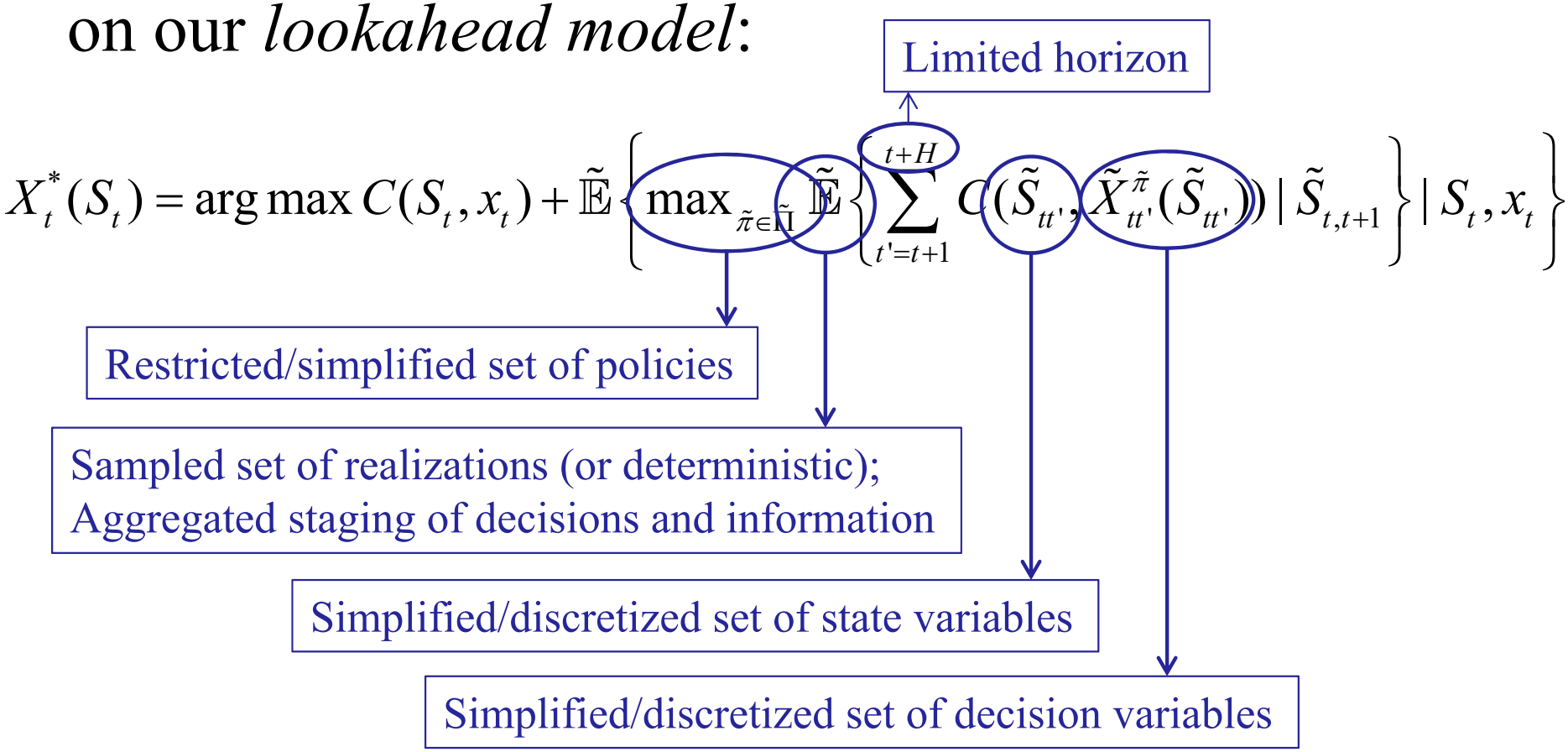
- Common simplifications in lookahead models:

  » Ignore the updating of estimates of parameters $\tilde{\theta}_{tt'} = \bar{\theta}_t$

  » Ignore the adaptive updating of forecasts $\tilde{f}^{D}_{tt'} = f^{D}_{t}$

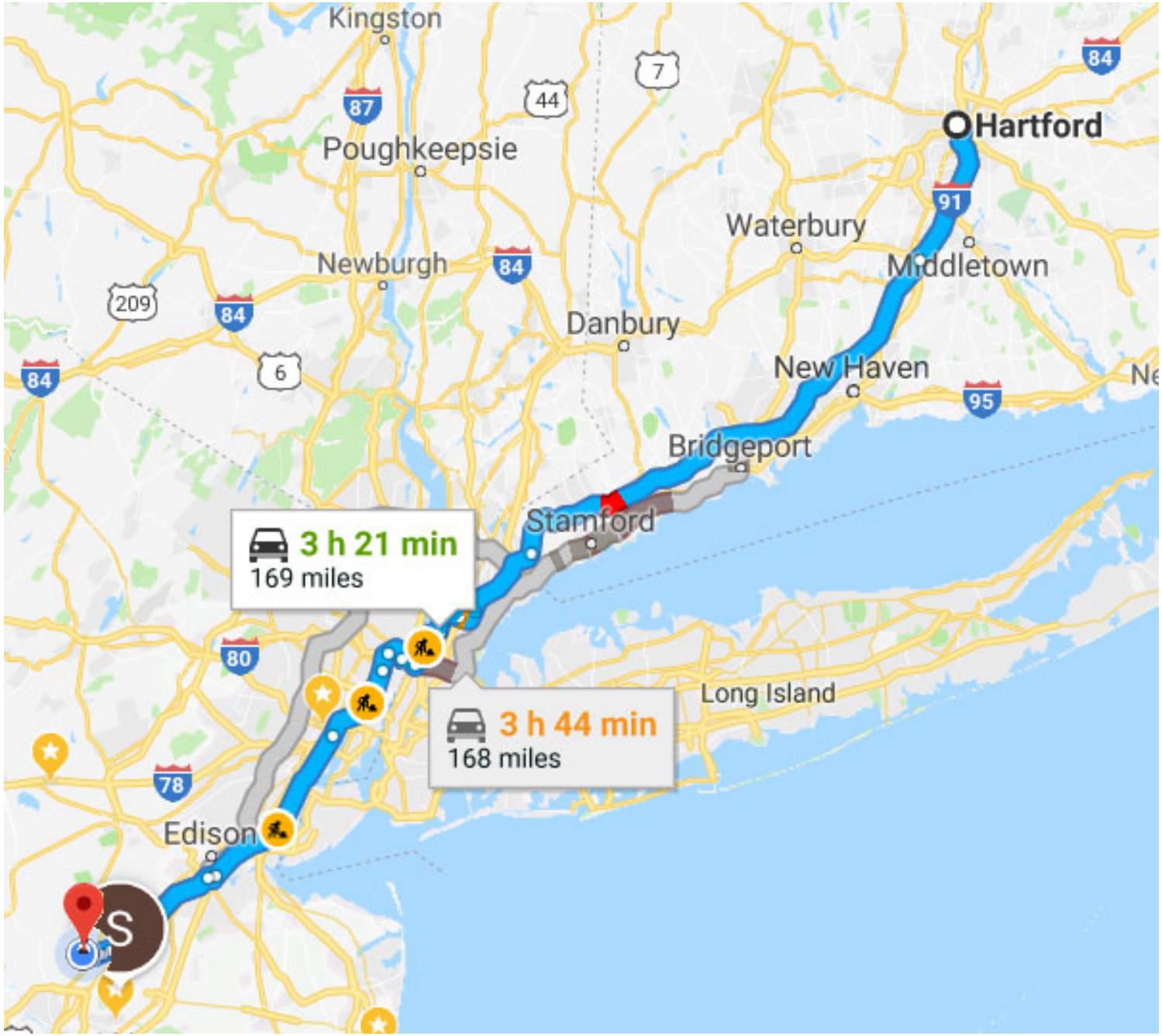
Lookahead policies

- We can use this notation to create a policy based on our *lookahead model*:

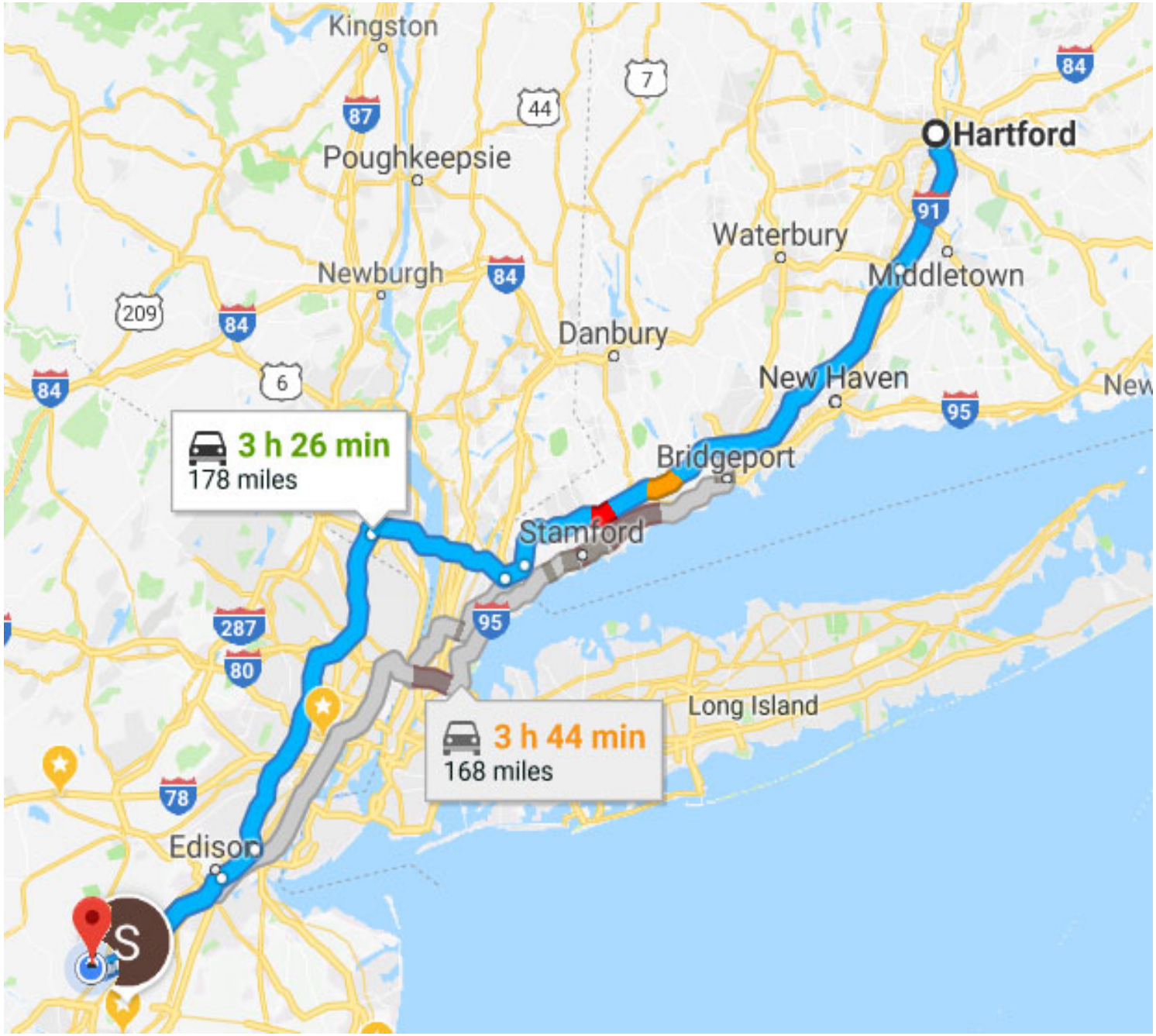


» Simplest lookahead is deterministic.

Dynamic shortest paths



Dynamic shortest paths



Dynamic shortest paths

- The problem
  - » Finding the best path through a stochastic dynamic network.
- The policy
  - » We can try to solve a stochastic lookahead model (perhaps using approximate dynamic programming).
  - » We can solve a deterministic lookahead model using point estimates of travel times that are updated from time to time.
  - » We can solve a parameterized lookahead model, where we use the θ-percentile of the travel time on each link.

Dynamic shortest paths

- Modeling:
  - » Objective function
    - Cost per period

$$\mathrm{C}\left(\mathrm{S_t}, \mathrm{X_t^\pi}(S_t)\right) = (X_t^\pi\ (S_t))^T \hat{c}_t$$
$$= \sum_j x_{t,i_t,j}^\pi\ \hat{c}_{t,i_t,j}$$
$$= \text{Costs incurred at time t.}$$

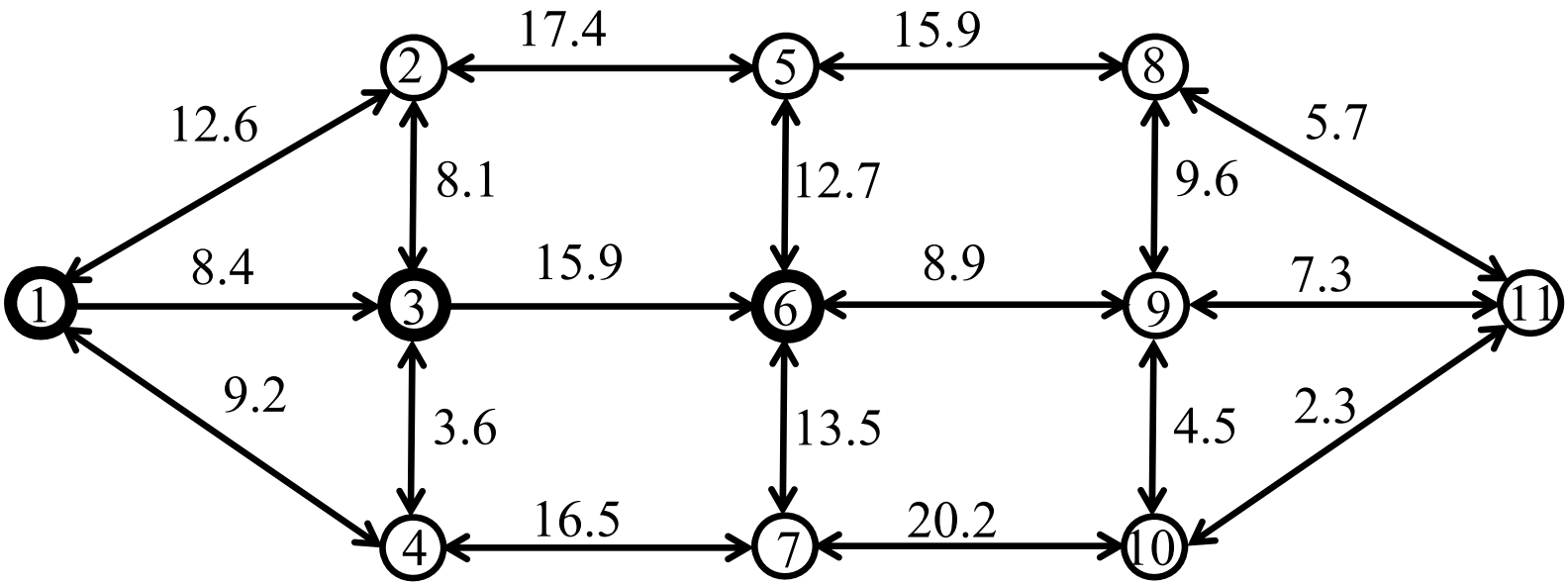
    - Total costs:

$$\min_\pi \mathbb{E} \sum_{t=0}^{T} \mathrm{C}(\mathrm{S_t}, \mathrm{X_t^\pi}(S_t))$$

  - » This is the base model.
  - » We are going to build a policy based on a lookahead model.

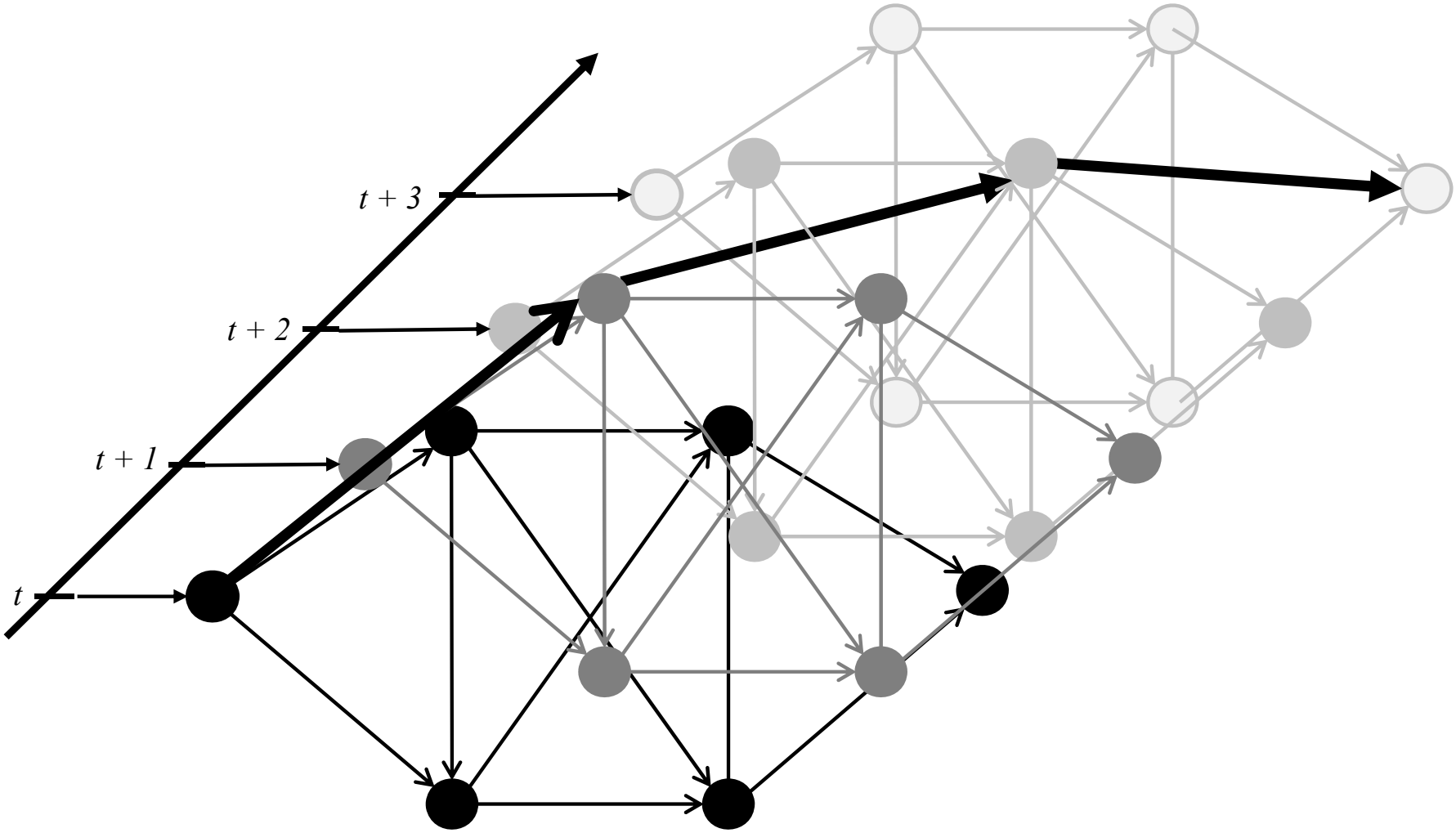
Dynamic shortest paths

- Estimates of travel times over the network as of time *t*. These estimates are updated over time.



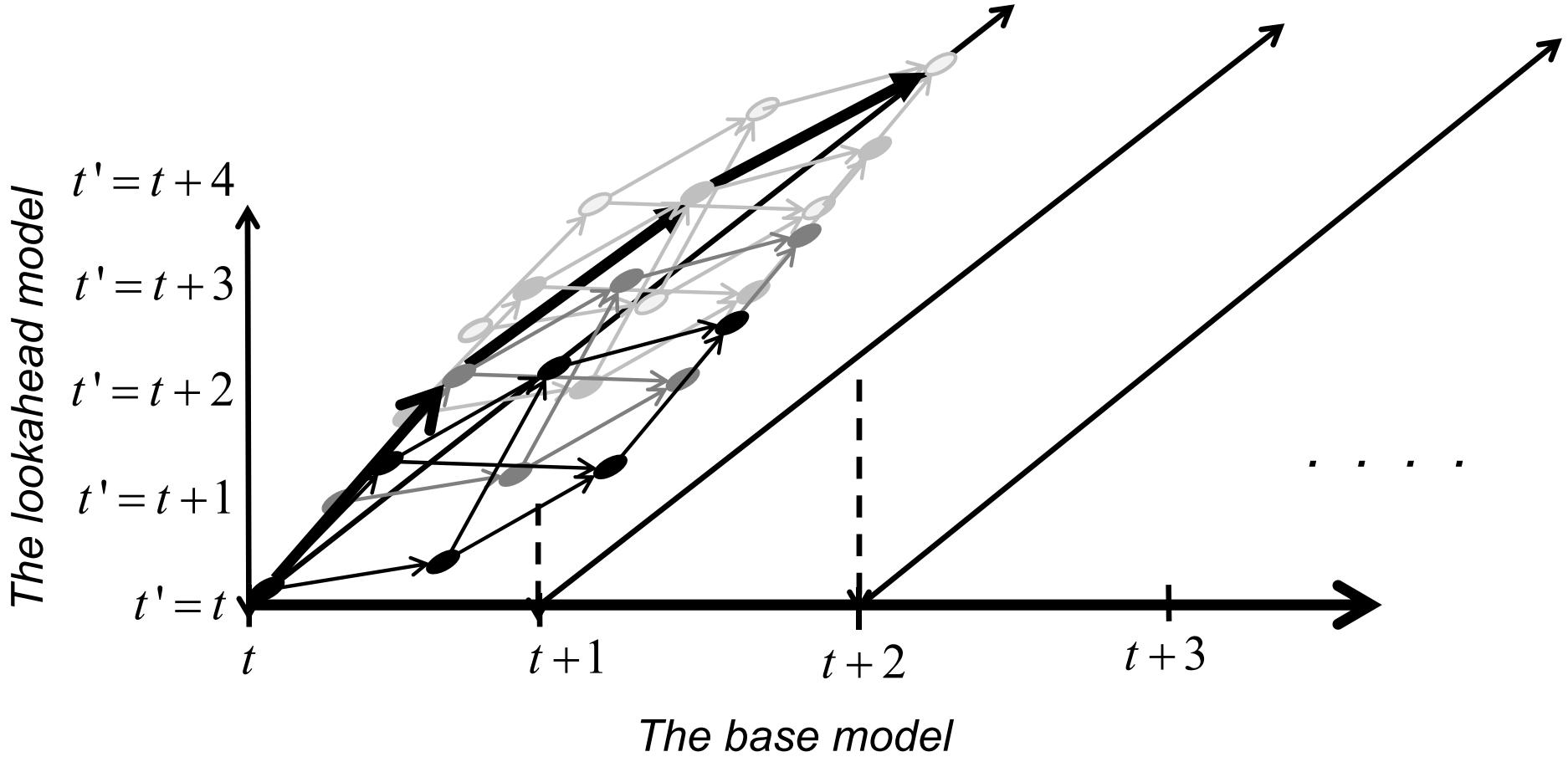
Dynamic shortest paths

- A time-dependent, deterministic network



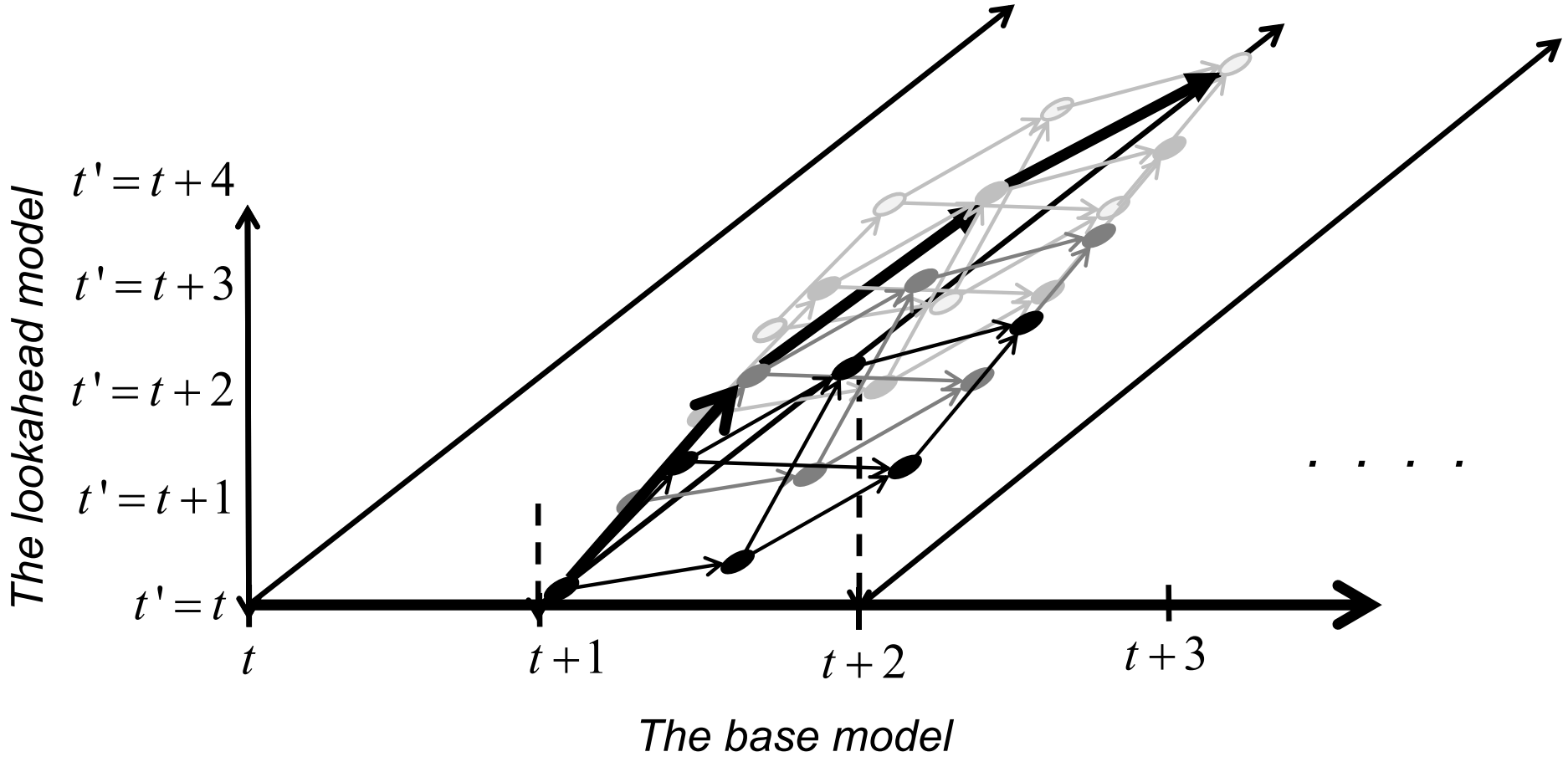
Dynamic shortest paths

- A time-dependent, deterministic lookahead network



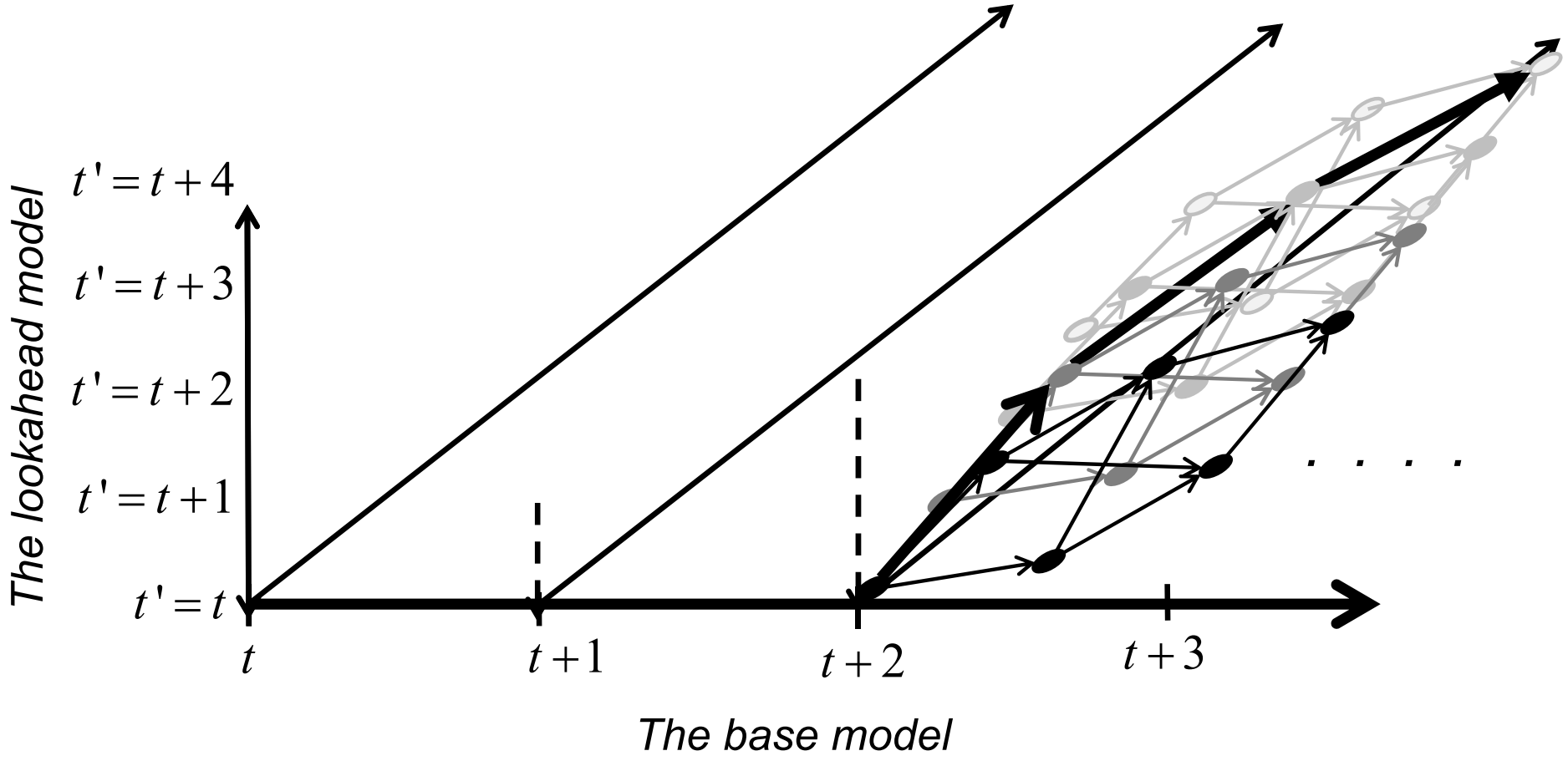
Dynamic shortest paths

- A time-dependent, deterministic lookahead network



Dynamic shortest paths

- A time-dependent, deterministic lookahead network



Dynamic shortest paths

- Simulating a lookahead policy

We would like to compute

Choice of link from shortest path problem

$$F^{\pi} = \mathbb{E}\sum_{t=0}^{T}\sum_{i,j} X^{\pi}_{t,ij}(S_t)\hat{c}_{t,ij}$$

but this is intractable.

Let ω be a sample realization of costs

$$\hat{c}_{t,t',ij}(\omega), \hat{c}_{t+1,t',ij}(\omega), \hat{c}_{t+2,t',ij}(\omega), \ldots$$

Now simulate the policy

$$\hat{F}^{\pi}(\omega^n) = \sum_{t=0}^{T}\sum_{i,j} X^{\pi}_{t,ij}(S_t(\omega^n))\hat{c}_{t,ij}(\omega^n)$$

Finally, get the average performance

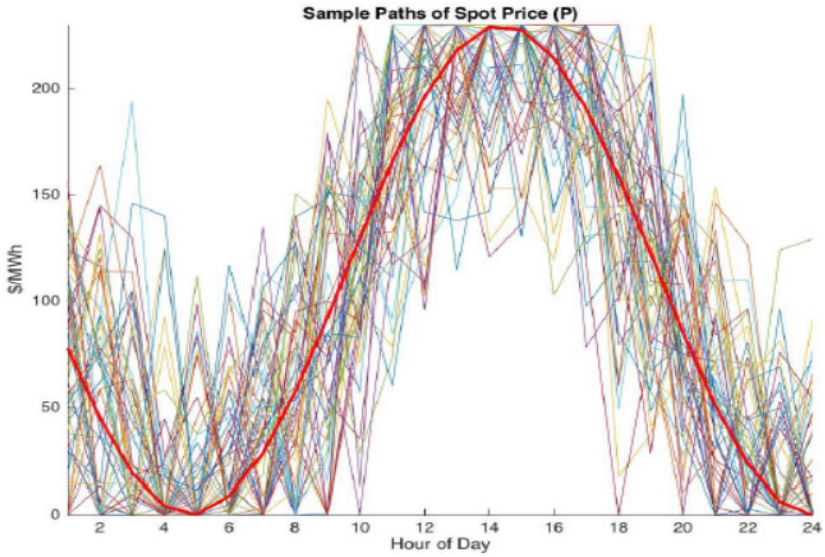
$$\bar{F}^{\pi} = \frac{1}{N}\sum_{n=1}^{N}\hat{F}^{\pi}(\omega^n)$$

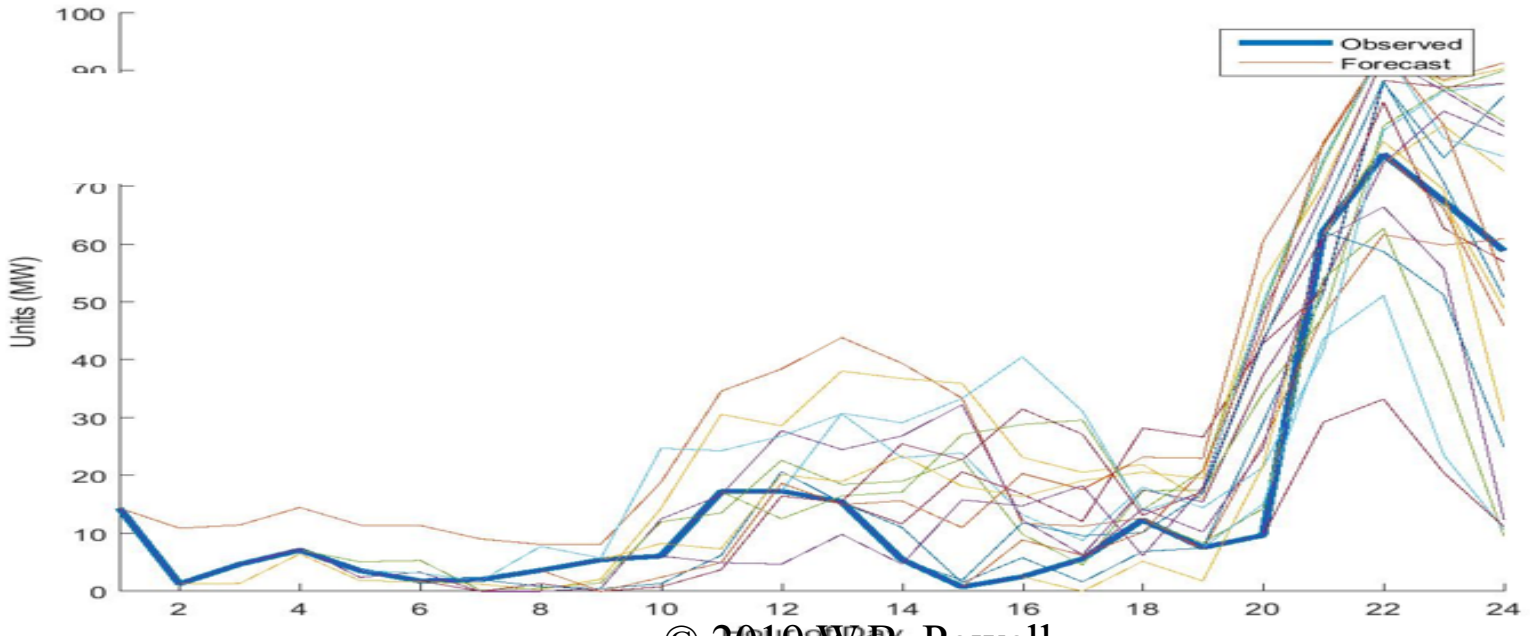
Outline

- The four classes of policies
  - » Policy function approximations (PFAs)
  - » Cost function approximations (CFAs)
  - » Value function approximations (VFAs)
  - » Direct lookahead policies (DLAs)
  - » Hybrid direct lookahead/CFA
  - » Any of the four classes may work best!

Hybrid direct lookahead/CFA

- An energy storage problem:





Four (meta)classes of policies

Policy search

1) **Policy function approximations (PFAs)**

» Lookup tables, rules, parametric/nonparametric functions

2) **Cost function approximation (CFAs)**

» $X^{CFA}(S_t \mid \theta) = \arg\max_{x_t \in \bar{X}_t^{\pi}(\theta)} \bar{C}^{\pi}(S_t, x_t \mid \theta)$

3) **Policies based on value function approximations (VFAs)**

» $X_t^{VFA}(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \bar{V}_t^{x} \left(S_t^{x}(S_t, x_t) \right) \right)$

Lookahead approximations

4) **Direct lookahead policies (DLAs)**

» *Deterministic lookahead/rolling horizon proc./model predictive control*

$$X_t^{LA-D}(S_t) = \arg\max_{\tilde{x}_{tt}, \ldots, \tilde{x}_{t,t+H}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}, \tilde{x}_{tt'})$$

» *Chance constrained programming*

$$P[A_t x_t \le f(W)] \le 1 - \delta$$

» *Stochastic lookahead /stochastic prog/Monte Carlo tree search*

$$X_t^{LA-S}(S_t) = \arg\max_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{\tilde{\omega} \in \tilde{\Omega}_t} p(\tilde{\omega}) \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}(\tilde{\omega}), \tilde{x}_{tt'}(\tilde{\omega}))$$

» *"Robust optimization"*

$$X_t^{LA-RO}(S_t) = \arg\max_{\tilde{x}_{tt}, \ldots, \tilde{x}_{t,t+H}} \min_{w \in W_t(\theta)} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}(w), \tilde{x}_{tt'}(w))$$

Hybrid direct lookahead/CFA

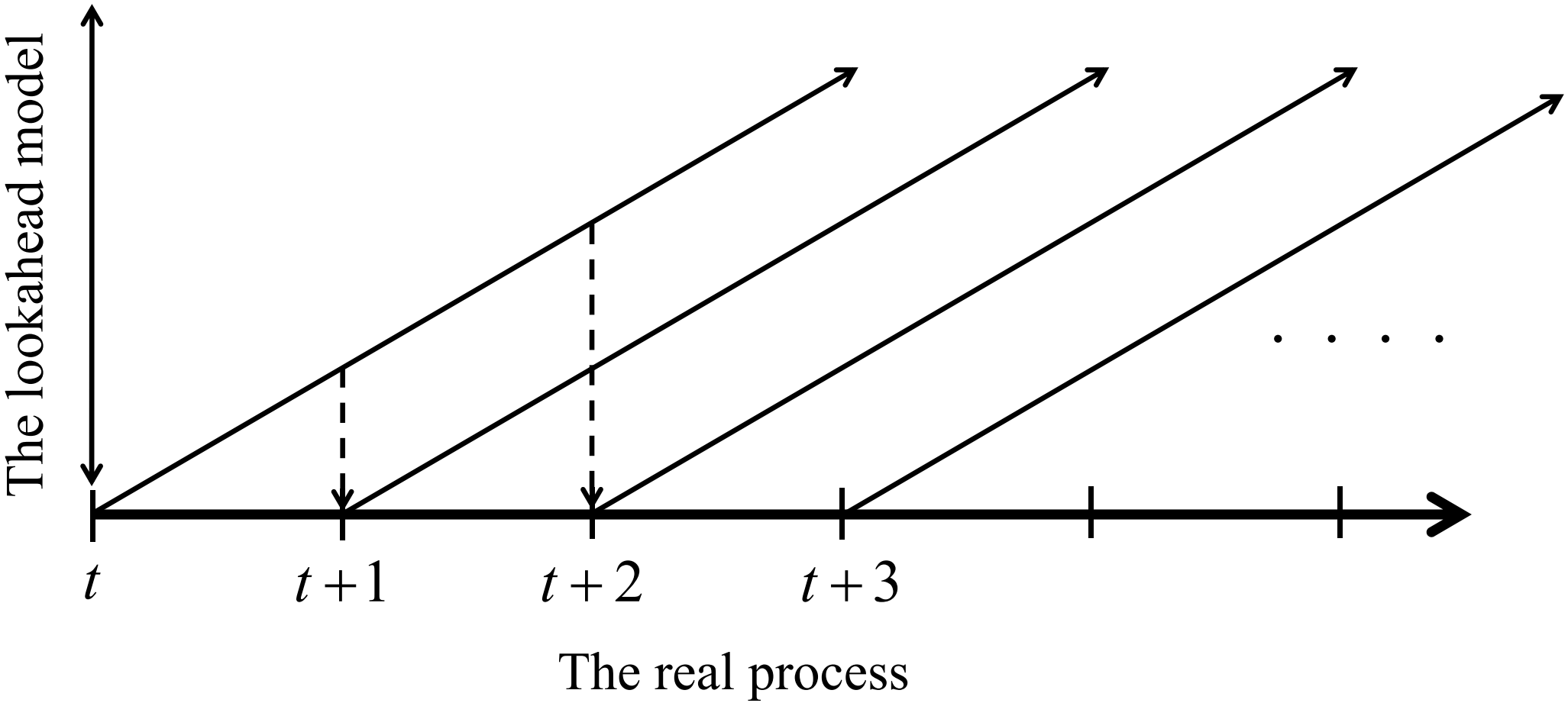
- Benchmark policy – Deterministic lookahead

$$X_t^{\text{D-LA}}(S_t) = \underset{x_t,(\tilde{x}_{tt'},t'=t+1,\ldots,t+H)}{\text{argmin}} \left(C(S_t,x_t) + \left[\sum_{t'=t+1}^{t+H} \tilde{c}_{tt'}\tilde{x}_{tt'} \right] \right)$$

$$\begin{aligned}
\tilde{x}_{tt'}^{wd} + \beta\tilde{x}_{tt'}^{rd} + \tilde{x}_{tt'}^{gd} &\le f_{tt'}^{D} \\
\tilde{x}_{tt'}^{gd} + \tilde{x}_{tt'}^{gr} &\le f_{tt'}^{G} \\
\tilde{x}_{tt'}^{rd} + \tilde{x}_{tt'}^{rg} &\le \tilde{R}_{tt'} \\
\tilde{x}_{tt'}^{wr} + \tilde{x}_{tt'}^{gr} &\le R^{\max} - \tilde{R}_{tt'} \\
\tilde{x}_{tt'}^{wr} + \tilde{x}_{tt'}^{wd} &\le f_{tt'}^{E} \\
\tilde{x}_{tt'}^{wr} + \tilde{x}_{tt'}^{gr} &\le \gamma^{charge} \\
\tilde{x}_{tt'}^{rd} + \tilde{x}_{tt'}^{rg} &\le \gamma^{discharge}
\end{aligned}$$

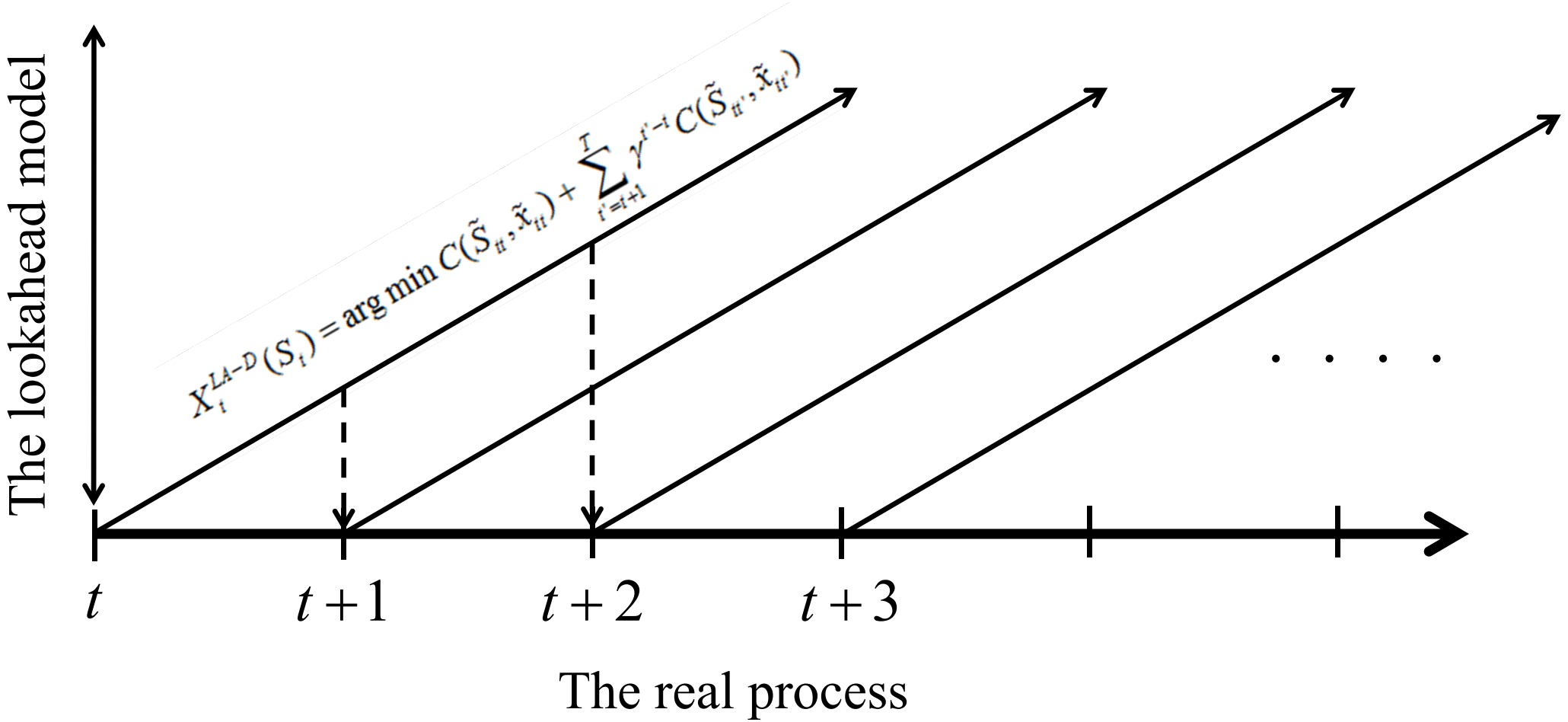
Hybrid direct lookahead/CFA

- Lookahead policies peek into the future
  - » Optimize over deterministic lookahead model



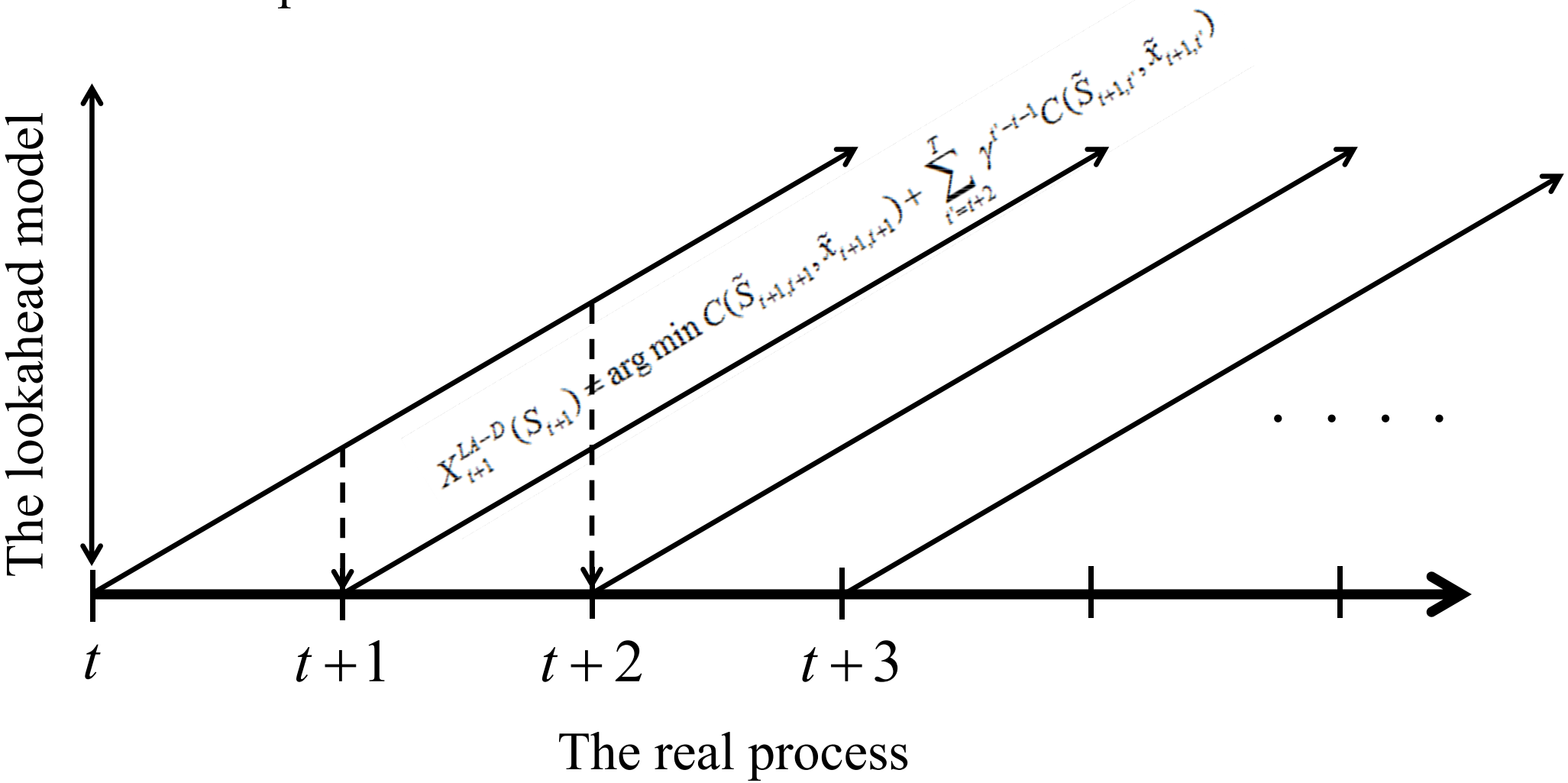
Hybrid direct lookahead/CFA

- Lookahead policies peek into the future
  - » Optimize over deterministic lookahead model



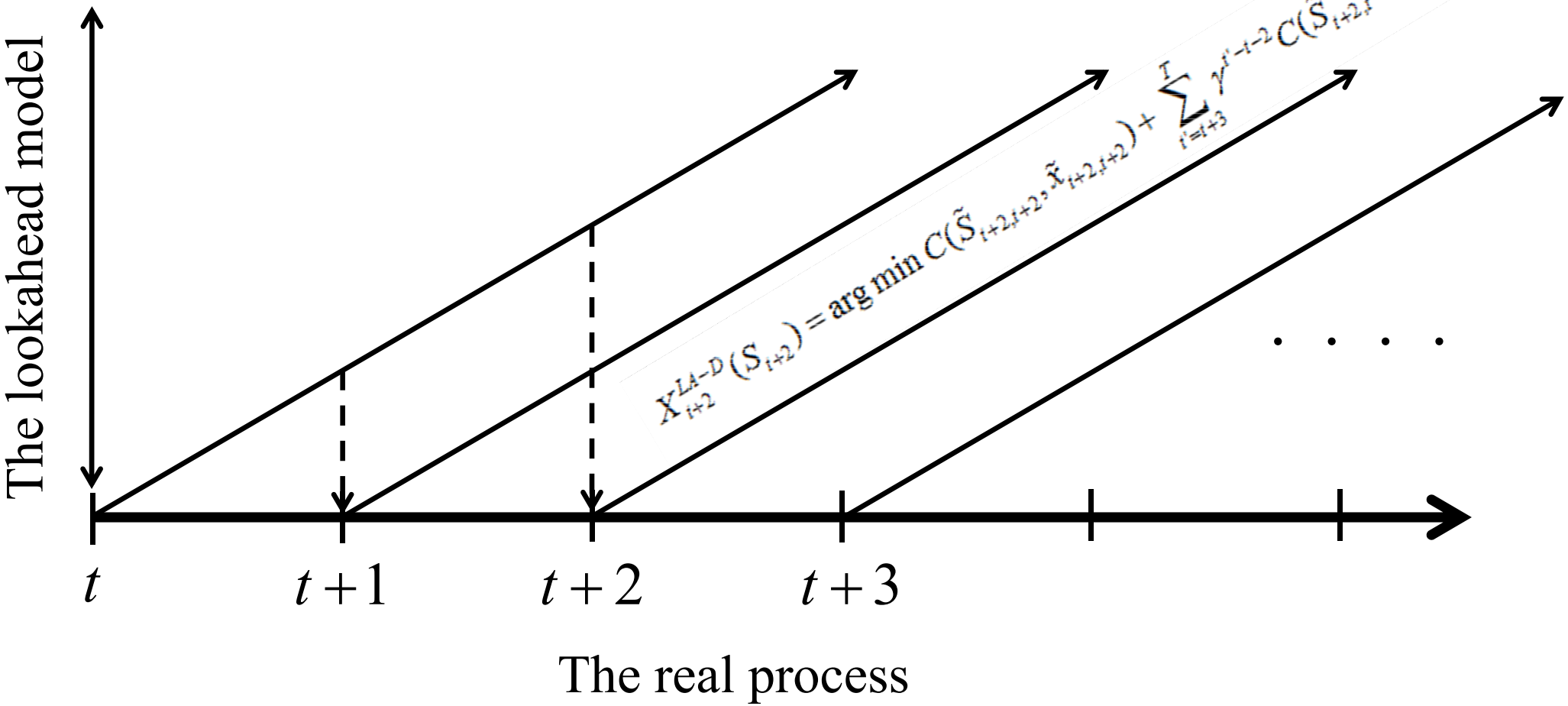
Hybrid direct lookahead/CFA

- Lookahead policies peek into the future
  - » Optimize over deterministic lookahead model



Hybrid direct lookahead/CFA

- Lookahead policies peek into the future
  - » Optimize over deterministic lookahead model



Hybrid direct lookahead/CFA

- Benchmark policy – Deterministic lookahead

$$X^{DLA}(S_t|\theta) = \underset{x_t,(x_{tt'},t'=t+1,\ldots,t+H)}{\text{argmin}} \left(C(S_t,x_t) + \left[\sum_{t'=t+1}^{t+H} \tilde{c}_{tt'}\tilde{x}_{tt'} \right] \right)$$

$$\tilde{x}^{wd}_{tt'} + \beta\tilde{x}^{rd}_{tt'} + \tilde{x}^{gd}_{tt'} \le f^{D}_{tt'}$$

$$\tilde{x}^{gd}_{tt'} + \tilde{x}^{gr}_{tt'} \le f^{G}_{tt'}$$

$$\tilde{x}^{rd}_{tt'} + \tilde{x}^{rg}_{tt'} \le \tilde{R}_{tt'}$$

$$\tilde{x}^{wr}_{tt'} + \tilde{x}^{gr}_{tt'} \le R^{\max} - \tilde{R}_{tt'}$$

$$\tilde{x}^{wr}_{tt'} + \tilde{x}^{wd}_{tt'} \le \theta_{t'-t} f^{E}_{tt'}$$

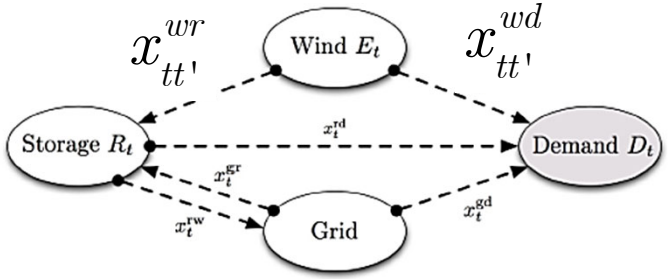
$$\tilde{x}^{wr}_{tt'} + \tilde{x}^{gr}_{tt'} \le \gamma^{charge}$$

$$\tilde{x}^{rd}_{tt'} + \tilde{x}^{rg}_{tt'} \le \gamma^{discharge}$$

Hybrid direct lookahead/CFA

- Parametric cost function approximations
  - » Replace the constraint

$$\tilde{x}^{wr}_{tt'} + \tilde{x}^{wd}_{tt'} \leq f^{E}_{tt'}$$



  with:

  - » Lookup table modified forecasts (one adjustment term for each time $\tau = t' - t$ in the future):

$$x^{wr}_{tt'} + x^{wd}_{tt'} \leq \theta_{t'-t} f^{E}_{tt'}$$

  - » We can simulate the performance of a parameterized policy

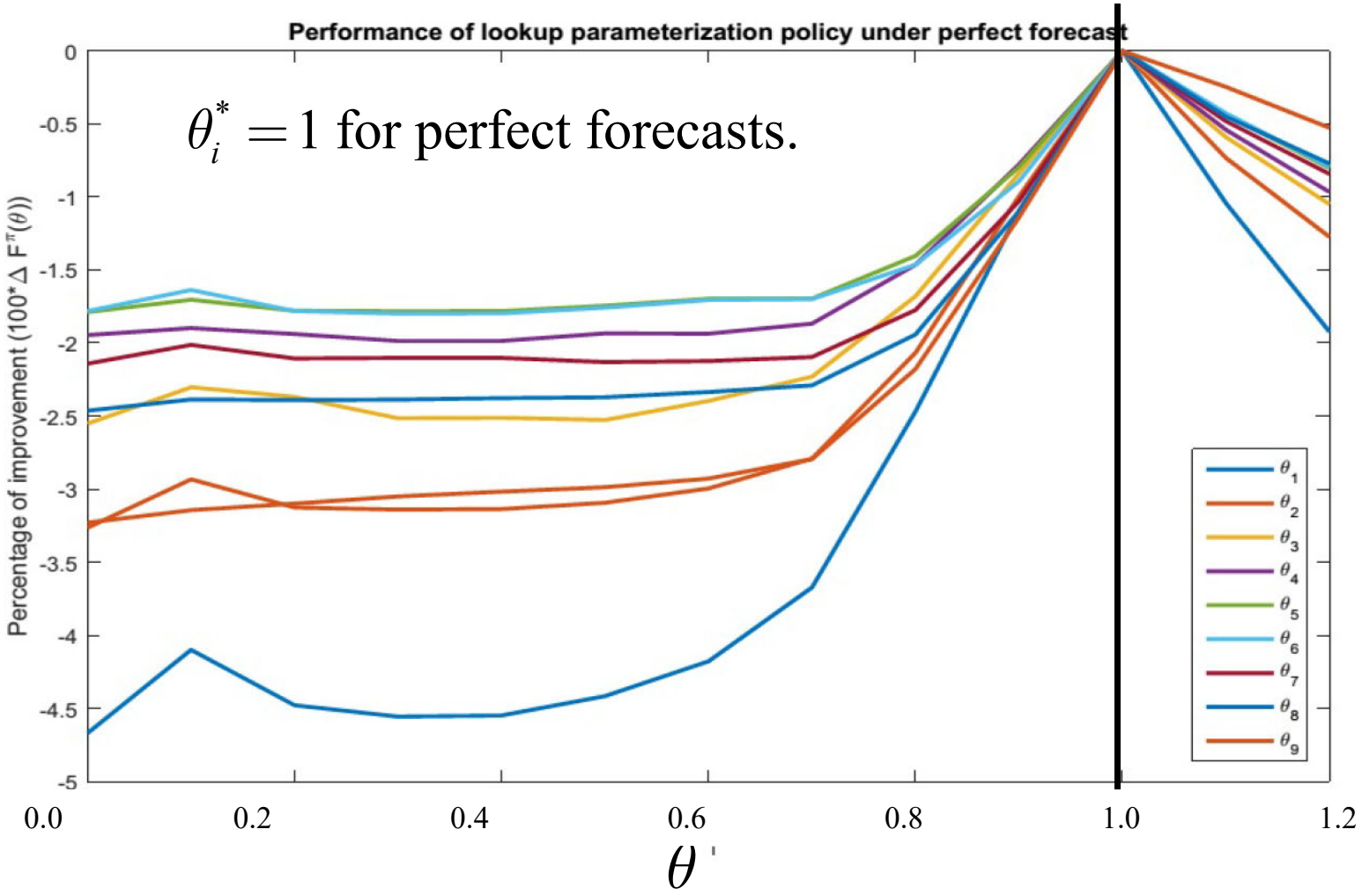
$$\bar{F}(\theta, \omega) = \sum_{t=0}^{T} C\left(S_t(\omega), X^{\pi}_t(S_t(\omega) \mid \theta)\right)$$

  - » The challenge is to optimize the parameters:

$$\min_{\theta} \mathbb{E} \sum_{t=0}^{T} C\left(S_t, X^{\pi}_t(S_t \mid \theta)\right)$$

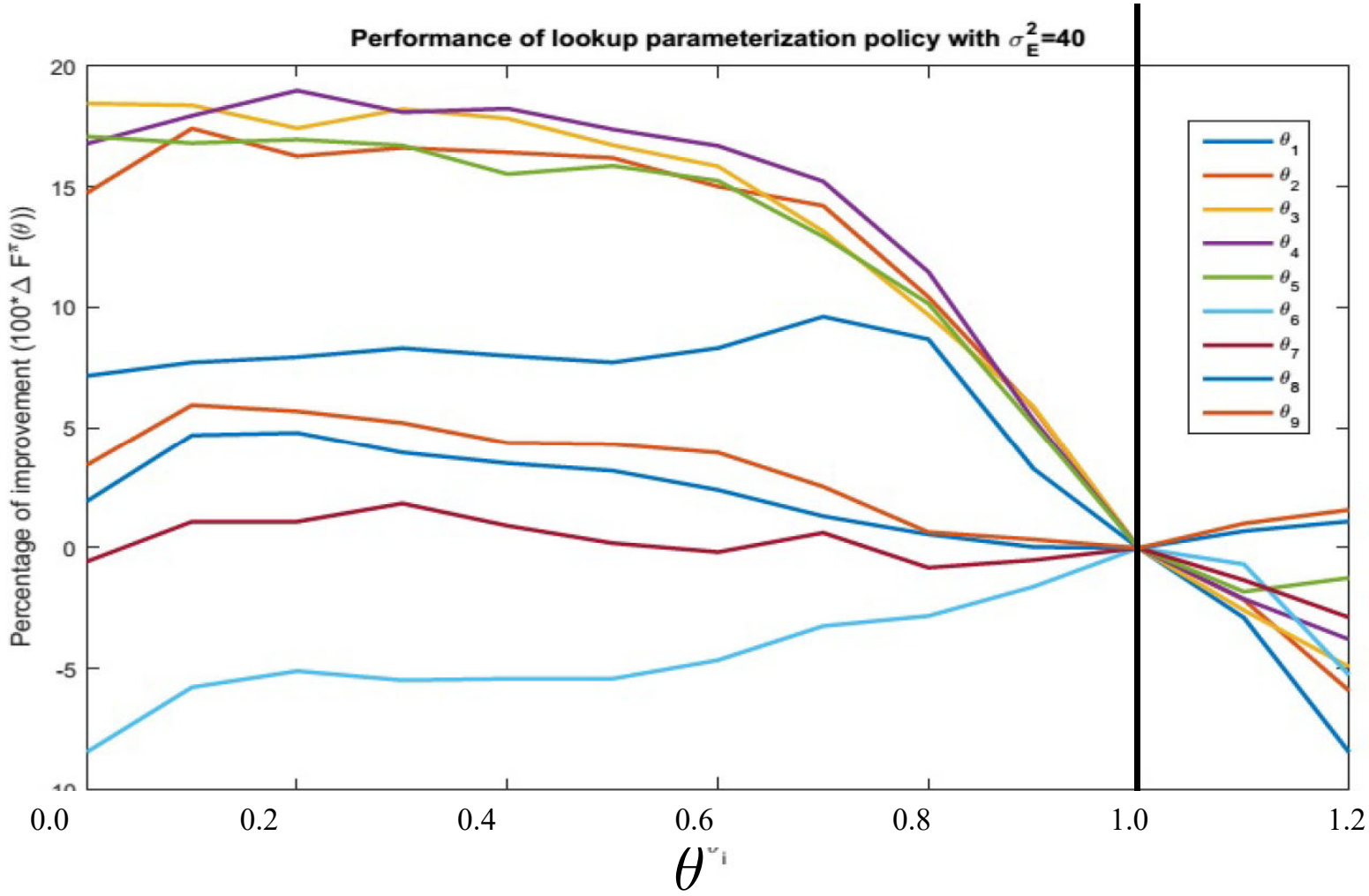
Hybrid direct lookahead/CFA

- One-dimensional contour plots – perfect forecast



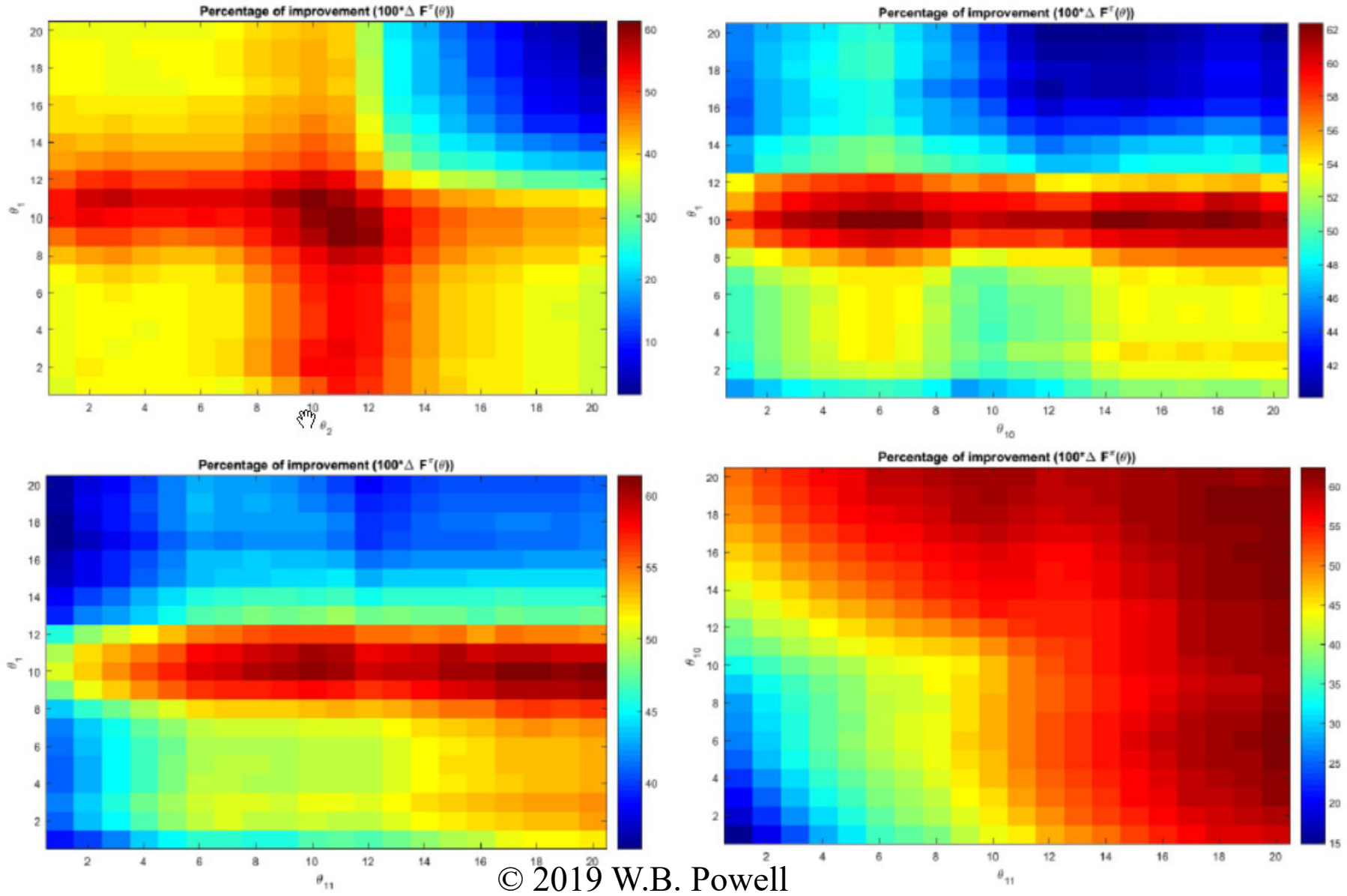
Hybrid direct lookahead/CFA

- One-dimensional contour plots-uncertain forecast



Hybrid direct lookahead/CFA

- 2-D contours for uncertain forecasts



Hybrid direct lookahead/CFA

- Simultaneous perturbation stochastic approximation
  - » Let:
    - x^n be a p – dimensional vector.
    - δ^n be a scalar perturbation
    - Z^n be a p –dimensional vector, with each element drawn from a normal (0,1) distribution.
  - » We can obtain a sampled estimate of the gradient $\nabla_x F(x^n, W^{n+1})$ using two function evaluations: $F(x^n + \delta^n Z^n)$ and $F(x^n - \delta^n Z^n)$

$$\nabla_x F(x^n, W^{n+1}) = \begin{bmatrix} \dfrac{F(x^n + \delta^n Z^n) - F(x^n - \delta^n Z^n)}{2\delta^n Z_1^n} \\ \dfrac{F(x^n + \delta^n Z^n) - F(x^n - \delta^n Z^n)}{2\delta^n Z_2^n} \\ \vdots \\ \dfrac{F(x^n + \delta^n Z^n) - F(x^n - \delta^n Z^n)}{2\delta^n Z_p^n} \end{bmatrix}$$

Hybrid direct lookahead/CFA

- Algorithmic issues:
  - » Stepsize rules
    - We use a standard gradient-based method:

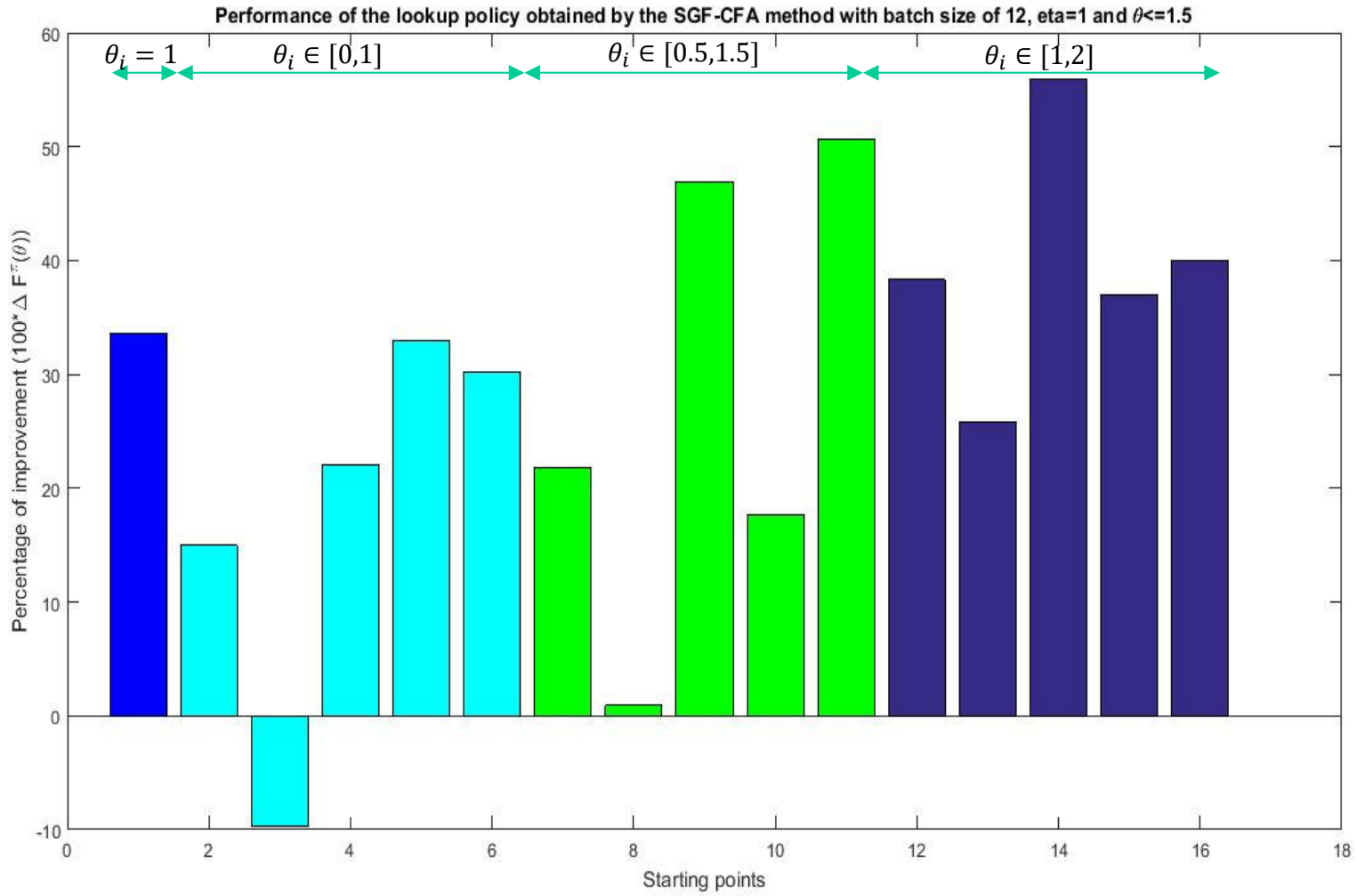
$$x^{n+1} = x^n + \alpha \nabla_x F\left(x^n, W^{n+1}\right)$$

    - We face the usual challenge of choosing stepsize rules (we used RMSProp), and tuning the parameter(s) in the stepsize rule (this is an optimization problem within the optimization problem).
  - » We use "mini-batches" to reduce the noise when evaluating the function in SPSA.
  - » We also used gradient smoothing:

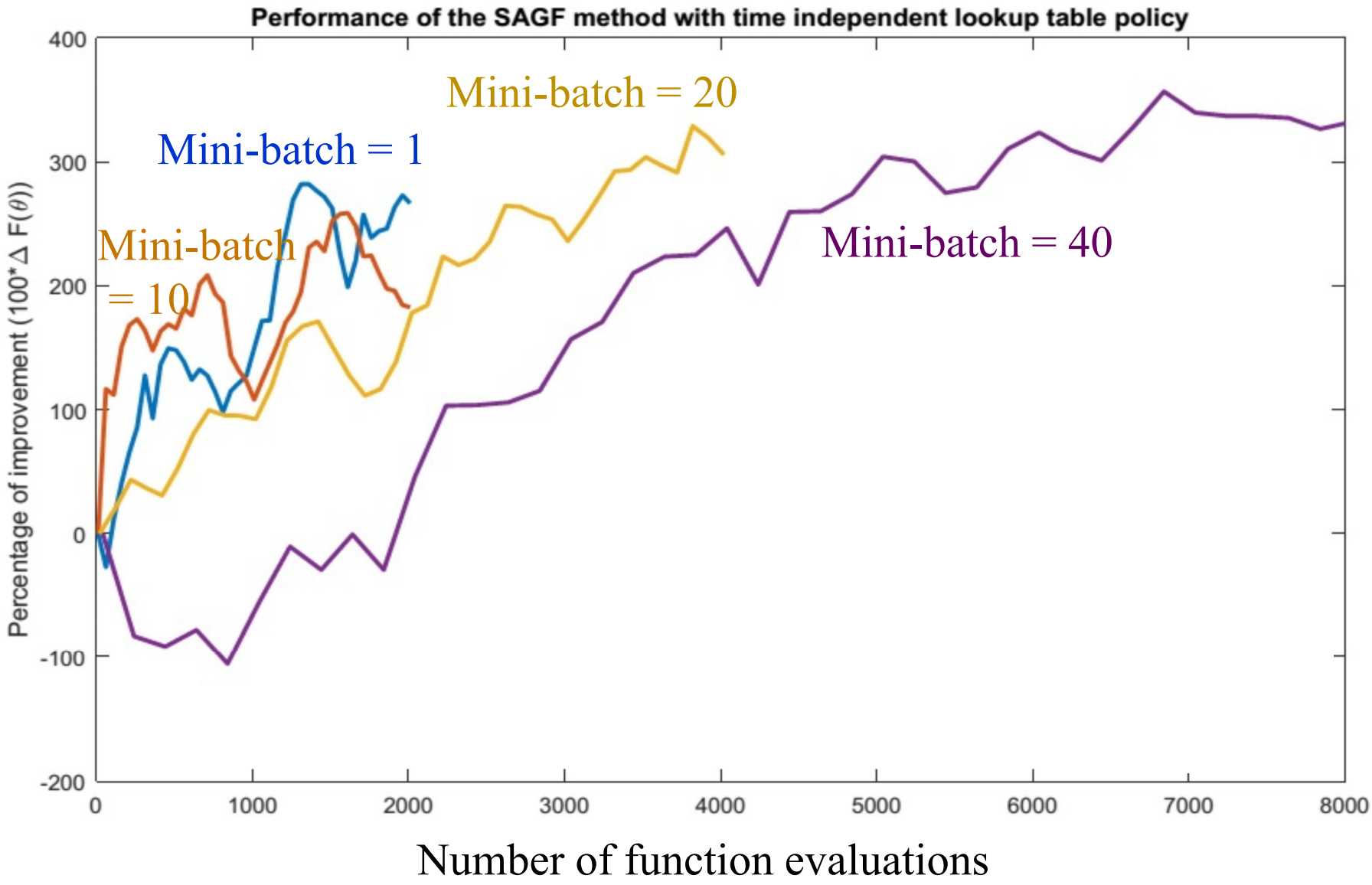
$$\bar{g}^{n+1} = (1-\eta)\bar{g}^n + \eta \nabla_x F(x^n, W^{n+1})$$
$$x^{n+1} = x^n + \alpha_n \bar{g}^n$$

Hybrid direct lookahead/CFA

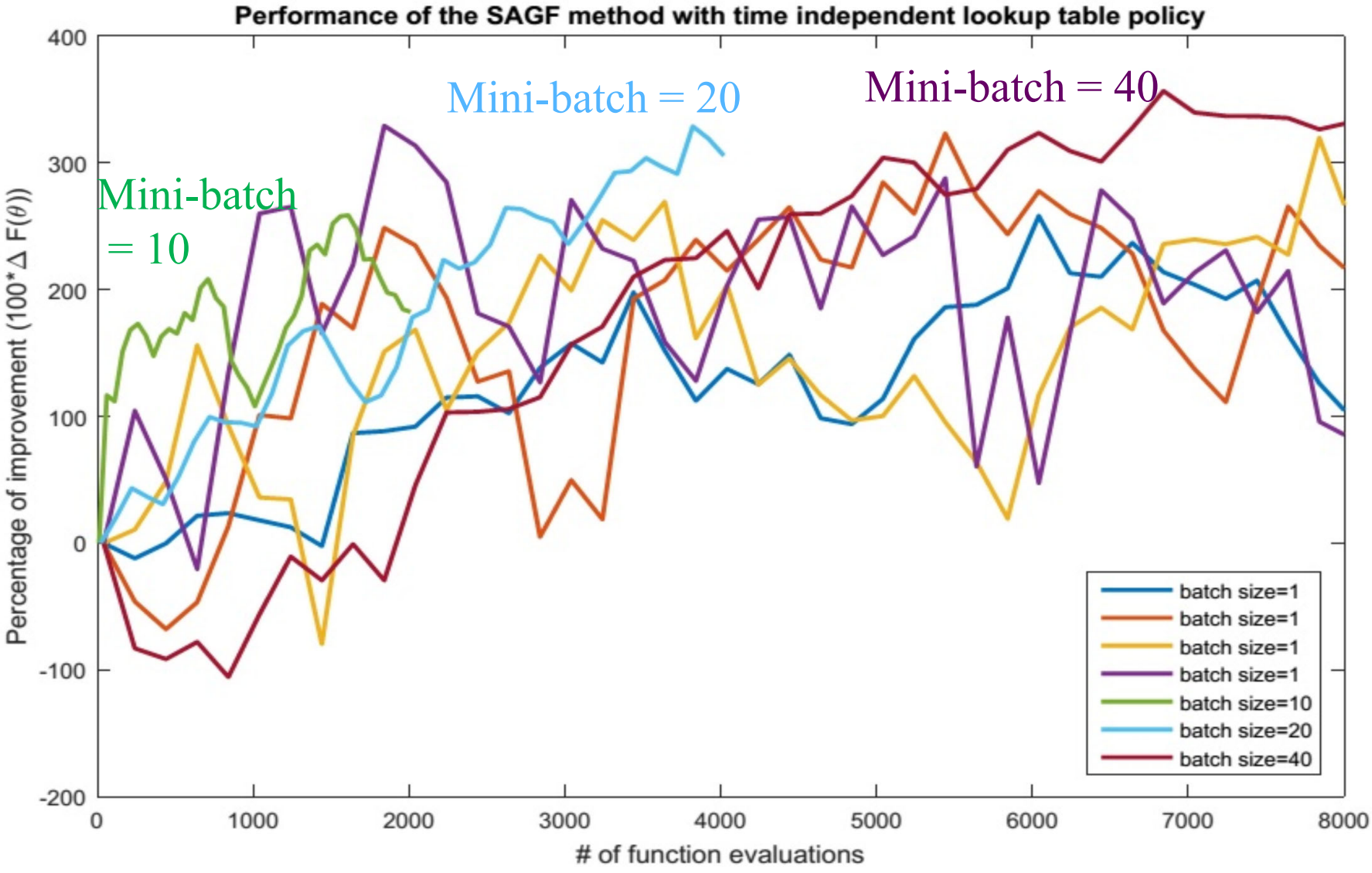
- Tuning the parameters
  - » Stepsizes tuned for region [0,2].



Hybrid direct lookahead/CFA



Hybrid direct lookahead/CFA



Outline

- The four classes of policies
  - » Policy function approximations (PFAs)
  - » Cost function approximations (CFAs)
  - » Value function approximations (VFAs)
  - » Direct lookahead policies (DLAs)
  - » Hybrid direct lookahead/CFA
  - » Any of the four classes may work best!

An energy storage problem

- Consider a basic energy storage problem:

» We are going to show that with minor variations in the characteristics of this problem, we can make *each* class of policy work best.

An energy storage problem

- We can create distinct flavors of this problem:
  - » Problem class 1 – Best for PFAs
    - Highly stochastic (heavy tailed) electricity prices
    - Stationary data
  - » Problem class 2 – Best for CFAs
    - Stochastic prices and wind (but not heavy tailed)
    - Stationary data
  - » Problem class 3 - Best for VFAs
    - Stochastic wind and prices (but not too random)
    - Time varying loads, but inaccurate wind forecasts
  - » Problem class 4 – Best for deterministic lookaheads
    - Relatively low noise problem with accurate forecasts
  - » Problem class 5 – A hybrid policy worked best here
    - Stochastic prices and wind, nonstationary data, noisy forecasts.

An energy storage problem

- The policies
  - » The PFA:
    - Charge battery when price is below p1
    - Discharge when price is above p2
  - » The CFA
    - Optimize over a horizon H; maintain upper and lower bounds (u, l) for every time period except the first (note that this is a hybrid with a lookahead).
  - » The VFA
    - Piecewise linear, concave value function in terms of energy, indexed by time.
  - » The lookahead (deterministic)
    - Optimize over a horizon H (only tunable parameter) using forecasts of demand, prices and wind energy
  - » The lookahead CFA
    - Use a lookahead policy (deterministic), but with a tunable parameter that improves robustness.

An energy storage problem

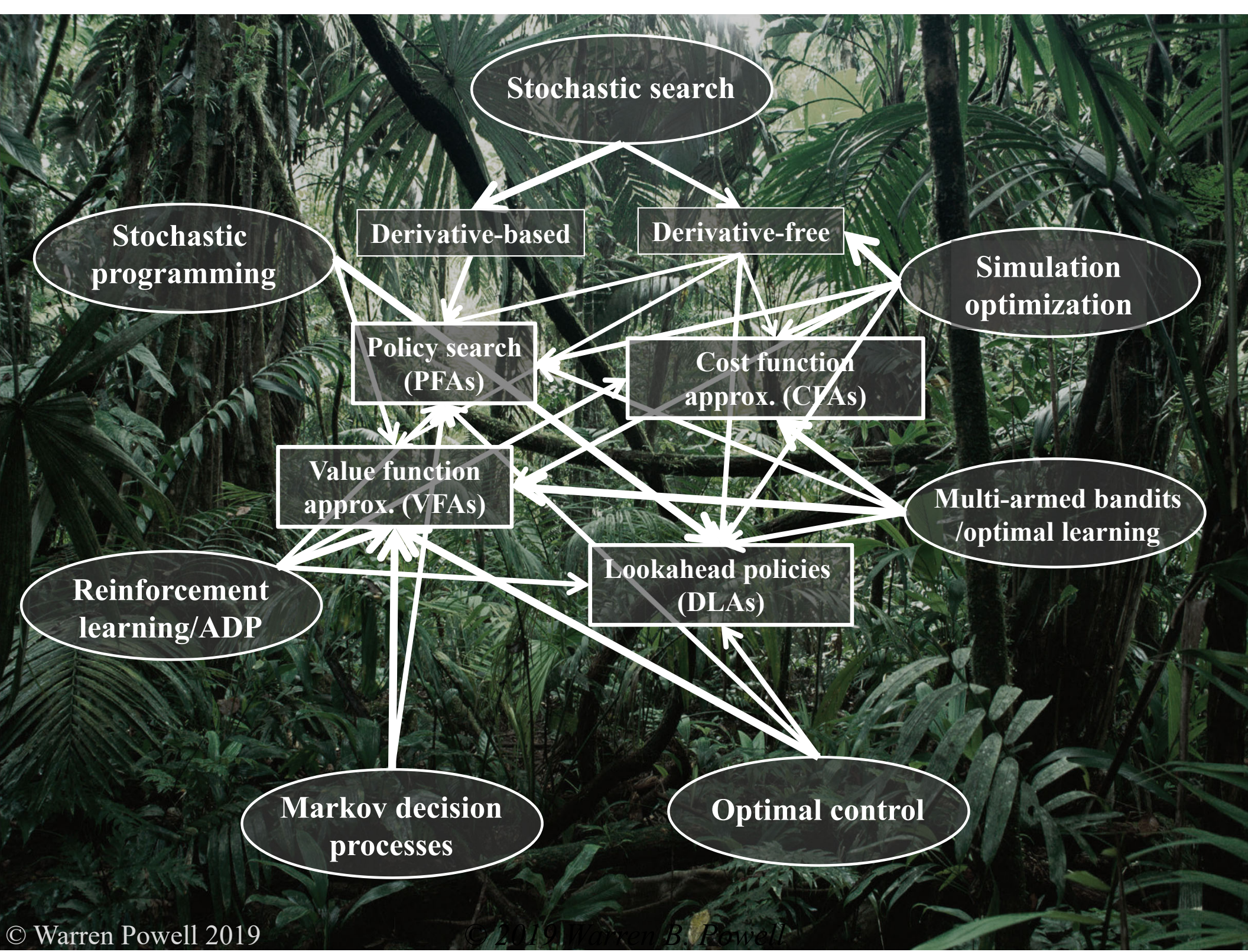
- Each policy is best on certain problems
  - » Results are percent of *posterior* optimal solution

Problem:	Problem description	PFA	CFA Error correction	VFA	Determ. Lookahead	CFA Lookahead
A	A stationary problem with heavy-tailed prices, relatively low noise, moderately accurate forecasts.	**0.959**	**0.839**	**0.936**	**0.887**	**0.887**
B	A time-dependent problem with daily load patterns, no seasonalities in energy and price, relatively low noise, less accurate forecasts.	**0.714**	**0.752**	**0.712**	**0.746**	**0.746**
C	A time-dependent problem with daily load, energy and price patterns, relatively high noise, forecast errors increase over horizon.	**0.865**	**0.590**	**0.914**	**0.886**	**0.886**
D	A time-dependent problem, relatively low noise, very accurate forecasts.	**0.962**	**0.749**	**0.971**	**0.997**	**0.997**
E	Same as (C), but the forecast errors are stationary over the planning horizon.	**0.865**	**0.590**	**0.914**	**0.922**	**0.934**

  - » … any policy might be best depending on the data.

Joint research with Prof. Stephan Meisel, University of Muenster, Germany.


John R. Birge
François Louveaux
Introduction to Stochastic Programming
Second Edition
Michael C. Fu Editor
Handbook of Simulation Optimization
Princeton Series in APPLIED MATHEMATICS
Robust Optimization
Introduction to Decision Analysis
A Practitioner's Guide to Improving Decision Quality
SECOND EDITION
Approximate Dynamic Programming
Solving the Curses of Dimensionality
Warren B. Powell
Optimal Learning
MULTI-ARMED BANDIT ALLOCATION INDICES
SECOND EDITION
John Gittins, Kevin
and Richard W
VOLUME 2 • 4th EDITION
Dynamic Programming and Optimal Control
APPROXIMATE DYNAMIC PROGRAMMING
Dimitri P. Bertsekas
WILEY
SECOND EDITION
Model Predictive Control
OPTIMAL CONTROL
Frank L. Lewis
MORGAN&CLAYPOOL PUBLISHERS
Active Learning
Burr Settles
INTRODUCTION TO STOCHASTIC SEARCH AND OPTIMIZATION
Estimation, Simulation, and Control
JAMES C. SPALL
Athena Scientific
WILEY
Markov Decision Processes
Discrete Stochastic Dynamic Programming
MARTIN L. PUTERMAN
Online Computation and Competitive Analysis
Allan Borodin
Ran El-Yaniv
43
Jiongmin Yong
Xun Yu Zhou
Stochastic Controls
Hamiltonian Systems and HJB Equations
Reinforcement Learning
An Introduction
second edition
Richard S. Sutton and Andrew G. Barto
STOCHASTIC SIMULATION OPTIMIZATION
An Optimal Computing Budget Allocation
Chun-Hung Chen • Loo Hay Lee



Outline

- Elements of a dynamic model
- An energy storage illustration
- Modeling uncertainty
- Designing policies
- Educational materials

Educational materials

- More information is available at
  - » Jungle.princeton.edu
- Scroll down to “Educational materials” to find:
  - » A five page book chapter (very quick read)
  - » A 25 page article in EJOR
  - » Material for undergraduate course:
    - *Sequential decision analytics and modeling*
  - » Material for graduate level course
    - *Reinforcement Learning and Stochastic Optimization*

Educational materials

- Undergraduate course: *Sequential Decision Analytics and Modeling.*
  - » There is an undergraduate level book that can be accessed at jungle.Princeton.edu or directly from:
    - http://tinyurl.com/sequentialdecisionanalytics
  - » This uses a teach-by-example style.
  - » Prerequisite is an undergraduate course in probability and statistics (one chapter uses linear programming, but can be read without a course in linear programming)
  - » There is an undergraduate course with complete lecture notes in powerpoint at
    - http://www.castlelab.princeton.edu/orf-411/

Can be accessed at ***jungle.princeton.edu***

SEQUENTIAL DECISION ANALYTICS AND MODELING:

Modeling exercises with python

Warren B. Powell

This book can be accessed by going to

https://tinyurl.com/sequentialdecisionanalytics

If you would like to write your own chapter, please go to

https://tinyurl.com/sequentialdecisionanalyticspub

May 27, 2019

Sequential decision analytics

- In addition:

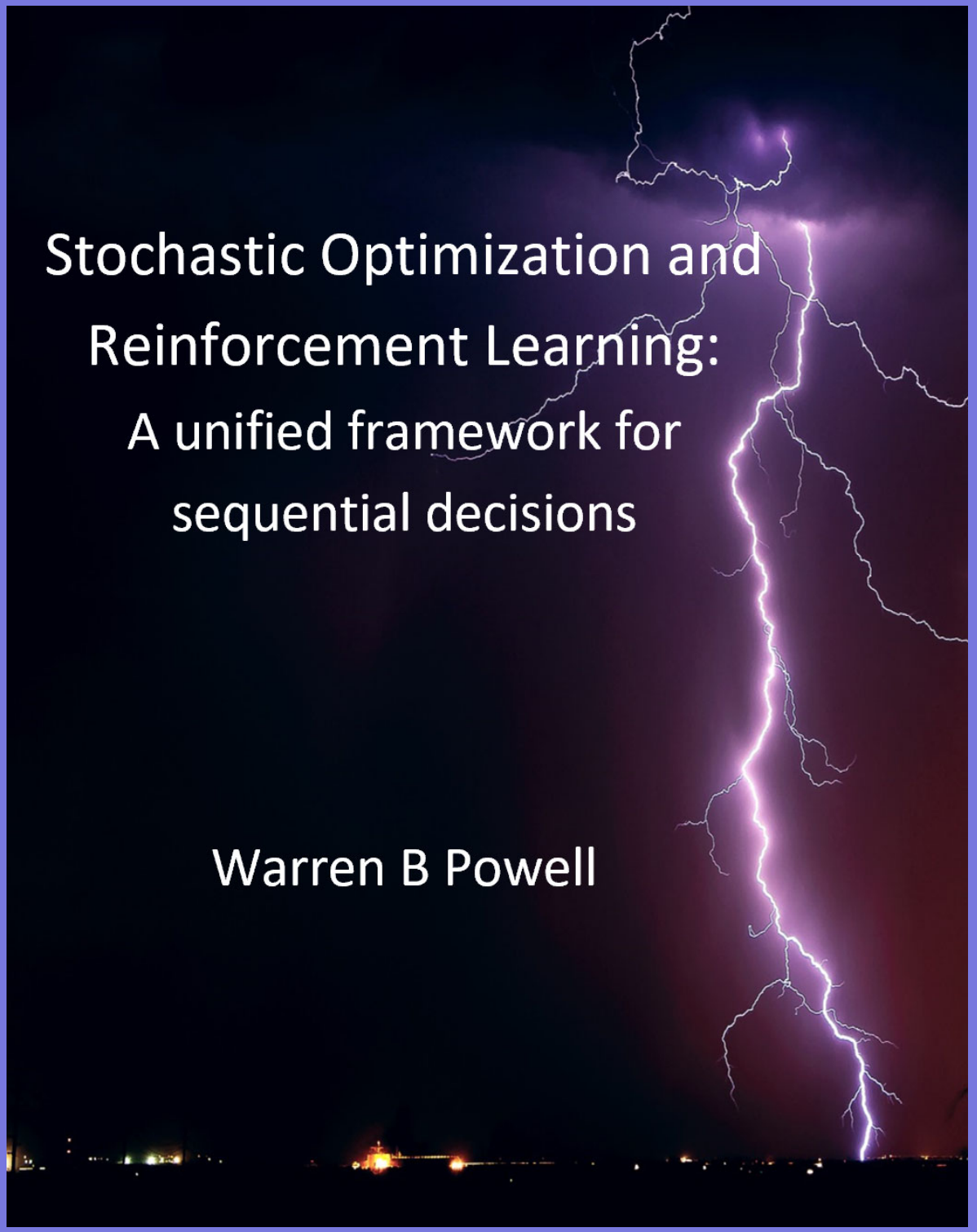
  » There is a link to a python library of exercises on github. There are 10 modules for 10 different chapters.

  » Each chapter follows a specific modeling style. You may write your own chapters at
  - http://tinyurl.com/sequentialdecisionanalyticspub

  All material is available at ***jungle.Princeton.edu***

Sequential decision analytics

- There is also a 700+ page graduate level book written entirely around the unified framework.
  - » The book can be downloaded from *jungle.princeton.edu*
  - » The prerequisite is also an undergraduate course in probability and statistics, but the presentation is much more methods driven.
  - » This is a work in progress. I hope to finish the book by fall, 2020.


Stochastic Optimization and
Reinforcement Learning:
A unified framework for
sequential decisions
Warren B Powell

Thank you!

More material is available at:

jungle.princeton.edu

Please be sure you are on the signup list so I can send you followup information (link is at the top of jungle.Princeton.edu).

Look under "Educational materials"

This presentation will be posted at jungle.Princeton.edu soon.