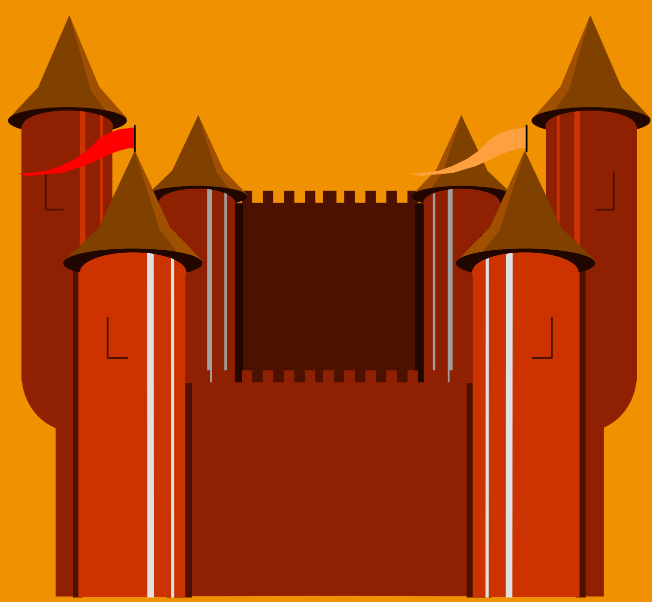
From Multiarmed Bandits to Stochastic Optimization

Multiarmed Bandits Workshop
Rotterdam, NL

May 24, 2018



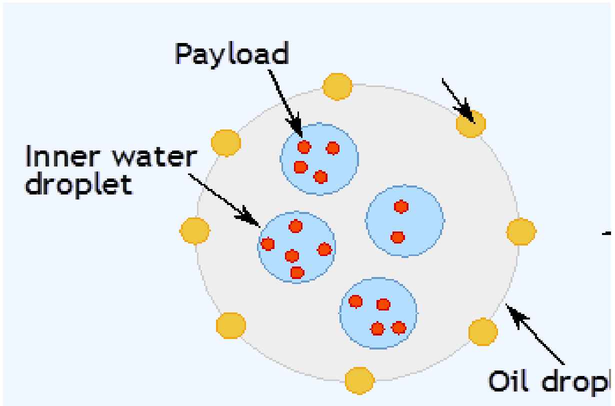
Warren B. Powell

Princeton University
Department of Operations Research and Financial Engineering


$1.036.03
5000
triple
DOUBLE

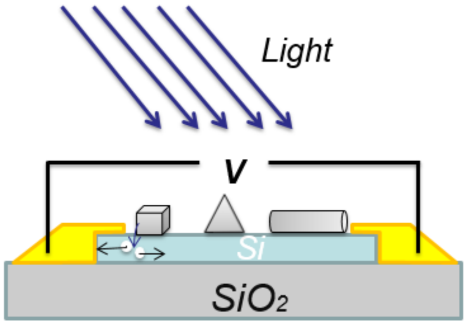
Materials science

» Optimizing payloads: reactive species, biomolecules, fluorescent markers, …



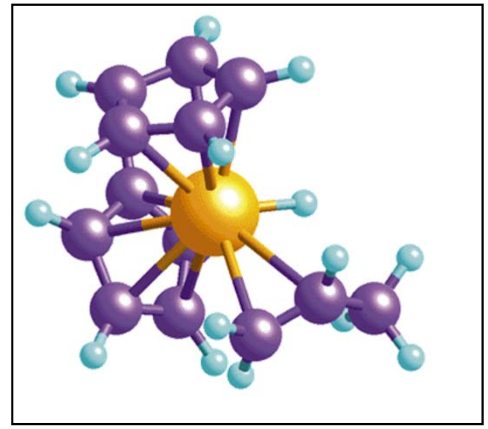
» Controllers for robotic scientist for materials science experiments

» Optimizing nanoparticles to maximize photoconductivity

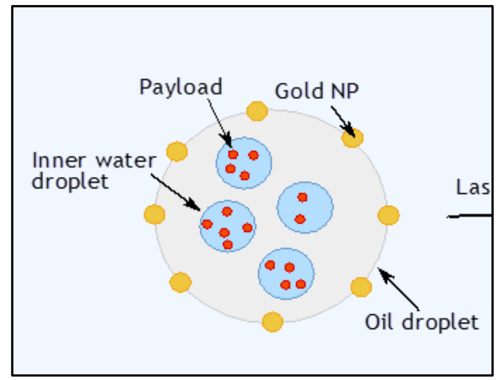


Learning problems

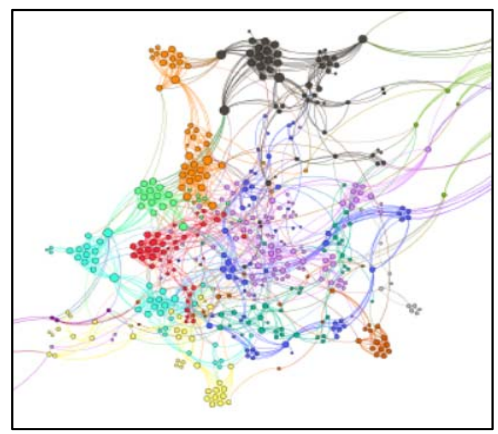
- Health sciences
  - » Sequential design of experiments for drug discovery



  - » Drug delivery – Optimizing the design of protective membranes to control drug release

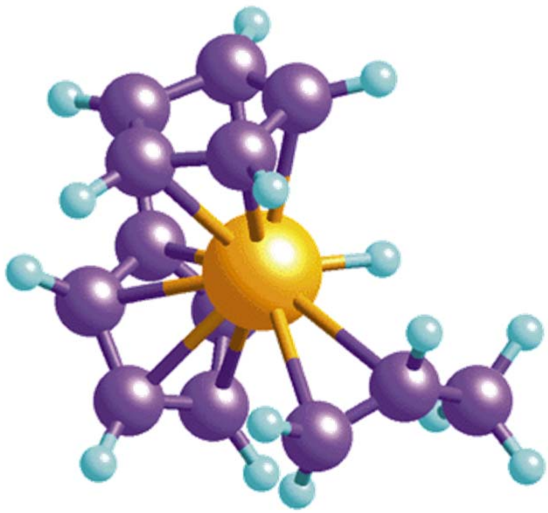


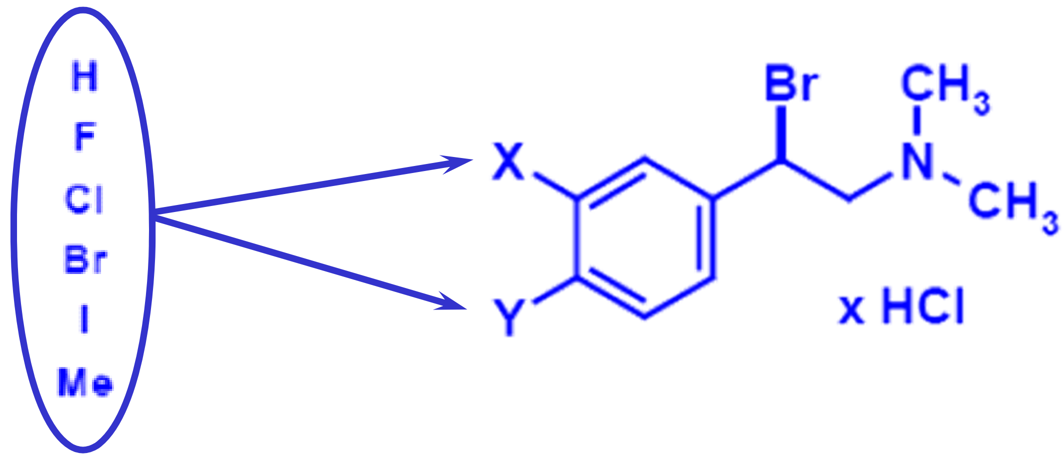
  - » Medical decision making – Optimal learning for medical treatments.



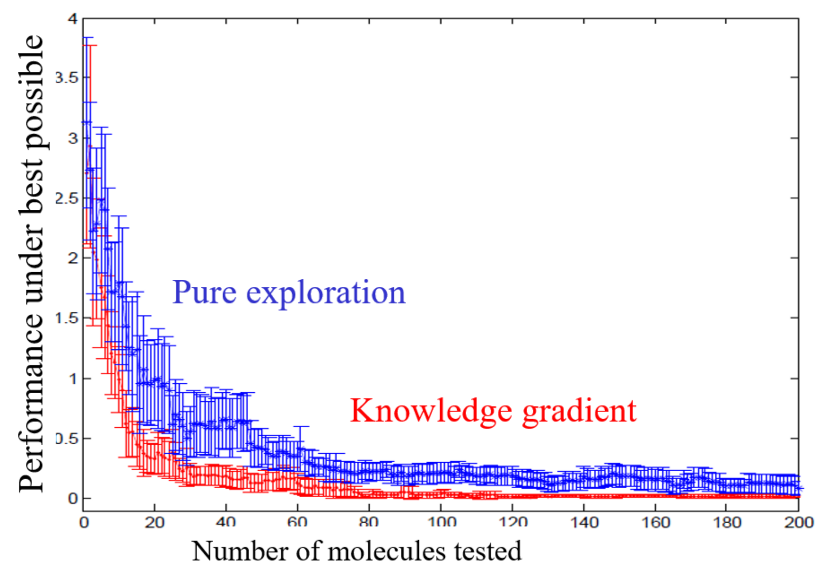
Drug discovery

- Optimizing the configuration of molecules





Design of effective policies can accelerate the search process for new drugs.



Optimal learning in diabetes

- How do we find the best treatment for diabetes?
  - » The standard treatment is a medication called metformin, which works for about 70 percent of patients.
  - » What do we do when metformin does not work for a patient?
  - » There are about 20 other treatments, and it is a process of trial and error. Doctors need to get through this process as quickly as possible.

OPTIMAL DOSING APPLIED TO
GLYCEMIC CONTROL FOR TYPE 2 DIABETES

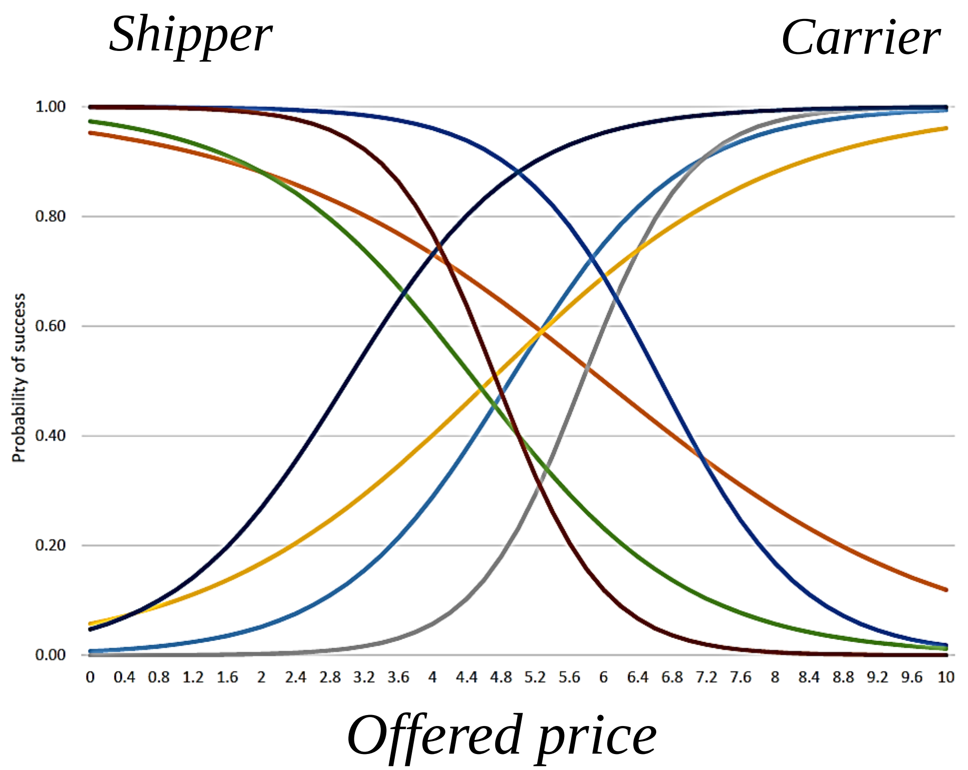
KATIE W. HSIH
ADVISOR: WARREN B. POWELL

Truckload brokerages

- Now we have a logistic curve for each origin-destination pair (i,j)

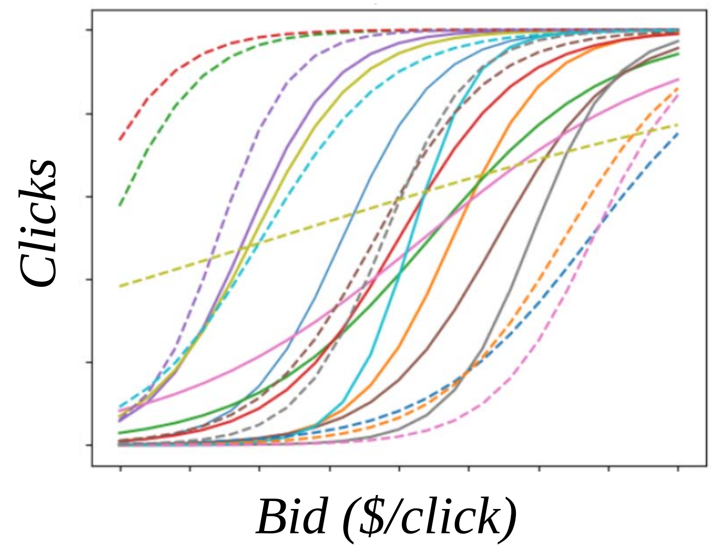
$$P^Y(p,a\,|\,\theta) = \frac{e^{\theta_{ij}^0 + \theta_{ij} p + \theta_{ij}^a a}}{1 + e^{\theta_{ij}^0 + \theta_{ij} p + \theta_{ij}^a a}}$$

- Number of offers for each (i,j) pair is relatively small.
- Need to generalize the learning across "traffic lanes."
- Slides that follow are from senior thesis of Connor Werth '2017



Ad-click optimization

- Optimizing bids for internet ads
  - » In partnership with Roomsage.com
  - » Developed Princeton ad-click game
  - » Teams compete to find best policy



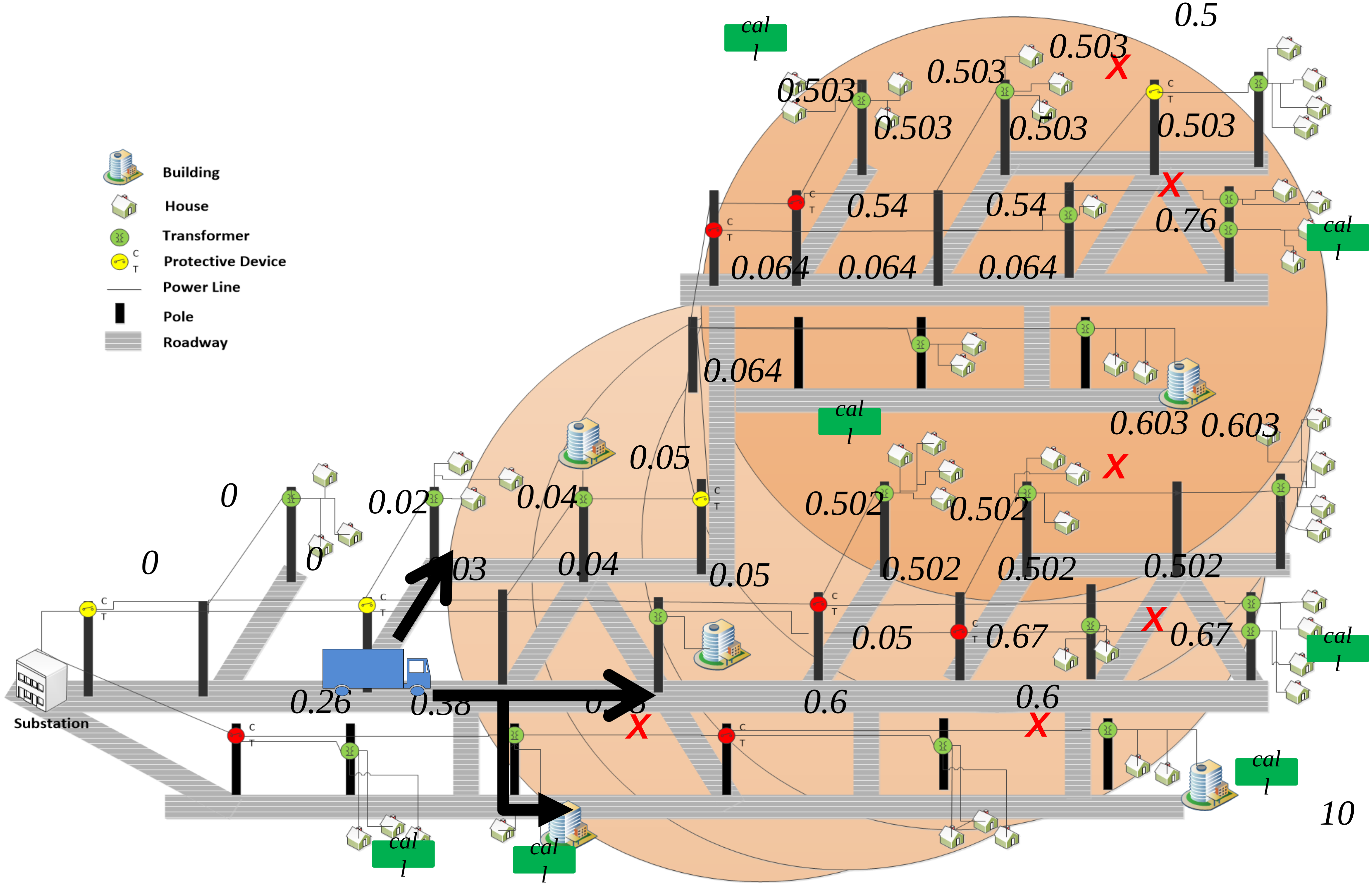
Policy	profit
PresidentBidness_LA_1	10528
MaxBidder_LAPS_alpha	8439
PresidentBidness_PS_1	5553
Weebs_LA_EZPolicy	3458
MaxBidder_PS_alpha	2573
Weebs_LA_MetropolisHastings	1740
AKCB_LA_1	1471
pbchen_PS_s4real	790
BaoWang_PS_WeGo2	599
MnM_LAPS_M	219
MmegwaWagnerinterval_estimation	61
AKCB_PS_1	0
ohiustina_LA_3	0

Emergency storm response

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

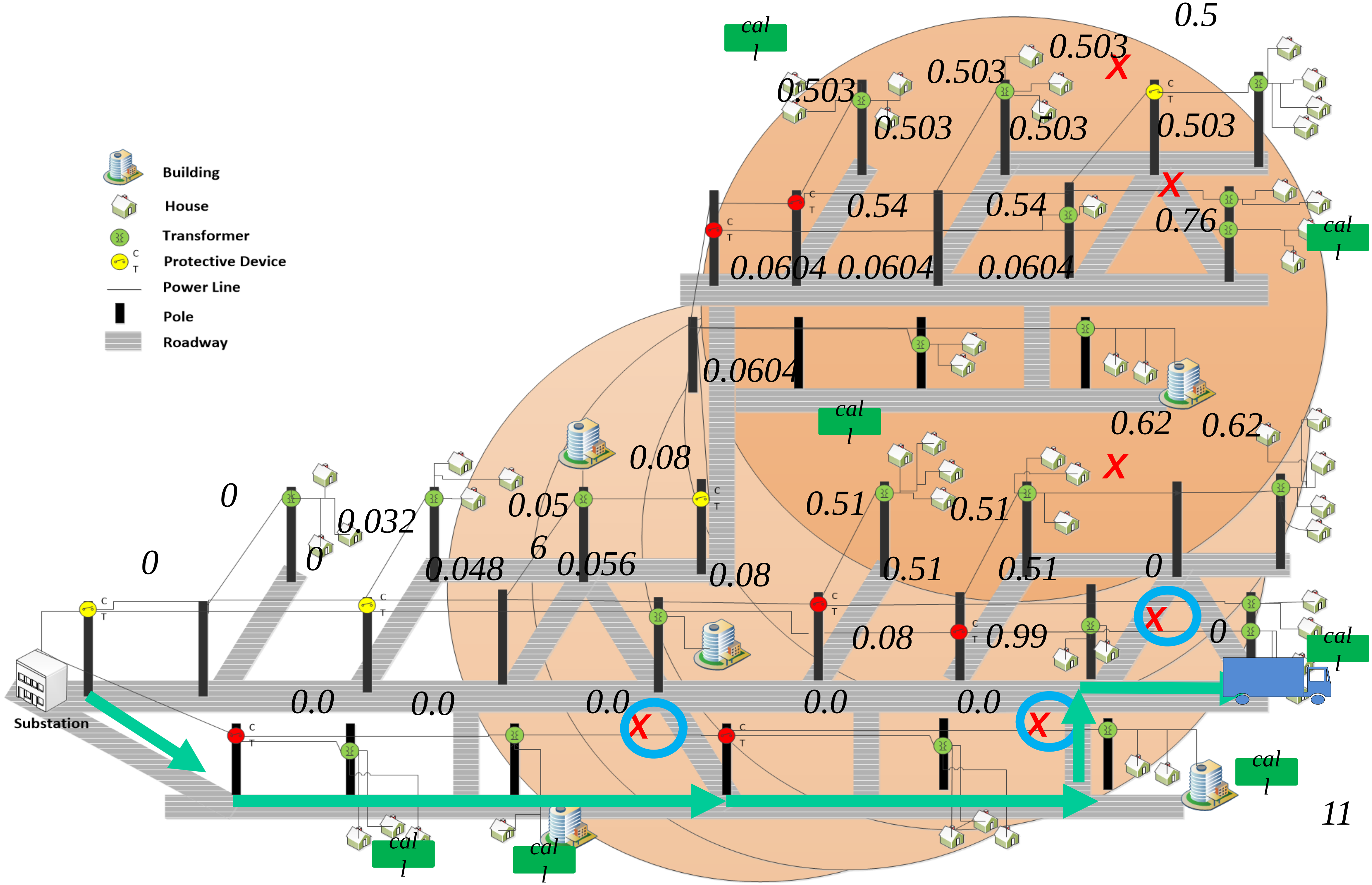
- The power grid
  » Loss of power creates cascading failures (lack of fuel, inability to pump water)
  » How to plan?
  » How to react?

Emergency storm response

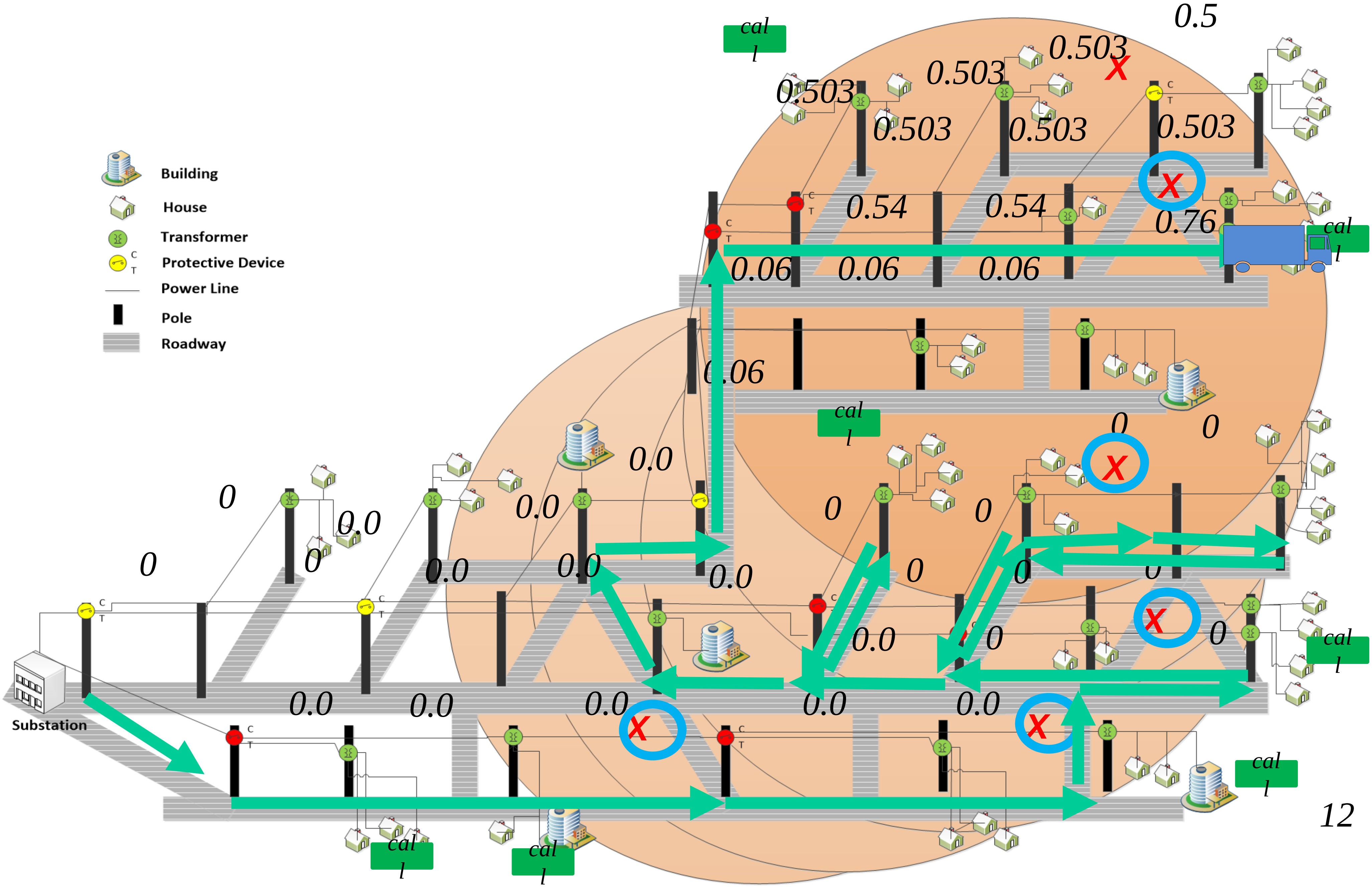


10

Emergency storm response



Emergency storm response



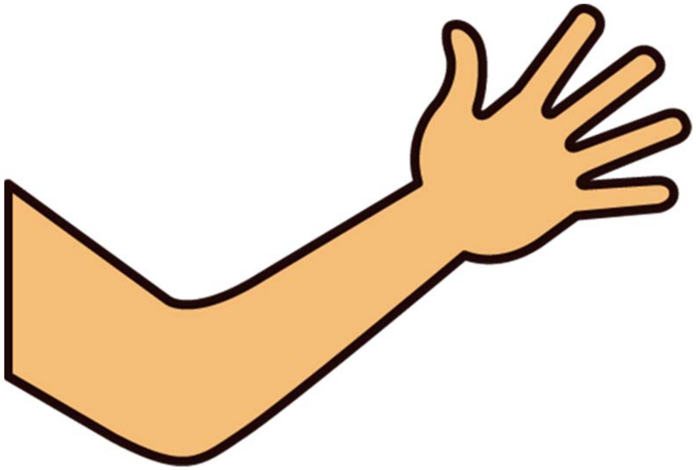
The “bandit” vocabulary

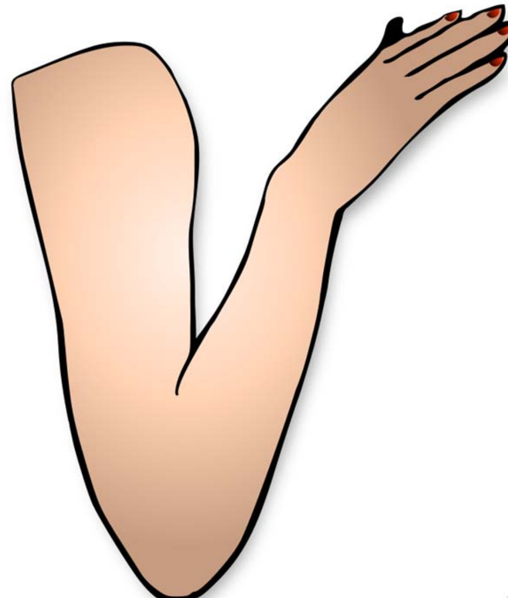
Bandit problem	Description
Multiarmed bandits	Basic problem with discrete alternatives, online (cumulative regret) learning, lookup table belief model with independent beliefs
Restless bandits	Truth evolves exogenously over time
Adversarial bandits	Distributions from which rewards are being sampled can be set by arbitrarily by an adversary
Continuum-armed bandits	Arms are continuous
X-armed bandits	Arms are a general topological space
Contextual bandits	Exogenous state is revealed which affects the distribution of rewards
Dueling bandits	The agent gets a relative feedback of the arms as opposed to absolute feedback
Arm-acquiring bandits	New machines arrive over time
Intermittent bandits	Arms are not always available
Response surface bandits	Belief model is a response surface (typically a linear model)

The "bandit" vocabulary

Bandit problem	Description
Linear bandits	Belief is a linear model
Dependent bandits	A form of correlated beliefs
Finite horizon bandits	Finite-horizon form of the classical infinite horizon multi-armed bandit problem
Parametric bandits	Beliefs about arms are described by a parametric belief model
Nonparametric bandits	Bandits with nonparametric belief models
Graph-structured bandits	Feedback from neighbors on graph instead of single arm
Extreme bandits	Optimize the maximum of recieved rewards
Quantile-based bandits	The arms are evaluated in terms of a specified quantile
Preference-based bandits	Find the correct ordering of arms
Best-arm bandits	Identify the optimal arm with the largest confidence given a fixed budget

Arms…





… and bandits

Multiarmed bandit problems

- What is a "bandit problem"?
  - » The literature seems to characterize a "bandit problem" as any problem where a policy has to balance exploration vs. exploitation.
  - » But this means that a bandit "problem" is defined by how it is solved. E.g., if you use a pure exploration policy, is it a bandit problem?
- My definition:
  - » Any sequential decision problem which involves learning, and where we have direct or indirect control over the information that is collected.

Multiarmed bandit problems

- Dimensions of a “bandit” problem:
  - » The “arms” (decisions) may be
    - Binary (A/B testing, stopping problems)
    - Discrete alternatives (drug, catalyst, …)
    - Continuous choices (price)
    - Vector-valued (basketball team, products, movies, …)
    - Multiattribute (attributes of a movie, song, person)
    - Static vs. dynamic choice sets
    - Sequential vs. batch
  - » Information (what we observe)
    - Success-failure/discrete outcome
    - Exponential family (e.g. Gaussian, exponential, …)
    - Heavy-tailed (e.g. Cauchy)
    - Data-driven (distribution unknown)
    - Stationary vs. nonstationary processes
    - Lagged responses?
    - Adversarial?

Multiarmed bandit problems

- Dimensions of a "bandit" problem:
  - » Belief models
    - Lookup tables (these are most common)
      - – Independent or correlated beliefs
    - Parametric models
      - – Linear or nonlinear in the parameters
    - Nonparametric models
      - – Locally linear
      - – Deep neural networks/SVM
    - Bayesian vs. frequentist
  - » Objective function
    - Expected performance (e.g. regret)
    - Offline (final reward) vs. online (cumulative reward)
      - – Just interested in final design?
      - – Or optimizing while learning?
    - Risk metrics

Outline

- Elements of a sequential decision model
- Mixed state problems
- Designing policies
- Searching for the best policy

Outline

- Elements of a sequential decision model
- Mixed state problems
- Designing policies
- Searching for the best policy

Modeling

- *Any* sequential decision problem consists of five core elements:
  - » State variables
  - » Decision variables
  - » Exogenous information
  - » Transition function
  - » Objective function

Modeling dynamic problems

- 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

Energy in storage

$I_t =$ Information state

Prices

Weather

$B_t =$ Belief state ("state of knowledge")

Belief about traffic delays

Belief about the status of equipment

Modeling dynamic problems

- The state variable:
  - » The initial state S^0 contains:
    - All deterministic parameters
    - Initial values of dynamic parameters
    - Prior distribution of belief about unknown parameters
  - » The dynamic state $S^n, n > 0$, contains
    - All information that changes over time.
    - Physical state

$$R^{n+1} = R^n + x^n + \hat{R}^{n+1}$$

    - Information state

$$p^{n+1} = p^n + \hat{p}^{n+1}$$

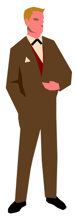
    - Belief state (Bayesian updating):

$$\bar{\mu}_x^{n+1} = \frac{\beta^n \bar{\mu}_x^n + \beta^W W^{n+1}}{\beta^n + \beta^W}$$

$$\beta_x^{n+1} = \beta_x^n + \beta^W$$

Modeling dynamic problems

- 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 $X^{\pi}(s)$ (or $A^{\pi}(s)$ or $U^{\pi}(s)$), where π specifies the type of policy. "π" carries information about the type of function f, and any tunable parameters $\theta \in \Theta^{f}$.

The decision variables

- Styles of decisions
  - » Binary

    $x \in X = \{0,1\}$
  - » Finite

    $x \in X = \{1, 2, \ldots, M\}$ $\leftarrow$ Classic bandit model
  - » Continuous scalar

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

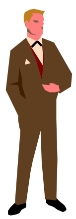
    $x = (x_1, \ldots, x_K), \quad x_k \in \mathbb{R}$
  - » Discrete vector

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

    $x = (a_1, \ldots, a_I), \quad a_i$ is a category (e.g. patient attributes)

Modeling dynamic problems

- 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.

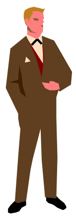
Below, we let ω represent a sequence of actual observations $W_1, W_2, ...$

$W_t\left(\omega\right)$ refers to a sample realization of the random variable W_t.

Modeling dynamic problems

- The transition function

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




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

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

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

$$\bar{\mu}_x^{n+1} = \frac{\beta^n \bar{\mu}_x^n + \beta^W W^{n+1}}{\beta^n + \beta^W}$$

$$\beta_x^{n+1} = \beta_x^n + \beta^W$$

Bayesian updating of belief

Also known as the:

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

Modeling stochastic, dynamic problems

- The universal objective function
  - » Cumulative reward (classical bandit objective)

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

  - » Final reward ("best arm" bandit objective)

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

Given a *system model* (transition function)

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

and a stochastic process:

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


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
VOLUME 2 • 4th EDITION
Dynamic Programming and Optimal Control
APPROXIMATE DYNAMIC PROGRAMMING
Dimitri P. Bertsekas
Athena Scientific
Wiley Series in Probability and Statistics
Optimal Learning
WILEY
SECOND EDITION
Model Predictive Control
OPTIMAL CONTROL
WILEY
INTRODUCTION TO STOCHASTIC SEARCH AND OPTIMIZATION
Estimation, Simulation, and Control
JAMES C. SPALL
MULTI-ARMED BANDIT ALLOCATION INDICES
SECOND EDITION
John Gittins, Kevin Glazebrook and Richard Weber
Reinforcement Learning
Introduction
Richard S. Sutton and Andrew G. Barto
WILEY
Markov Decision Processes
Discrete Stochastic Dynamic Programming
MARTIN L. PUTERMAN
43
Jiongmin Yong
Xun Yu Zhou
Stochastic Controls
Hamiltonian Systems and HJB Equations
Modelling and Applied Probability
Online Computation and Competitive Analysis
Allan Borodin
Ran El-Yaniv
Vol. 1
STOCHASTIC SIMULATION OPTIMIZATION
An Optimal Computing Budget Allocation
Chun-Hung Chen • Loo Hay Lee

Outline

- Elements of a sequential decision model
- Mixed state problems
- Designing policies
- Searching for the best policy

Modeling dynamic problems

- Some major problem classes
  - » Pure physical state $S^n = (R^n)$
    - Inventory problems
    - Stochastic shortest path problems
  - » Physical plus information $S^n = (R^n, I^n)$
    - Inventory with exogenous prices, weather, …
  - » Pure belief states $S^n = (B^n)$
    - These are classical bandit problems
    - Different types of belief models
  - » Belief plus information $S^n = (I^n, B^n)$
    - Patient arriving to doctor's office who then prescribes a drug.
    - "Contextual bandit problems"
  - » Everything: $S^n = (R^n, I^n, B^n)$
    - Revenue management
    - Clinical trials

Modeling dynamic problems

- Mixed state problems (physical and belief state)

  » Clinical trials
    - Learning the performance of a new drug (belief state)
    - Tracking the number of patients signed up (physical state)

  » Revenue management for hotels
    - Learning market response to price (belief state)
    - Tracking how many rooms have been reserved (physical state)

  » An energy storage problem…

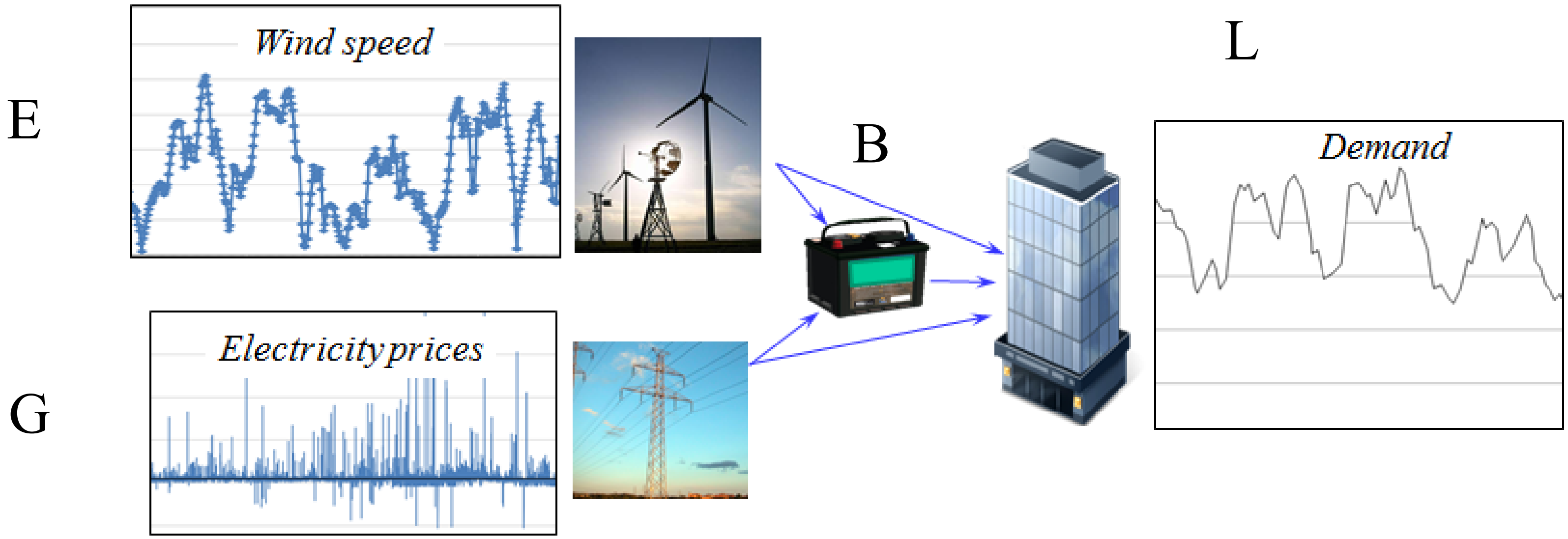
An energy storage problem

- Consider a basic energy storage problem:

» We have to manage the flows of energy (blue lines) while managing different sources of uncertainty.

An energy storage problem

- Transition function without learning



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

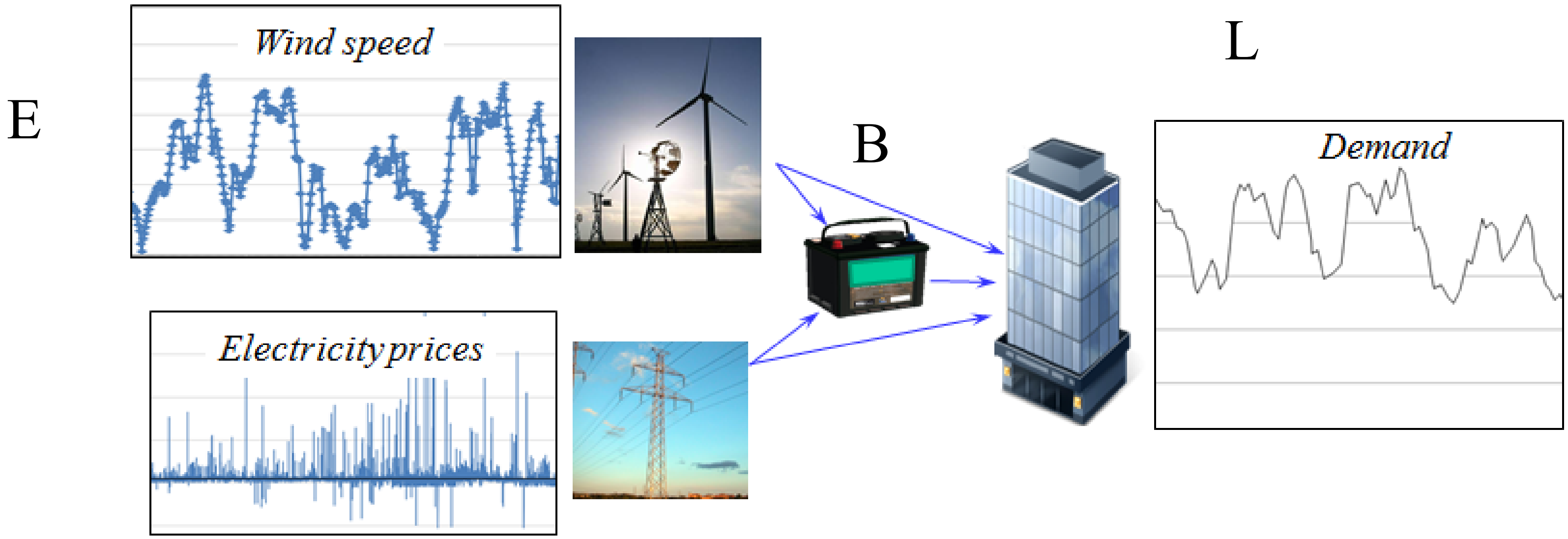
$$p_{t+1} = \theta_0 p_t + \theta_1 p_{t-1} + \theta_2 p_{t-2} + \varepsilon^p_{t+1}$$

$$D_{t+1} = f^D_{t,t+1} + \varepsilon^D_{t+1}$$

$$R^{battery}_{t+1} = R^{battery}_t + x_t$$

An energy storage problem

- Transition function with passive learning



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

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

$$D_{t+1} = f^{D}_{t,t+1} + \varepsilon^{D}_{t+1}$$

$$R^{battery}_{t+1} = R^{battery}_{t} + x_t$$

Learning in stochastic optimization

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

$$\bar{F}^n(x|\bar{\theta}_t) = \bar{\theta}_{t0} p_t + \bar{\theta}_{t1} p_{t-1} + \bar{\theta}_{t2} p_{t-2}$$

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

$$\bar{\theta}_{t+1} = \bar{\theta}_t - \frac{1}{\gamma_{t+1}} B_t \phi_t \varepsilon_{t+1} \qquad \phi_t = \begin{pmatrix} p_t \\ p_{t-1} \\ p_{t-2} \end{pmatrix}$$

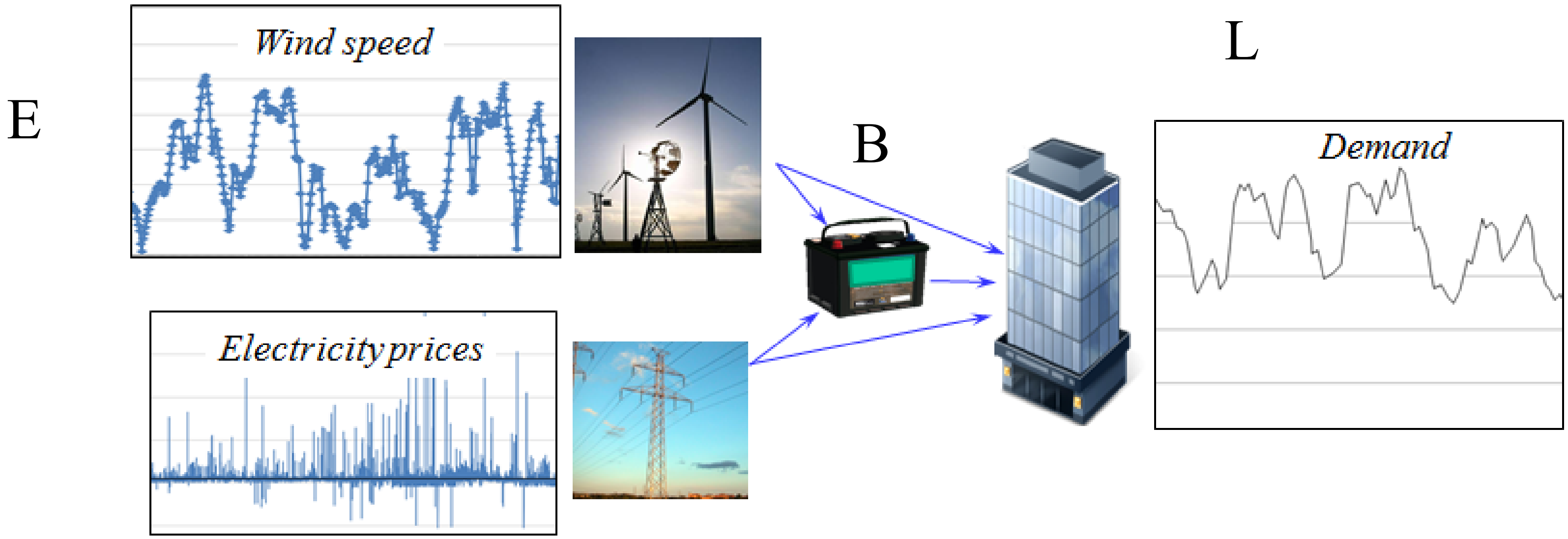
$$\varepsilon_{t+1} = \bar{F}_t(x_t | \bar{\theta}_t) - p_{t+1},$$

$$B_{t+1} = B_t - \frac{1}{\gamma_{t+1}} \left(B_t \phi (\phi)^T B_t \right)$$

$$\gamma_{t+1} = 1 + (\phi)^T B_t \phi$$

An energy storage problem

- Transition function with active learning



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

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

$$D_{t+1} = f^{D}_{t,t+1} + \varepsilon^{D}_{t+1}$$

$$R^{battery}_{t+1} = R^{battery}_{t} + x_t$$

Outline

- Elements of a sequential decision model
- Mixed state problems
- Designing policies
- Searching for the best policy

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

- Two fundamental strategies:

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

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

2) 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:
  1a) 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 - $x_t = X^{PFA}(S_t \mid \theta) = \sum_{f \in F} \theta_f \phi_f(S_t)$
      - Neural networks
    - Locally/semi/non parametric
      - Requires optimizing over local regions

  1b) Cost function approximations (CFAs)
    - Optimizing a deterministic model modified to handle uncertainty (buffer stocks, schedule slack)

$$X^{CFA}(S_t \mid \theta) = \arg\max_{x_t} \left(\bar{\mu}_{tx} + \theta \sigma_{tx} \right)$$

Designing policies

- Lookahead policies

  2a) Value function approximations

  We approximate the impact of a decision on the future

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

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

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

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

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

Designing policies

- Lookahead policies

  2a) Value function approximations

  We approximate the impact of a decision on the future

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

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

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

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

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

Designing policies

- Lookahead policies

  2a) Value function approximations

  We approximate the impact of a decision on the future

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

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

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

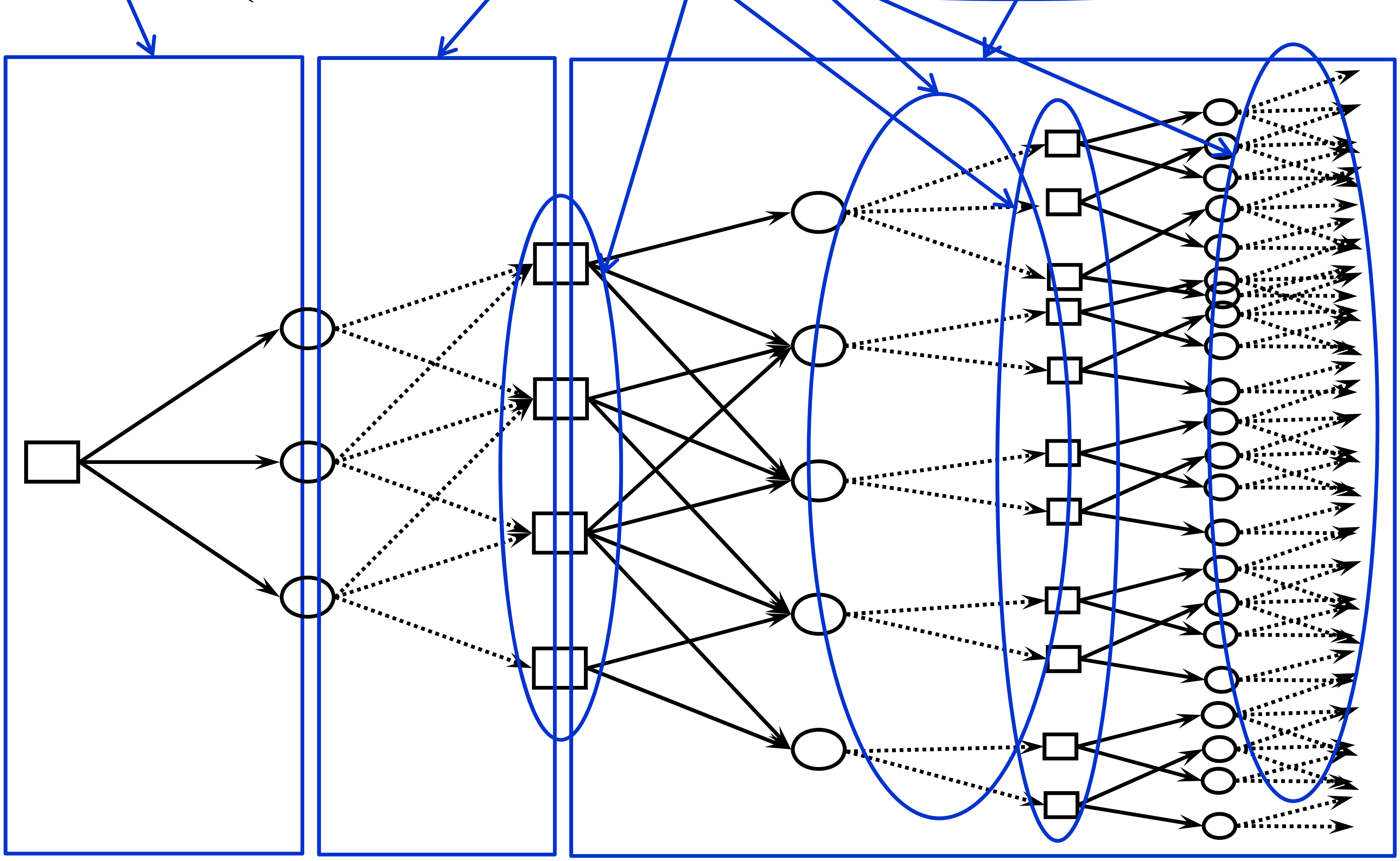
$$X_t^{VFA}(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \mathbb{E}\left\{ \overline{V}_{t+1}(S_{t+1}) \,|\, S_t, x_t \right\} \right)$$

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

Designing policies

- 2b) Direct lookahead policies

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



Designing policies

- 2b) Direct lookahead policies
  - » We replace the exact lookahead…

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

… with an approximation called the *lookahead model*:

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

  - » *A lookahead policy* works by approximating the *lookahead model.*

Designing policies

- Types of lookahead approximations
  - » One-step lookahead – Widely used in pure learning policies:
    - Bayes greedy/naïve Bayes
    - Thompson sampling
    - Value of information (knowledge gradient)
  - » Multi-step lookahead
    - Deterministic lookahead, also known as model predictive control, rolling horizon procedure
    - Stochastic lookahead:
      - Two-stage (widely used in stochastic linear programming)
      - Multistage
        - » Monte carlo tree search (MCTS) for discrete action spaces
        - » Multistage scenario trees (stochastic linear programming) – typically not tractable.

Four (meta)classes of policies

Policy search

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

» Lookup tables, rules, parametric/nonparametric functions

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

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

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*

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

» *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"*

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

Four (meta)classes of 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"*

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

Four (meta)classes of policies

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

» Lookup tables, rules, parametric/nonparametric functions

Imbedded optimization

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

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

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*

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

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

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

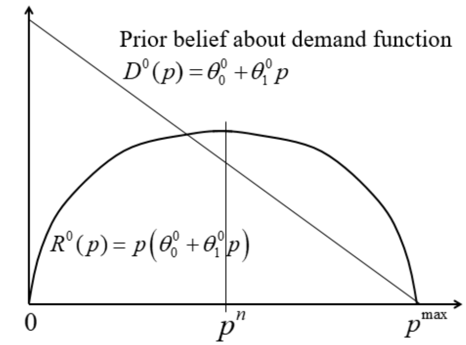
» *"Robust optimization"*

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

Policies for pure learning problems

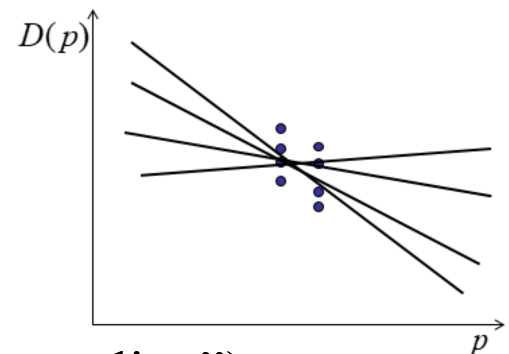
- 1) Policy function approximation (PFA)
  - » Revenue maximization problem
    - Demand function

$$D(p \mid \bar{\theta}^n) = \bar{\theta}_1^n - \bar{\theta}_2^n p$$



    - Revenue

$$R(p \mid \bar{\theta}^n) = pD(p) = \bar{\theta}_1^n p - \bar{\theta}_2^n p^2$$



    - PFA policy – pure exploitation

$$p^n = \frac{\bar{\theta}_1^n}{2\bar{\theta}_2^n}$$

    - PFA policy with active exploration ("excitation policy")

$$p^n = \frac{\bar{\theta}_1^n}{2\bar{\theta}_2^n} + \varepsilon^n \qquad \varepsilon^n \sim N(0, \sigma^\varepsilon)$$

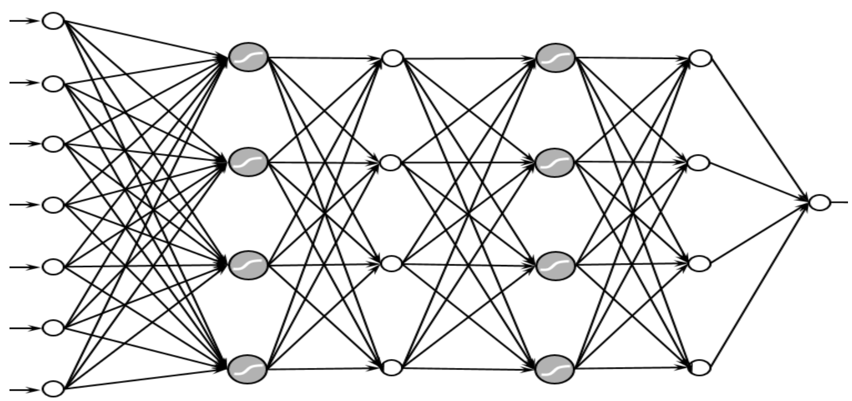
    - Need to tune σ^ϵ

Policies for pure learning problems

- 1) Policy function approximation (PFA)
  - » Linear decision rules ("affine policies")

$$X^{PFA}(S^n \mid \theta) = \theta_0 + \theta_1\phi_1(S^n) + \theta_2\phi_2(S^n) + ... + \theta_F\phi_F(S^n)$$

  - » Neural networks

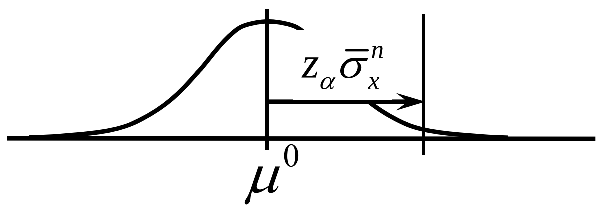


Policies for pure learning problems

- 2) 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



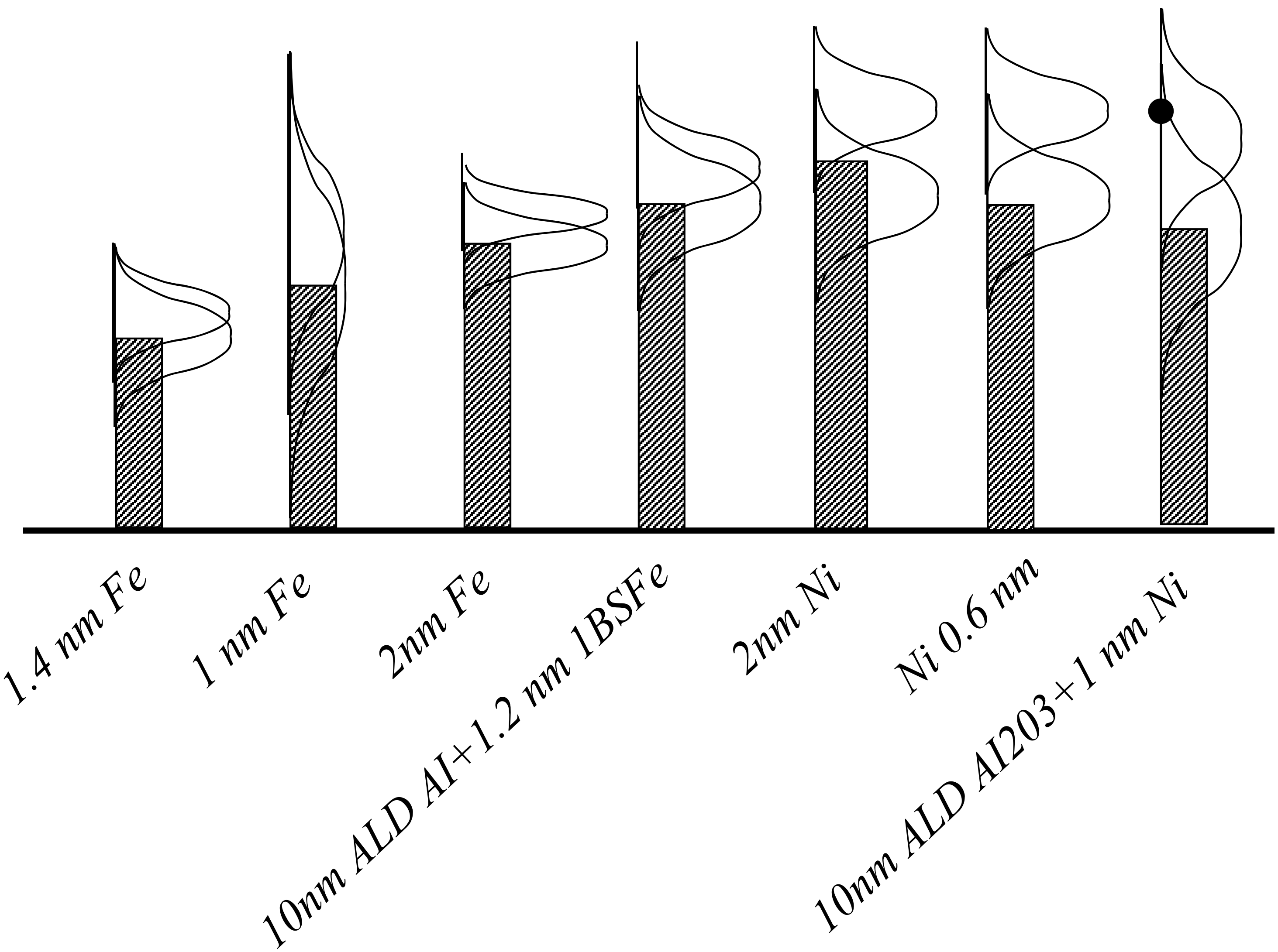
$$X^{IE}(S^n \mid \theta^{IE}) = \arg\max_x \left(\bar{\mu}_x^n + \theta^{IE} \bar{\sigma}_x^n \right)$$

  - » Boltzmann exploration ("soft max")
    - Choose x with probability: $P_x^n(\theta) = \dfrac{e^{\theta \bar{\mu}_x^n}}{\sum_{x'} e^{\theta \bar{\mu}_{x'}^n}}$

$$X^{Boltz}(S^n | \theta) = \arg\max_x \{x | P_x^n(\theta) \leq U\}.$$

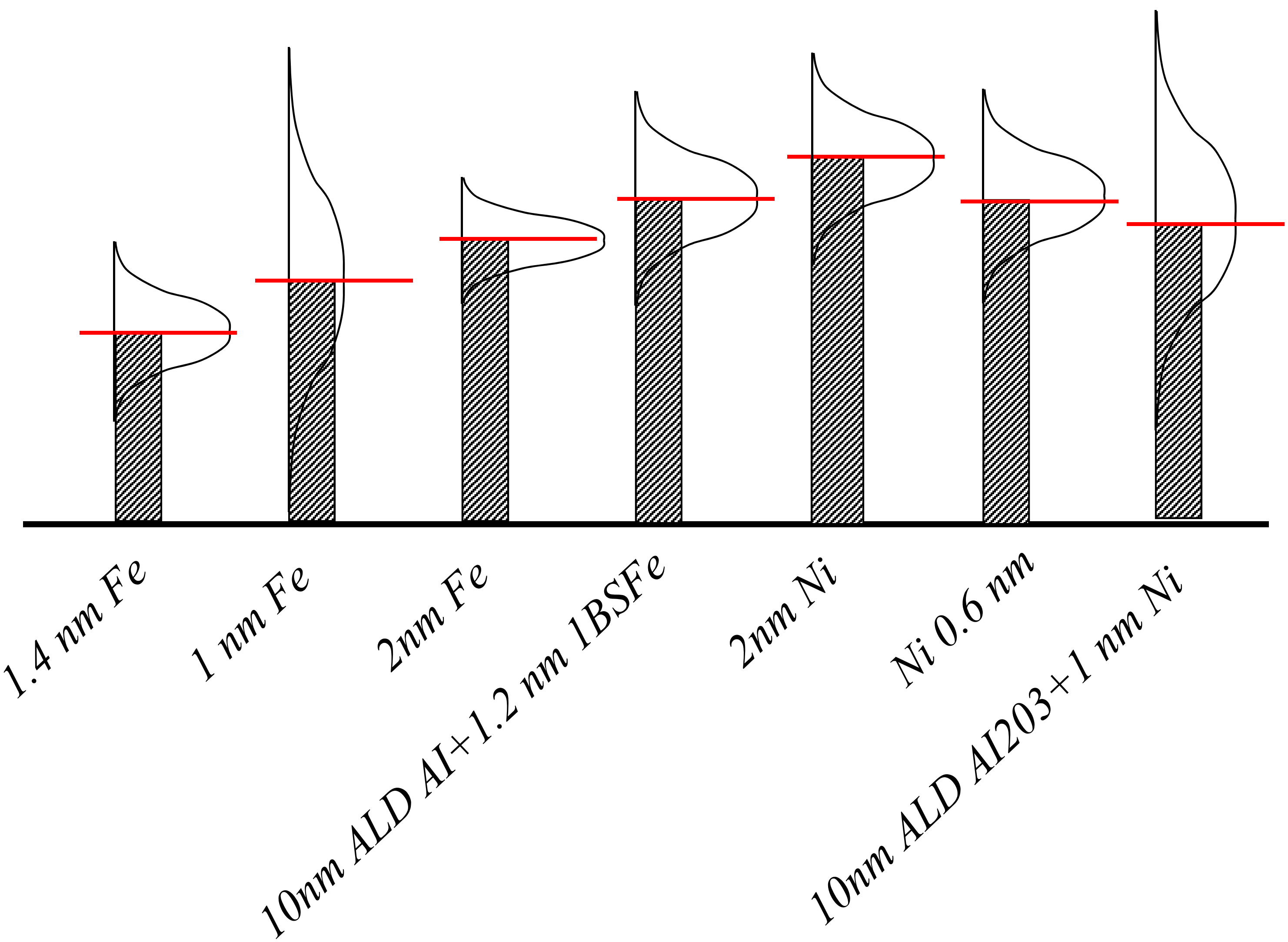
Policies for pure learning problems

- A learning problem with correlated beliefs



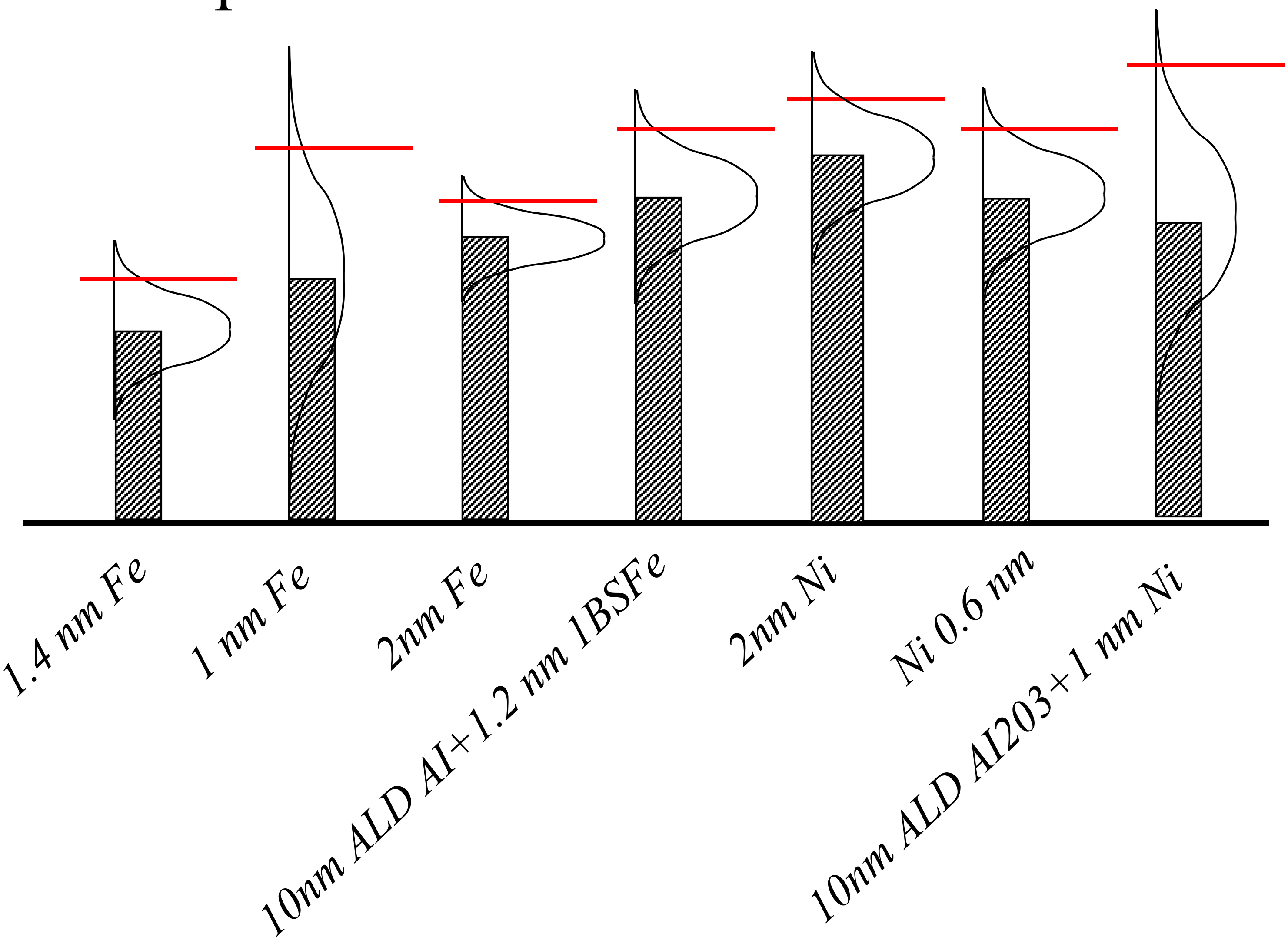
Policies for pure learning problems

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



Policies for pure learning problems

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



Policies for pure learning problems

- PFAs and CFAs have to be tuned

  » Final reward ("offline learning")

$$\max_{\theta^{IE}} \mathbb{E}F(x^{\pi,N},\hat{W}) = \mathbb{E}_{\mu}\mathbb{E}_{W^1,\ldots,W^N|\mu}\mathbb{E}_{\hat{W}}(x^{\pi,N}(\theta^{IE}),\hat{W})$$

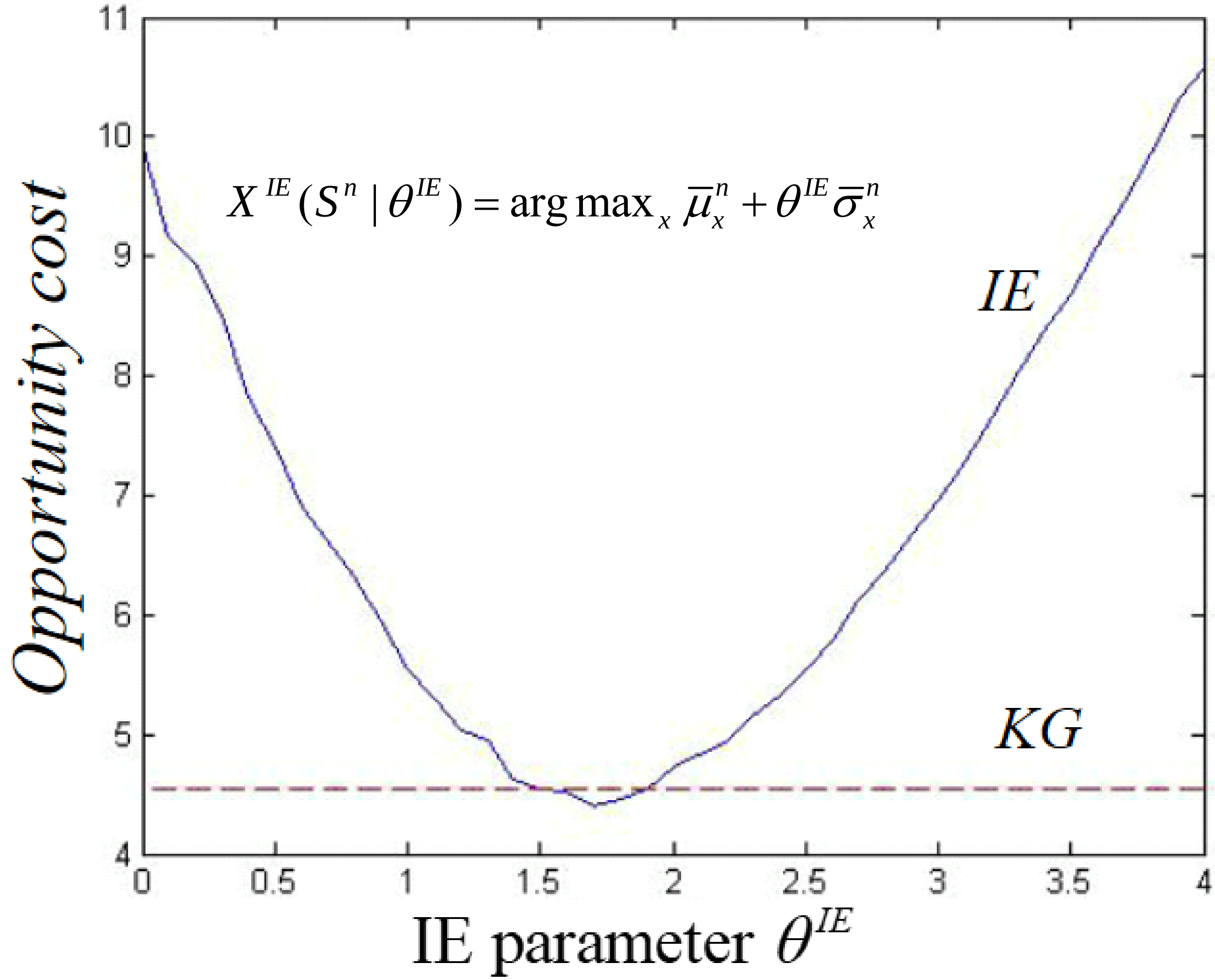
  » Cumulative reward ("online learning")

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

  » Both require searching over tunable parameters.
    - Offline tuning is classical stochastic search
    - Online tuning is a relatively open research area

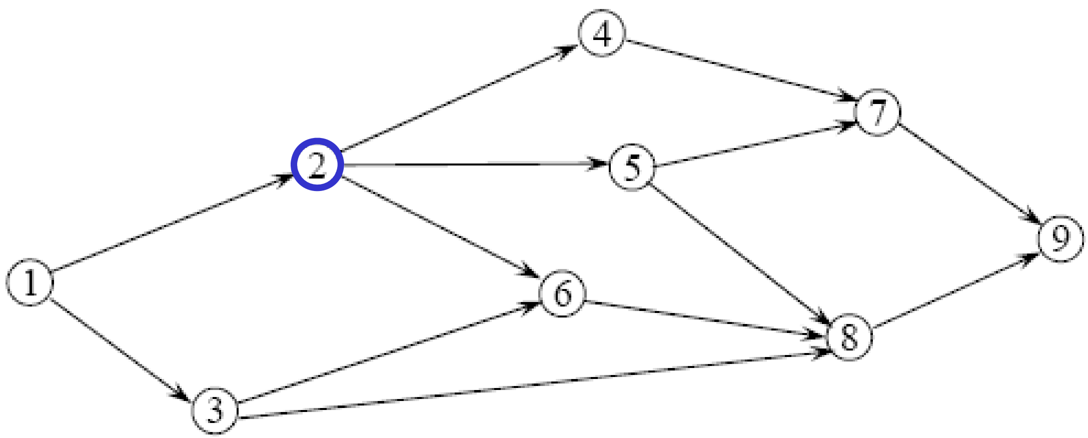
Cost function approximations

- Tuning the interval estimation policy



Policies for pure learning problems

- 3) Policies based on value function approximations
  - » VFAs using a physical state problem

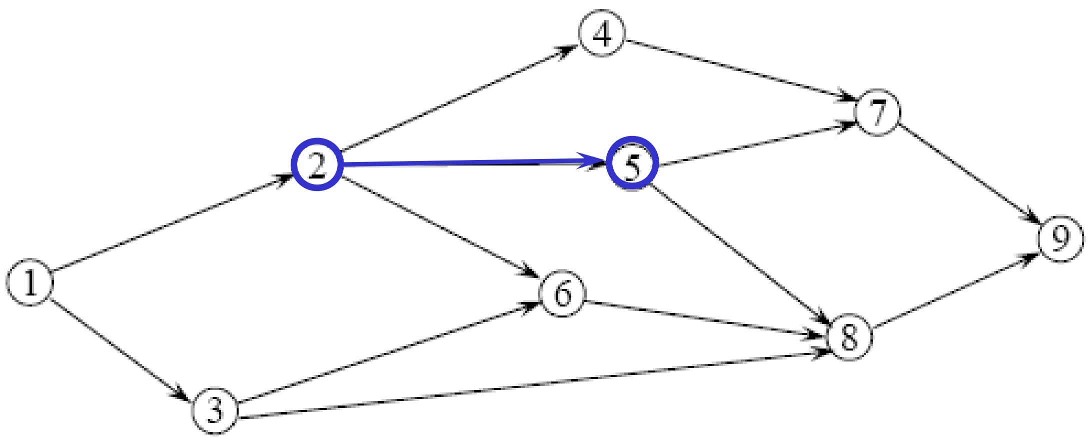


$$V^n(S^n) = \max_x \left(C(S^n, x) + E\left\{ V^{n+1}(S^{n+1}) \mid S^n \right\} \right)$$

Current node (e.g. node 2)

Policies for pure learning problems

- 3) Policies based on value function approximations
  » VFAs using a physical state problem



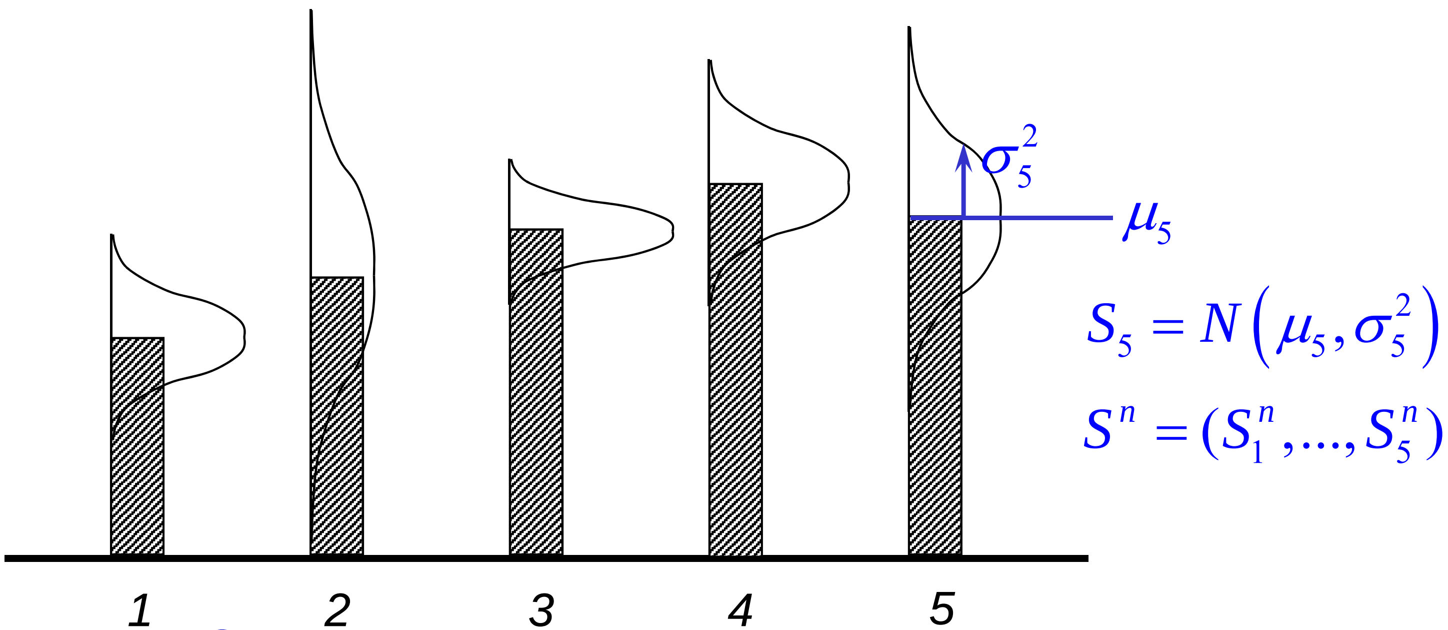
$$V^n(S^n) = \max_x \left(C(S^n, x) + E\left\{ V^{n+1}(S^{n+1}) \mid S^n \right\} \right)$$

Decision to go to a node (e.g. 5)

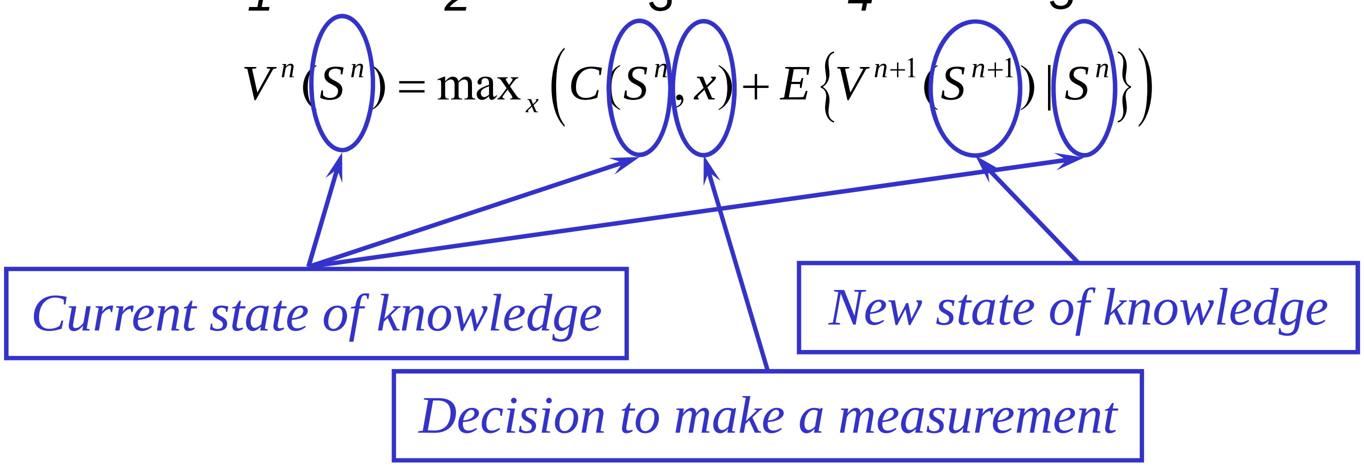
Downstream node (e.g. 5)

Policies for pure learning problems

- 3) Policies based on value function approximations
  - » VFAs using a learning problem



$$V^n(S^n) = \max_x \left(C(S^n, x) + E\left\{ V^{n+1}(S^{n+1}) \mid S^n \right\} \right)$$



Policies for pure learning problems

- 3) Policies based on value function approximations
  - » Illustration: finding the best drug in the set $\mathcal{X} \in \{x_1, x_2, \ldots, x_M\}$.
  - » After a test we observe success or failure:

$$W_x^{n+1} = \begin{cases} 1 & \text{Success} \\ 0 & \text{Failure} \end{cases} \quad \text{If } x^n = x$$

  - » Let ρ_x =Probability that drug x is successful. We assume that

$$\rho_x \mid S^n \sim Beta(\alpha_x^n, \beta_x^n)$$

where $S^n = (\alpha^n, \beta^n)$ is our belief state, with updating equations:

$$\alpha_x^{n+1} = \alpha_x^n + W_x^{n+1}, \quad \beta_x^{n+1} = \beta_x^n + (1 - W_x^{n+1})$$

Policies for pure learning problems

- 3) Policies based on value function approximations
  - » Bellman's equation:

$$V^n(\alpha^n, \beta^n) = \max_x \mathbb{E}\left[W_x^{n+1} + \gamma V^{n+1}(\alpha^n + W^{n+1}, \beta^n + 1 - W^{n+1}) \mid S^n\right]$$

  - » This can be solved for a stopping problem to determine when to stop testing a single drug.
  - » Problematic if α^n and β^n are vectors. Gittins developed a novel decomposition that allows us to solve this problem for one drug ("arm") at a time.

Policies for pure learning problems

- 3) Policies based on value function approximations
  - » For normally distributed rewards, Gittins (1974) showed that we can solve dynamic programs for each alternative.
  - » Produces a policy that looks like

$$X^{Gitt}(S^n) = \arg\max_x \left(\bar{\mu}_x^n + \sigma^W \Gamma\left(\frac{\sigma_x^n}{\sigma^W}, \gamma \right) \right)$$

where $\Gamma\left(\frac{\sigma_x^n}{\sigma^W}, \gamma \right)$ is the “Gittins index” obtained by solving a dynamic program for whether to continue or stop testing a single drug.

  - » Considered a computational breakthrough, but computing Gittins indices is still a challenge, and only applies to special cases.

Policies for pure learning problems

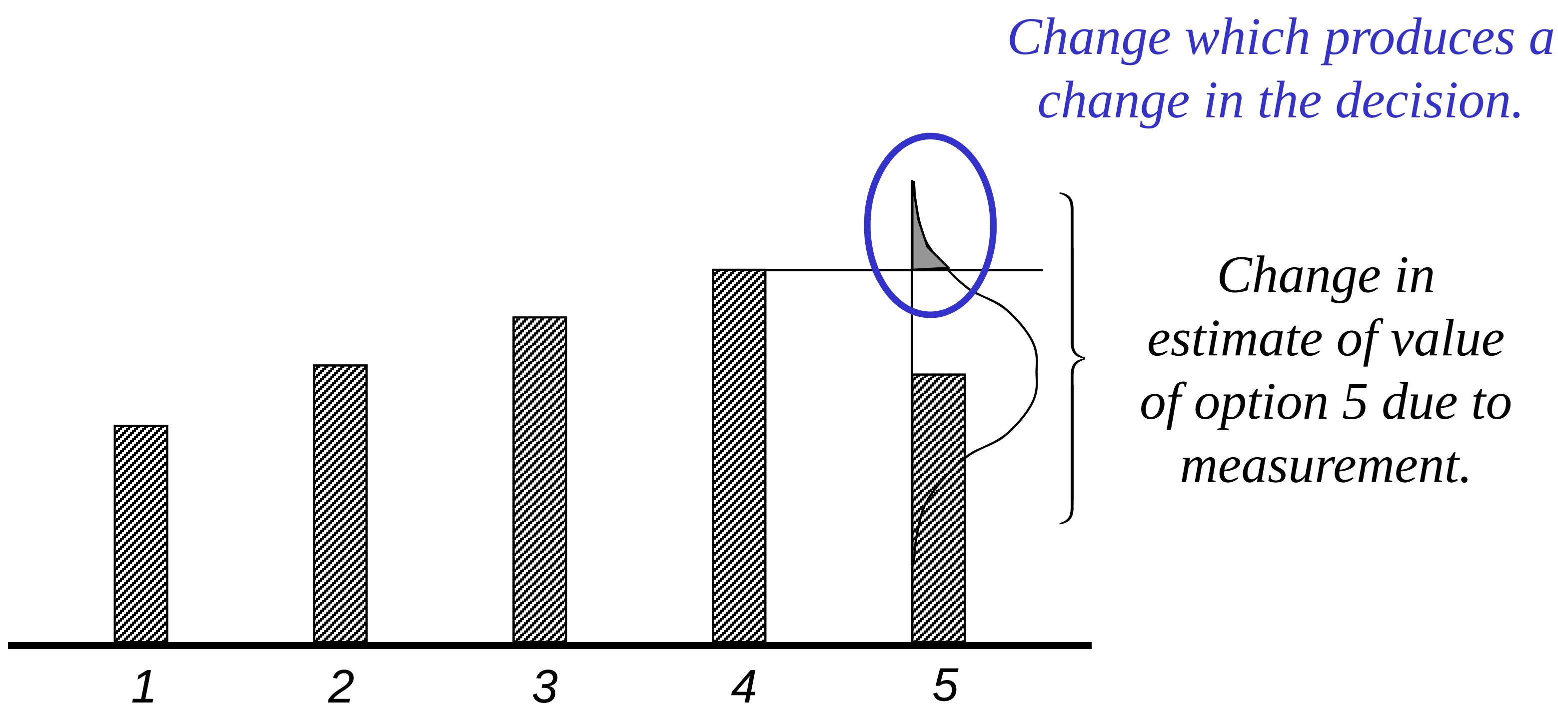
- 4) Policies based on direct lookaheads (DLA)
  - » The knowledge gradient for offline (final reward):

$$\nu_x^{KG,n} = E\left\{\max_y F(y, B^{n+1}(x))\right\} - \max_y F(y, B^n)$$

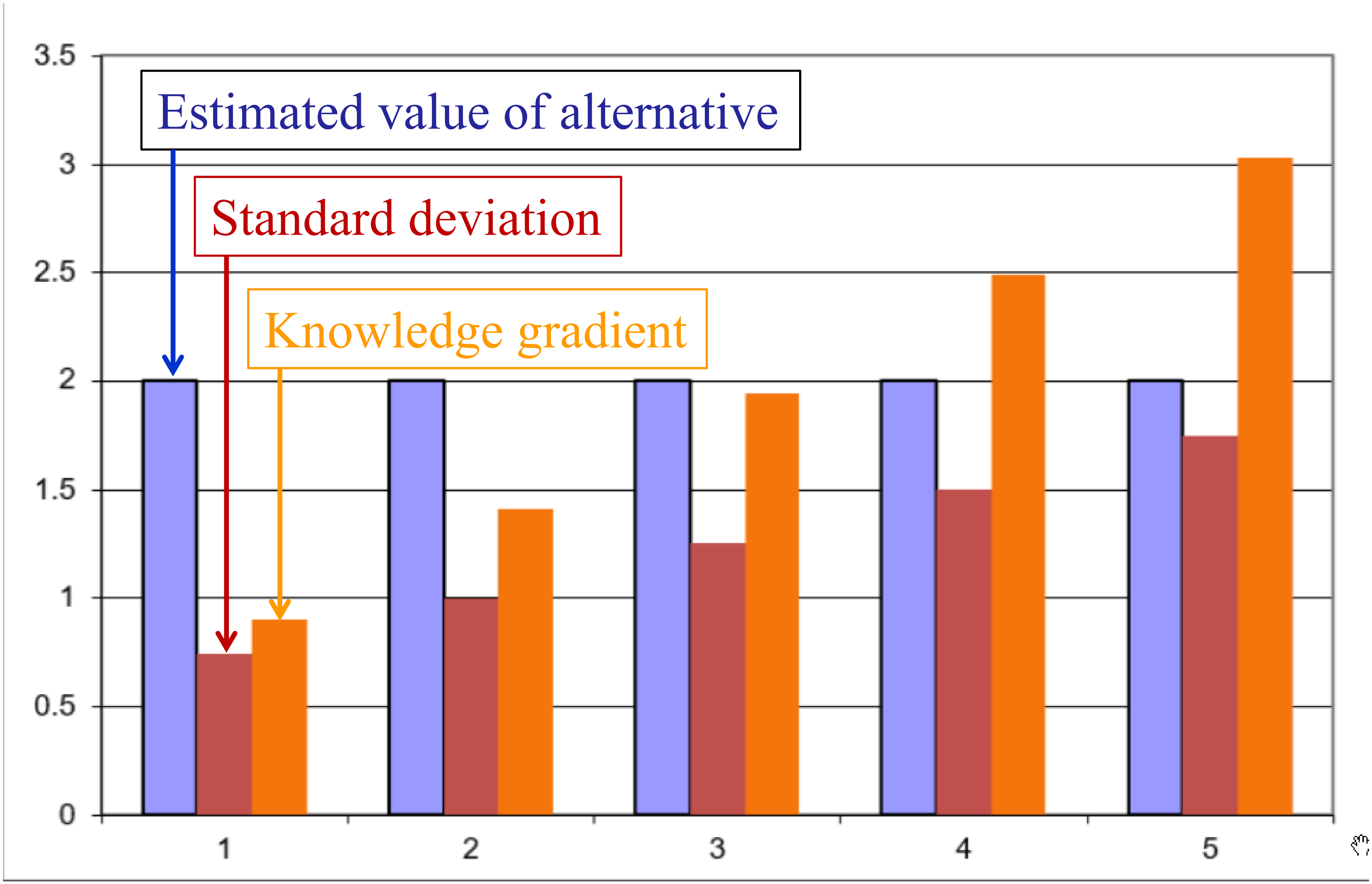
- *Finding the new design with our new belief (but without knowing the outcome of the experiment)*
- *Averaging over the possible outcomes of the experiment (and our different beliefs about parameters)*
- *Updated parameter estimates after running experiment with density x.*
- *Proposed experiment*
- *Choosing the best design given what we know now.*
- *Current belief state*

The knowledge gradient

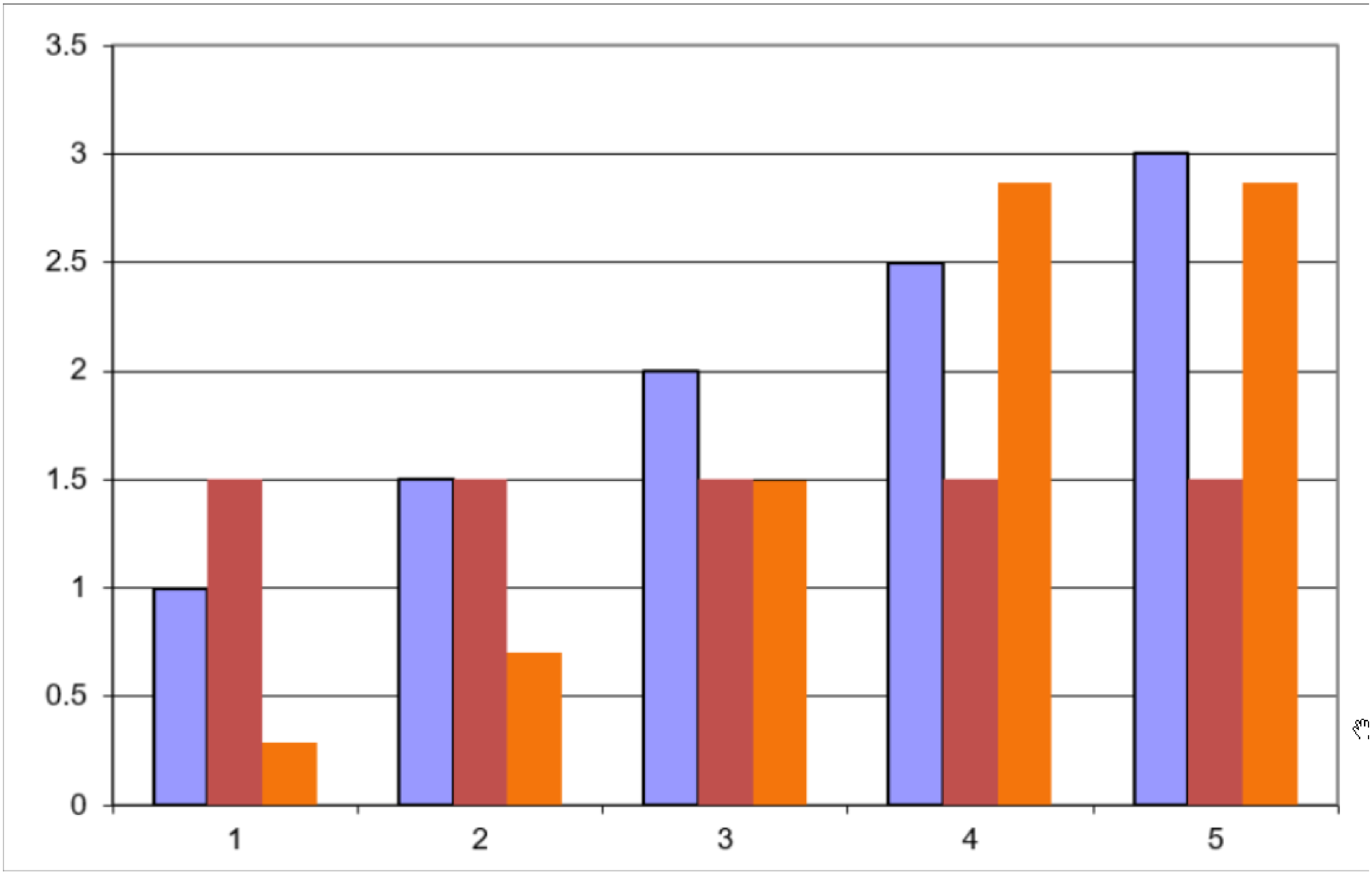
- 4) Policies based on direct lookaheads (DLA)
  - » The knowledge gradient computes the expected improvement from a single experiment



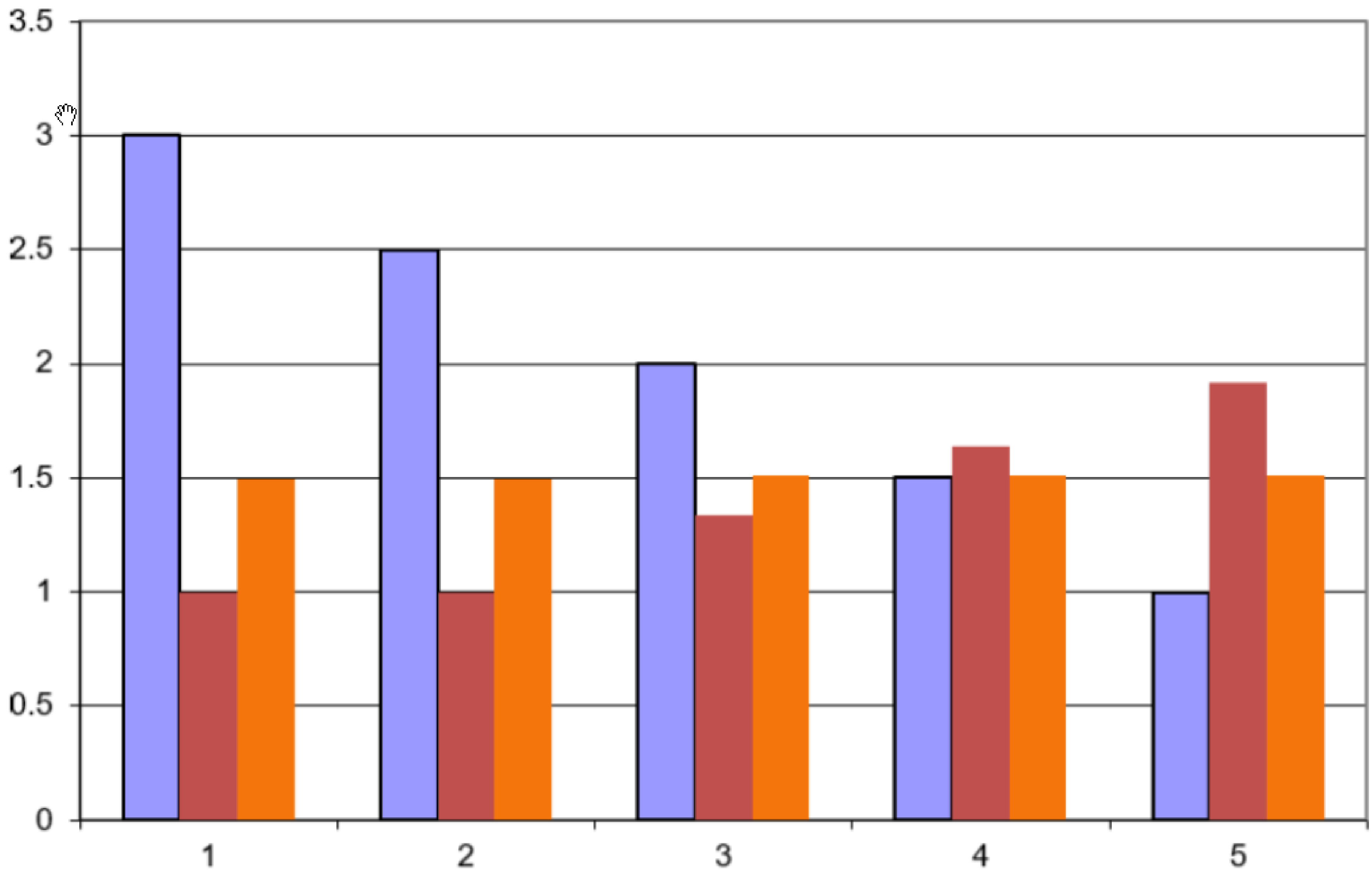
The knowledge gradient



The knowledge gradient



The knowledge gradient



The knowledge gradient

- Some properties of the knowledge gradient for offline (final reward) problems.
  - » $\nu_x^{KG,n} \geq 0$
  - » Asymptotically optimal (finds best *x* in the limit)
  - » Optimal (by construction) if budget =1.
  - » Optimal for all *n* if number of alternatives = 2 (e.g. A/B testing).
  - » Only stationary policy that is both myopically and asymptotically optimal.
- For online problems
  - » Asymptotically optimal (finds best *x* in the limit) as $\gamma \rightarrow 1$

SIAM J. CONTROL OPTIM.
Vol. 56, No. 2, pp. 1105–1129

FINITE-TIME ANALYSIS FOR THE KNOWLEDGE-GRADIENT POLICY*

YINGFEI WANG† AND WARREN B. POWELL‡

Abstract. We consider sequential decision problems in which we adaptively choose one of finitely many alternatives and observe a stochastic reward. We offer a new perspective on interpreting Bayesian ranking and selection problems as adaptive stochastic multiset maximization problems and derive the first finite-time bound of the knowledge-gradient policy for adaptive submodular objective functions. In addition, we introduce the concept of prior-optimality and provide another insight into the performance of the knowledge-gradient policy based on the submodular assumption on the value of information. We demonstrate submodularity for the two-alternative case and provide other conditions for more general problems, bringing out the issue and importance of submodularity in learning problems. Empirical experiments are conducted to further illustrate the finite-time behavior of the knowledge-gradient policy.

Key words. ranking and selection, sequential decision analysis, stochastic control

AMS subject classifications. 62F07, 62F15, 62L05, 93E35, 68W40, 68T05

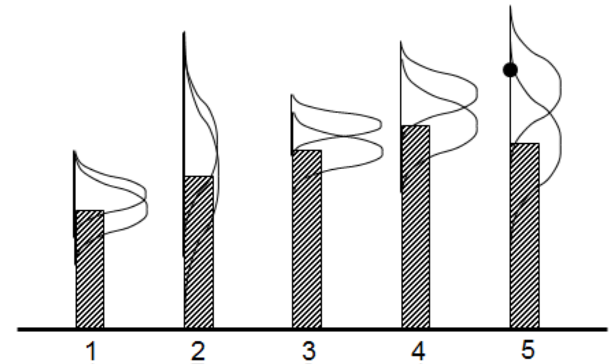
DOI. 10.1137/16M1073388

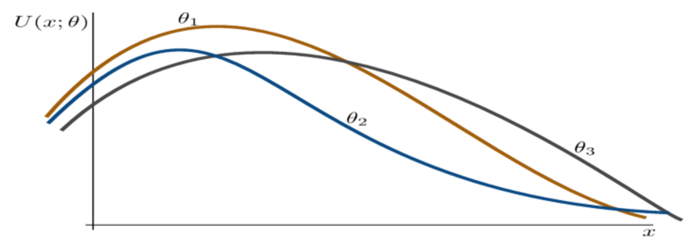
1. Introduction.

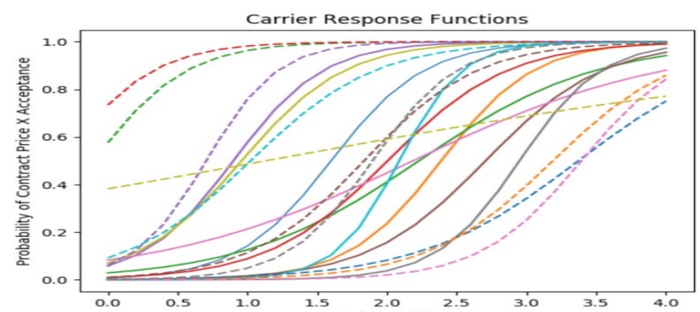
We consider sequential decision problems in which at each time step, we choose one of finitely many alternatives and observe a random reward. The rewards are independent of each other and follow some unknown probability distribution. One goal can be to identify the alternative with the best expected performance within a limited measurement budget, which is the objective of Bayesian ranking and selection problems. Ranking and selection problems are exam-

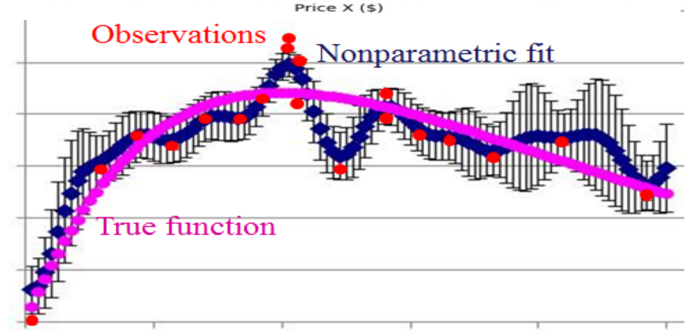
The knowledge gradient

- Different belief models
  - » Lookup tables
    - **Independent beliefs**
    - **Correlated beliefs**
  - » Linear parametric models
    - **Linear models**
    - **Sparse-linear**
    - **Tree regression**
  - » Nonlinear parametric models
    - **Logistic regression**
    - Neural networks
  - » Nonparametric models
    - **Gaussian process regression**
    - **Kernel regression**
    - Support vector machines
    - Deep neural networks



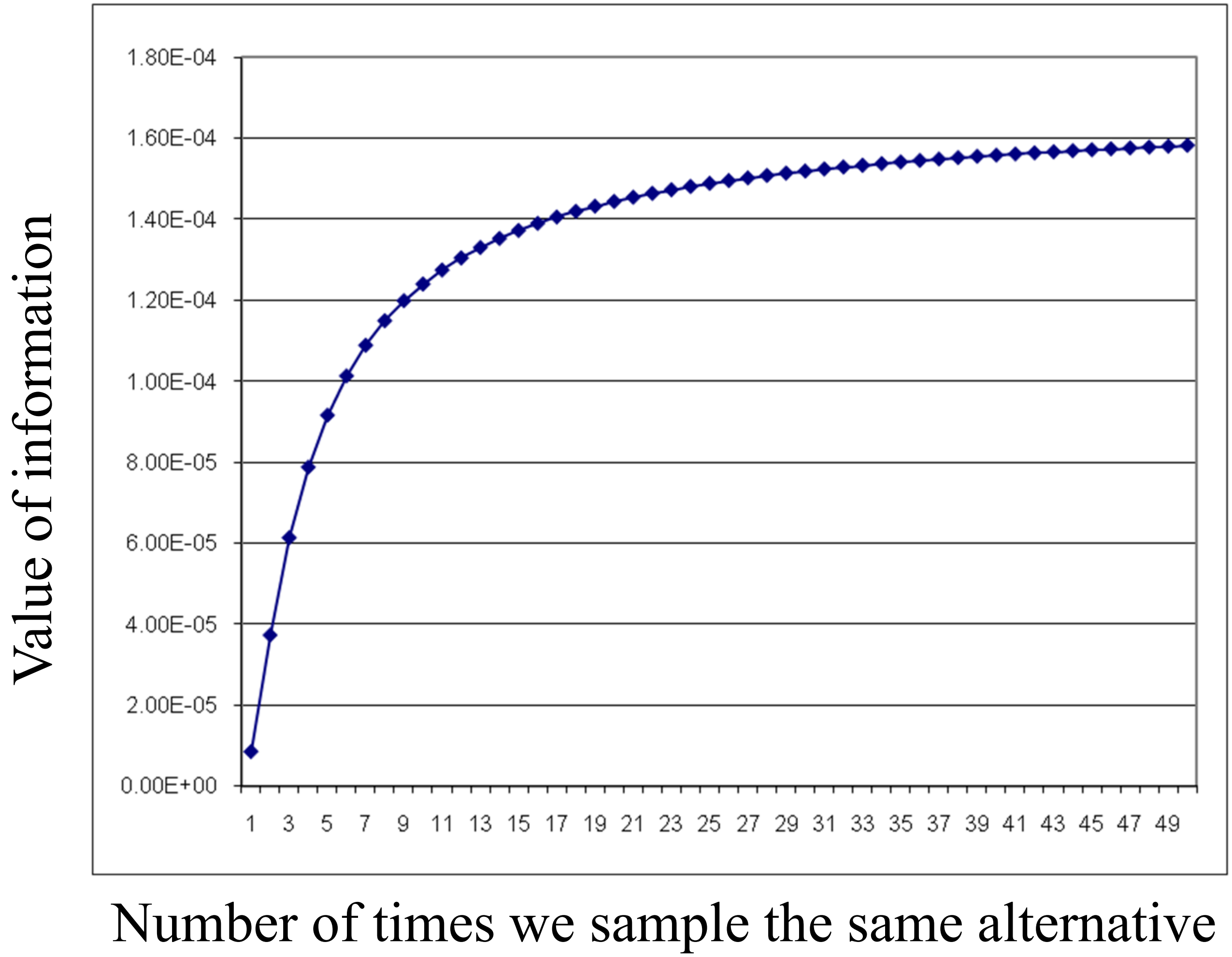






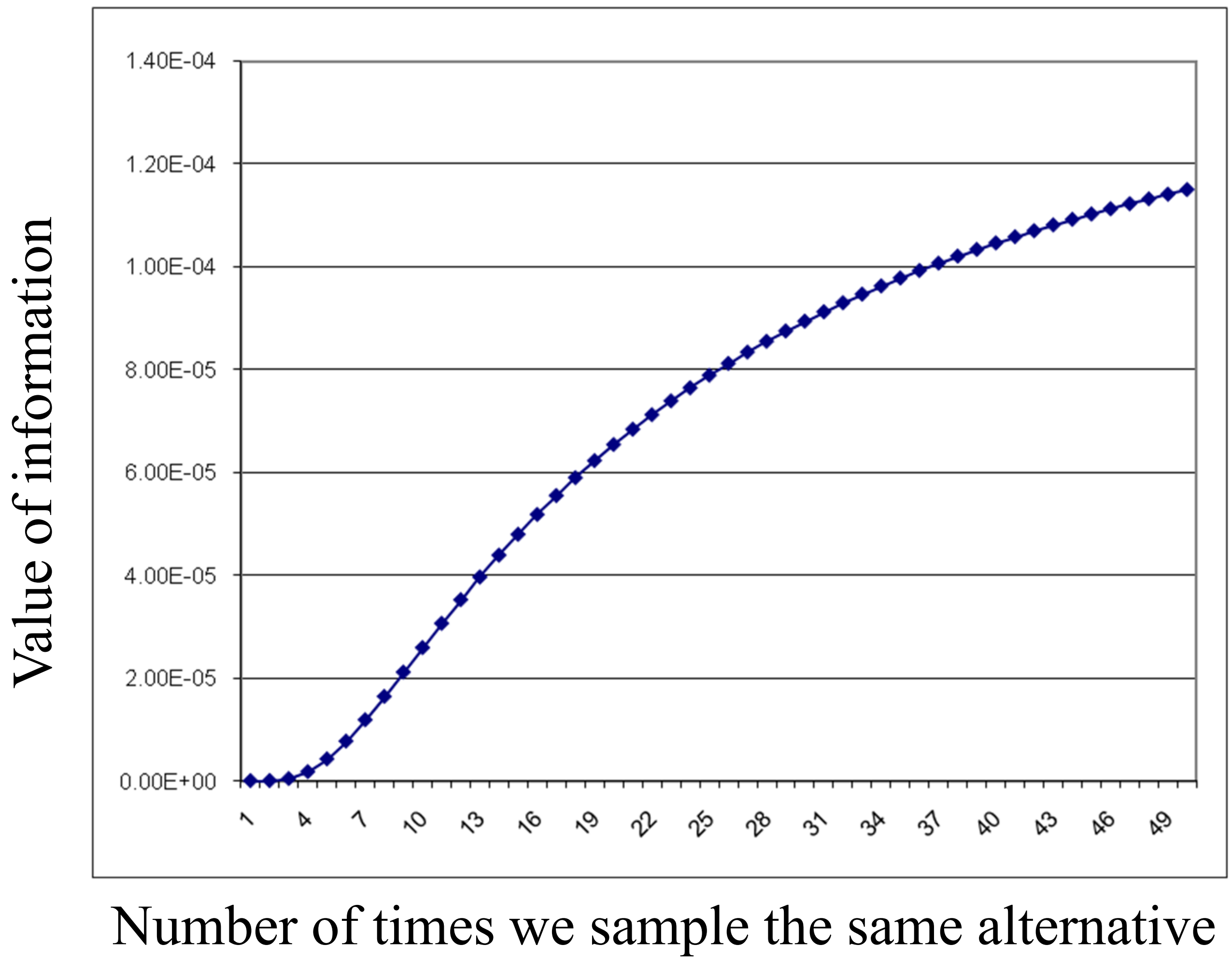
The knowledge gradient

- The marginal value of information
  » Repeatedly sampling the same alternative



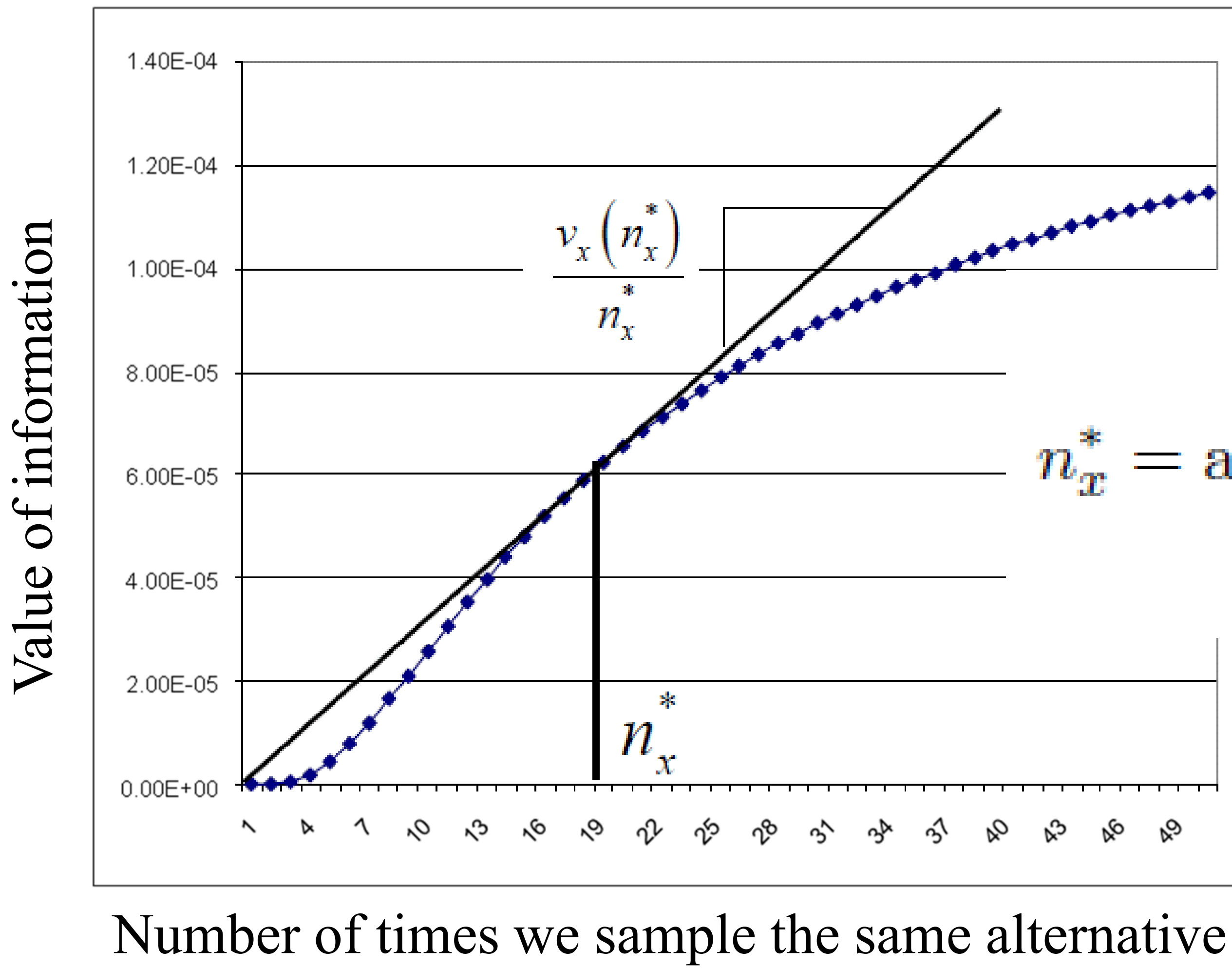
The knowledge gradient

- The marginal value of information
  - » The value of information may be concave if an experiment is noisy



The knowledge gradient

- The marginal value of information
  - » The value of information may be concave if an experiment is noisy



$$n_x^* = \arg\max_{n_x > 0} \frac{v_x(n_x)}{n_x}.$$

The knowledge gradient

- From offline to online learning
  - » The knowledge gradient computes the value of information for a terminal reward objective:

$$\nu_x^{KG,n} = E\left\{\max_y F(y, B^{n+1}(x))\right\} - \max_y F(y, B^n)$$

  - » Imagine that we have a budget of N experiments, and that we are summing rewards over this horizon. The value of information from a single experiment is now

$$\nu_x^{KG-OL,n} = \bar{\mu}_x^n + (N-n)\nu_x^{KG,n}$$

Expected reward

Remaining horizon

Offline KG

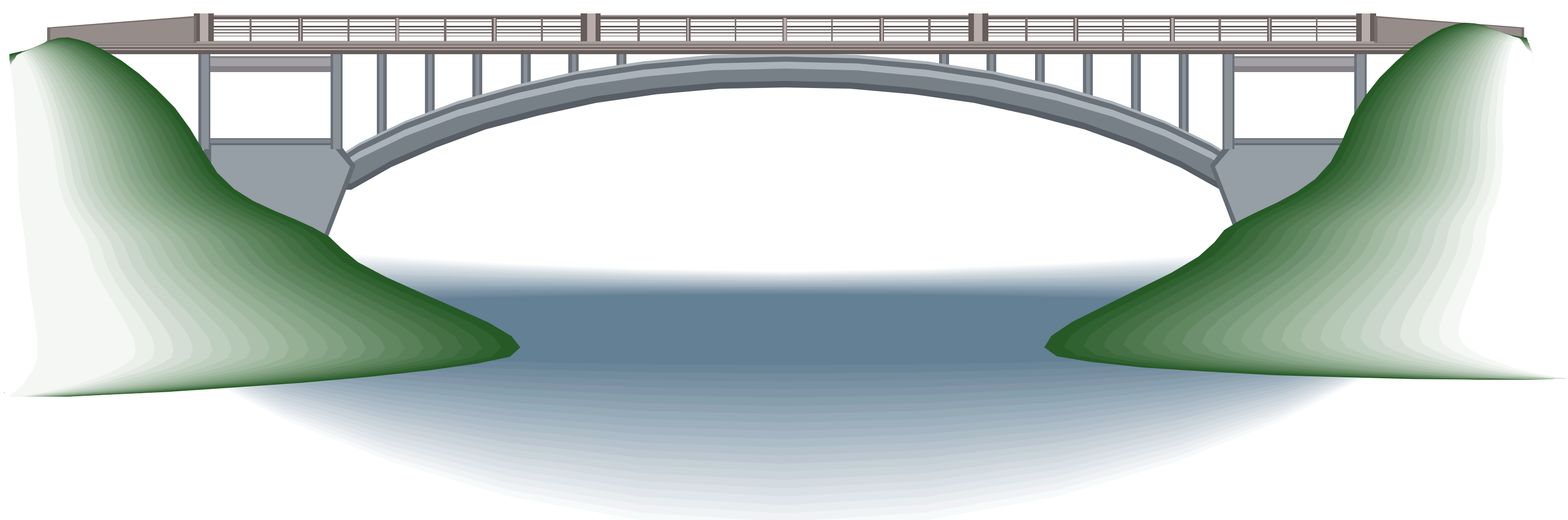
The knowledge gradient

- Knowledge gradient for offline and online learning

Offline learning — Online learning

$\nu_x^{KG,n}$

$\nu_x^{KG-OL,n} = \bar{\mu}_x^n + (N-n)\nu_x^{KG,n}$



» This bridges what have historically been fundamentally different fields.

Outline

- Elements of a sequential decision model
- Mixed state problems
- Designing policies
- Searching for the best policy

Designing policies

- Finding the best policy
  - » We have to first articulate our classes of policies

$$f \in \mathcal{F} = \{PFAs, CFAs, VFAs, DLAs\}$$

$$\theta \in \Theta^f = \text{Parameters that characterize each family.}$$

  - » So minimizing over $\pi \in \Pi$ means:

$$\Pi = \{ f \in \mathcal{F}, \theta \in \Theta^f \}$$

  - » We then have to pick an objective such as

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

or

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

Multiarmed bandit problems

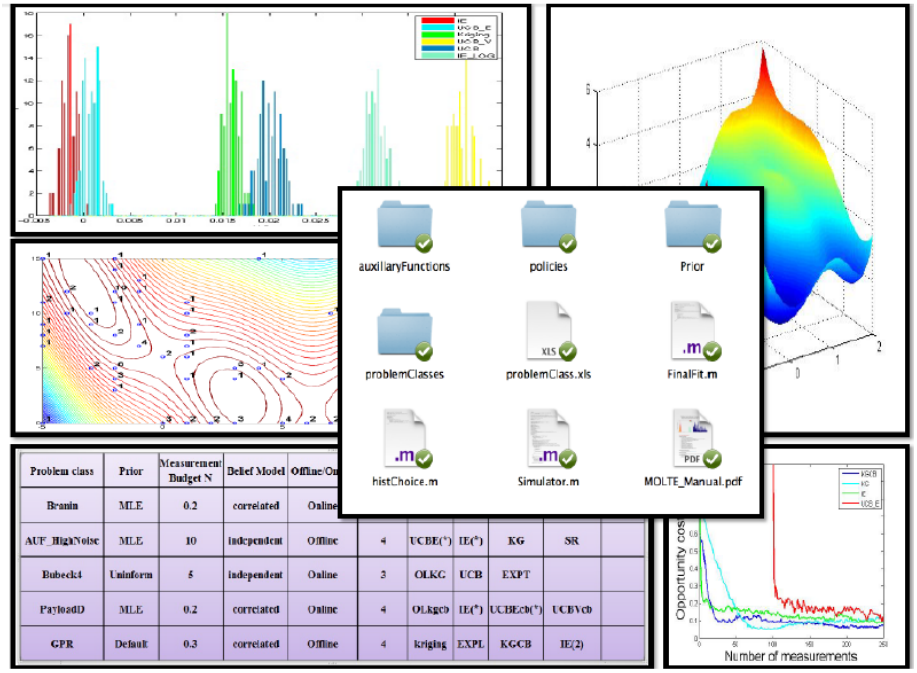
- Policy search class
  - » Policies tend to be relatively simple and easy to compute
  - » Well suited to rapid (e.g. internet speed) learning applications needing fast computation.
  - » Tuning is important, and typically requires a realistic simulator.
- Lookahead class
  - » Policies can be relatively complex to compute.
  - » Well suited to problems with expensive experiments.
  - » Typically avoids tuning, but may require a prior.

Multiarmed bandit problems

- Notes:
  - » *Any* of the four classes of policies may be appropriate depending on the characteristics of the problem.
  - » Active learning arises in many applications, but is often overlooked.
  - » The "bandit" culture of coming up with problem variations should be inherited by other communities.
  - » Bandit researchers often focus on good but not optimal policies (e.g. UCB policies) with good characteristics (e.g. robust across a wide range of distributions).

MOLTE

- Modular, optimal learning testing environment
  - » Matlab-based environment with modular library of problems and algorithms, each in its own .m file.
  - » User specifies in a spreadsheet which algorithms are run on which problems

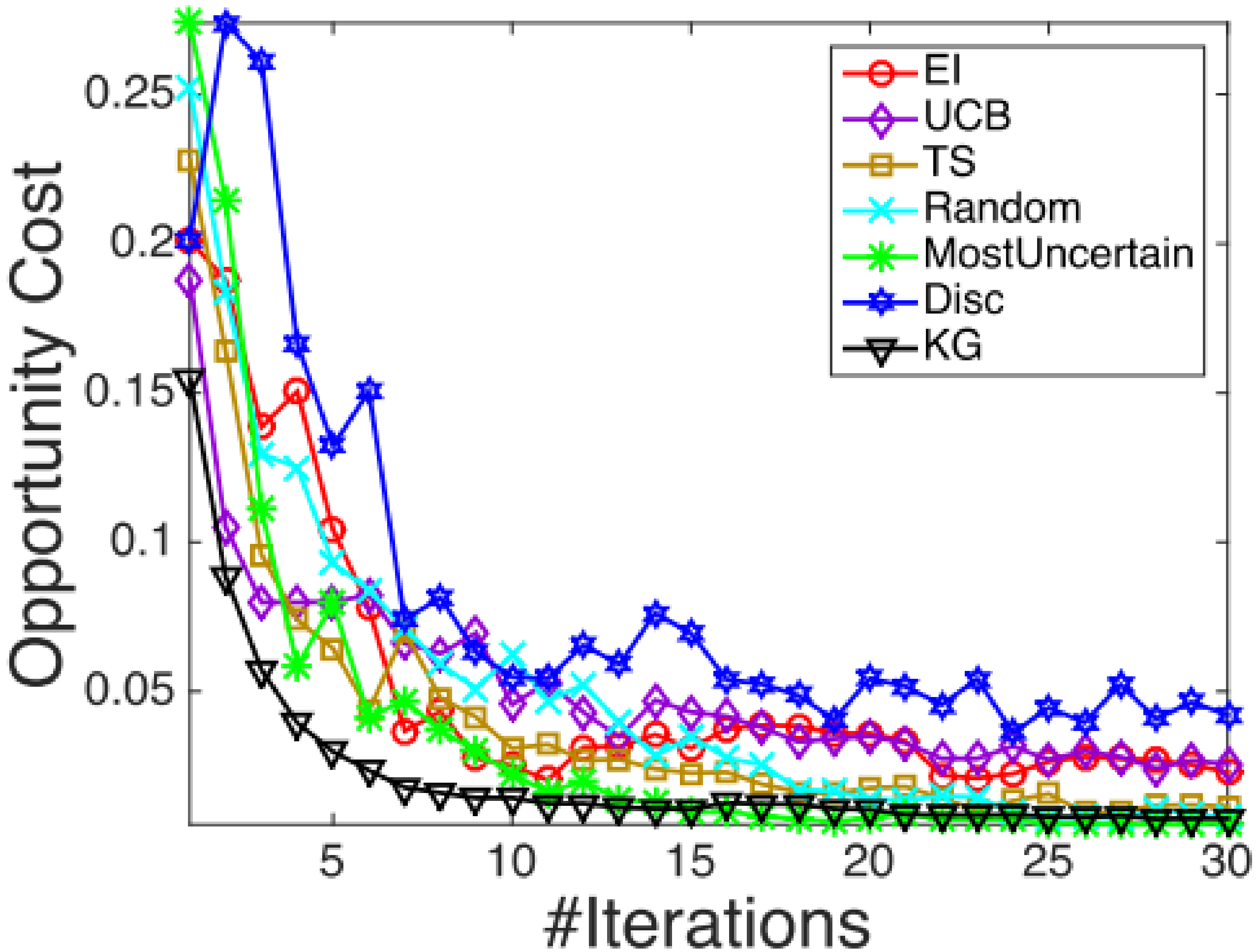


Problem class	Prior	Measurement Budget	Belief Model	Offline/ Online	Number of Policies				
PayloadD	MLE	0.2	independent	Offline	4	kriging	EXPL	IE(1.7)	Thompson Sampling
Branin	MLE	10	correlated	Online	4	OLkgcb	UCBEcb(*)	IE(2)	BayesUCB
Bubeck4	uninformative	5	independent	Online	4	OLKG	UCB	SR	UCBV
GPR	Default	0.3	correlated	Offline	4	kriging	kgcb	IE(*)	EXPT

http://www.castlelab.princeton.edu/software/

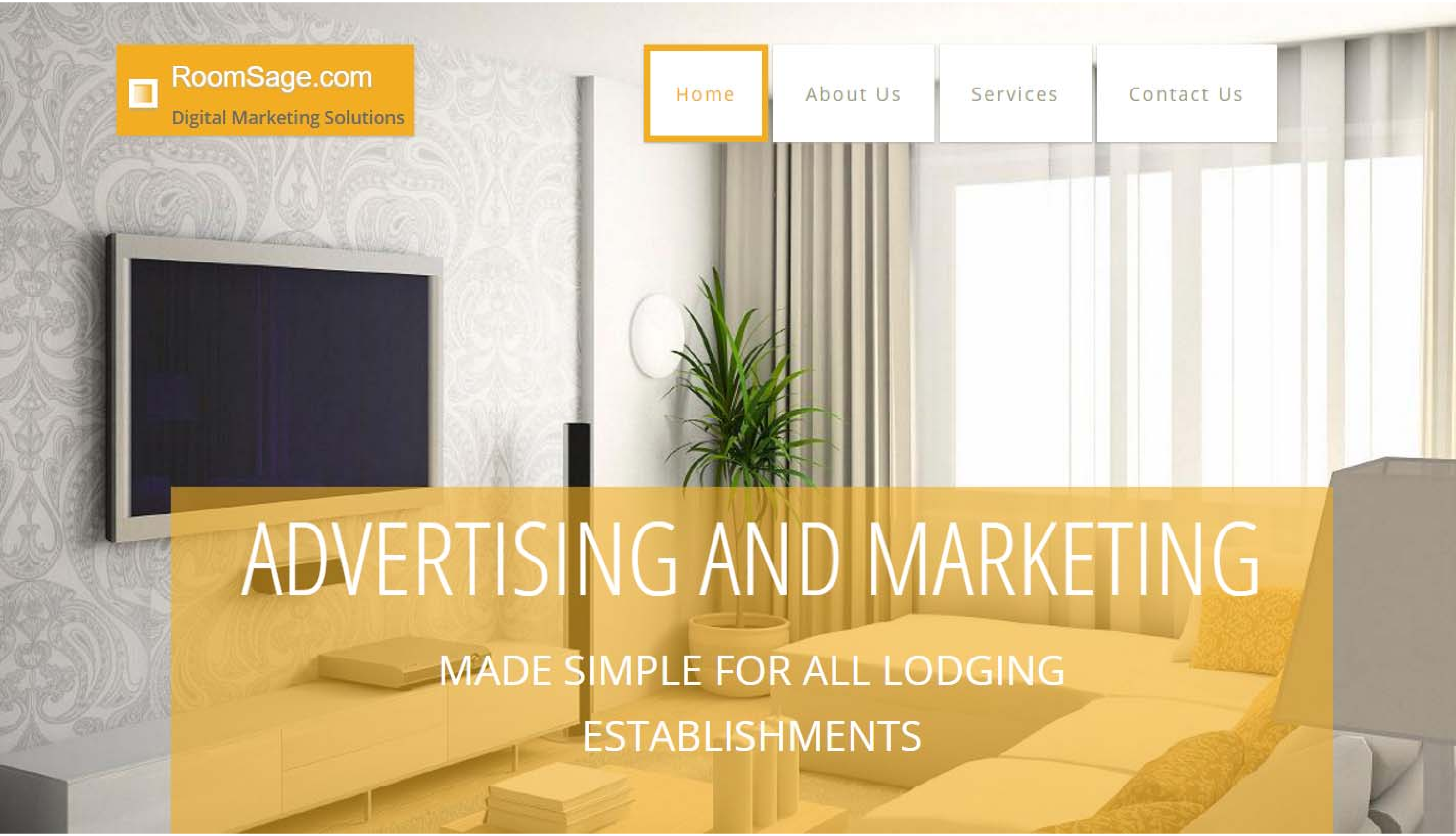
MOLTE

- Comparison on library problems



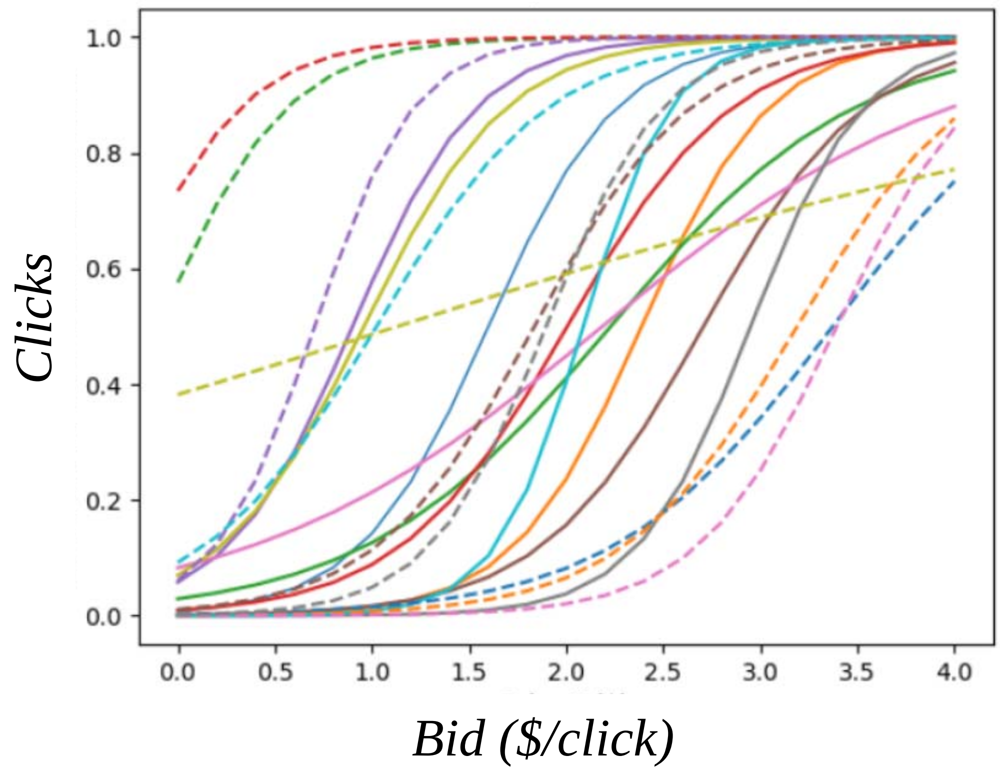
Princeton ad-click game

- In collaboration with Roomsage.com



Princeton ad-click game

- Learning the bid-response curve



  » Varies by hour of week

  » Response depends on location, age, gender, device

Princeton ad-click game

- The ad-click game:
  - » Learn the best policy for bidding for ads
  - » Bids compete in a simulated auction following the rules used by Google

Policy	profit
PresidentBidness_LA_1	10528
MaxBidder_LAPS_alpha	8439
PresidentBidness_PS_1	5553
Weebs_LA_EZPolicy	3458
MaxBidder_PS_alpha	2573
Weebs_LA_MetropolisHastings	1740
AKCB_LA_1	1471
pbchen_PS_s4real	790
BaoWang_PS_WeGo2	599
MnM_LAPS_M	219
MmegwaWagnerinterval_estimation	61
AKCB_PS_1	0
ohiustina_LA_3	0
ohiustina_PS_3	0
TnT_PS_M	0
ConnorDozie_PS	-7
pbchen_LA_s4real	-42
BaoWang_PS_WeGo	-54
ConnorDozie_LAPS	-1007
BreyerJohnson_LA_3	-1242
BreyerJohnson_PS_3	-7132
WagnerMMegwa_LAPS	-13344
tw5_PS	-27302

Thank you!

For more information, please visit:

http://www.castlelab.Princeton.edu

See "Courses" or the "jungle" webpages.