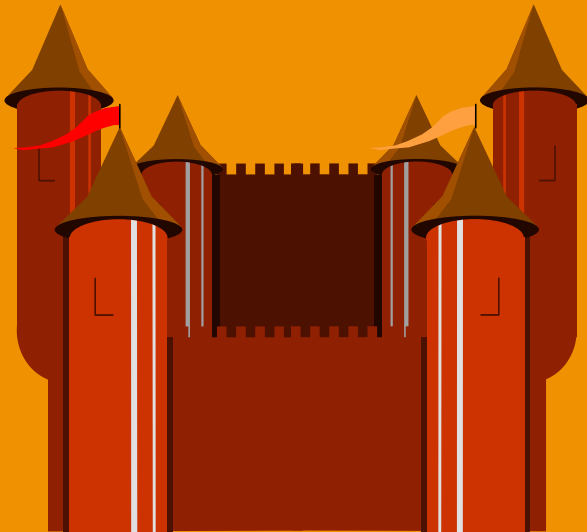


Adaptive learning algorithms for stochastic resource allocation

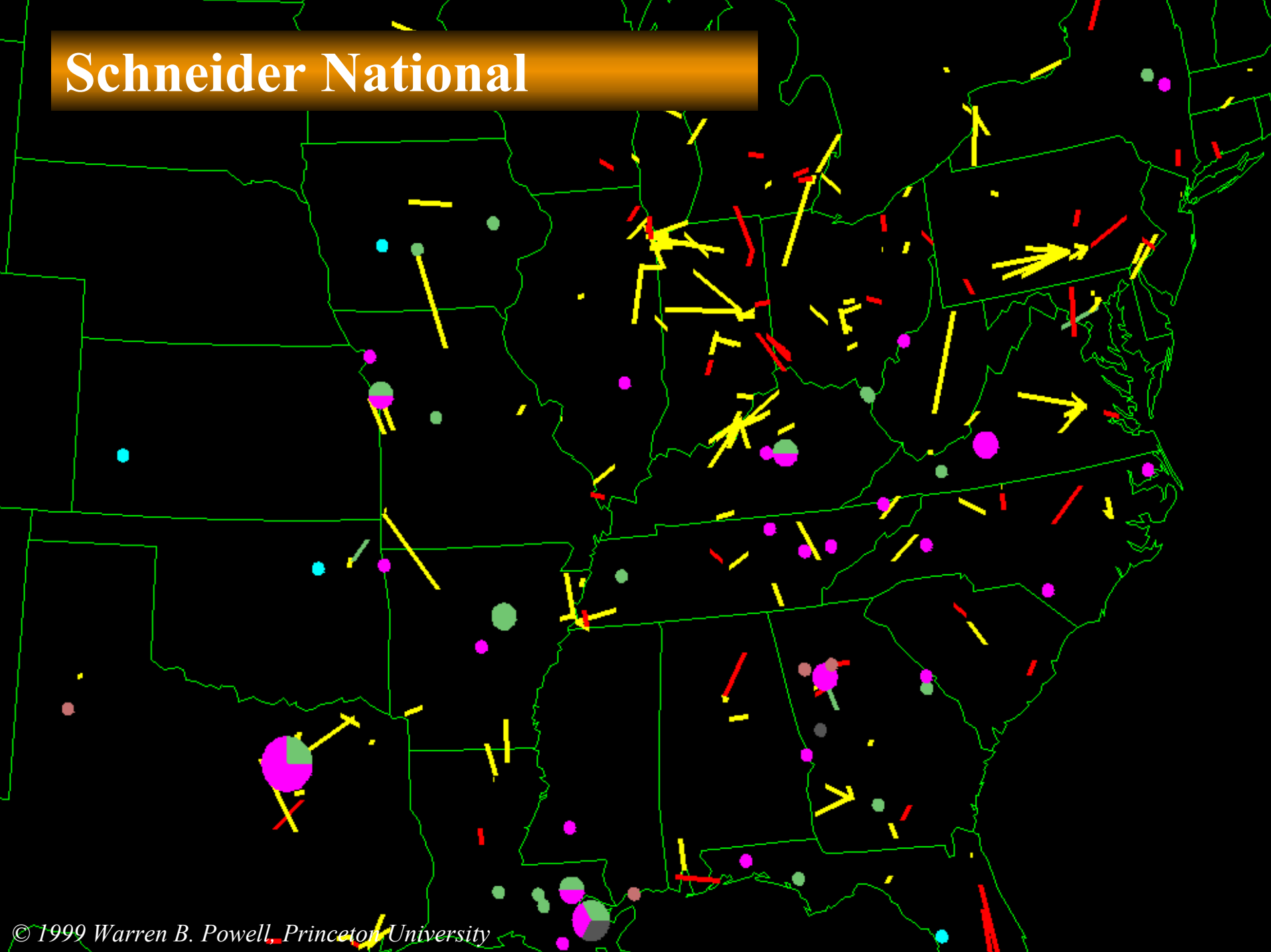
IMA Conference on
Supply Chain Management

September, 2002



Warren Powell
CASTLE Laboratory
Princeton University
<http://www.castlelab.princeton.edu>

Schneider National



Schneider National

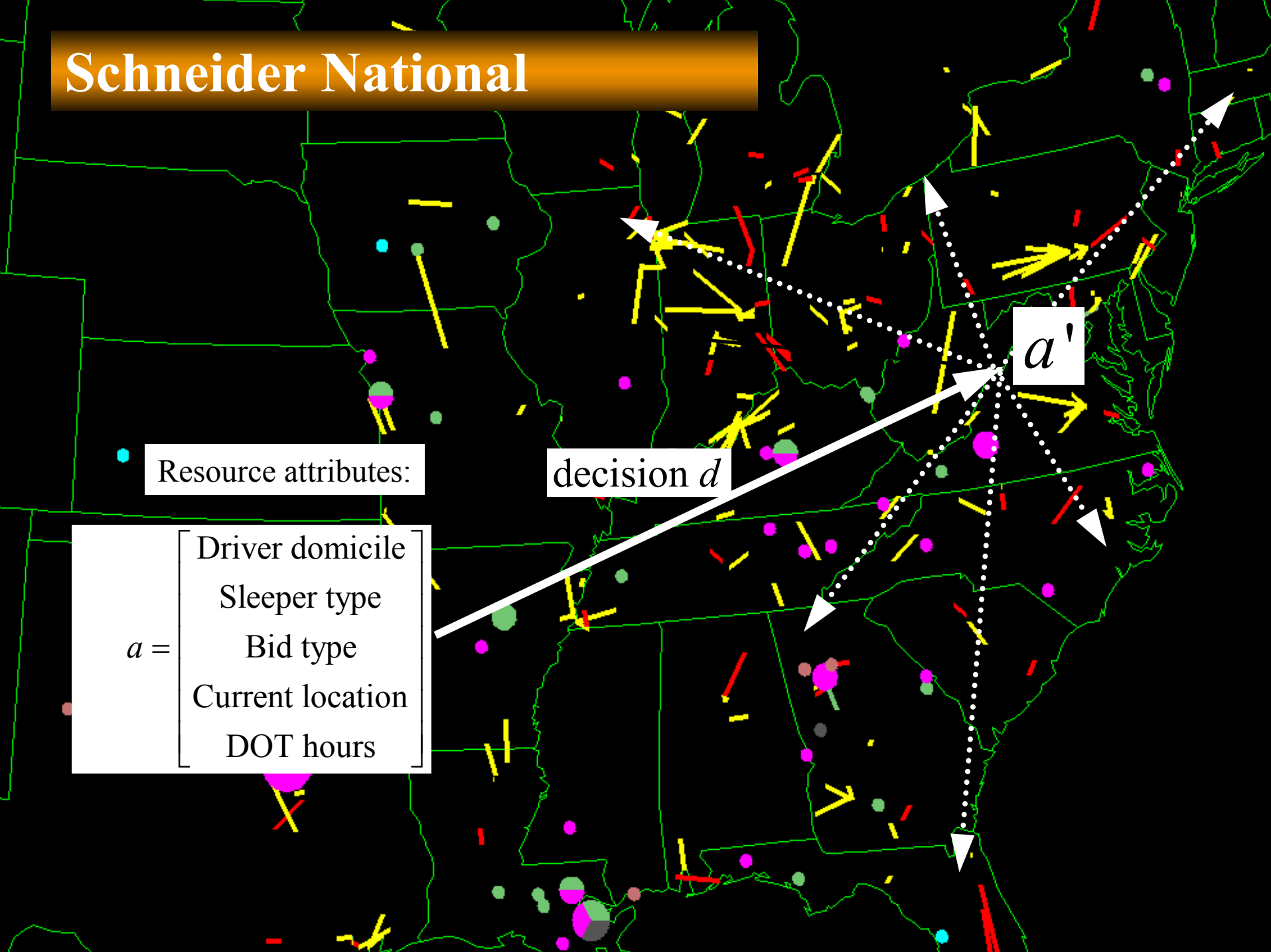
Resource attributes:

$a =$

Driver domicile
Sleeper type
Bid type
Current location
DOT hours

decision d

a'



Yellow Freight System

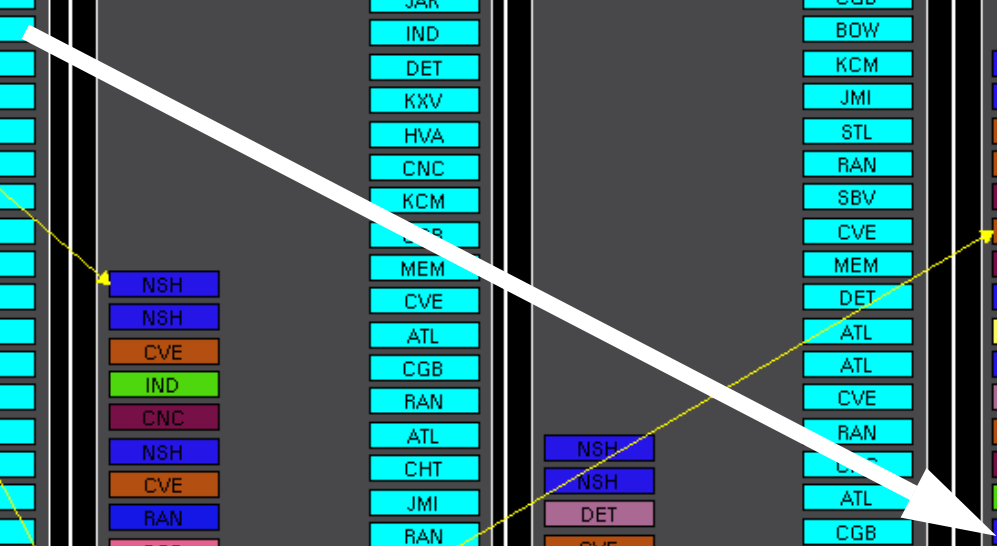


NSH	
NSH	ATL
NSH	JTC
CVE	CHT
NSH	KXV
IND	JMI
CNC	LOU
COL	KCM
CVE	BHM
RAN	CNC
COL	CGB
CVE	MEM
RAN	RAN
COL	CTI
NSH	ATL
RAN	RAN
CNC	MEM
CVE	MEM
COL	RAN
NSH	CVE
NSH	COL
CVE	CGB
CNC	MEM
CGB	CNC
CNC	KCM
JMI	MEM
DET	RAN
	JMI
	ATL
	COL
	RAN

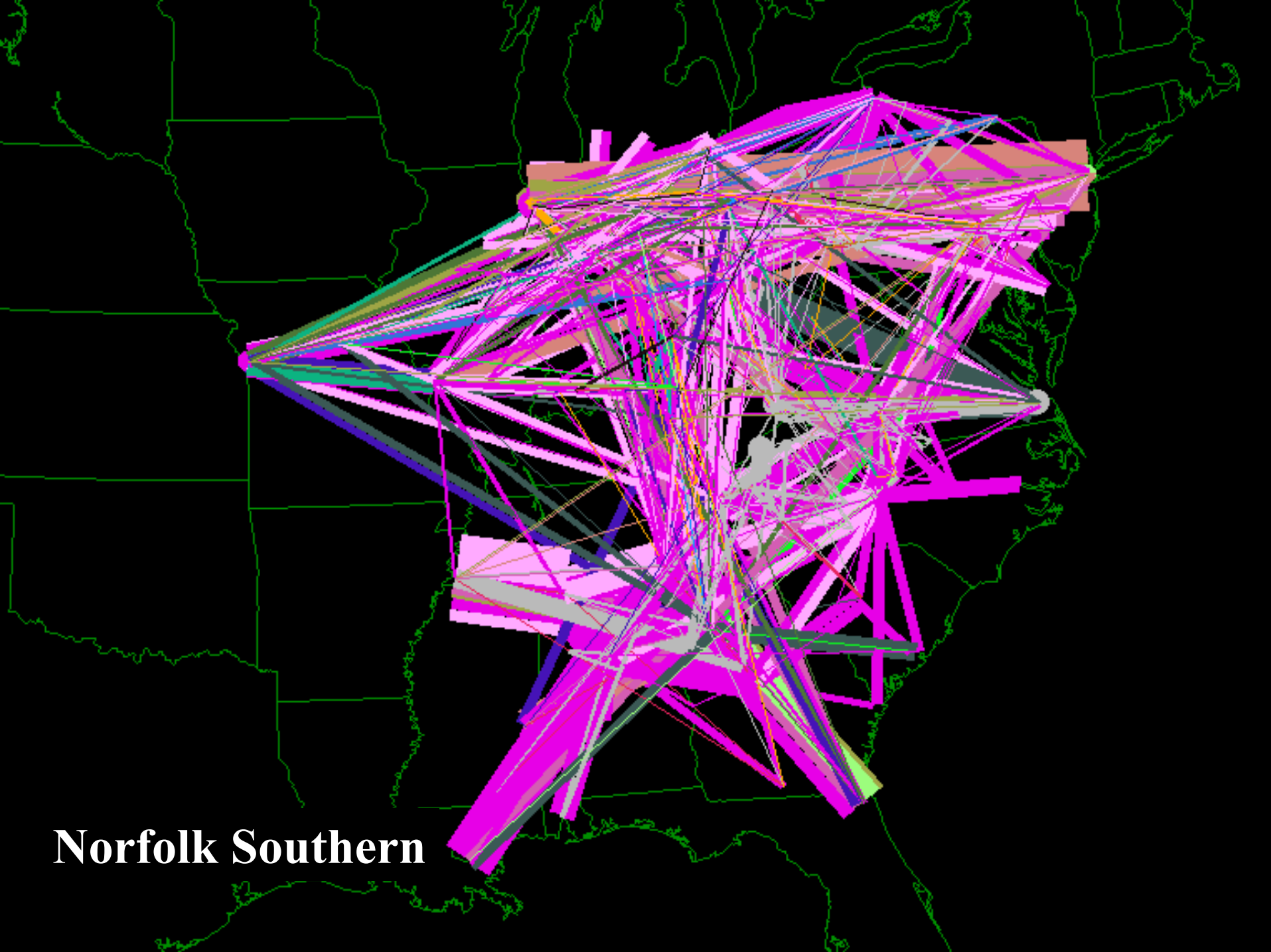
LOU	
COL	COL
COL	COL
NSH	
	STL
	JAK
	IND
	DET
	KXV
	HVA
	CNC
	KCM
	MEM
NSH	MEM
NSH	CVE
CVE	ATL
IND	CGB
CNC	RAN
NSH	ATL
CVE	CHT
RAN	JMI
CGB	RAN
COL	PHM
CNC	COL
GNR	RAN
CVE	RAN
NSH	CNC
CNC	MEM
NSH	CVE
IND	MEM
CVE	ATL
ATL	ATL
GNR	CGB
ATL	ATL
IND	MEM
MEM	CNC

NSH	
	COL
	DET
	EVL
	KXV
	CTI
	CGB
	BOW
	KCM
	JMI
	STL
	RAN
	SBV
	CVE
	MEM
	DET
	ATL
NSH	ATL
NSH	CVE
DET	RAN
CVE	ATL
CNC	CGB
IND	RAN
COL	ATL
NSH	CGB
DET	RAN
CVE	MEM
CNC	ATL
IND	MEM
COL	ATL
NSH	MEM
DET	ATL
CVE	MEM
CNC	ATL
CVE	MEM
MEM	KCM
IND	CNC

NSH	
	CTI
	JTC
	CKE
	LXK
	LOU
	BHM
	BOW
NSH	ATL
NSH	CTI
CVE	CVE
CVE	CNC
CNC	CGB
CVE	KCM
CNC	JMI
NSH	MEM
GNR	HVA
NSH	CHT
DET	CGB
CVE	JAK
CNC	RAN
IND	ATL
NSH	KCM
ATL	KXV
COL	RAN
ATL	JMI
NSH	CNC
IND	MEM
RAN	RAN
ATL	ATL
MEM	RAN
MBK	RAN
	MEM
	CTI
	CNC
	RAN
	CNC







Norfolk Southern



BNSF Coal Business Unit

Powder River & Illinois Coal Train Information

Current	Today's	
Loadings 6,808	Expectation 42	
64	42	
Yesterday's	MTD	January
Loadings	Actual	Train
46	42.71	40.66

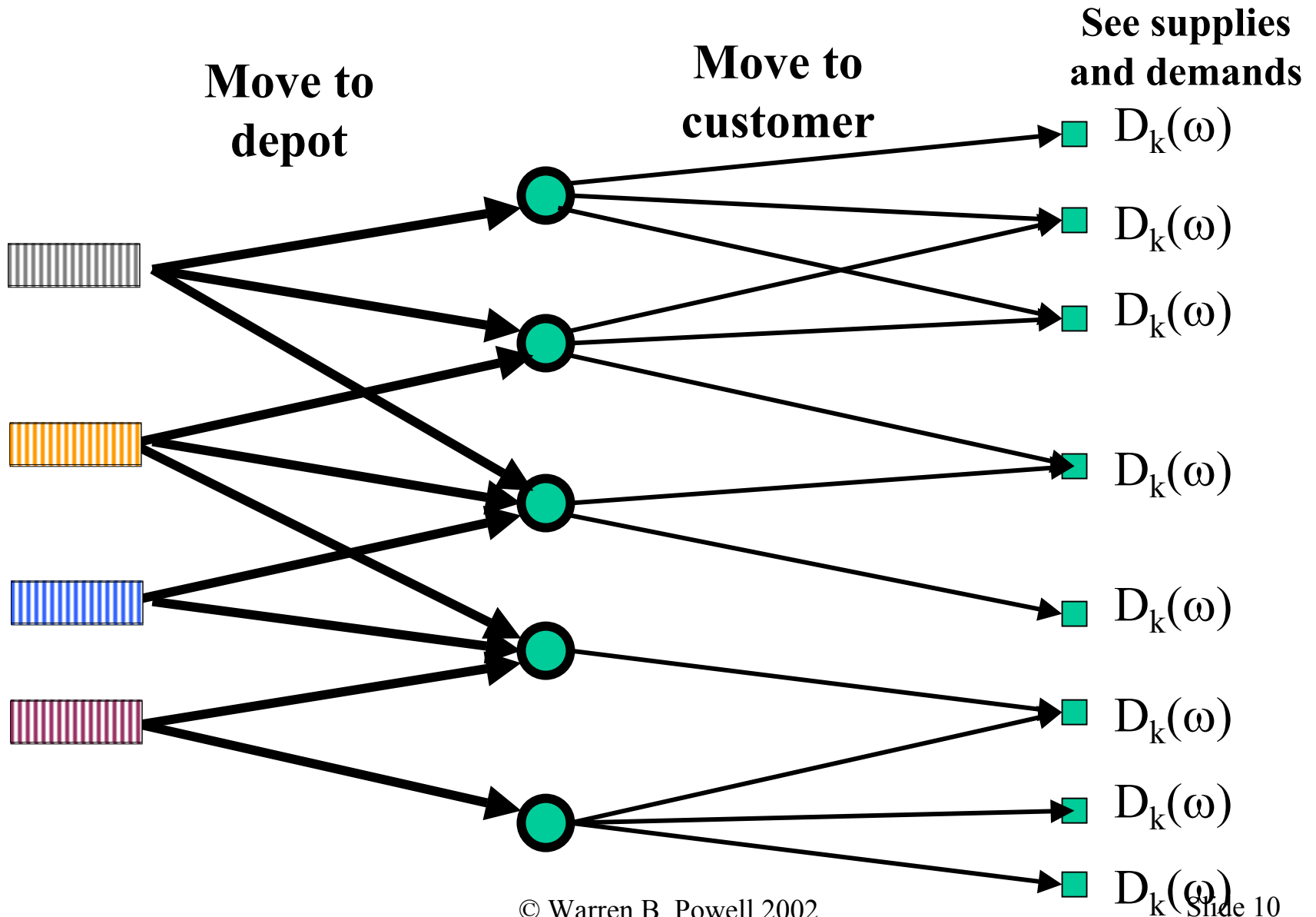


**Industrial Products
Service Summary**

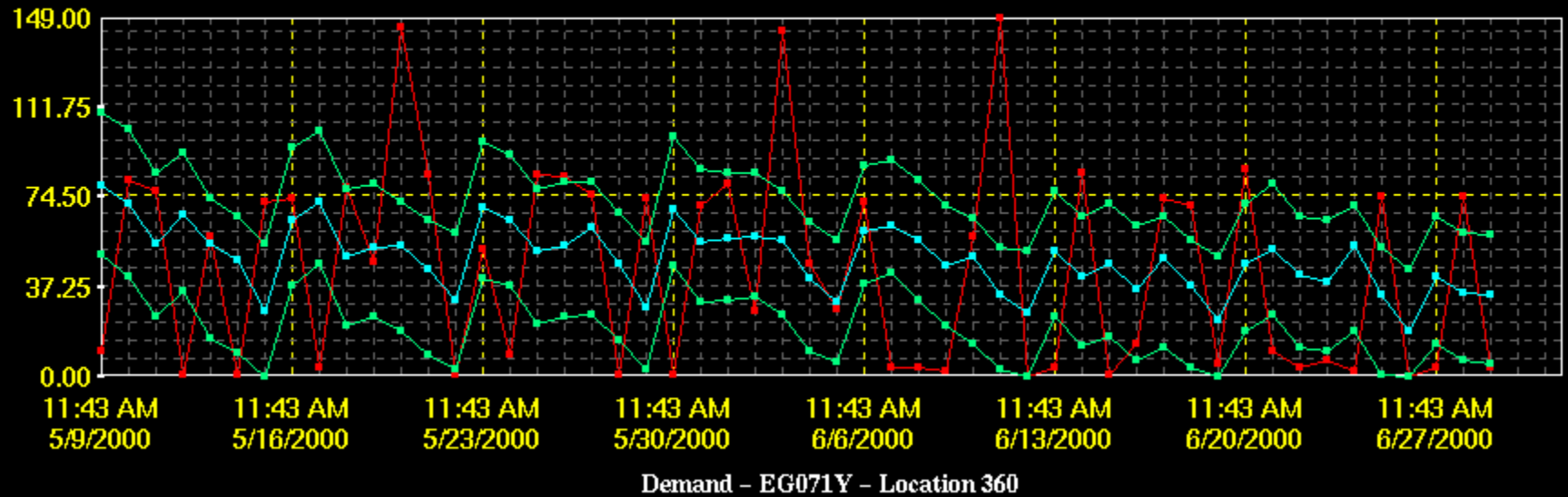
By Business Unit

47
46
45
44
43
42
41
40
39
38
37
36
35
34
33
32
31
30
29
28
27
26
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1

Distribution under uncertainty

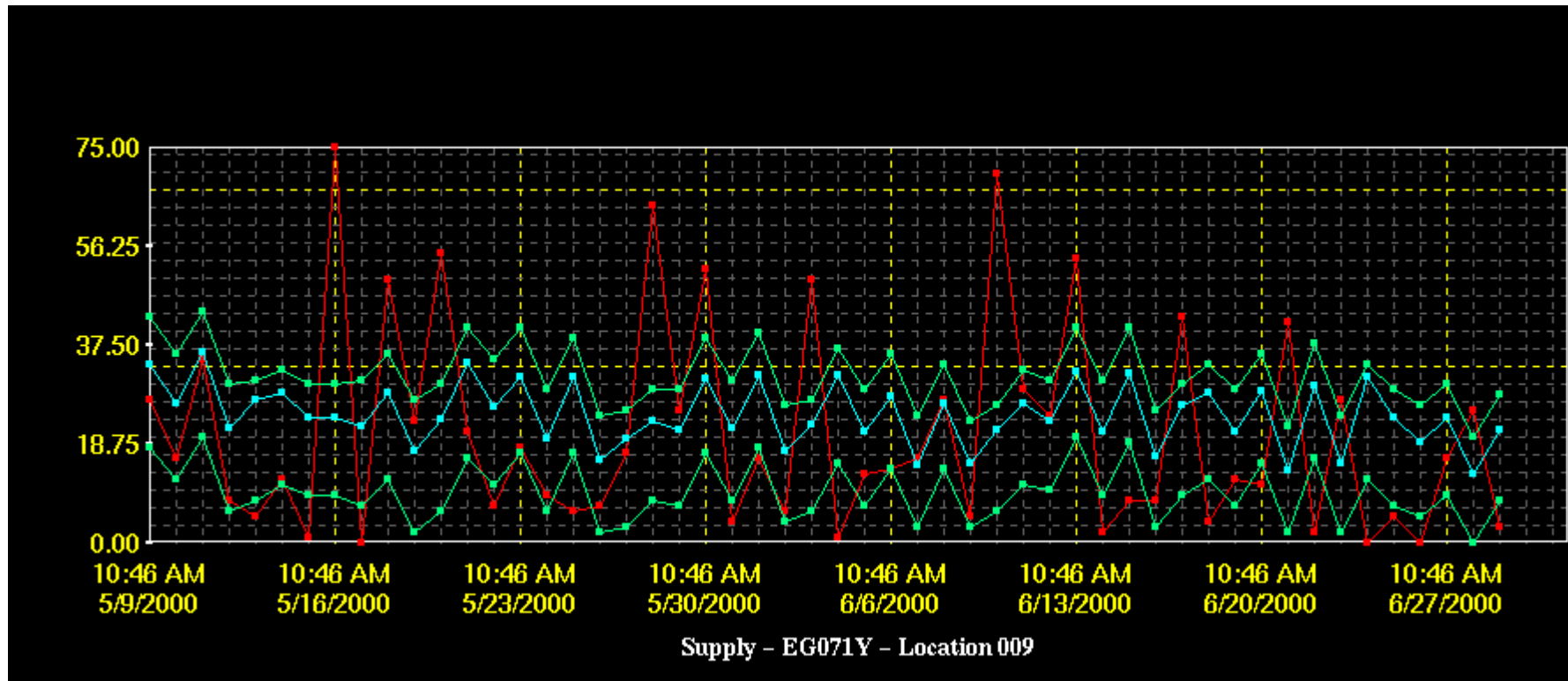


Car demand forecasting



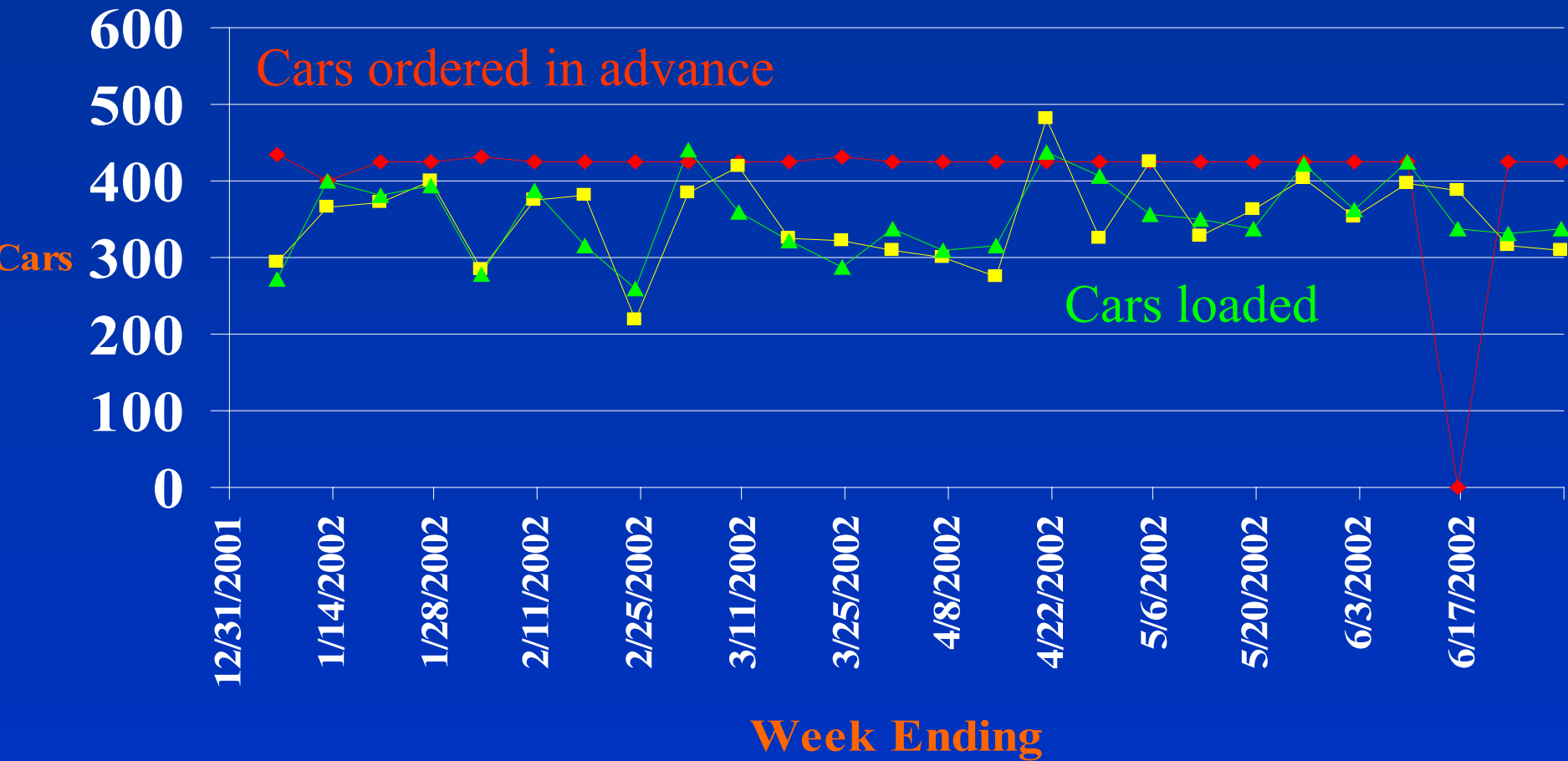
Actual vs. predicted car demands

Car supply forecasting



Actual vs. predicted car supplies

Cars ordered and the cars actually loaded



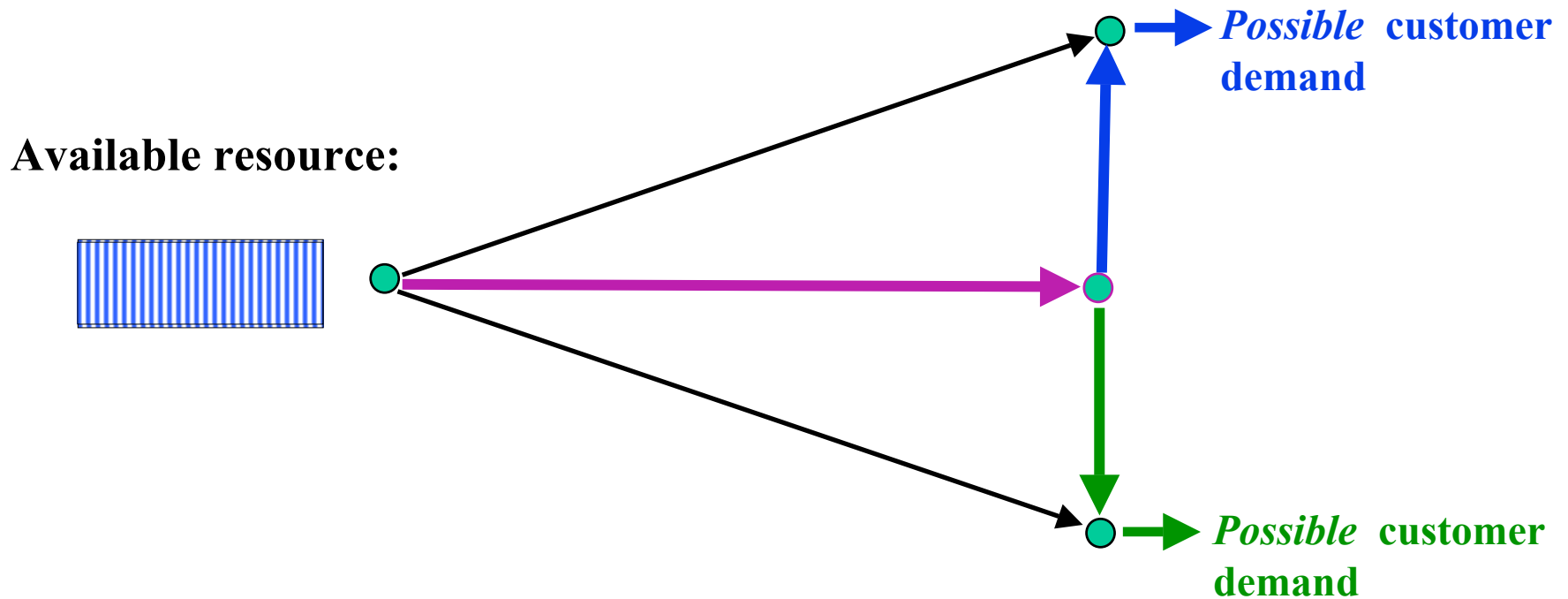
◆ Booked Orders

■ Cars Supplied

▲ Cars Loaded

The importance of uncertainty

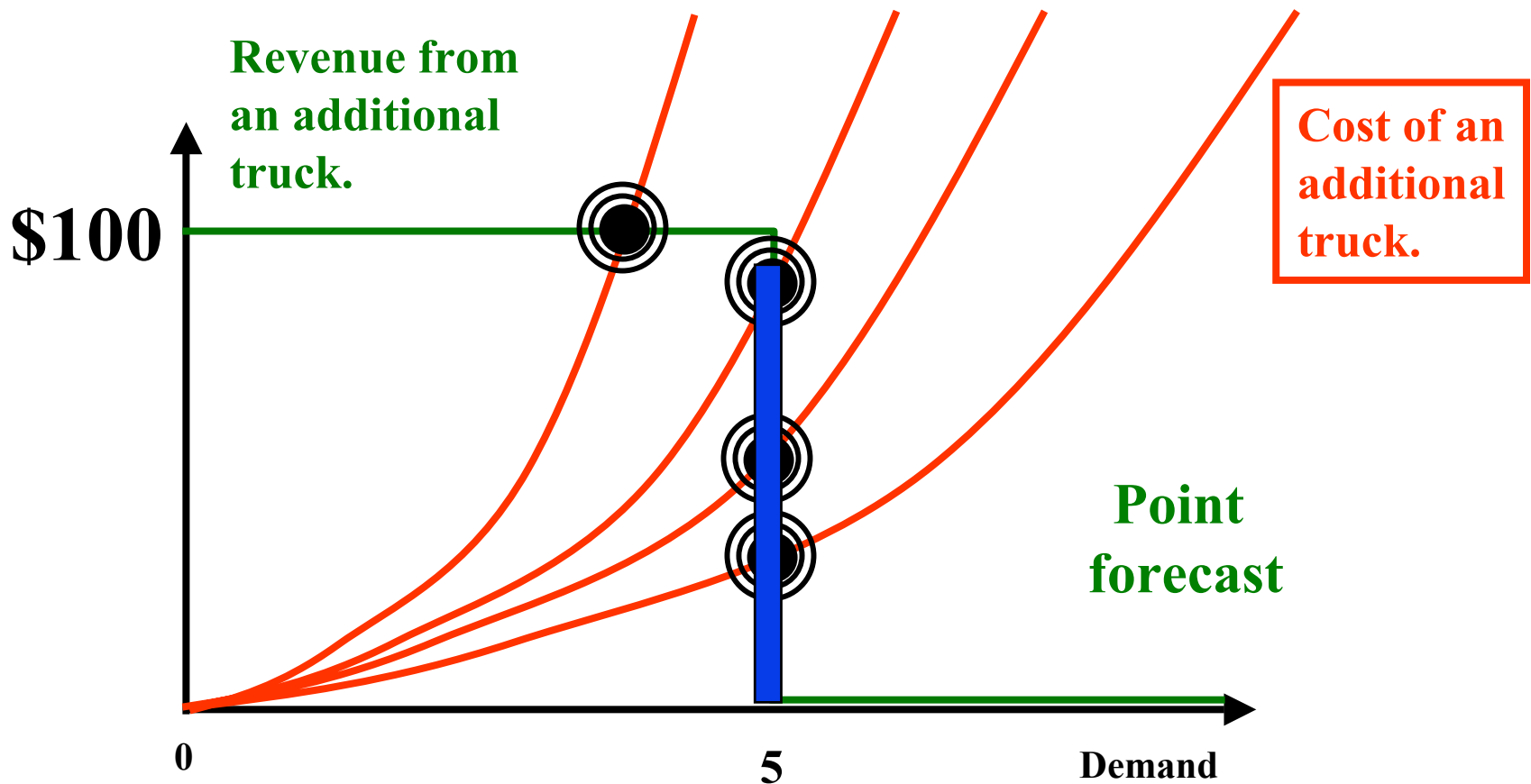
- Why do stochastic models work?
 - » Robust resource allocation:



The importance of uncertainty

■ Why do stochastic models work?

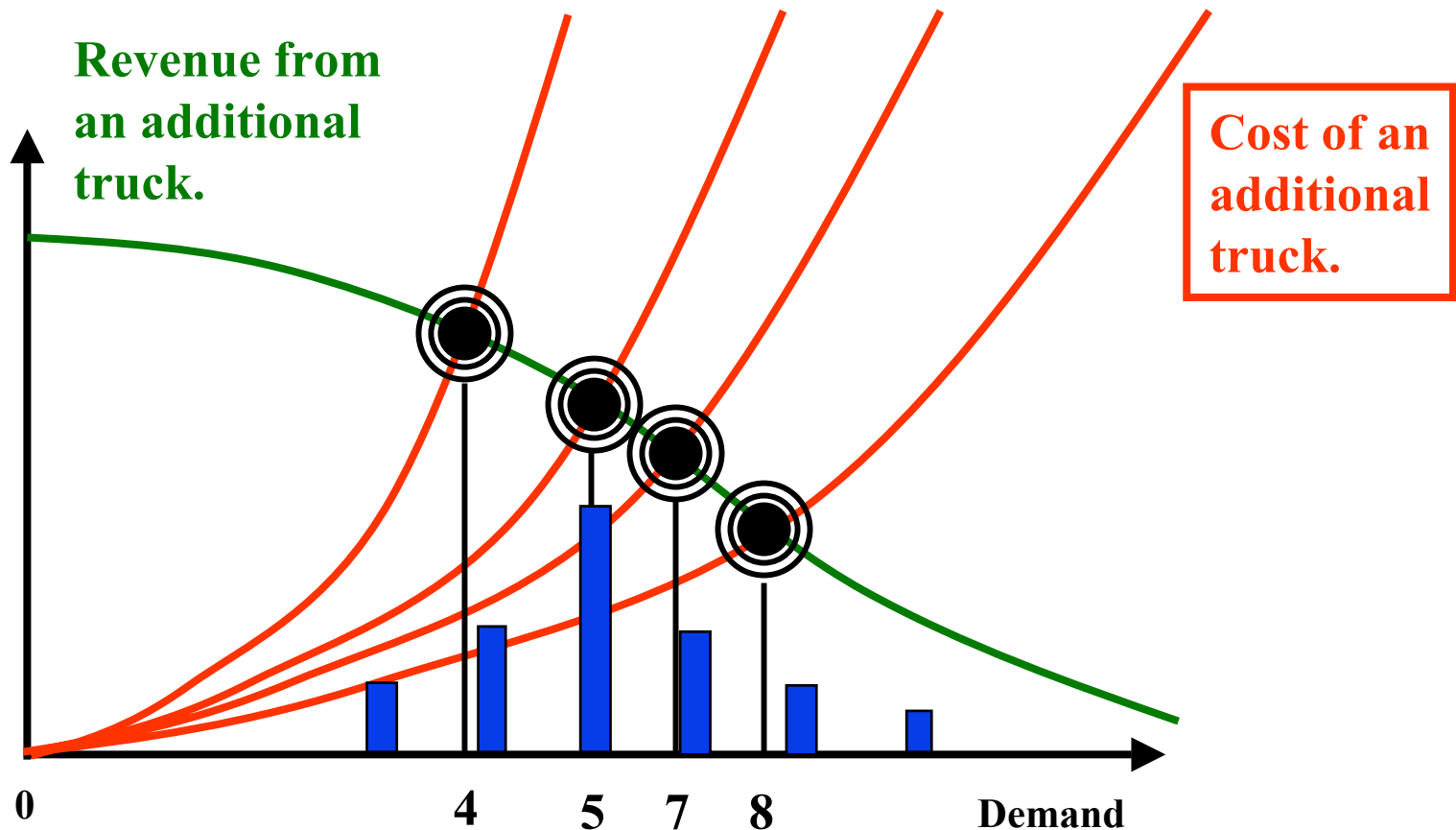
» The newsvendor effect:



The importance of uncertainty

■ Why do stochastic models work?

» The newsvendor effect:



The importance of uncertainty

■ Questions:

» Physical:

- How should I allocate my resources given the information that I have?

» Financial:

- What is the profitability of moving freight in a traffic lane, given the geographic and temporal patterns of demand?

» Informational:

- What is the cost of customers who book orders at the last minute?
- What is the value of advance information?

Outline

- Solution frameworks for stochastic optimization

Optimization frameworks

■ The objective function:

» Deterministic formulation:

$$\max_{x \in X} \sum_{t=0}^T c_t(x_t)$$

» Stochastic formulation:

$$\max_{\pi \in \Pi} E \left\{ \sum_{t=0}^T c_t(X_t^\pi) \right\}$$

Optimization frameworks

Multistage problems are typically solved as sequences of two-stage problems of the general form:

$$\min_{x \in \mathcal{X}} \left[c_t x_t + E \{ V(x_t, \omega) \} \right]$$

The diagram illustrates the decomposition of the multistage problem. The term $c_t x_t$ is circled in red and labeled "This period" with a red arrow. The term $E \{ V(x_t, \omega) \}$ is also circled in red and labeled "Next period" with a red arrow. The entire expression is enclosed in a larger red oval.

This equivalent to the classical problem:

$$\min_{x \in \mathcal{X}} E f(x, \omega)$$

A variety of methods have been proposed to solve problems of this form. What makes our problems hard is the mixture of uncertainty and integrality.

Optimization frameworks

■ Scenario methods:

Choose a sample $\hat{\Omega}_{t+1} \subset \Omega_{t+1}$ and solve:

$$x^{n+1} = \arg \max_{x_t \in \mathcal{X}, x_{t+1} \in \mathcal{X}_{t+1}} c_t x_t + \sum_{\hat{\omega}_{t+1} \in \hat{\Omega}_{t+1}} \hat{p}(\hat{\omega}_{t+1}) c_{t+1} x_{t+1}(\hat{\omega}_{t+1})$$

- Typically produces very large linear programs.
- LP relaxation is highly fractional – obtaining integer solutions is difficult (see Laporte and Louveaux).

Optimization frameworks

■ Nested Benders decomposition:

Choose a sample $\hat{\Omega}_{t+1} \subset \Omega_{t+1}$ and solve:

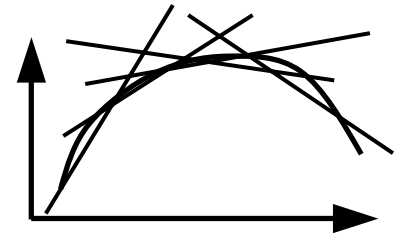
$$x^{n+1} = \arg \max_{x_t \in \mathcal{X}_t, x_{t+1} \in \mathcal{X}_{t+1}} c_t x_t + z_{t+1}$$

subject to:

$$z_{t+1} + \beta(\hat{\omega}_{t+1})x_t \leq \alpha(\hat{\omega}_{t+1}) \quad \text{for all } \hat{\omega} \in \hat{\Omega}_{t+1}$$

where $\beta(\hat{\omega}_{t+1})$ and $\alpha(\hat{\omega}_{t+1})$ are found from solving the dual for problem $t + 1$.

- Rate of convergence is very slow.
- Produces highly fractional solutions.



Optimization frameworks

■ Approximate (discrete) dynamic programming

Choose a sample $\hat{\Omega}_{t+1} \subset \Omega_{t+1}$ and solve:

$$x^{n+1} = \arg \max_{x_t \in \mathcal{X}} c_t(S_t, x_t) + \sum_{\hat{\omega}_{t+1} \in \hat{\Omega}_{t+1}} \hat{p}(\hat{\omega}_{t+1}) \hat{V}_{t+1}(S_{t+1}(\hat{\omega}_{t+1}))$$

- Discrete representation of states and actions. High dimensional action spaces are exponentially large.

Optimization frameworks

■ Stochastic gradient methods:

$$\hat{x}^{n+1} = x^n + \alpha^n \nabla f(x^n, \omega^n) \quad \text{Improvement step}$$

$$x^{n+1} = \Pi_X(\hat{x}^{n+1}) \quad \text{Projection back onto feasible region}$$

- Smoothing produces fractional solutions.

Optimization frameworks

■ Our basic strategy:

$$x^{n+1} = \arg \max_{x \in \mathcal{X}} c_t x_t + \hat{f}_t^n(R_t(x))$$

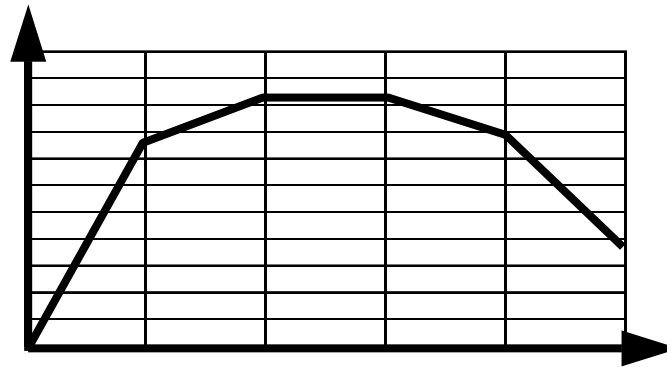
Piecewise linear,
separable approximation

where

$$R_t(x) = A_t x_t \quad \text{Lower dimensional state vector}$$

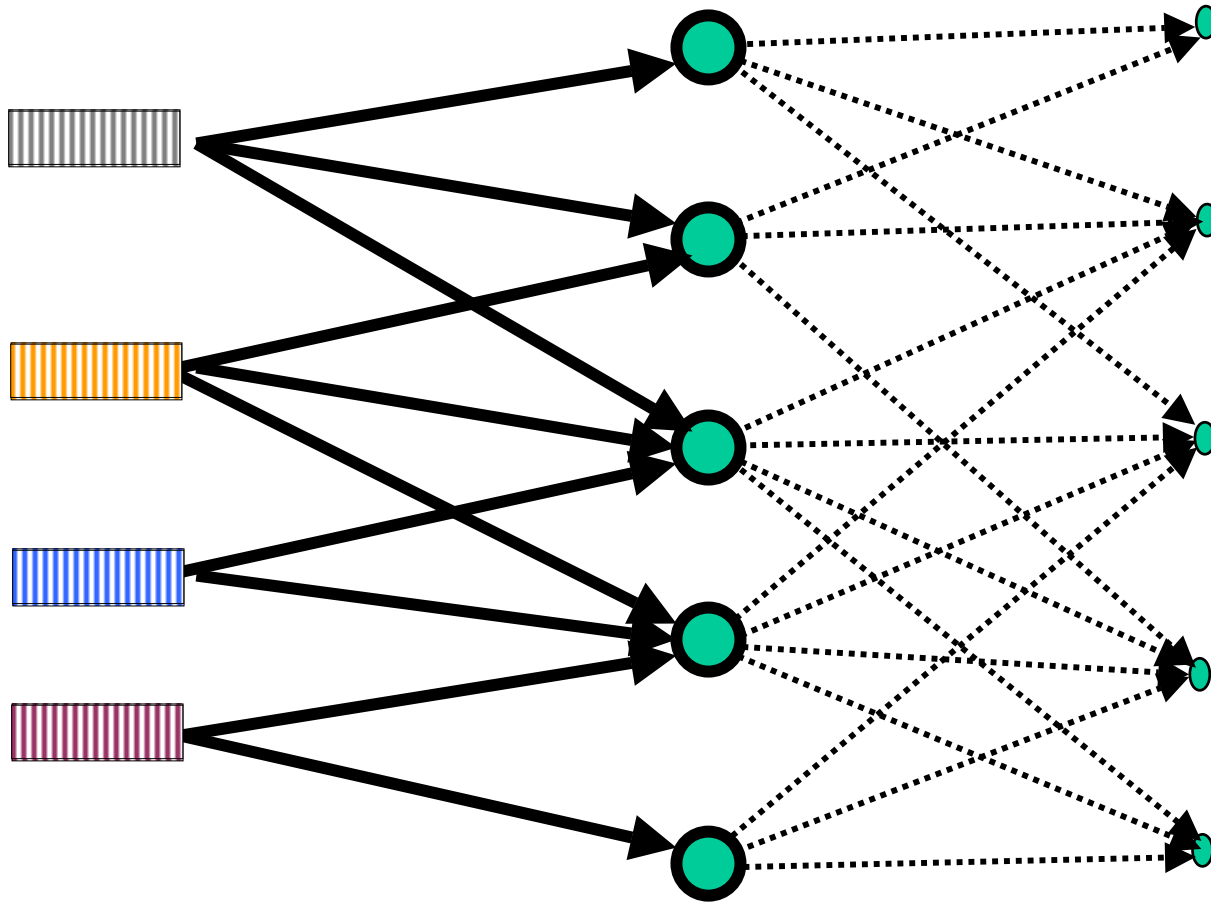
$$\hat{f}_t^n(R_t(x)) = \sum_{a \in \mathcal{A}} \hat{f}_{a,t}^n(R_{a,t}(x))$$

$$\hat{f}_t^n(R_t(x))$$



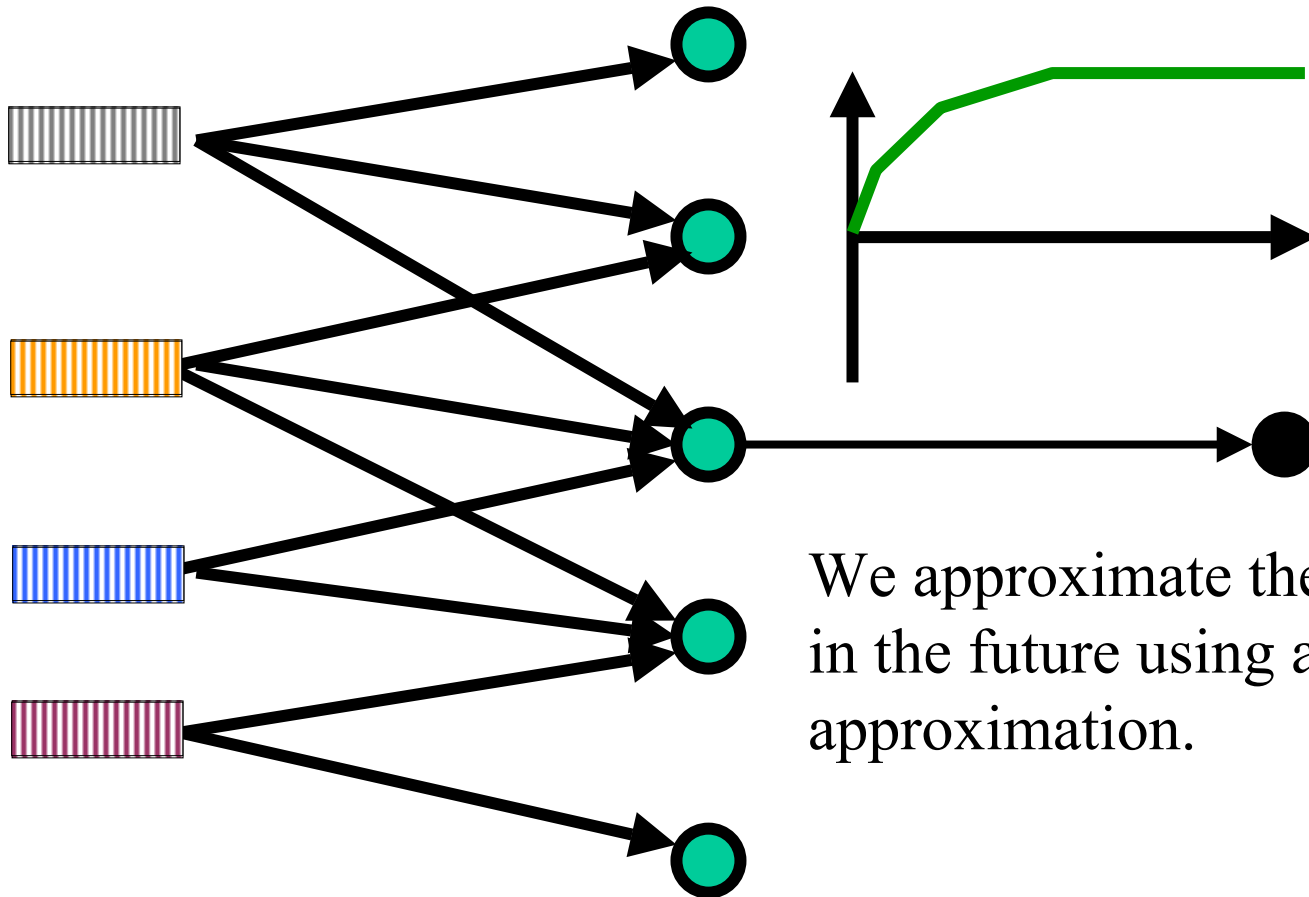
Optimization frameworks

- Two-stage resource allocation under uncertainty



Optimization frameworks

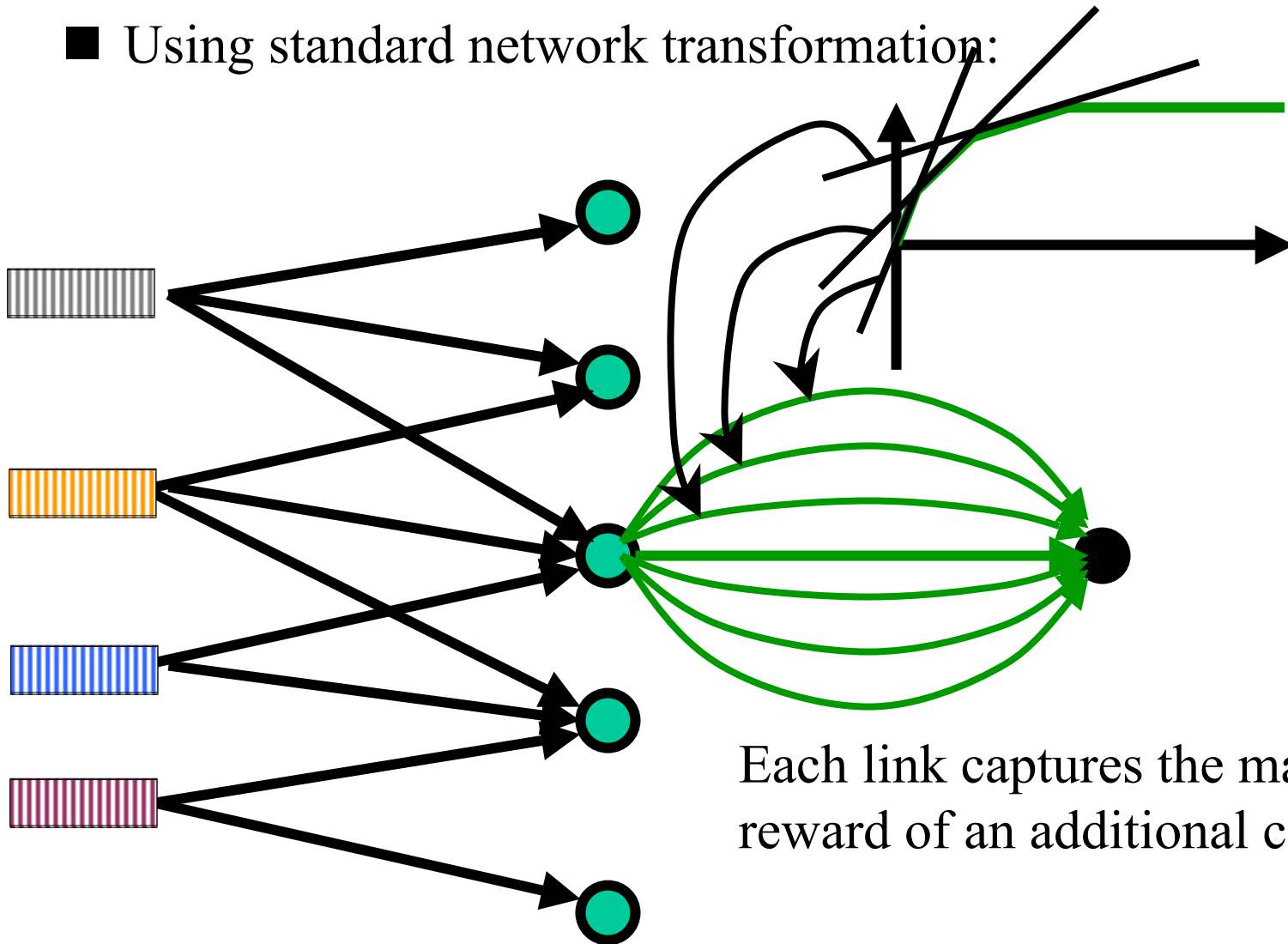
- Approximate the second stage using separable, piecewise linear function



We approximate the value of cars in the future using a separable approximation.

Optimization frameworks

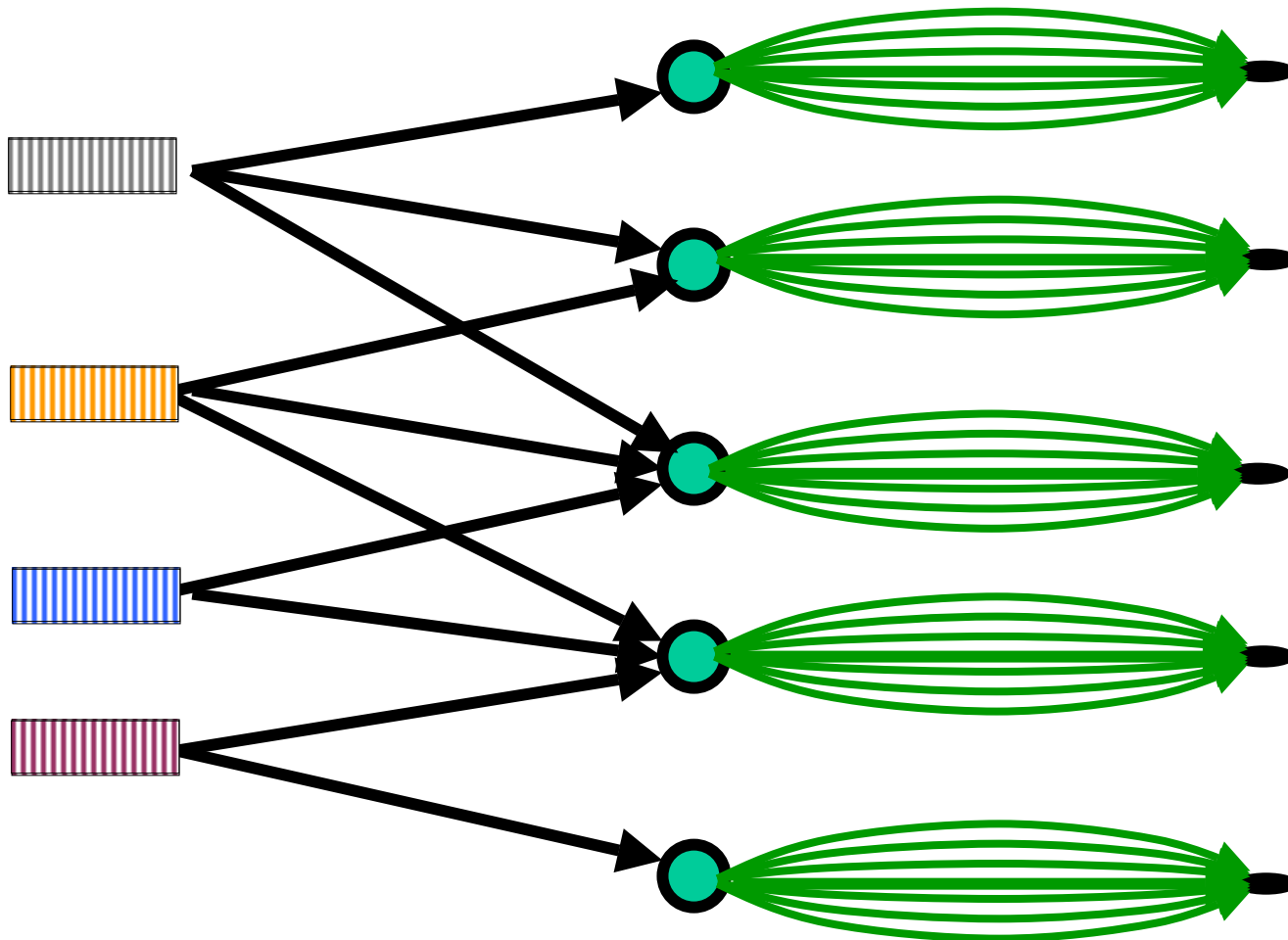
- Using standard network transformation:



Each link captures the marginal reward of an additional car.

Optimization frameworks

- ... to create sequences of *separable* approximations:

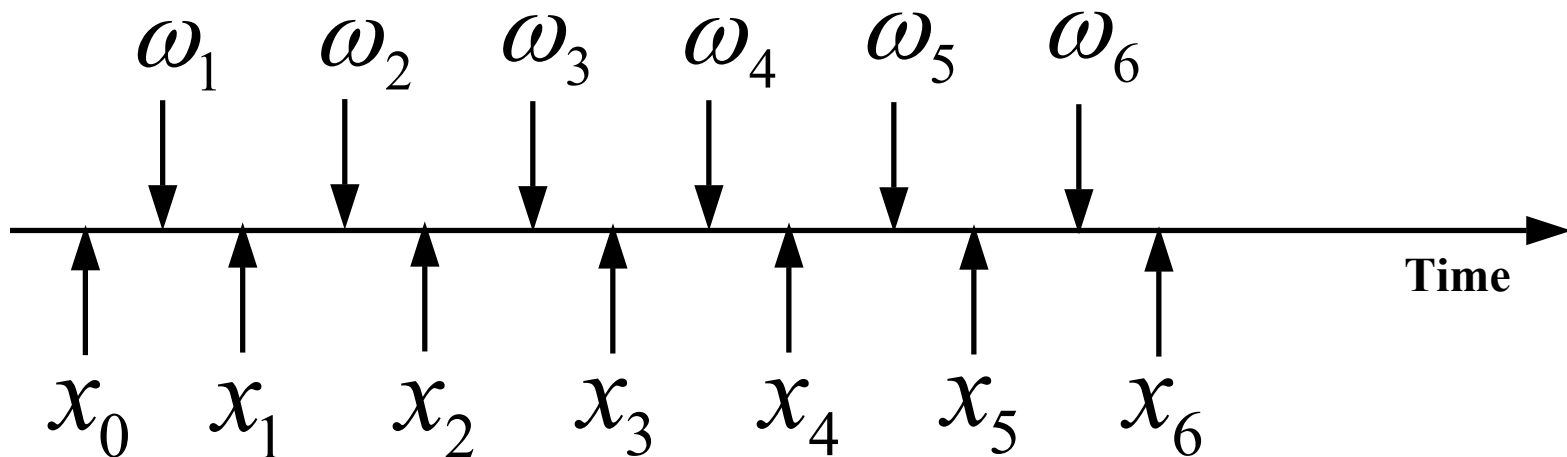


Outline

- An algorithmic strategy using adaptive dynamic programming

Adaptive dynamic programming

- Systems evolve through a cycle of exogenous and endogenous information



Adaptive dynamic programming

- We can address the problem using the basic DP equation:

$$V_t(S_t) = \max_{x \in X} c_t(S_t, x_t) + E \{V_{t+1}(S_{t+1}) | S_t\}$$

Three curses

- Problem: ~~Curse~~ of dimensionality

State space

Outcome space

Action space (feasible region)

Adaptive dynamic programming

■ Approximation methodology:

We start with:

$$V_t(S_t) = \max_{x_t} c_t(S_t, x_t) + E \{ V_{t+1}(S_{t+1}) | S_t \}$$

Can't compute this!!!

We solve this for a sample realization:

$$V_t(S_t, \omega_{t+1}) = \max_{x_t} c_t(S_t, x_t) + V_{t+1}(S_{t+1}(\omega_{t+1}))$$

Can't find this!!!

Now substitute in function approximations:

$$\tilde{V}_t(S_t, \omega_{t+1}) = \max_{x_t} c_t(S_t, x_t) + \hat{V}_{t+1}(S_{t+1}(\omega_{t+1}))$$

Adaptive dynamic programming

■ Challenge:

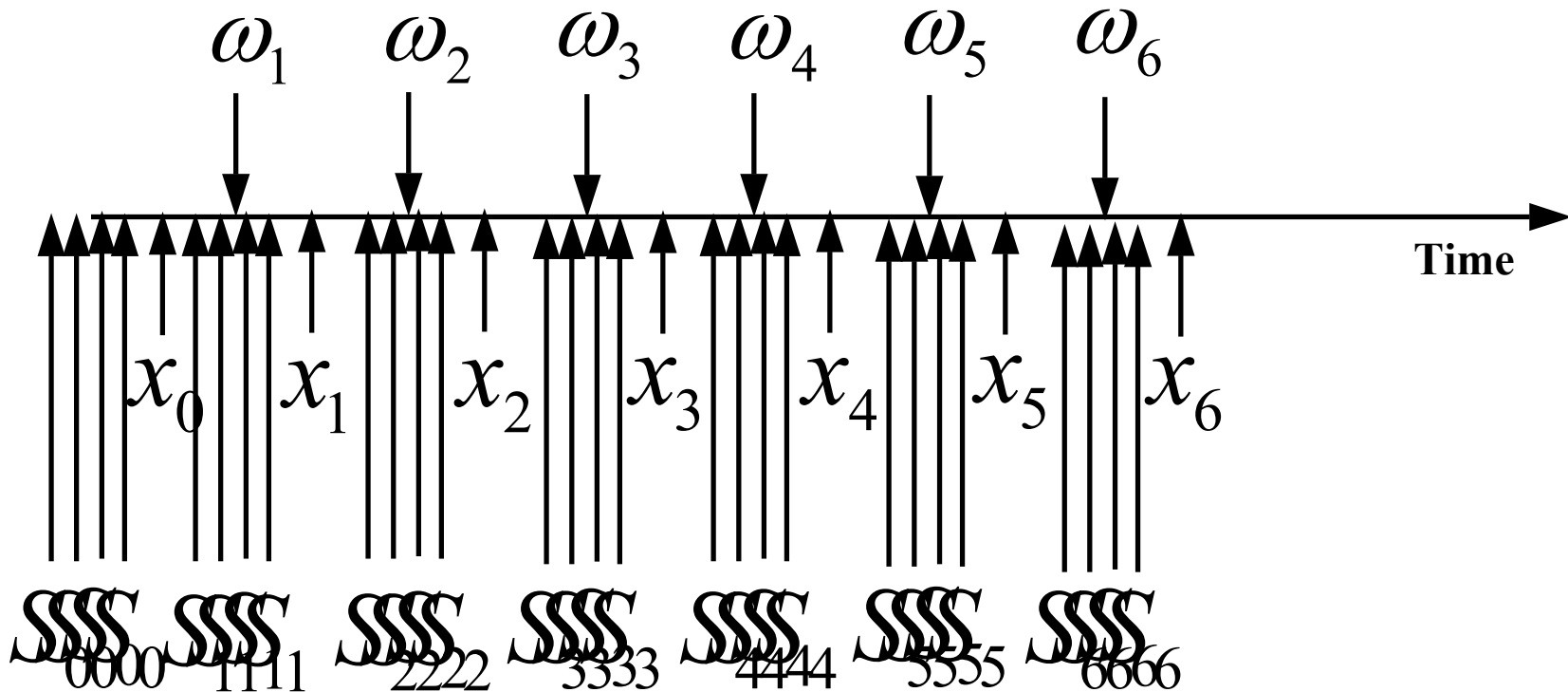
The optimal solution of:

$$\tilde{V}_t (S_t, \omega_{t+1}) = \max_{x_t} c_t (S_t, x_t) + \hat{V}_{t+1} (S_{t+1}(\omega_{t+1}))$$

is allowed to *see* ω_{t+1} ! This violates an information availability constraint.

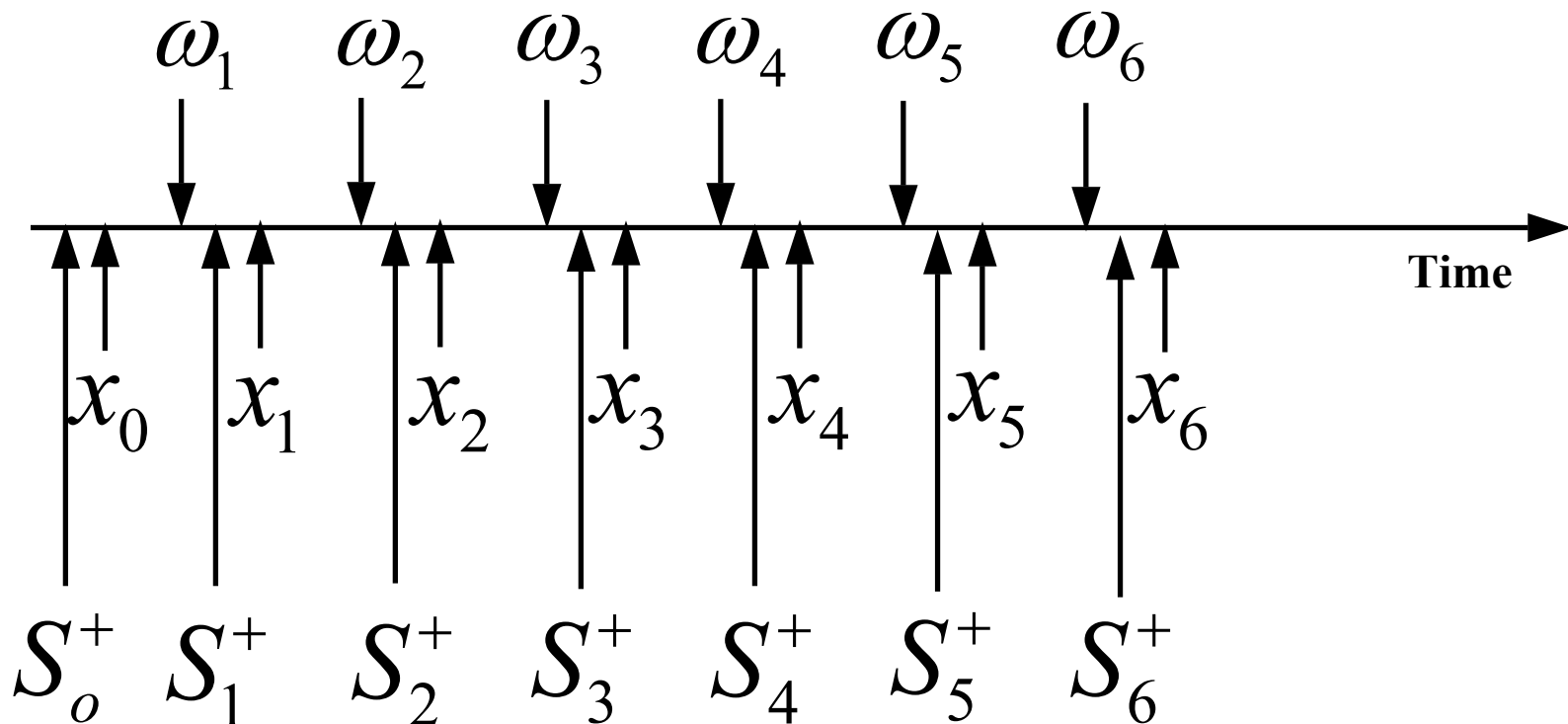
Adaptive dynamic programming

- Alternative: Change the definition of the state variable:



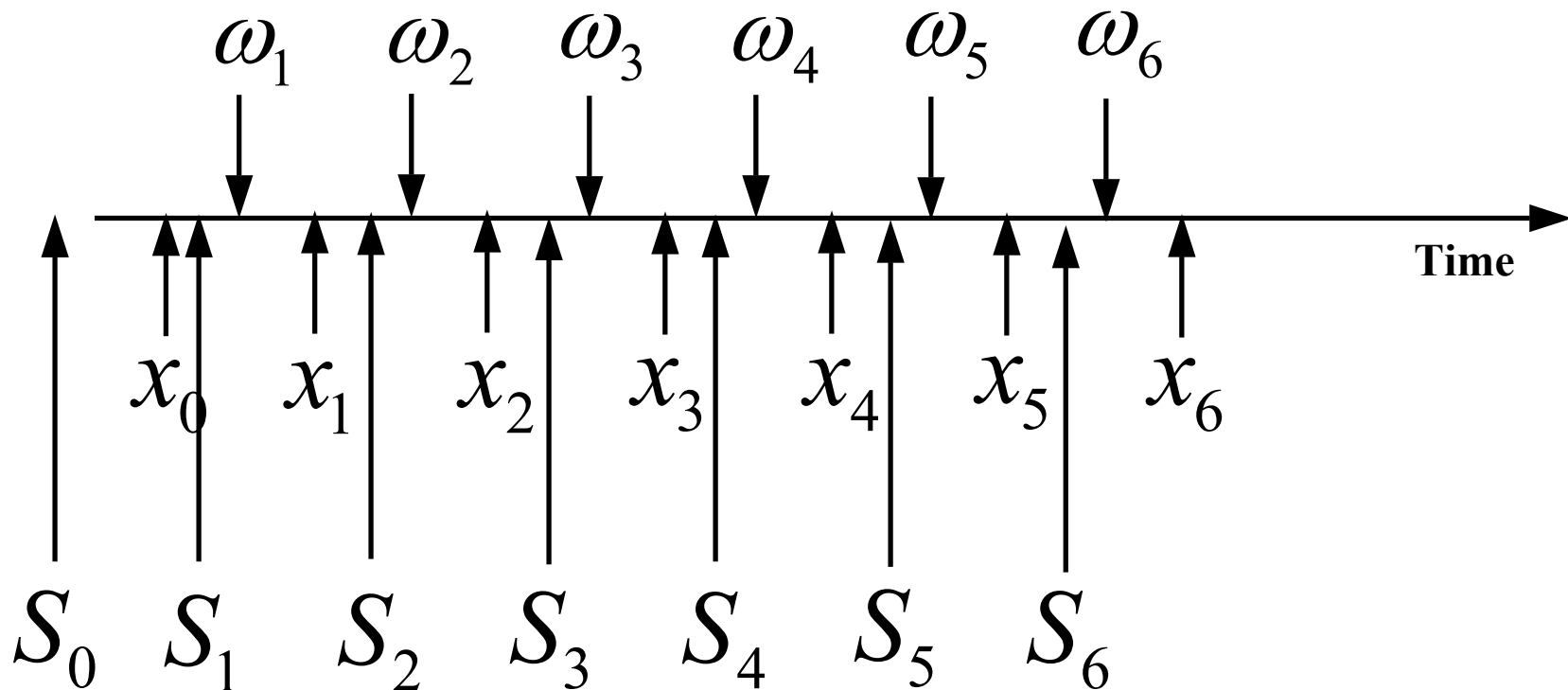
Adaptive dynamic programming

- We are going to call the state after ω_t is known the *complete* state variable, which we call S_t^+ .



Adaptive dynamic programming

- We call the state variable before ω_t is known the *incomplete* state variable, which we call S_t .



Adaptive dynamic programming

- Now our optimality equation looks like:

$$V_t(S_t) = E \left\{ \max_{x \in X} c_t(S_t, x_t) + V_{t+1}(S_{t+1}) \mid S_t \right\}$$

- And our conditional dynamic program looks like:

$$V_t(S_t, \omega_t) = \max_{x(\omega_t) \in X(\omega_t)} c_t(S_t, x_t(\omega_t), \omega_t) + V_{t+1}(S_{t+1}(\omega_t))$$

Adaptive dynamic programming

■ Approximation strategies

Linear (in the resource state):

$$\hat{V}_t(R_t) = \sum_{a \in \mathcal{A}} \hat{v}_{at} R_{at}$$

Nonlinear, separable:

$$\hat{V}_t(R_t) = \sum_{a \in \mathcal{A}} \hat{V}_{at}(R_{at})$$

Adaptive dynamic programming

- Solving the conditional problem using a functional approximation is usually pretty easy:

$$\max_{x_t} c_t(R_t, x_t, \omega_t) + \hat{V}_{t+1}(R_{t+1}(\omega_t))$$

subject to:

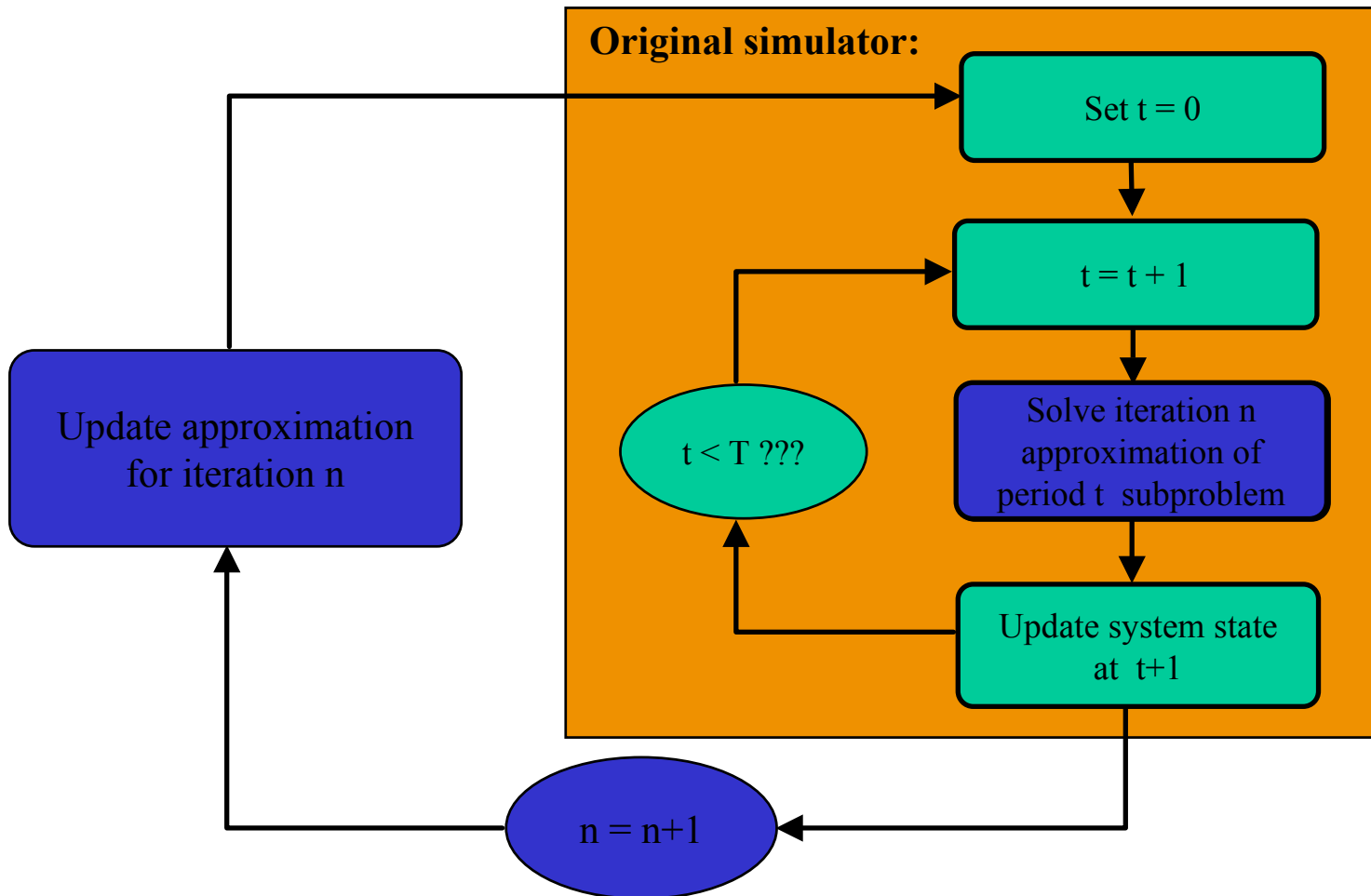
$$\text{Resource availability: } A_t x_t(\omega) = R_t \quad \text{Dual: } v_t(\omega)$$

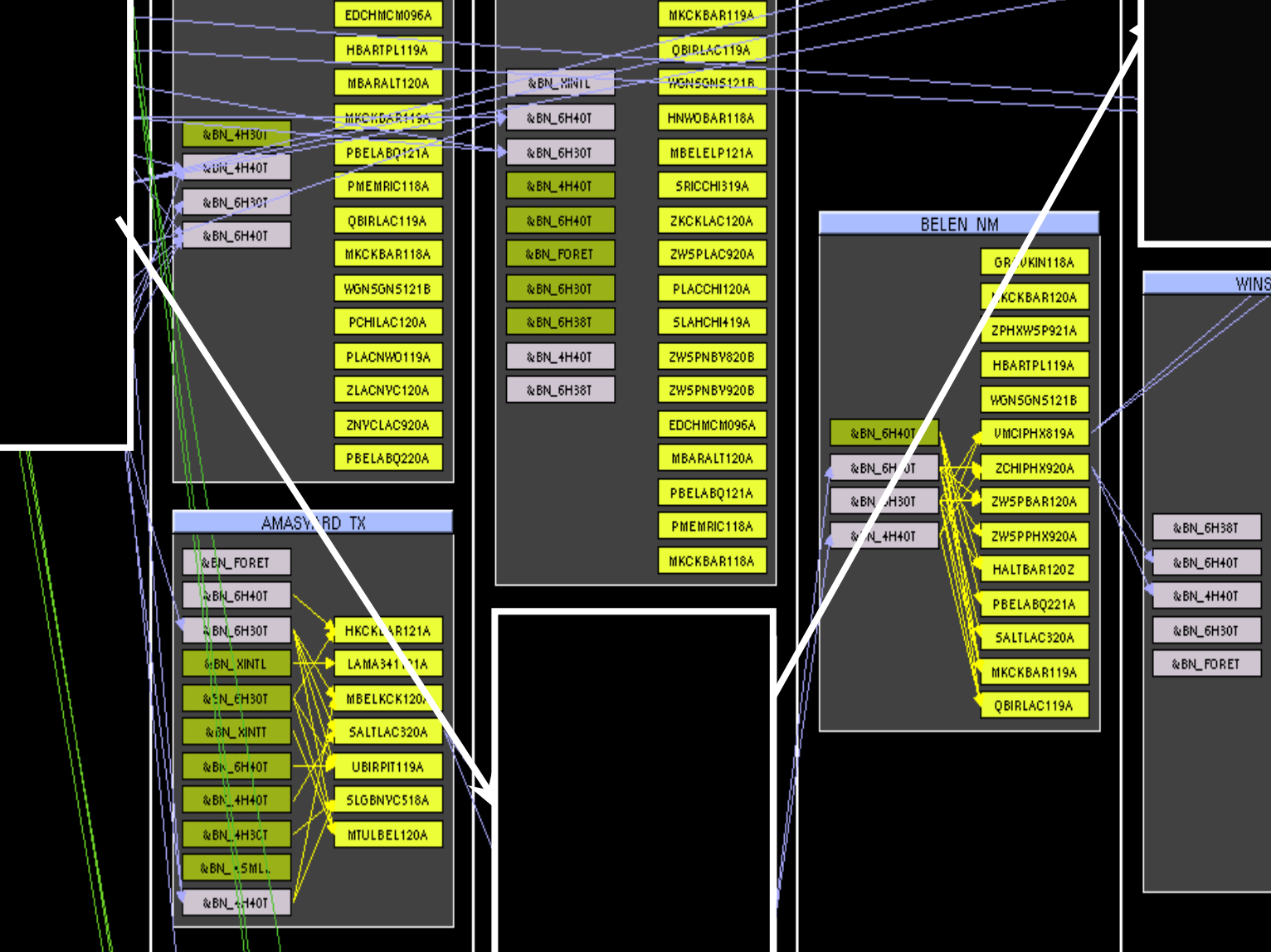
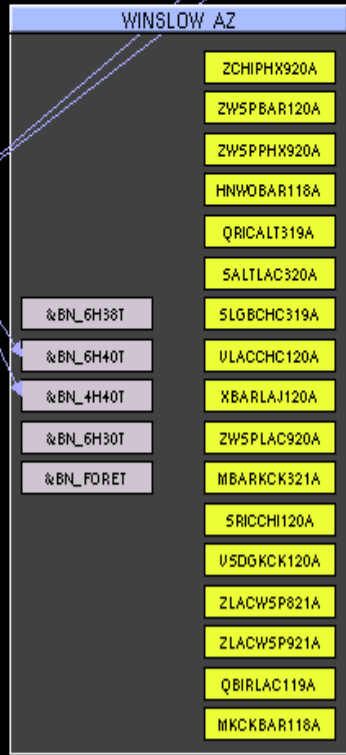
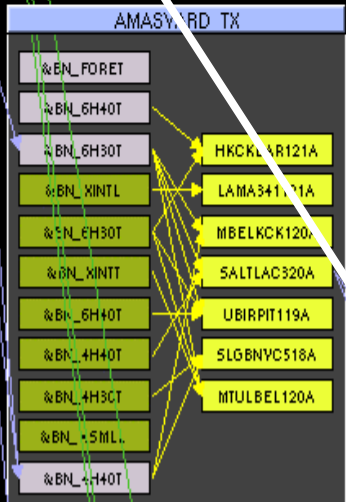
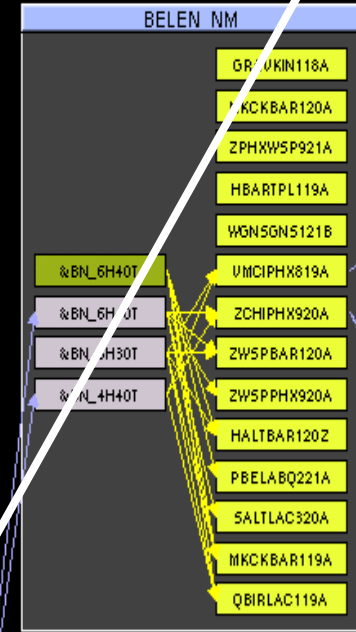
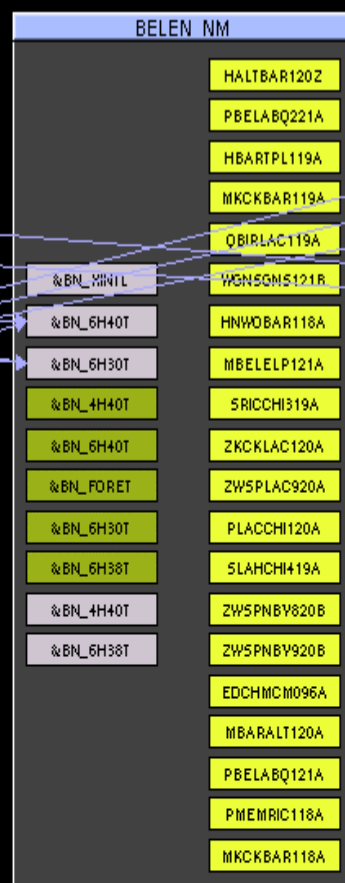
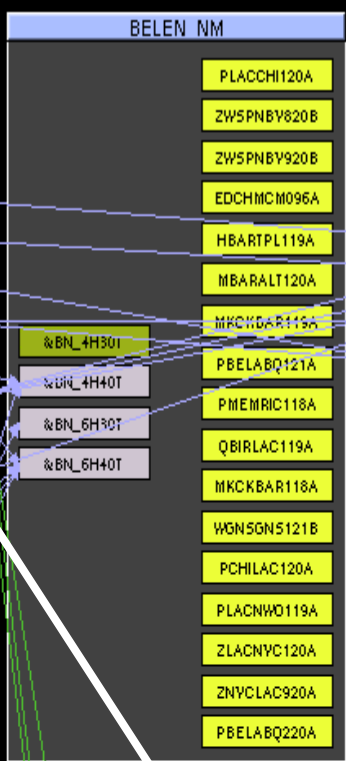
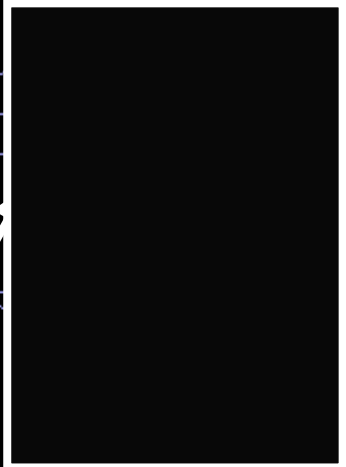
We now can use our solution to update the value function approximation:

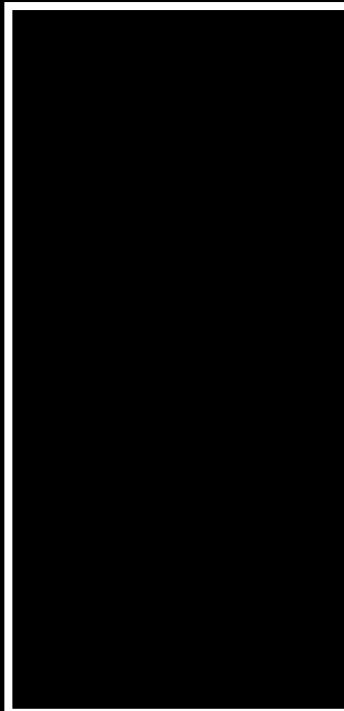
$$\hat{V}_t^{n+1} \leftarrow U^V(\hat{V}_t^n, v_t^n(\omega^n))$$

Adaptive dynamic programming

- The algorithm looks like an iterative simulator:



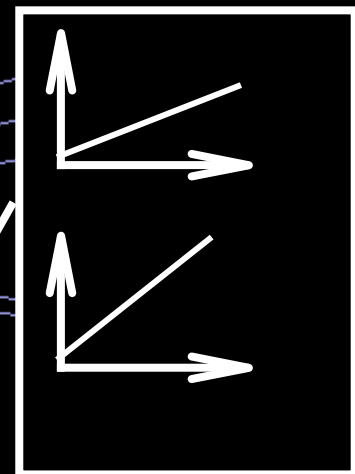




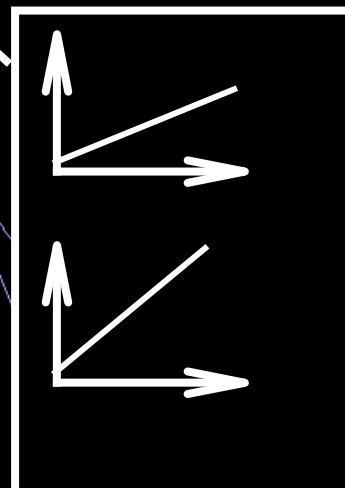
BELEN NM	
	PLACCHI120A
	ZW5PNBY820B
	ZW5PNBY920B
	EDCHMCM096A
	HBARTPL119A
	MBARALT120A
	MKCKBAR118A
&BN_4H30I	
&BN_4H40T	
&BN_6H30T	
&BN_6H40T	
	PBELABQ121A
	PMEMRIC118A
	QBIRLAC119A
	MKCKBAR118A
	WGN5GNS121B
	PCHILAC120A
	PLACNWD119A
	ZLACNYC120A
	ZNYCLAC920A
	PBELABQ220A

BELEN NM	
	HALTBAR120Z
	PBELABQ221A
	HBARTPL119A
	MKCKBAR119A
	QBIRLAC119A
	WGN5GNS121B
&BN_XINTL	
&BN_6H40T	
&BN_6H30T	
&BN_4H40T	
&BN_6H40T	
&BN_6H40T	
&BN_6H40T	
&BN_FORET	
&BN_6H30T	
&BN_6H38T	
&BN_4H40T	
&BN_6H38T	
	HNWOBAR118A
	MBELELP121A
	SRICCHI319A
	ZKCKLAC120A
	ZW5PLAC920A
	PLACCHI120A
	SLAHCHI119A
	ZW5PNBY820B
	ZW5PNBY920B
	EDCHMCM096A
	MBARALT120A
	PBELABQ121A
	PMEMRIC118A
	MKCKBAR118A

BELEN NM	
	TRAUKIN118A
	MKCKBAR120A
	ZPHXW5P921A
	HBARTPL119A
	WGN5GNS121B
&BN_6H30T	
&BN_4H40T	
&BN_6H30T	
&BN_4H40T	
	UMCIPHX819A
	ZCHIPX920A
	ZW5PBAR120A
	ZW5PPHX920A
	HALTBAR120Z
	PBELABQ221A
	SALTAC320A
	MKCKBAR119A
	QBIRLAC119A

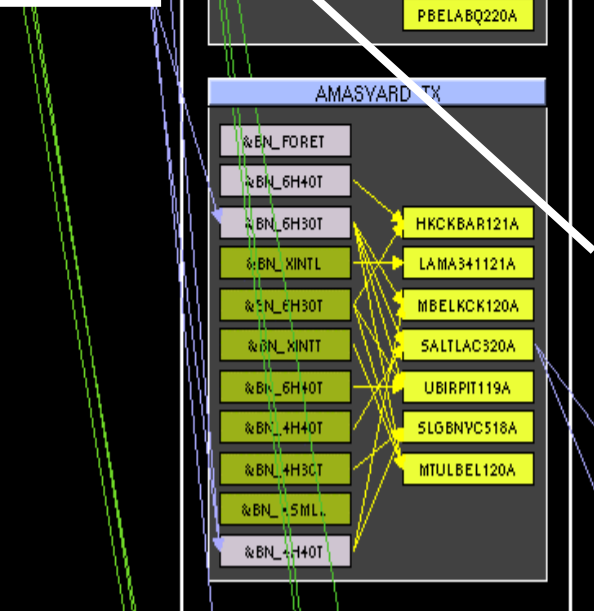


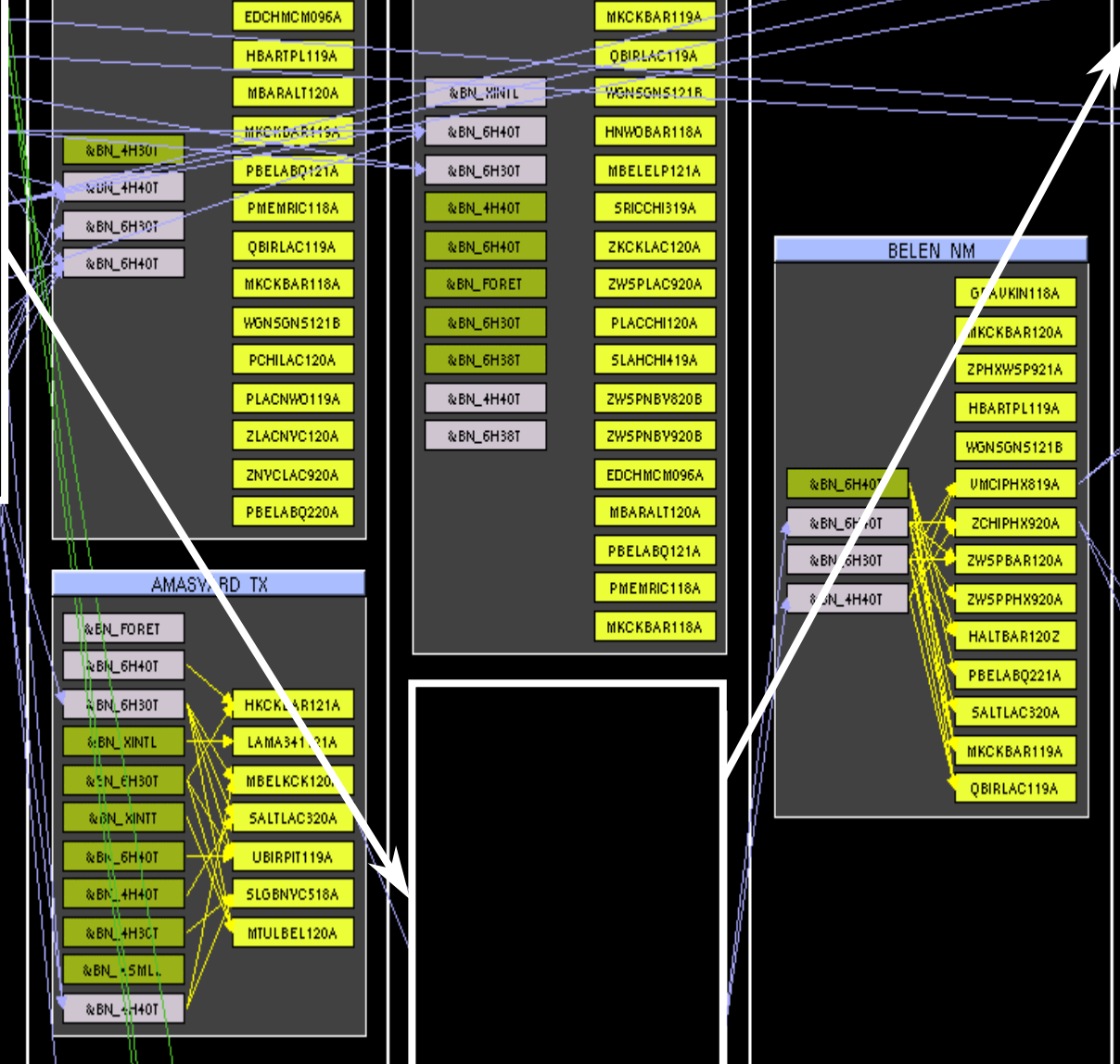
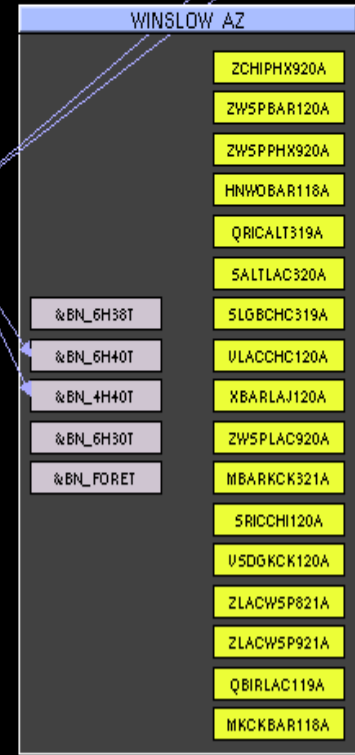
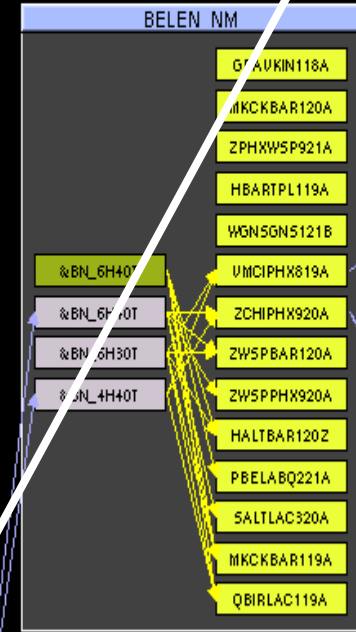
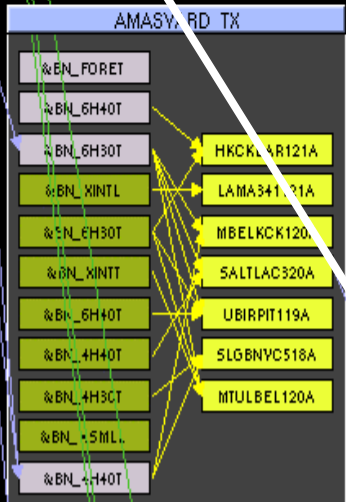
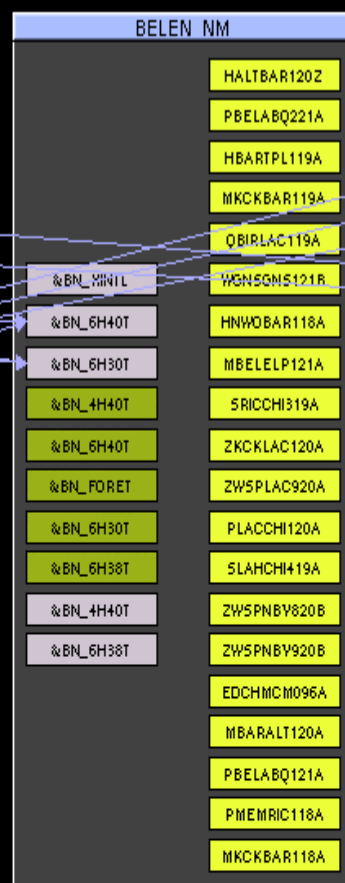
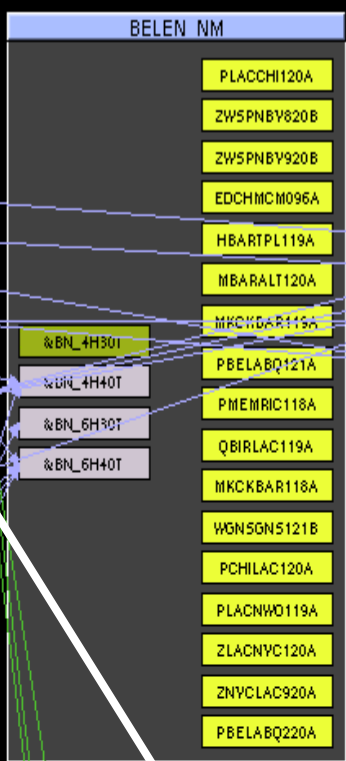
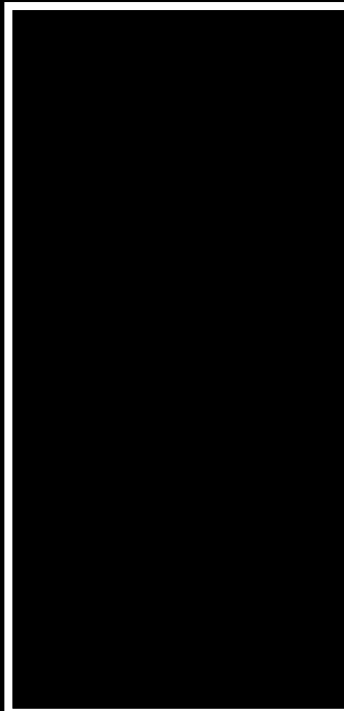
AMASYARD TX	
&BN_FORET	
&BN_6H40T	
&BN_6H30T	
&BN_XINTL	
&BN_6H30T	
&BN_XINTT	
&BN_6H40T	
&BN_4H40T	
&BN_4H30T	
&BN_XMILL	
&BN_4H40T	
	HKCKBAR121A
	LAMA341121A
	MBELKCK120A
	SALTAC320A
	UBIRPIT119A
	SLGBNYC518A
	MTULBEL120A

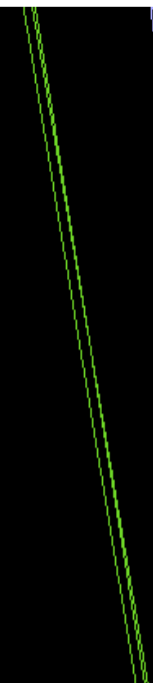
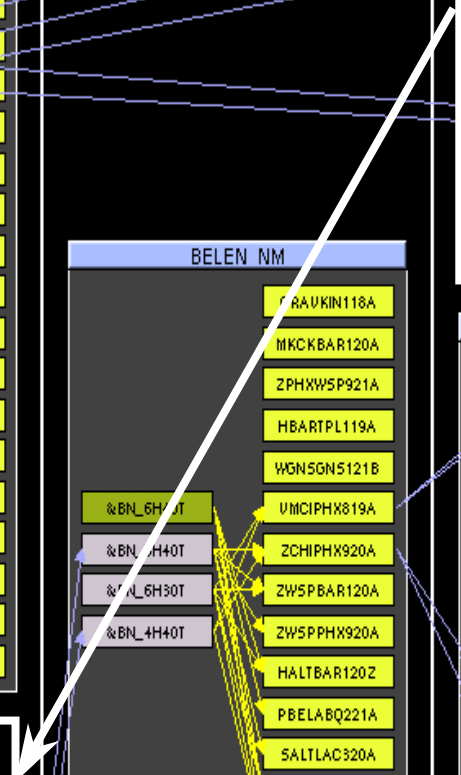
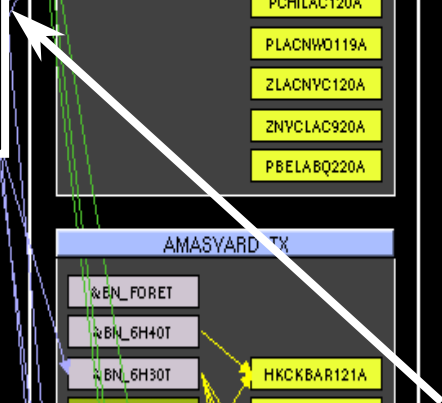
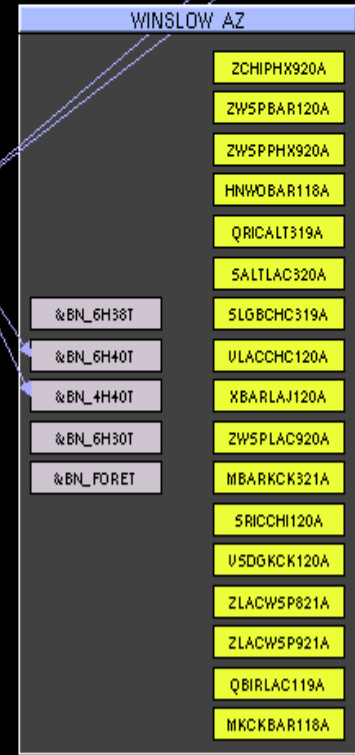
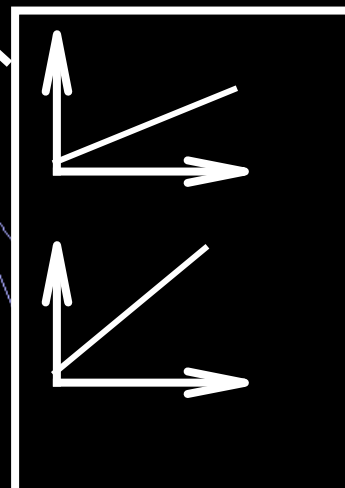
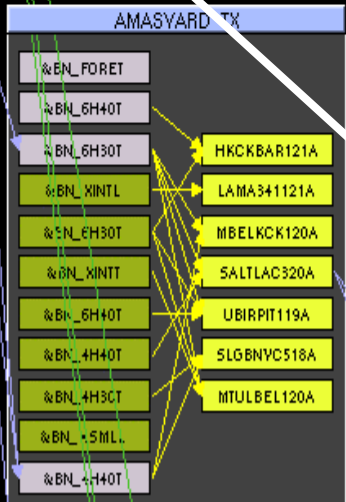
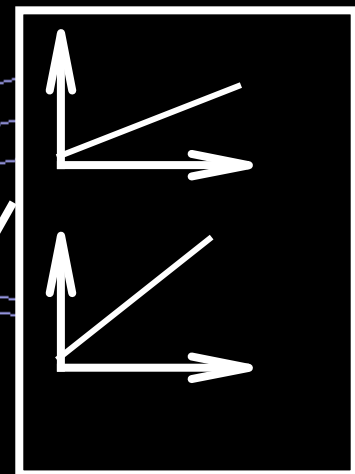
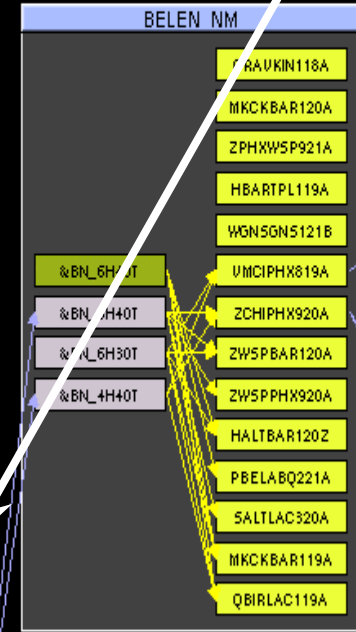
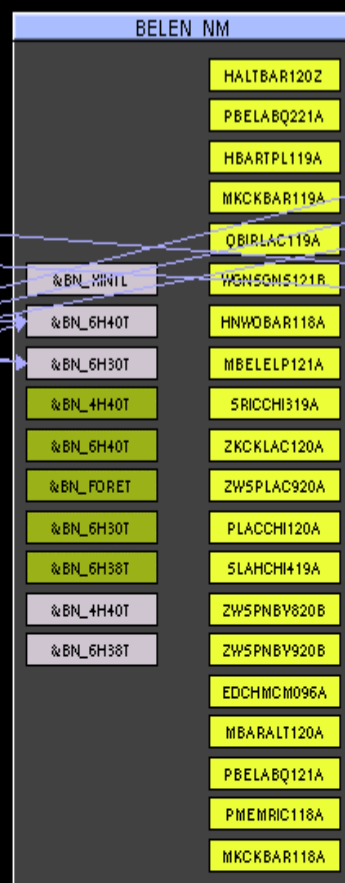
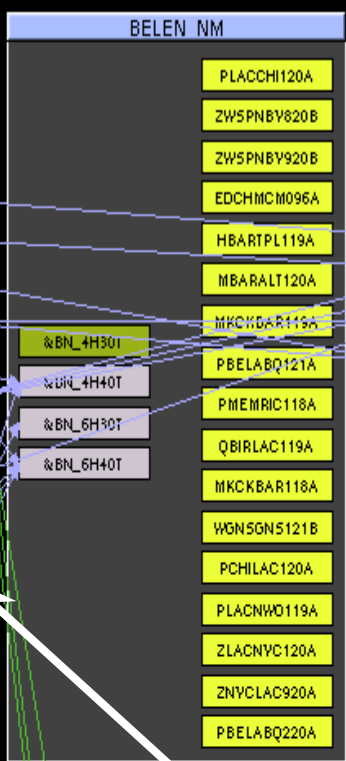
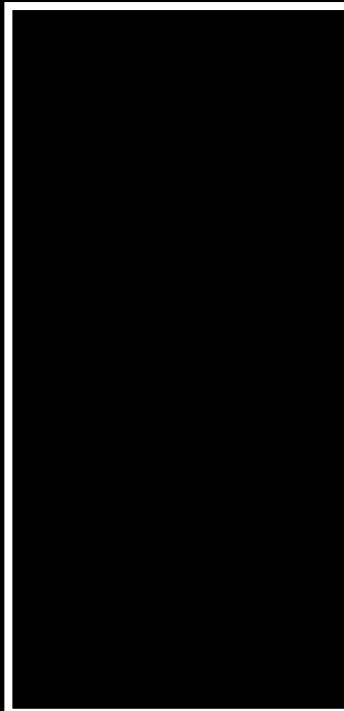


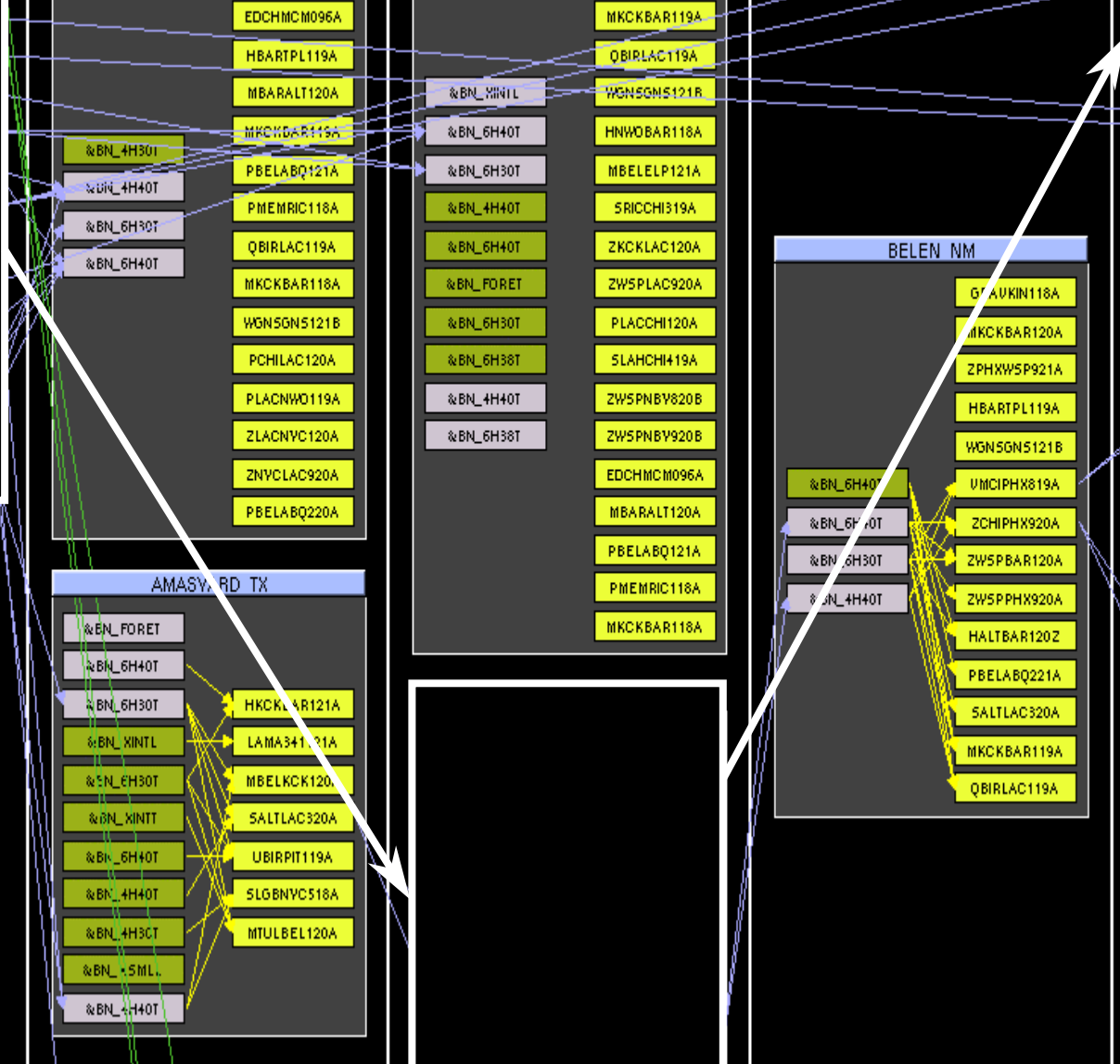
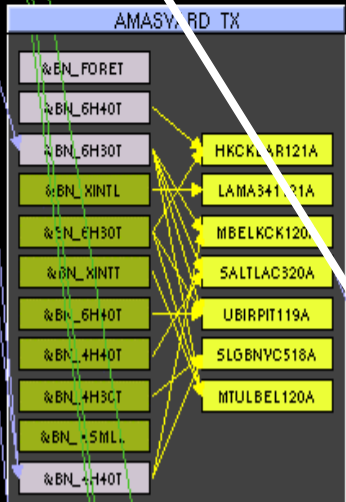
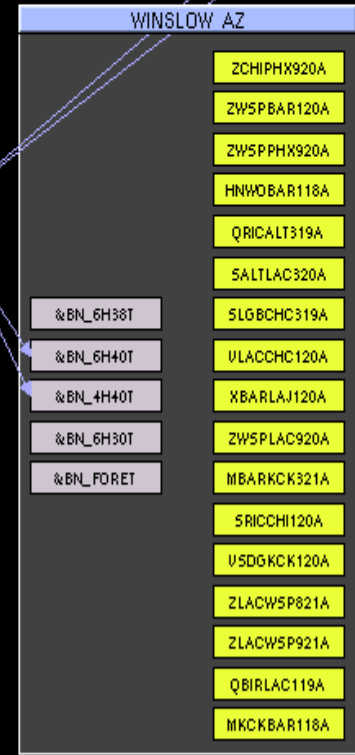
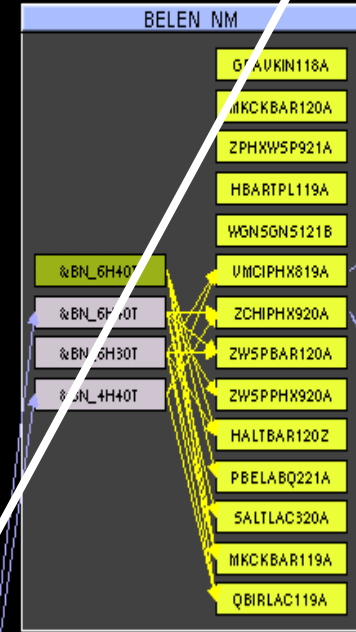
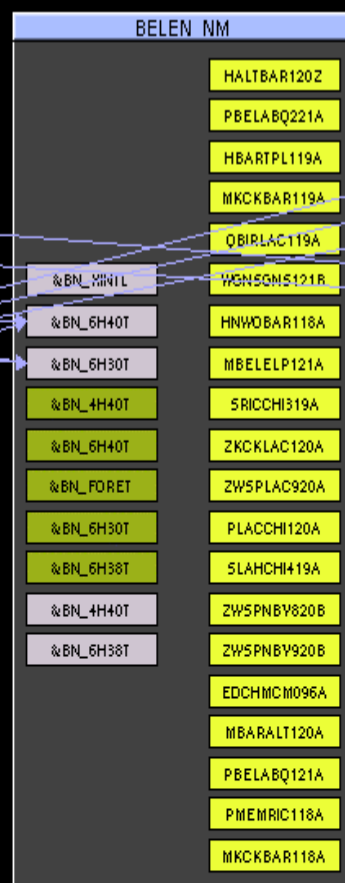
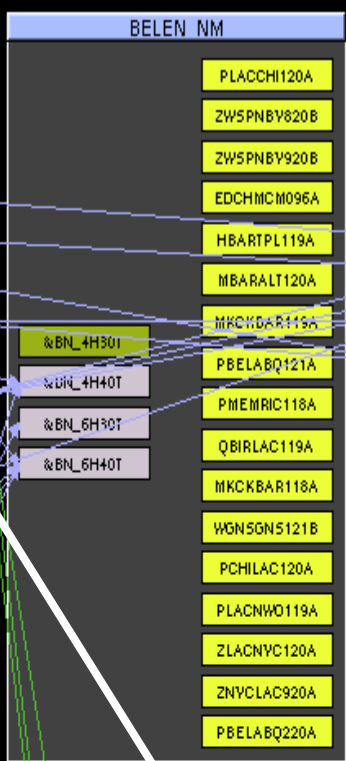
VAUGHN NM	
	WLOVAU121T

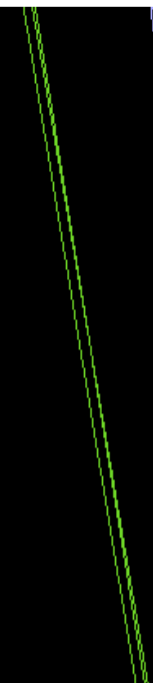
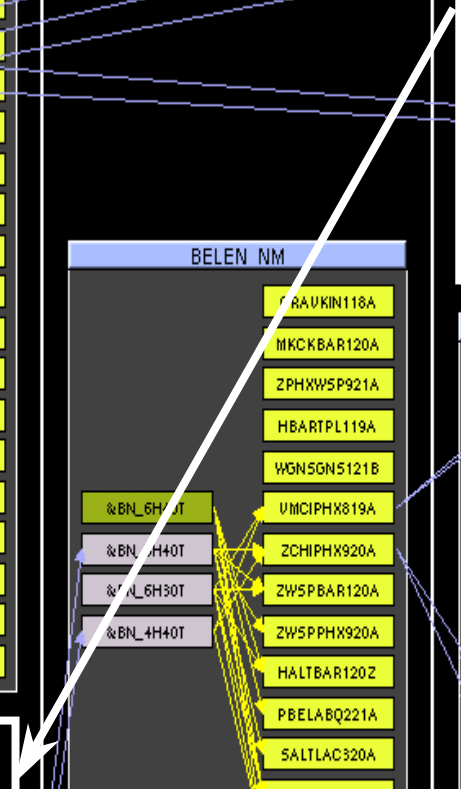
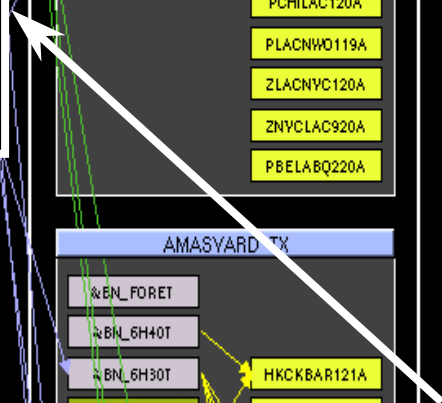
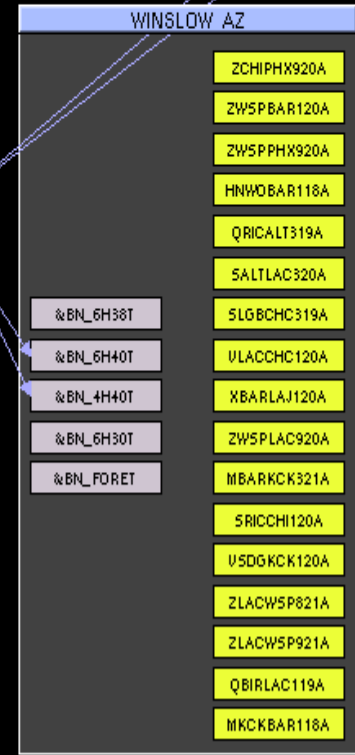
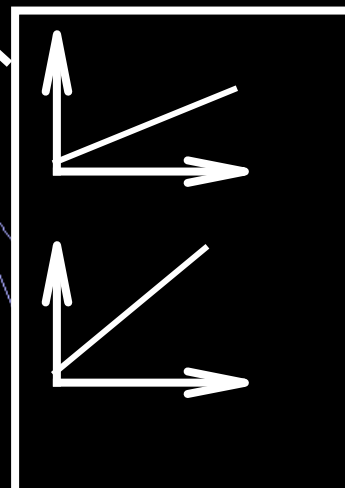
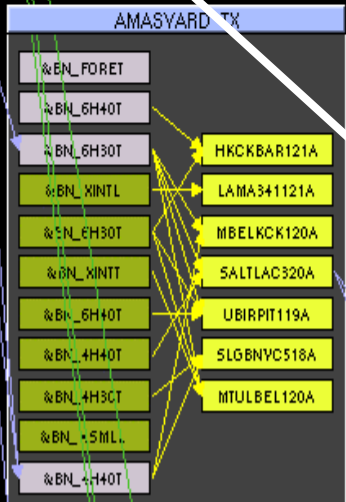
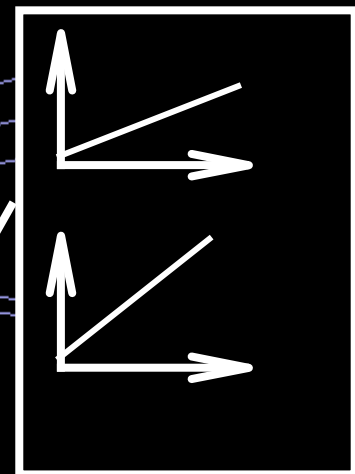
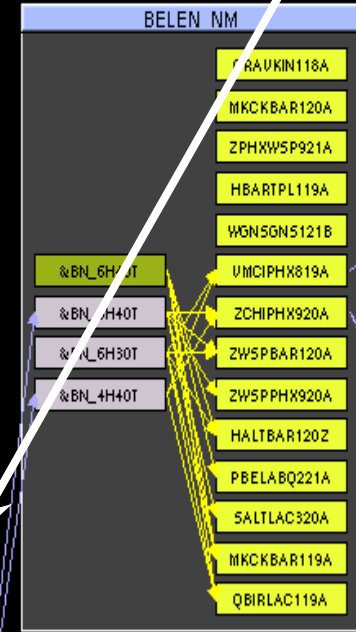
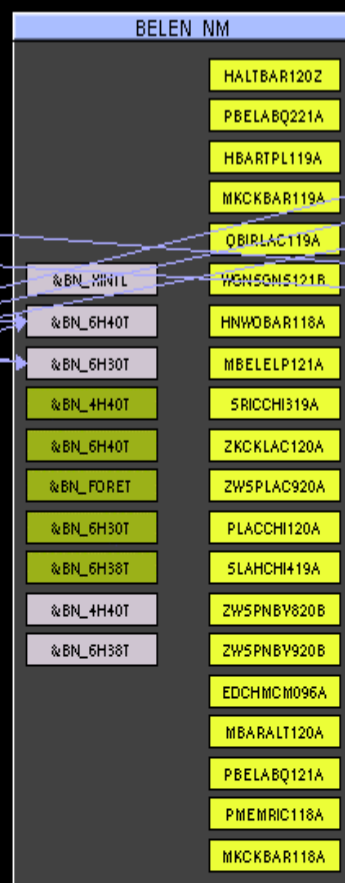
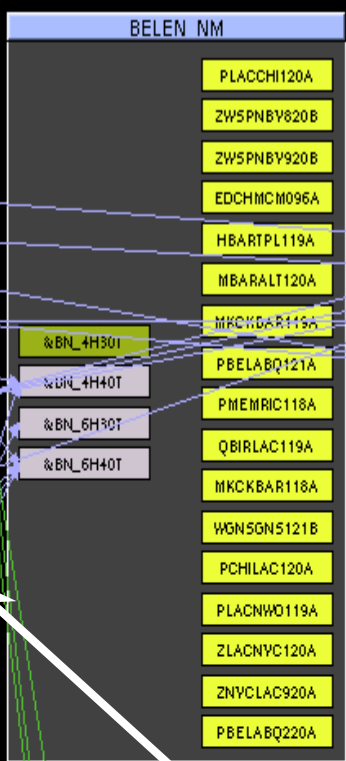
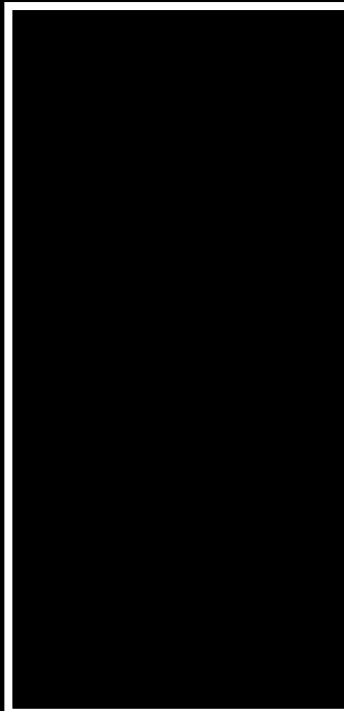
WINSLOW AZ	
	ZCHIPX920A
	ZW5PBAR120A
	ZW5PPHX920A
	HNWOBAR118A
	QRICALT319A
	SALTAC320A
&BN_6H38T	
&BN_6H40T	
&BN_4H40T	
&BN_6H30T	
&BN_FORET	
	SLGBCHC319A
	VLACCHC120A
	XBARLAJ120A
	ZW5PLAC920A
	MBARKCK321A
	SRICCHI120A
	USDGKCK120A
	ZLACW5P821A
	ZLACW5P921A
	QBIRLAC119A
	MKCKBAR118A







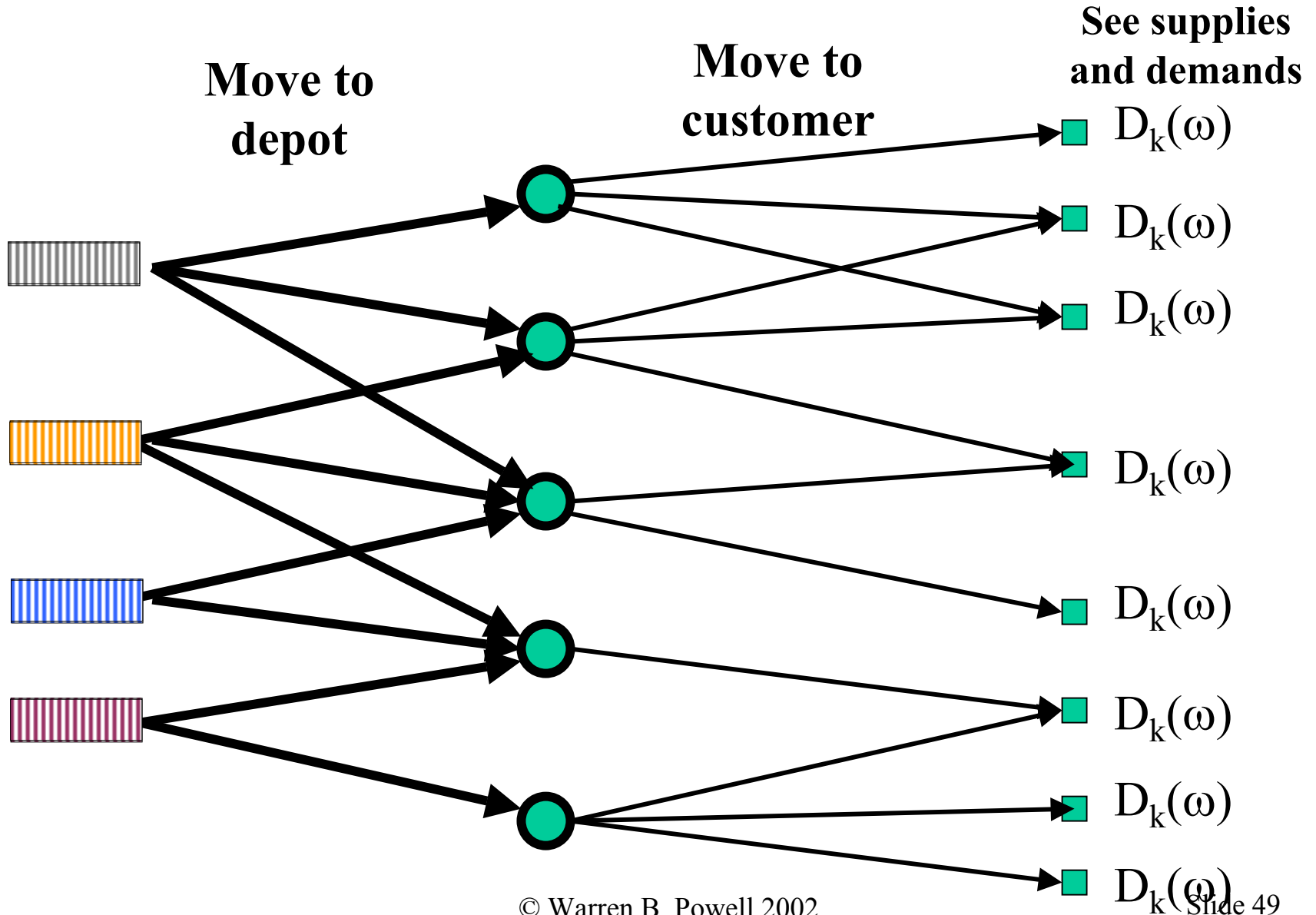




Outline

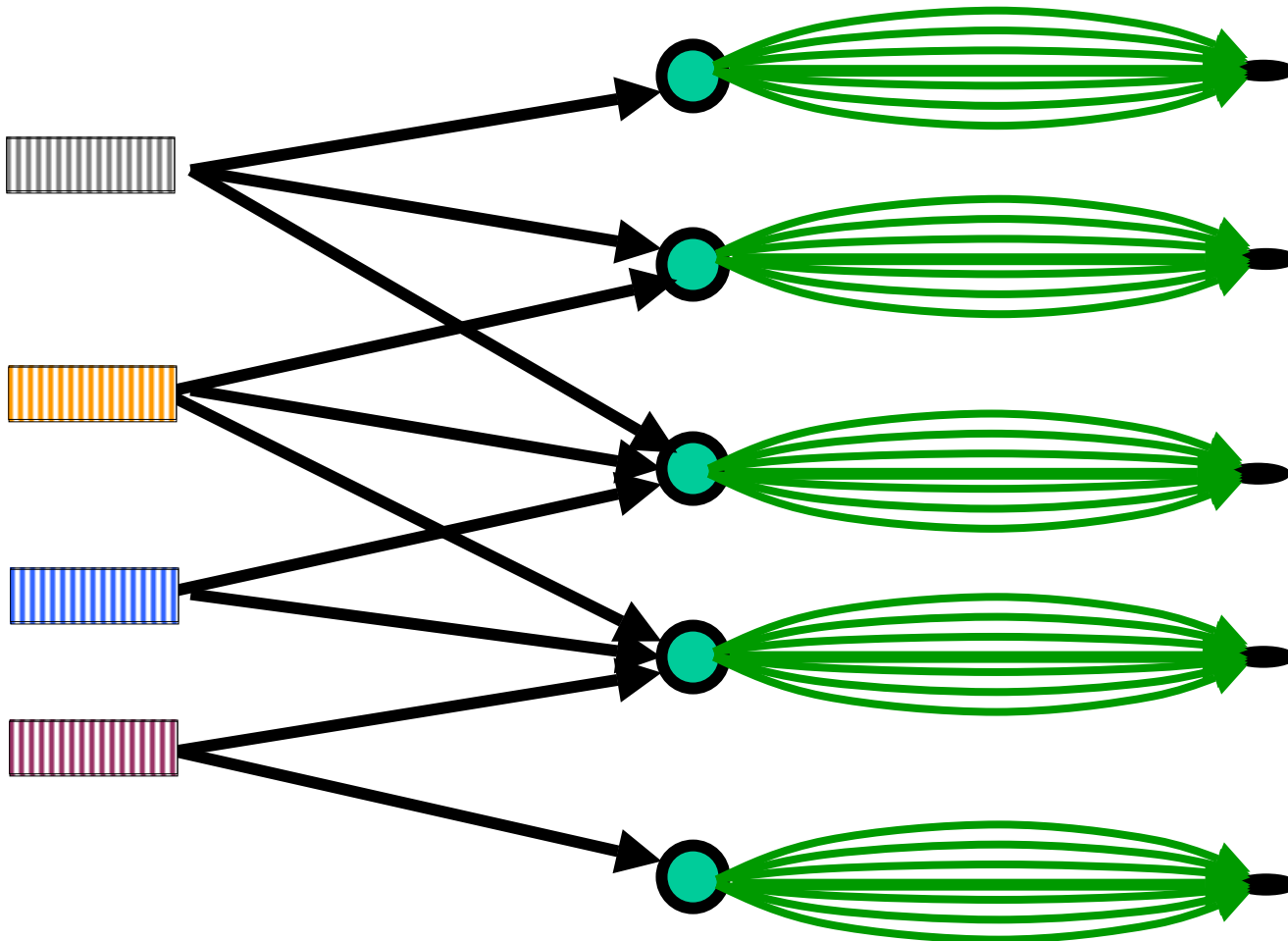
- The adaptive learning algorithm

Distribution under uncertainty



Distribution under uncertainty

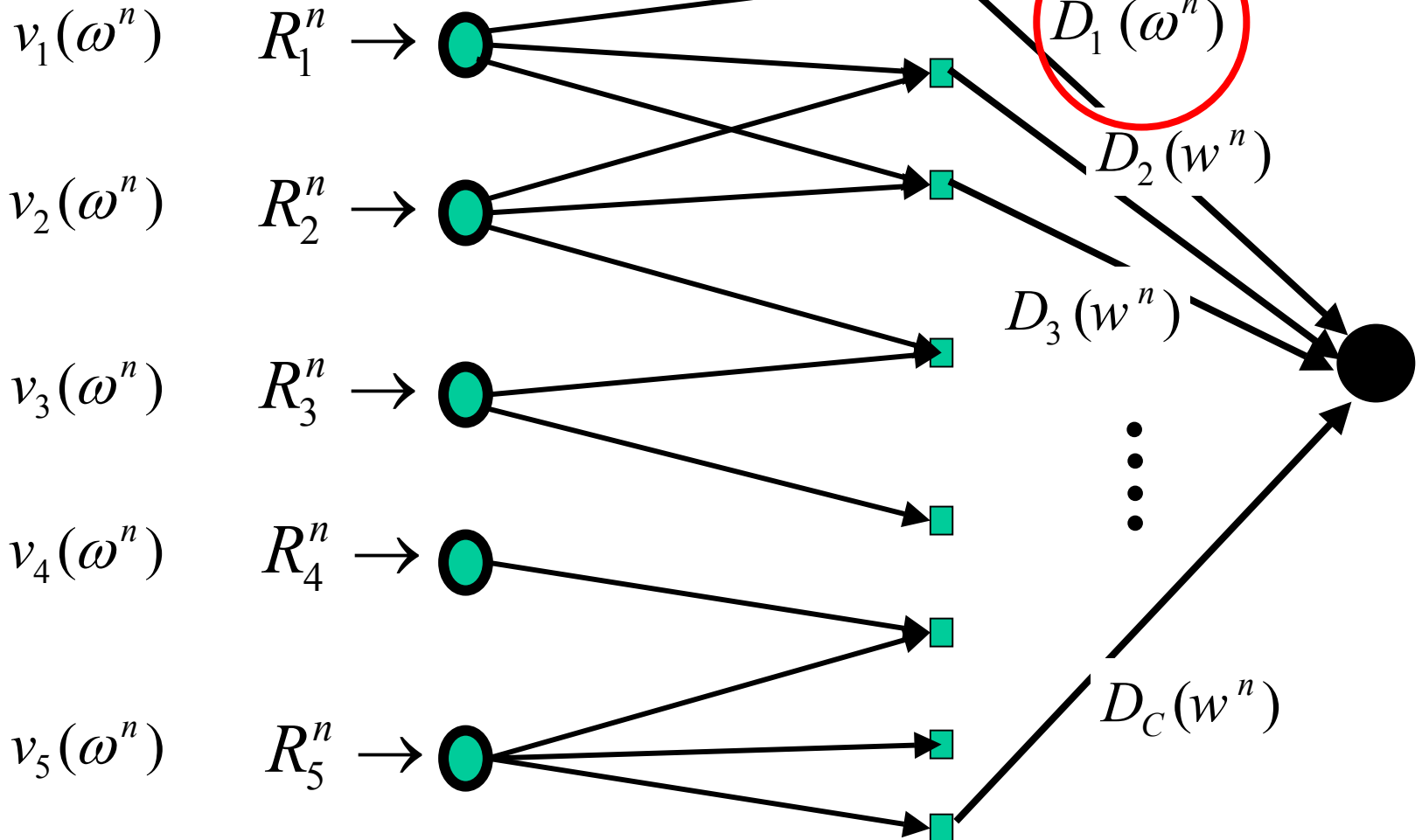
- We solve the first stage using an approximation of the second stage:



Distribution under uncertainty

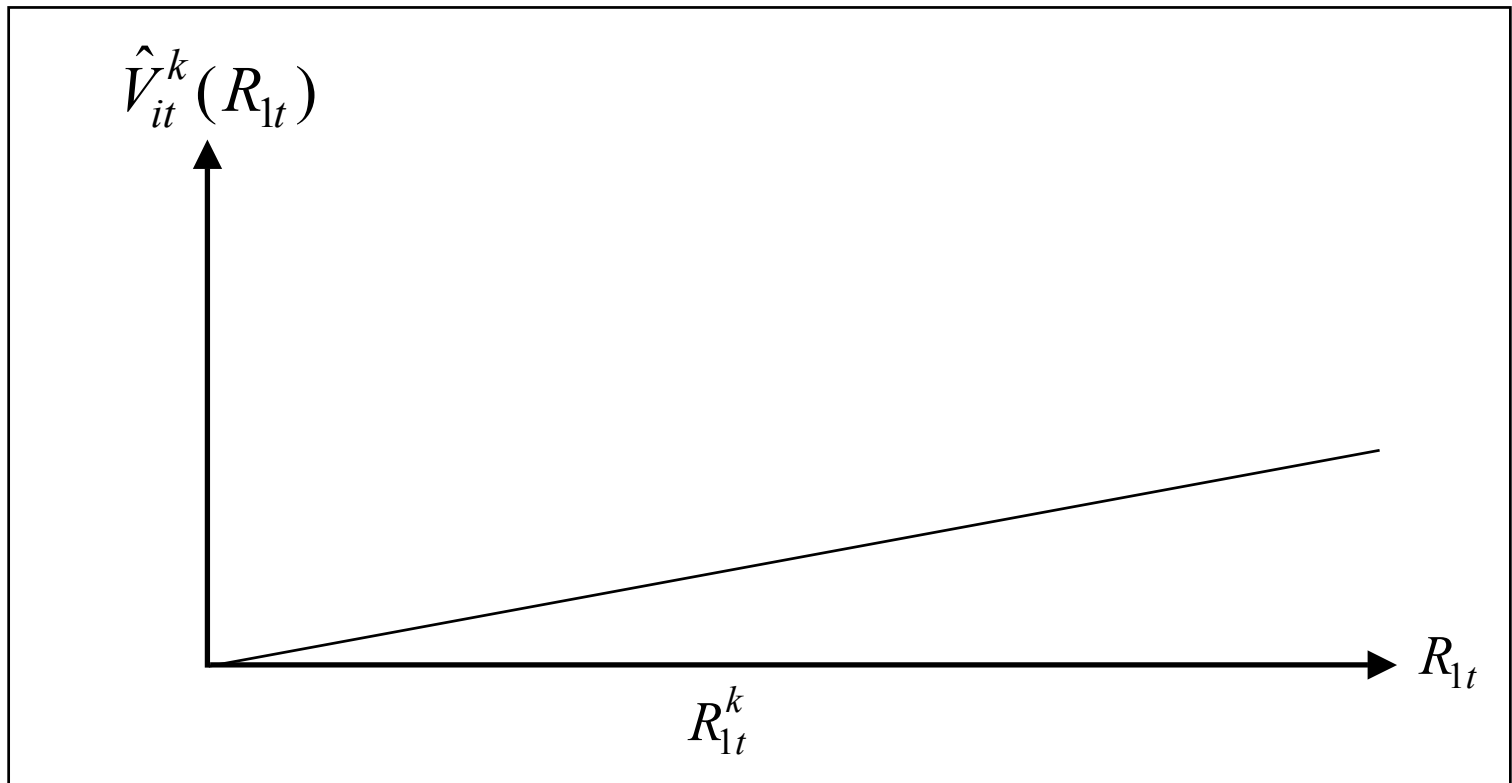
- We estimate the functions by sampling from our distributions. **Demand**

The value of
an additional car:



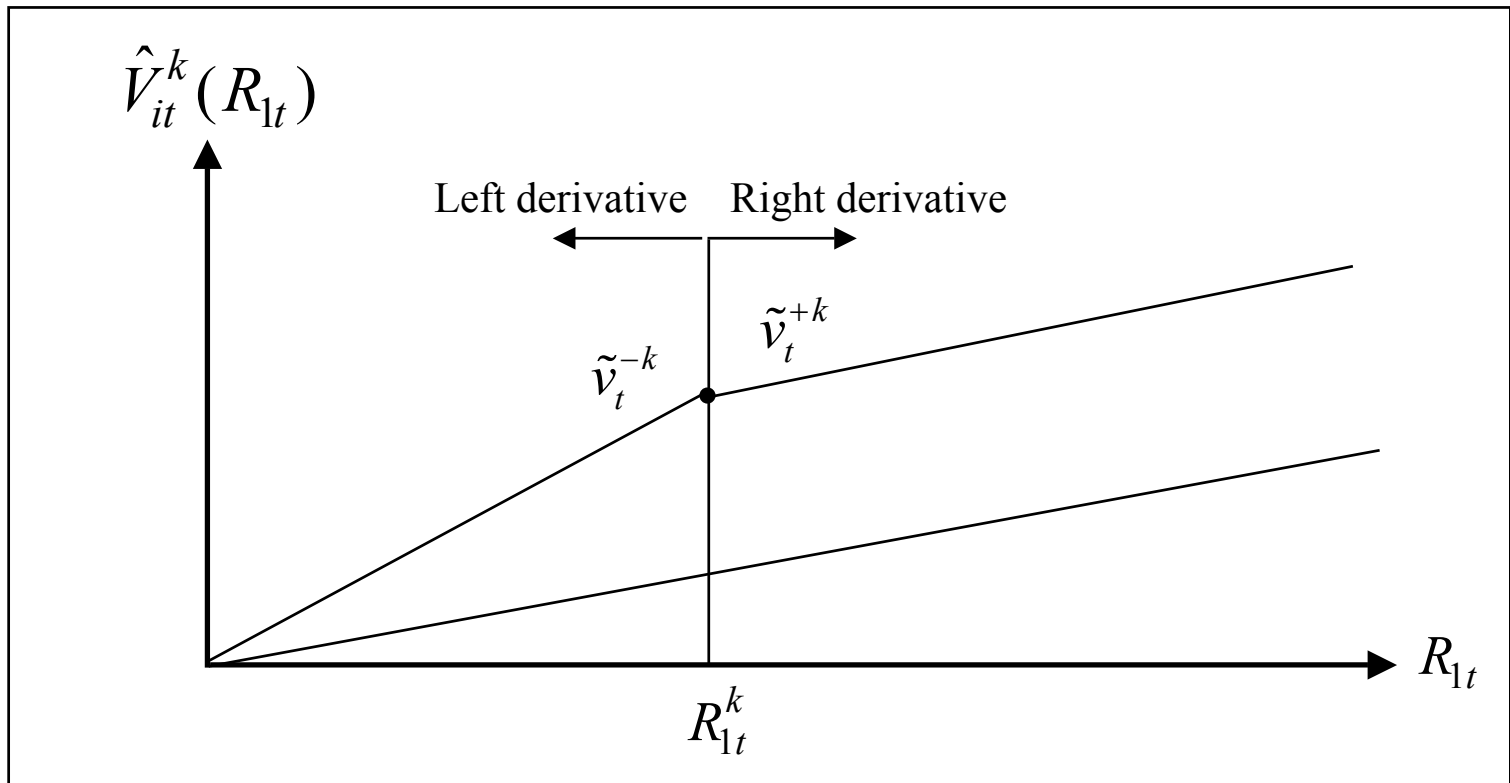
Distribution under uncertainty

- Left and right derivatives are used to build up a nonlinear approximation of the subproblem.



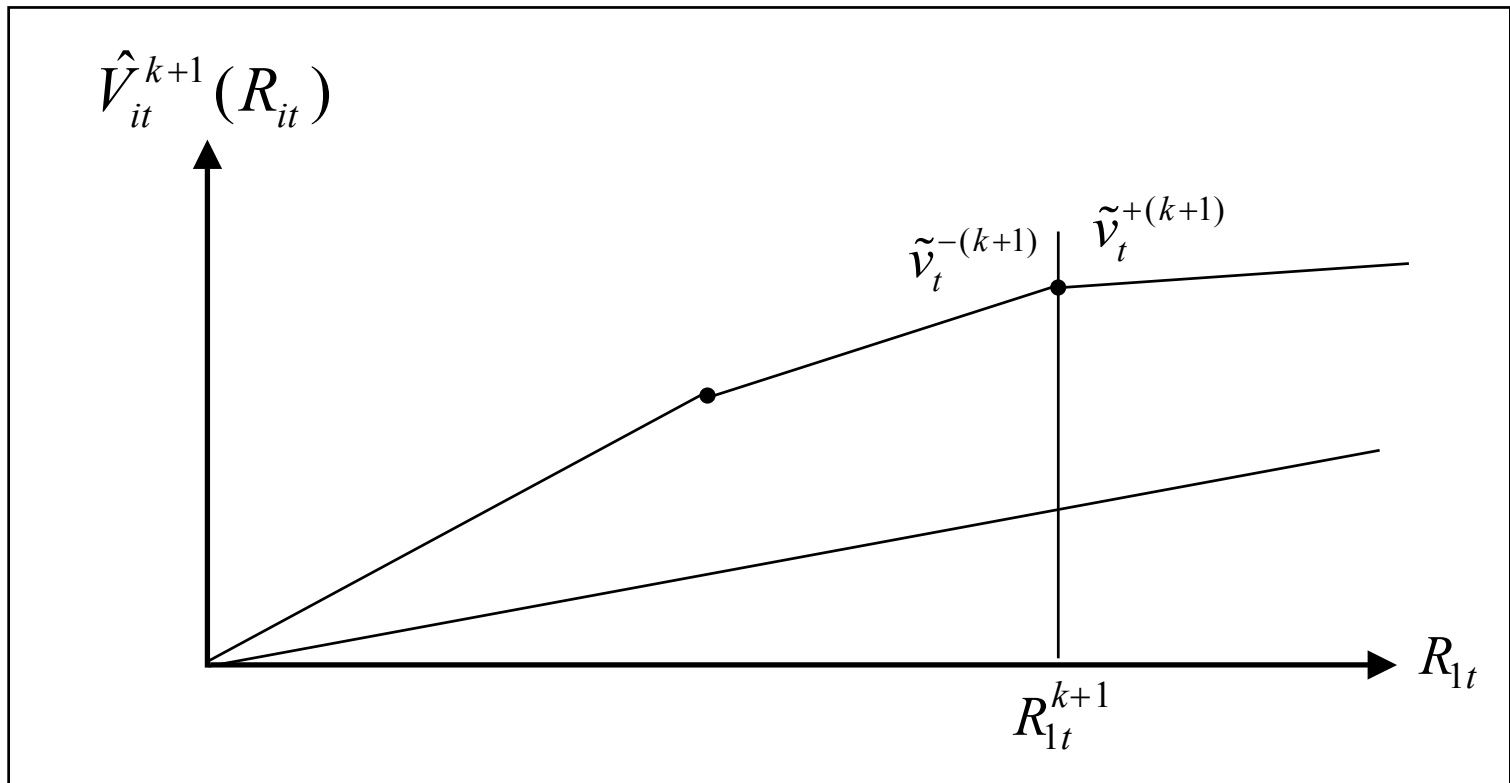
Distribution under uncertainty

- Left and right derivatives are used to build up a nonlinear approximation of the subproblem.

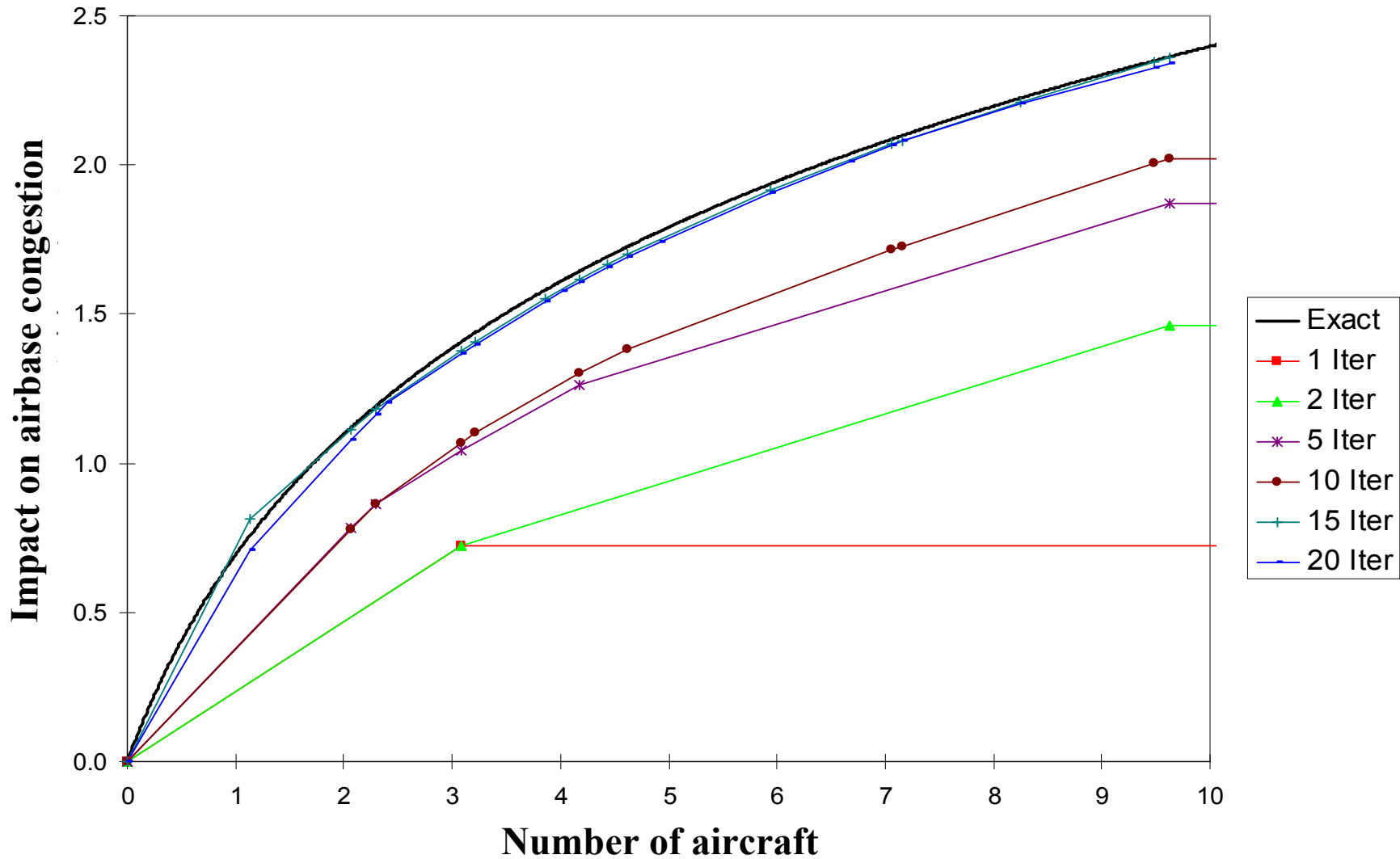


Distribution under uncertainty

- Each iteration adds new segments, as well as refining old ones.

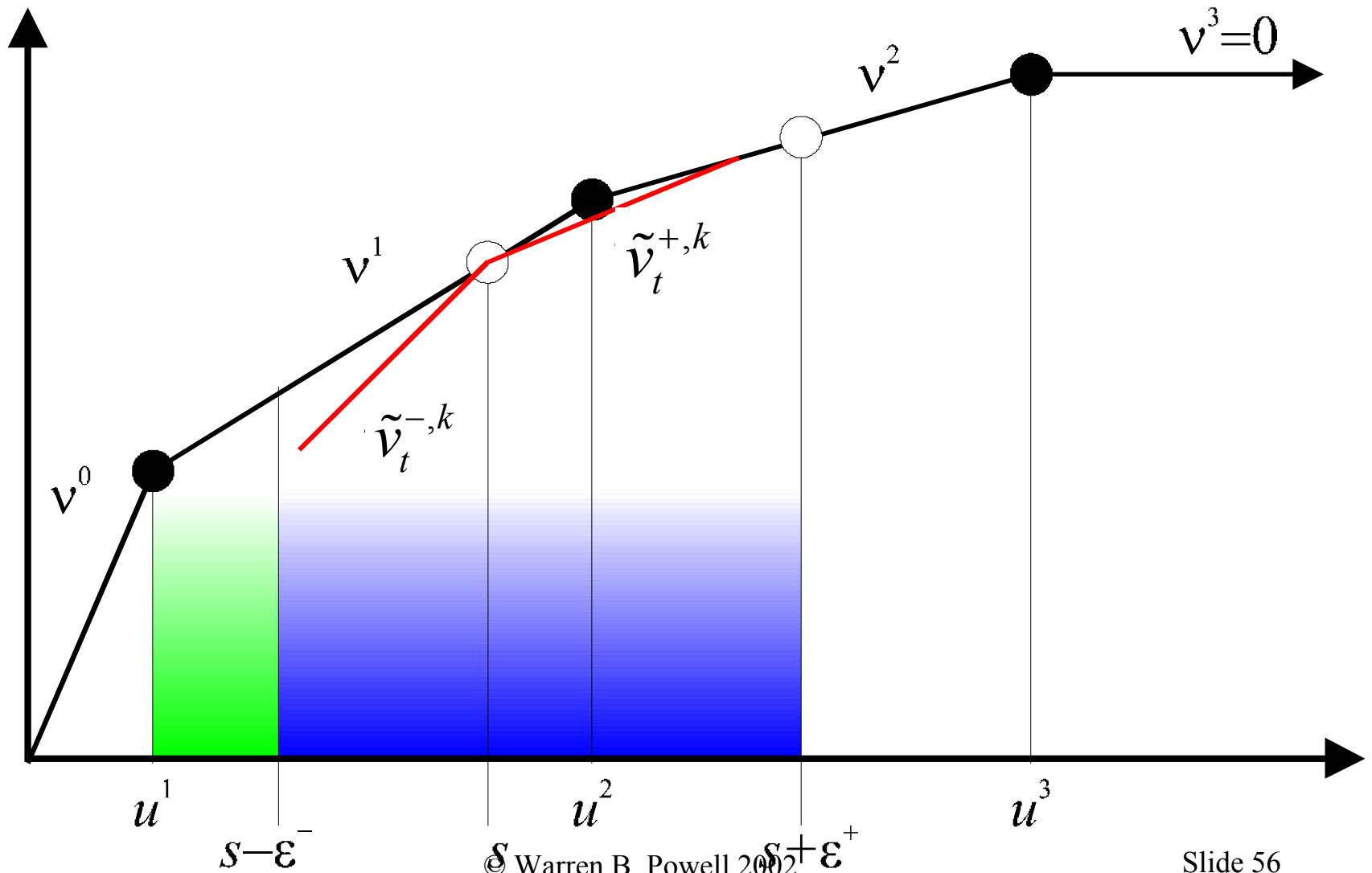


Nonlinear approximations

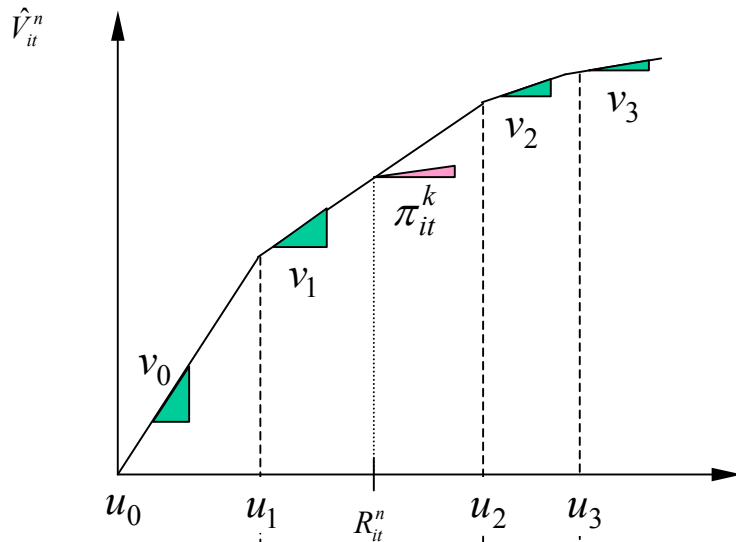


Nonlinear approximations

- It is important to maintain concavity:

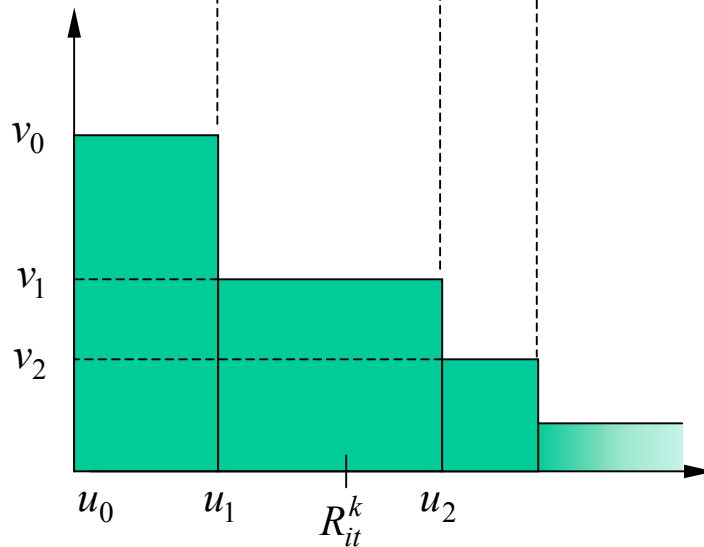


Value
function



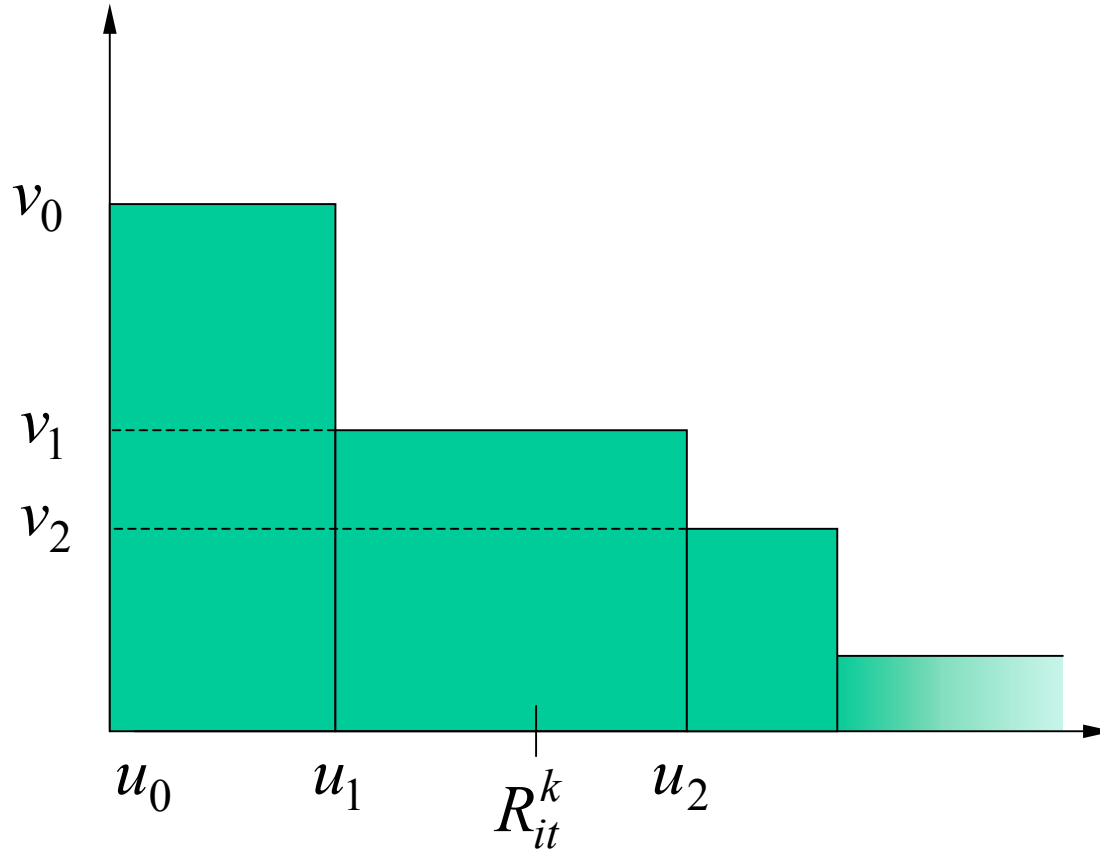
A concave function...

Slopes

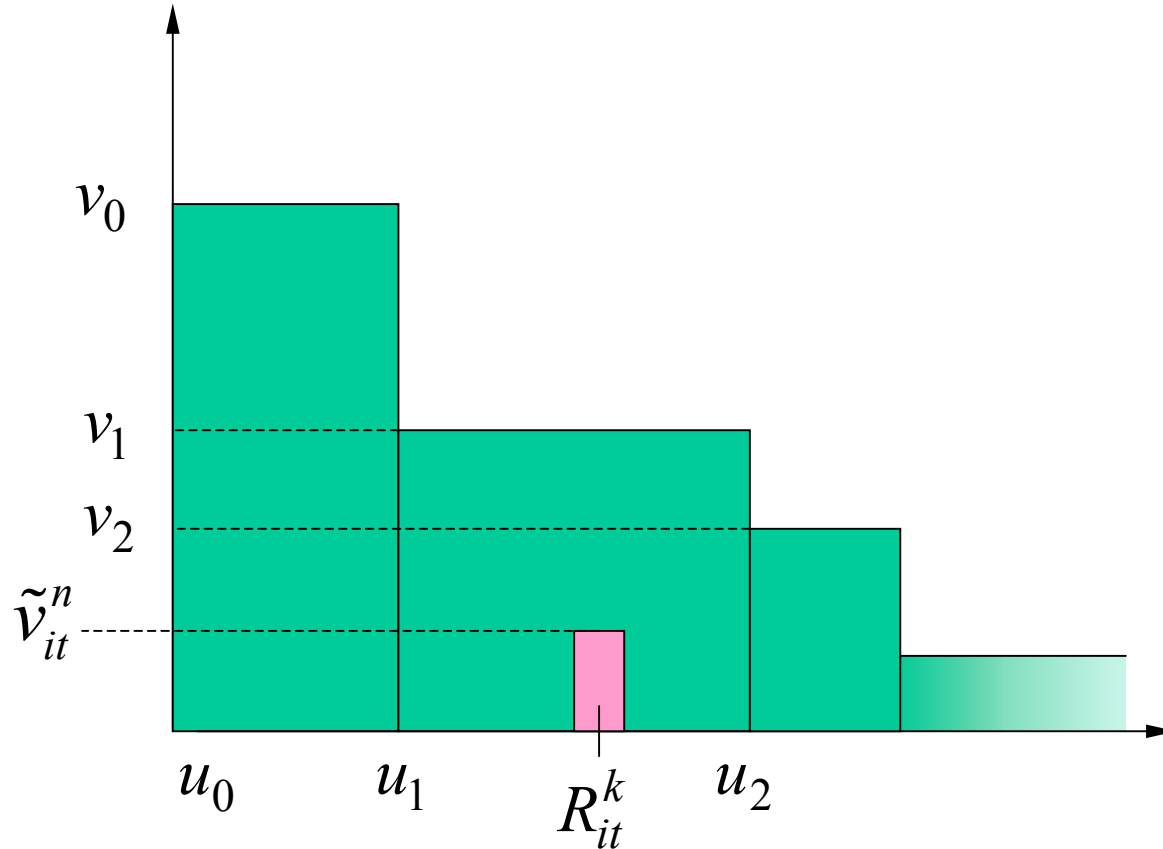


... has monotonically decreasing slopes. But updating the function with a stochastic gradient may violate this property.

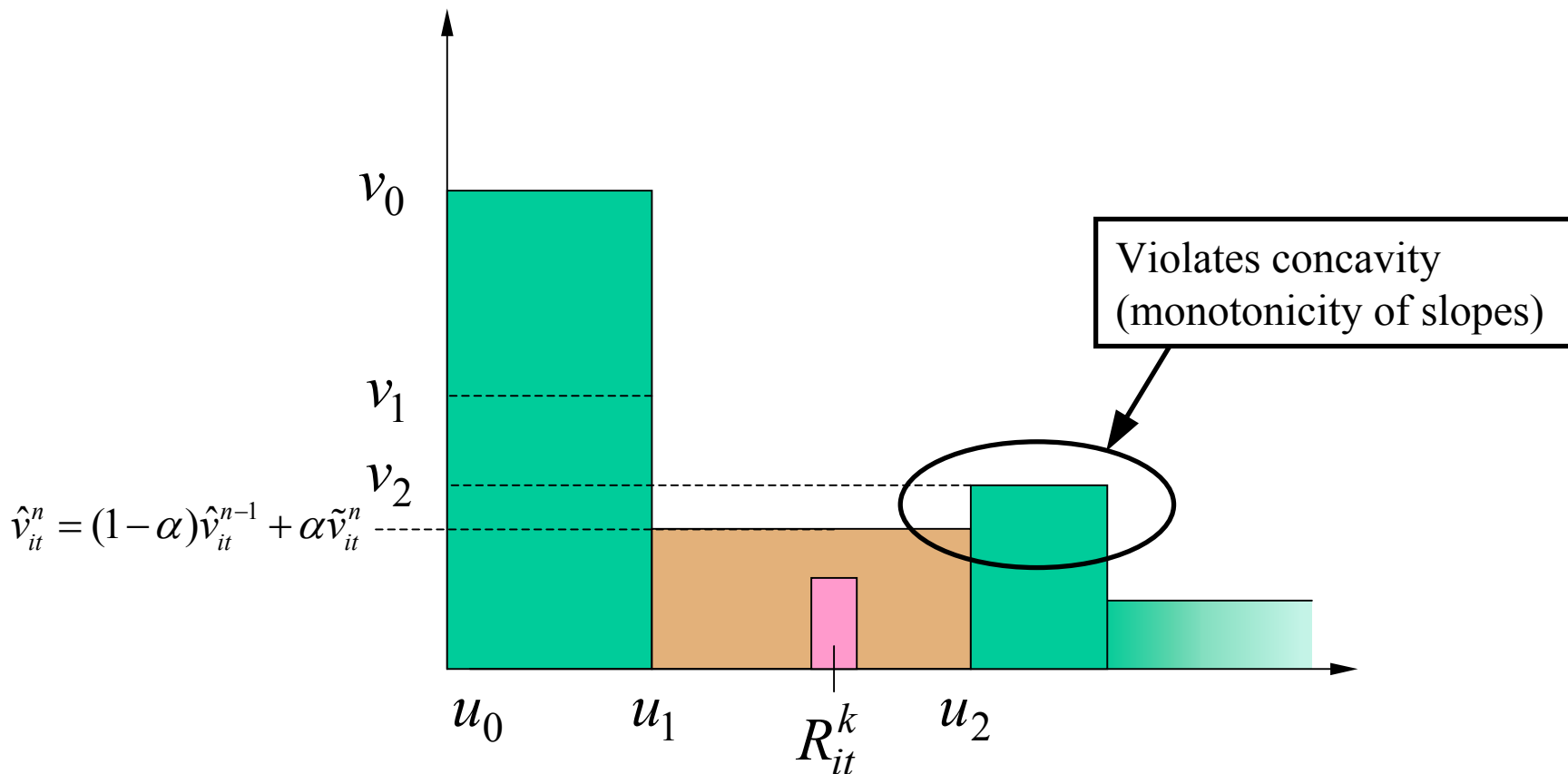
A leveling algorithm



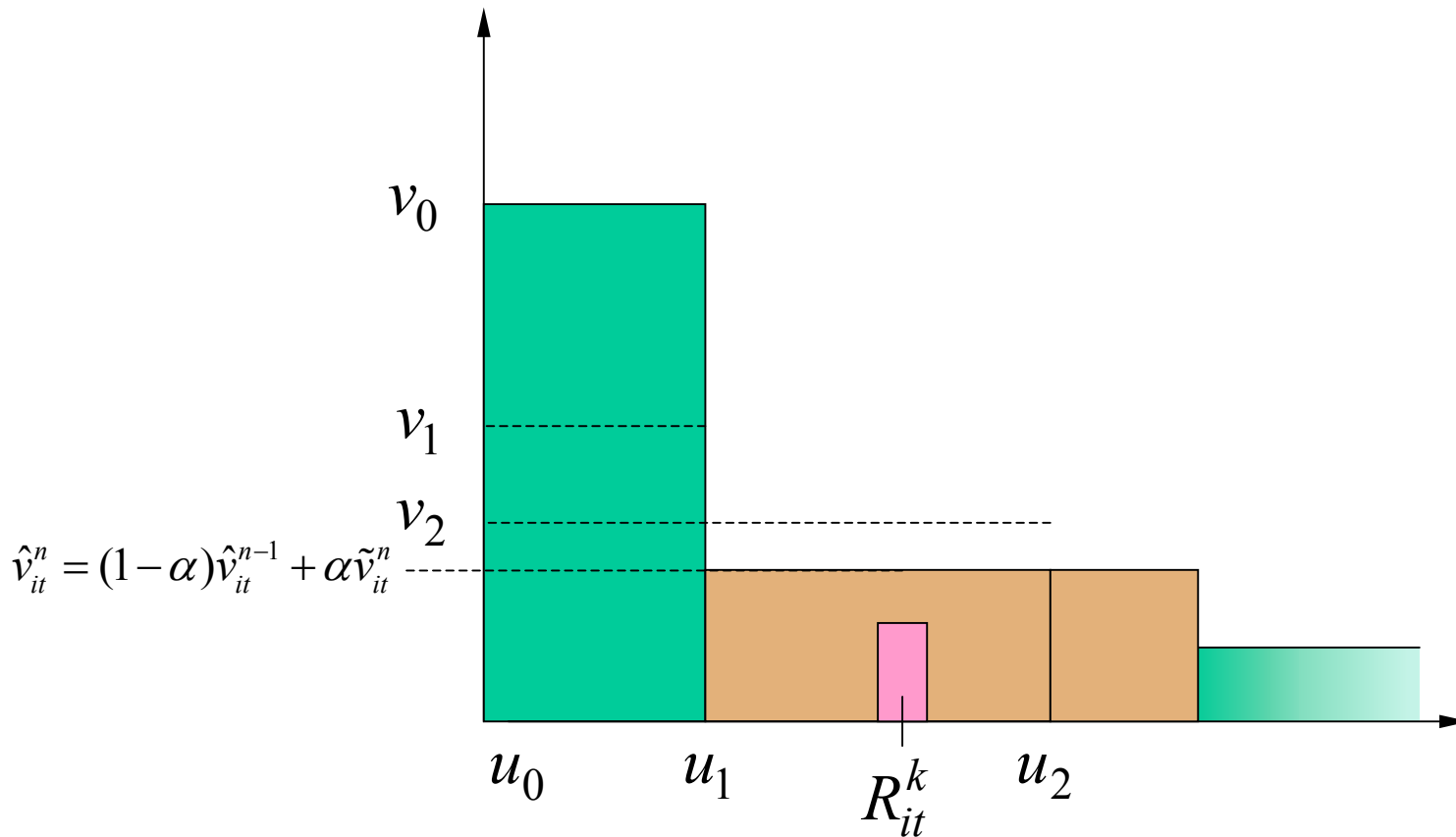
A leveling algorithm



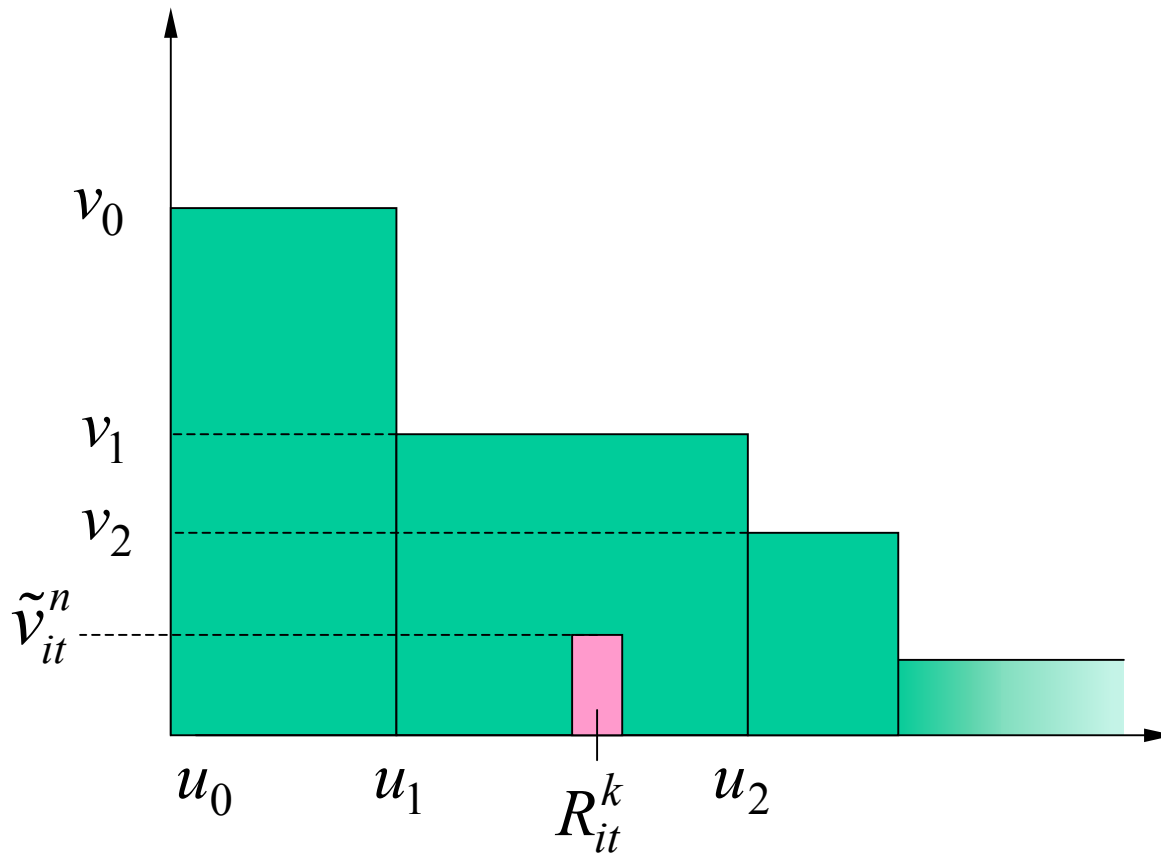
A leveling algorithm



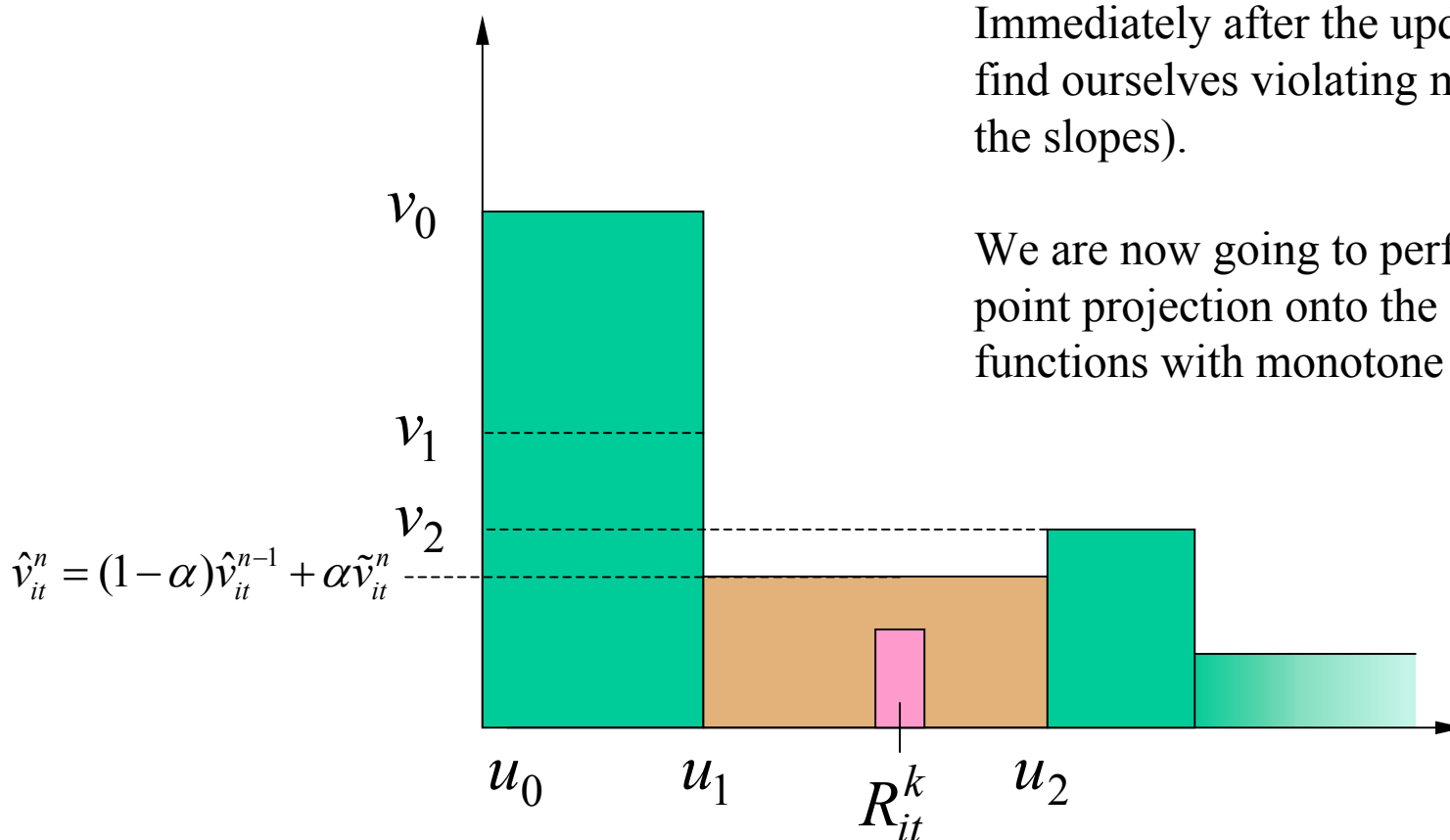
A leveling algorithm



A projection algorithm



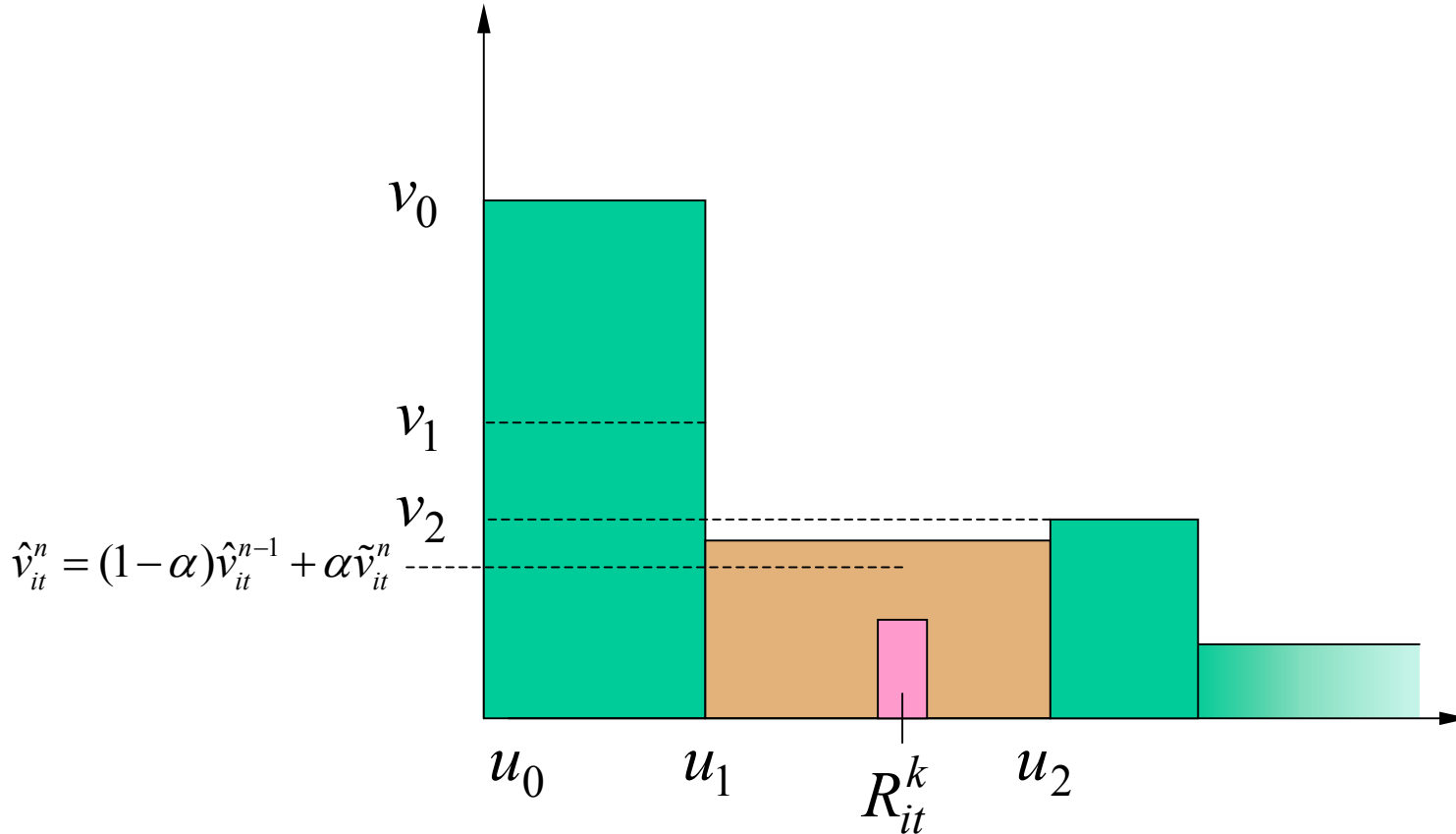
A projection algorithm



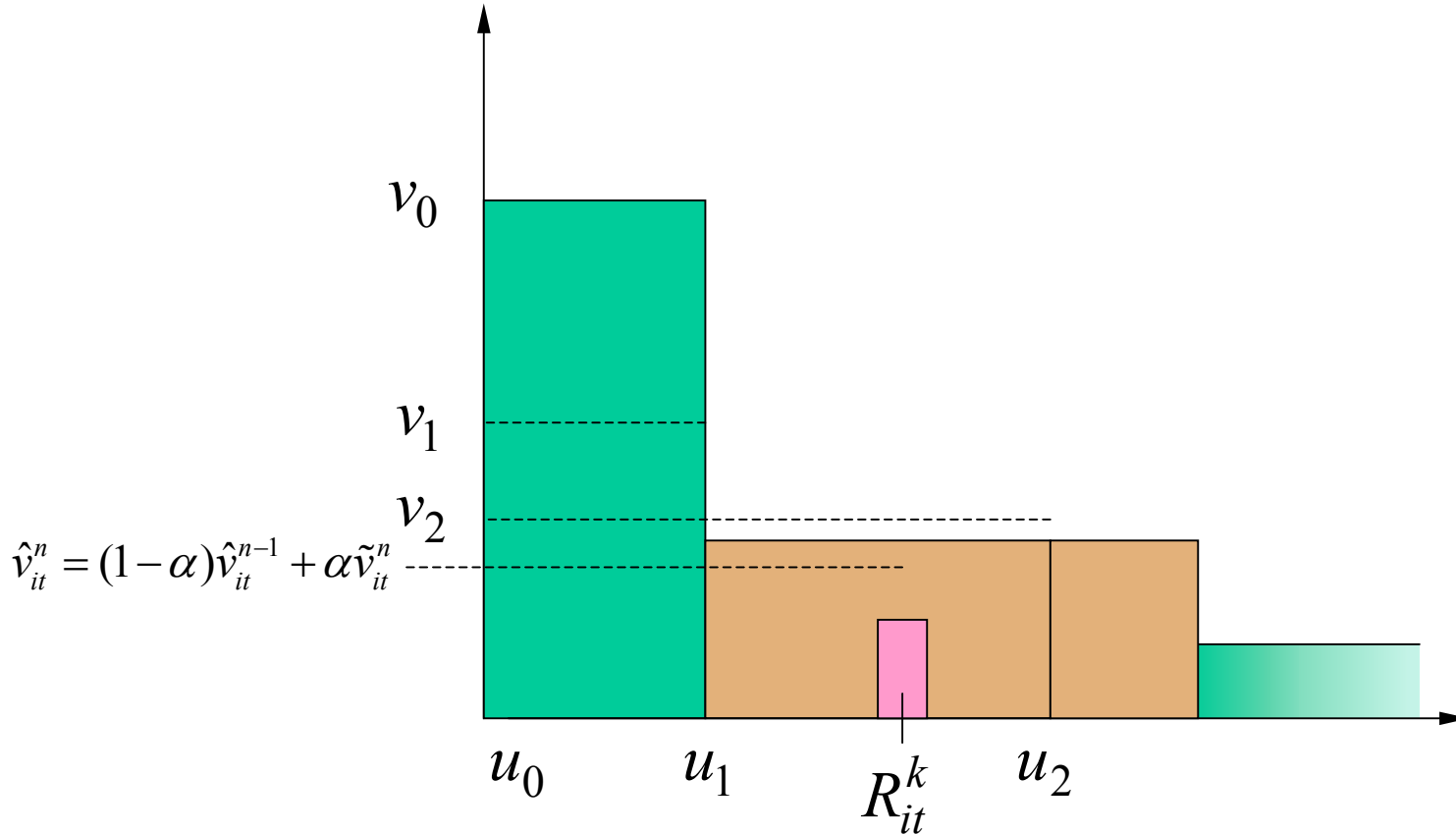
Immediately after the update, we gain find ourselves violating monotonicity (of the slopes).

We are now going to perform a nearest point projection onto the space of functions with monotone slopes.

A projection algorithm



A projection algorithm



Learning strategies

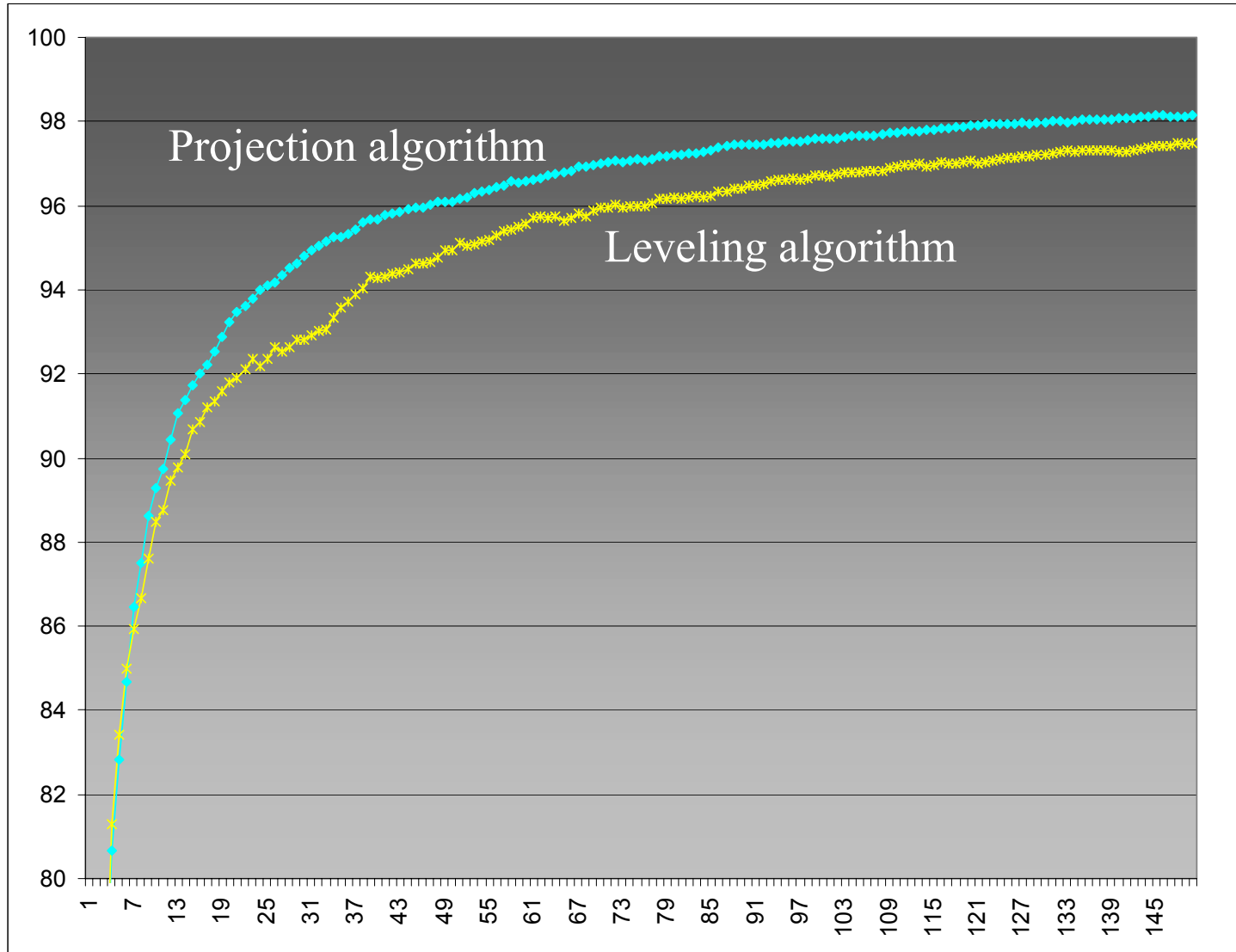
- Theorem (Topaloglu and Powell, 2001):

The leveling algorithm is a.s. convergent under the assumption that all points on the curve are sampled infinitely often.

- Theorem (Powell, Ruszczyński and Topaloglu, 2002):

The projection algorithm is convergent almost surely under the assumption that each point on the curve satisfies a "probing condition."

Comparison



Adaptive learning

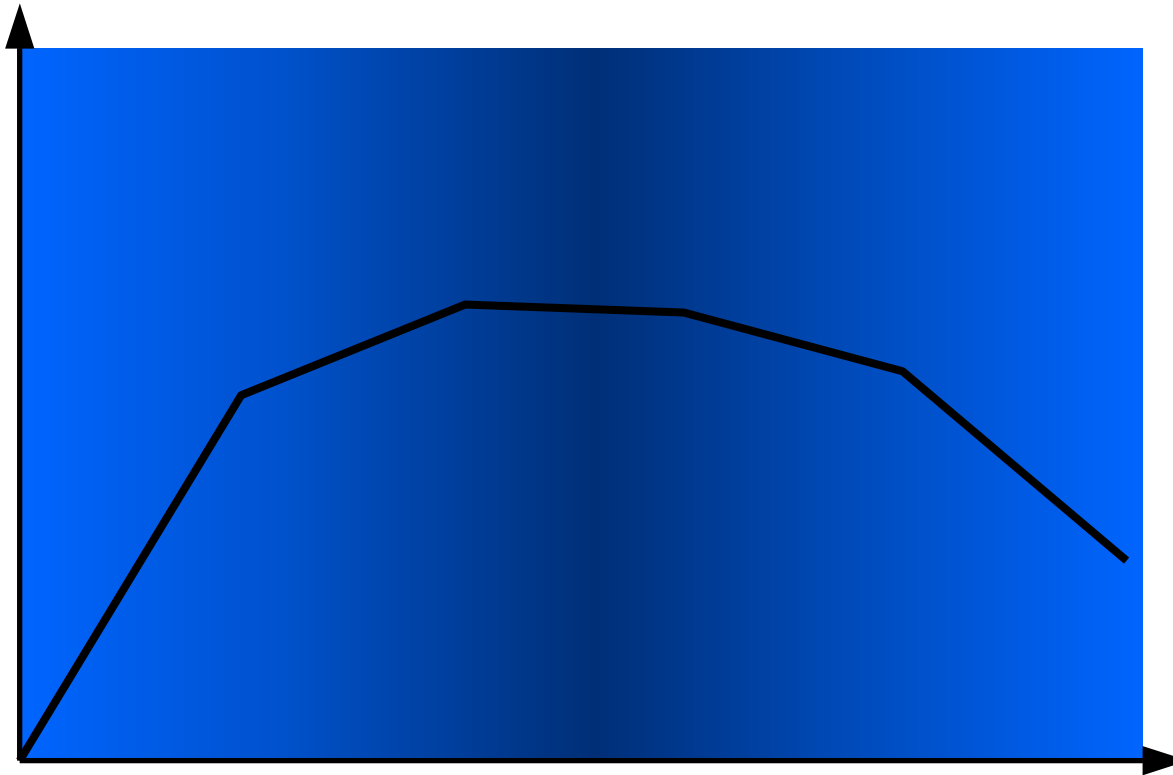
- A uniform sampling strategy:
 - » Second stage sampled based on exogenous distribution.



Adaptive learning

- An optimizing sampling strategy:

$$x^n = \arg \max_x c_0 x + \hat{V}^n(R(x))$$

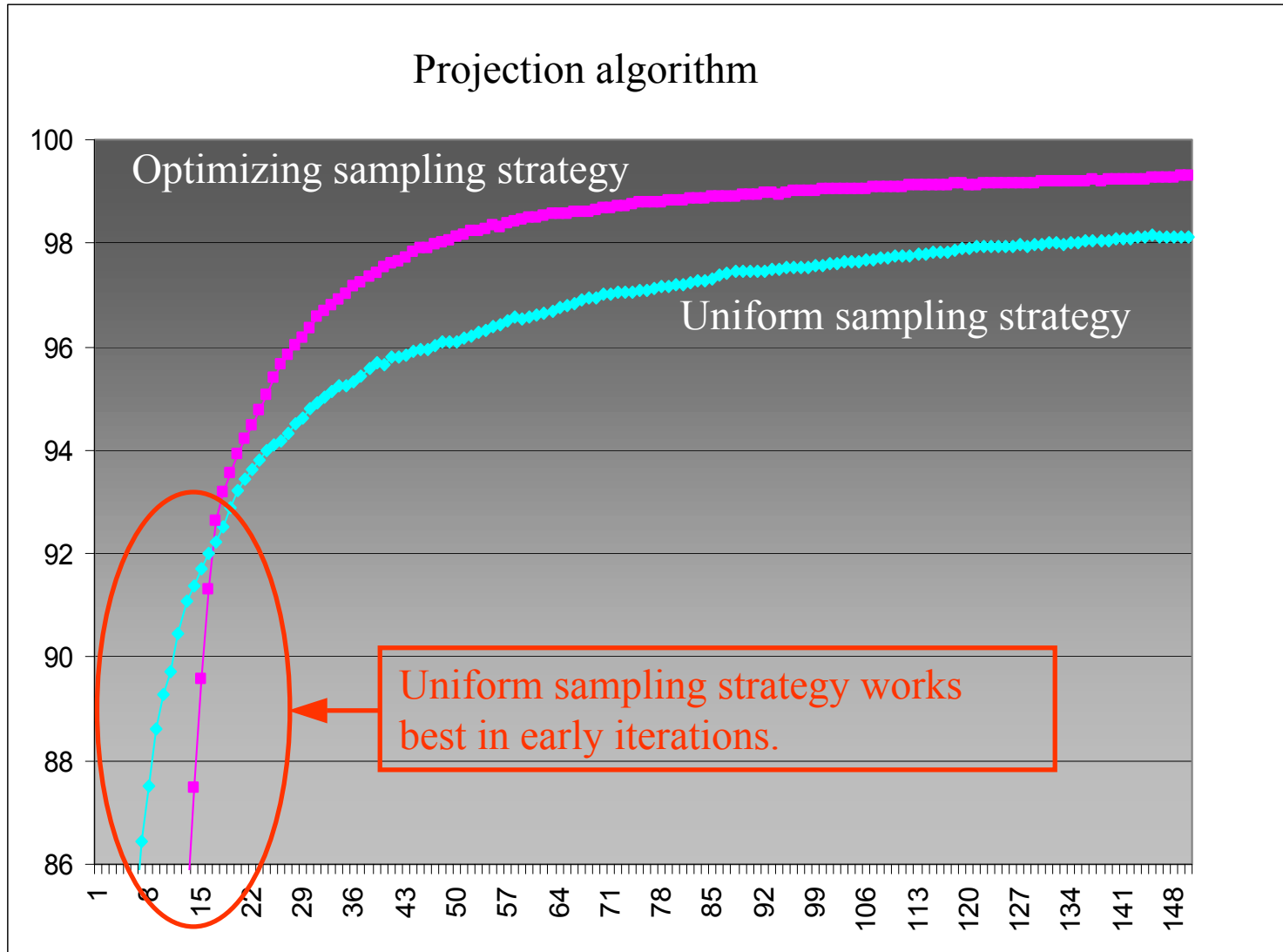


Learning strategies

- Theorem (Powell, Ruszczyński and Topaloglu, 2002):

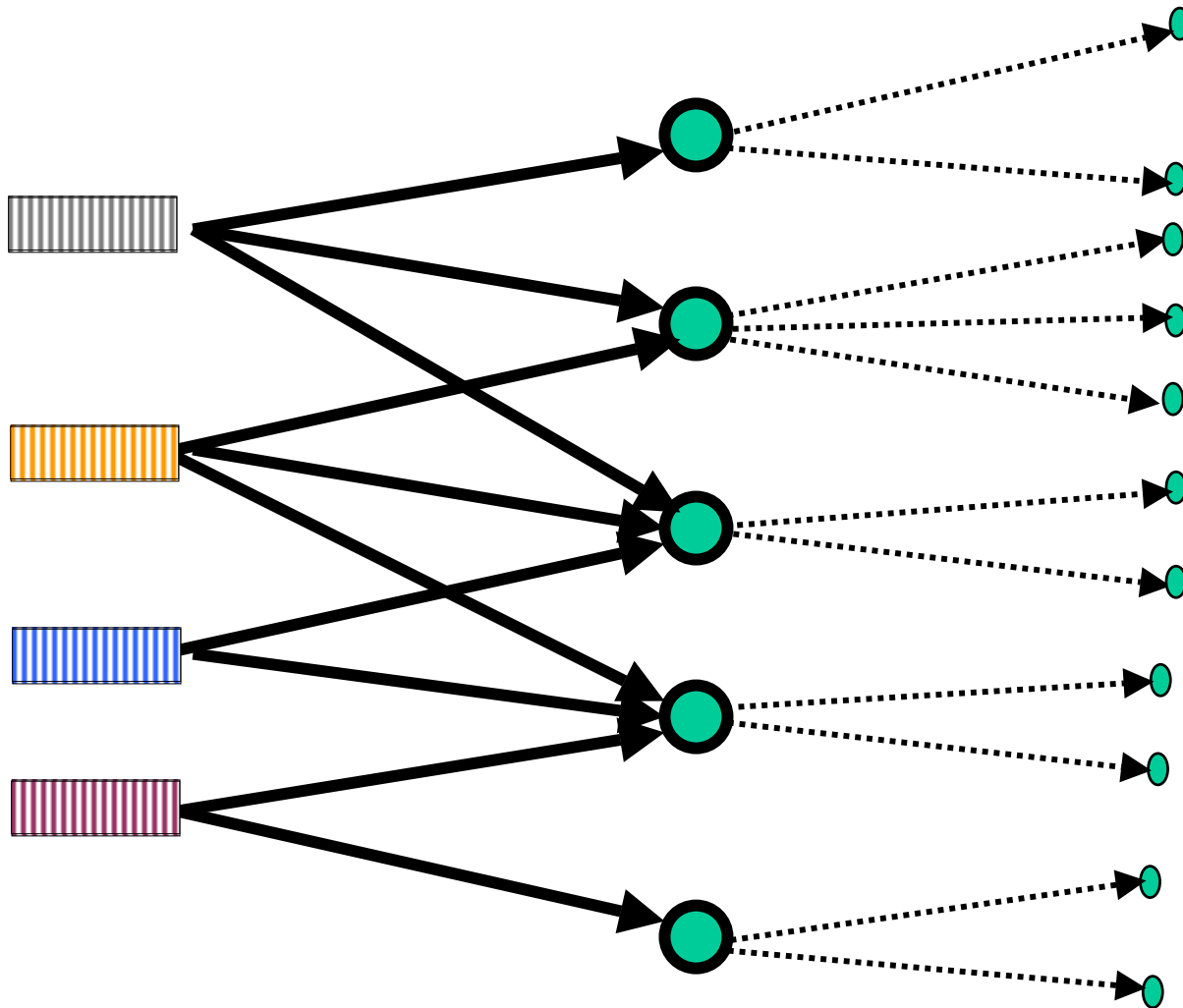
The sequence of solutions $(x^n) \xrightarrow{n \rightarrow \infty} x^$ almost surely under an optimizing algorithm, where the second stage points are sampled based on solutions of approximations of the problem at each iteration, and when the second stage recourse function is separable.*

Learning strategies



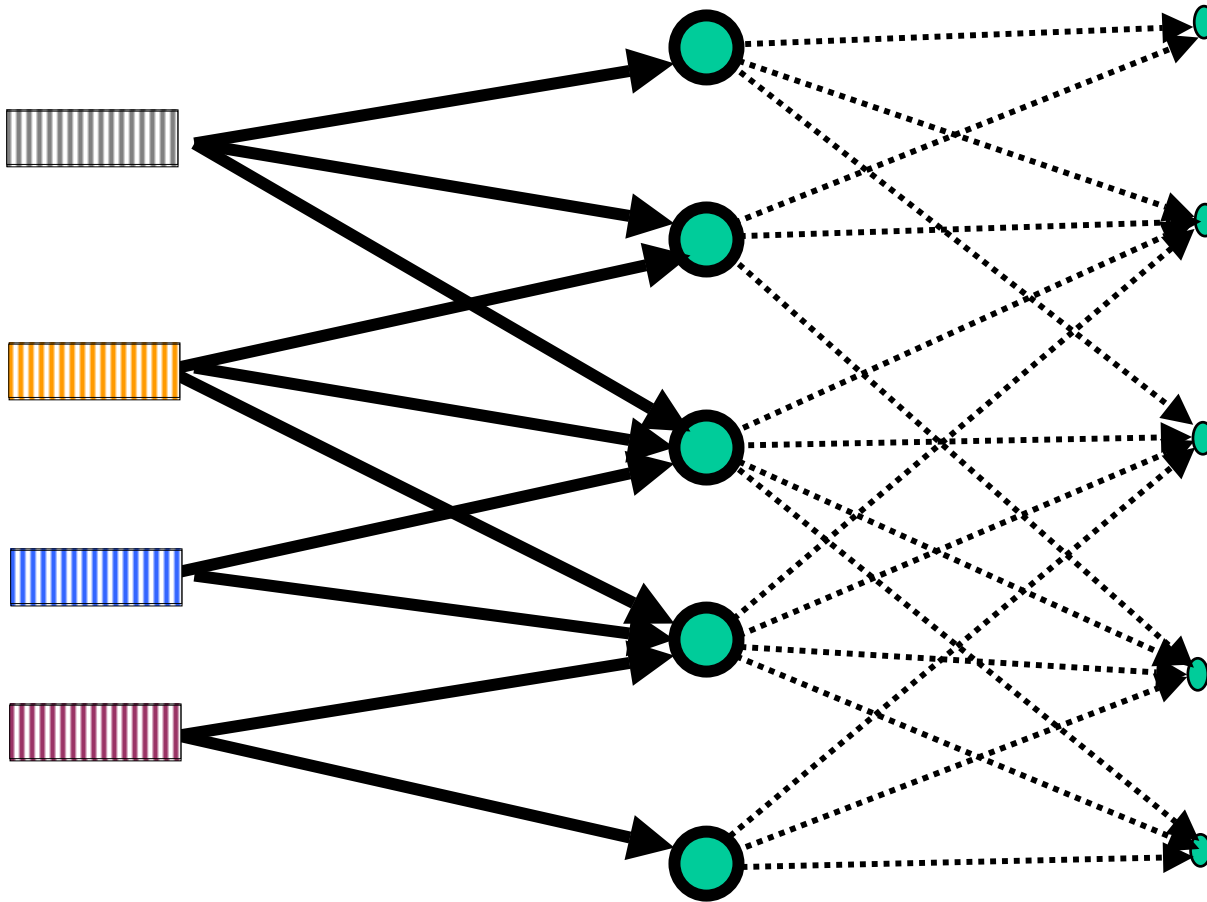
Two-stage experiments

- Second stage may be separable.....



Two-stage experiments

- ... or nonseparable (substitutable resources)



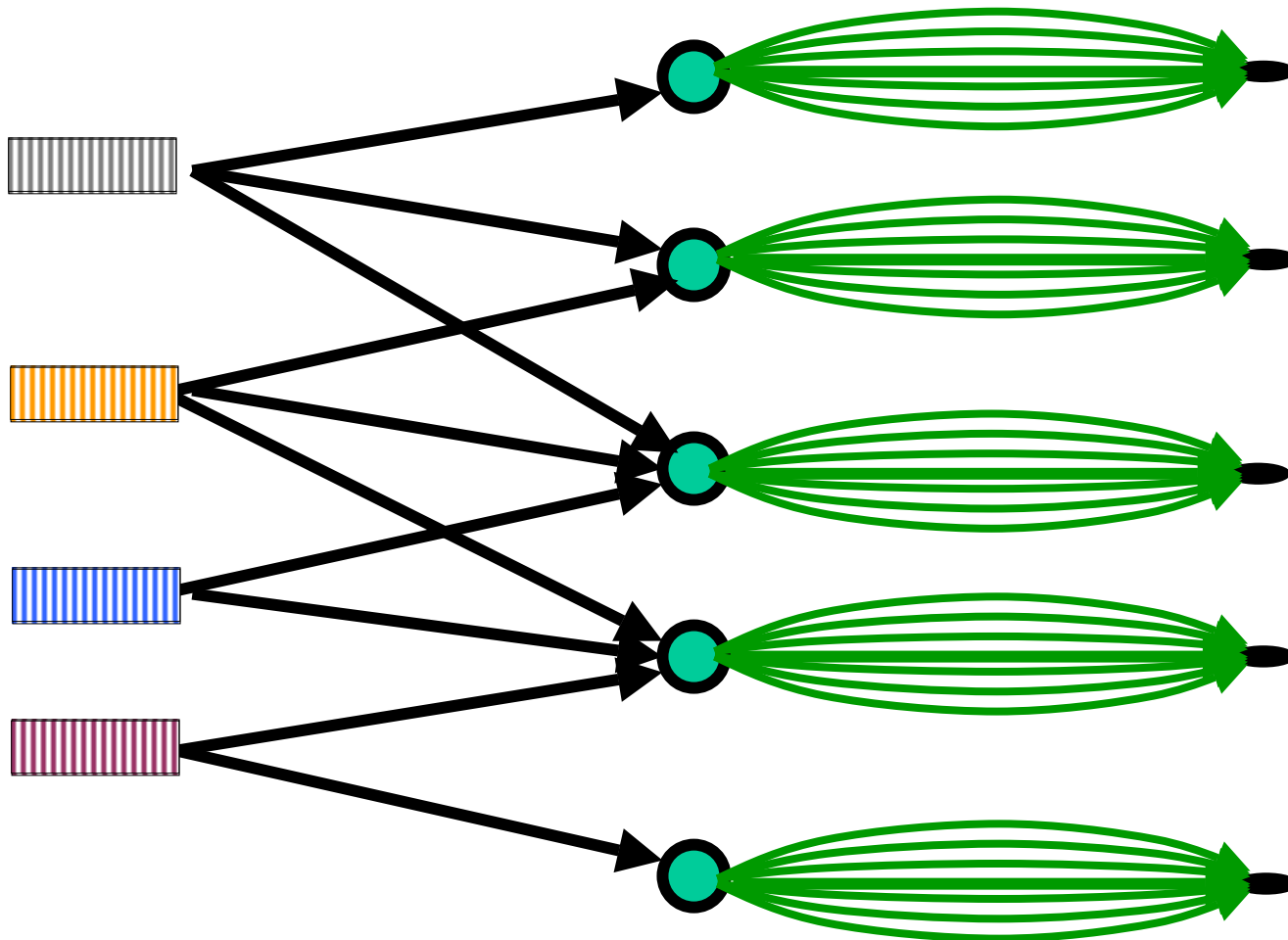
Two-stage experiments

■ Notes:

- » The use of separable approximations for two-stage problems can produce algorithms that produce solutions that converge *almost surely* to the optimal solution *if the second stage problem is continuously differentiable* (Cheung and Powell, *Operations Research*, 2000).
- » The use of separable approximations for two-stage problems will *not* generally produce optimal solutions when the second stage is nondifferentiable and nonseparable.

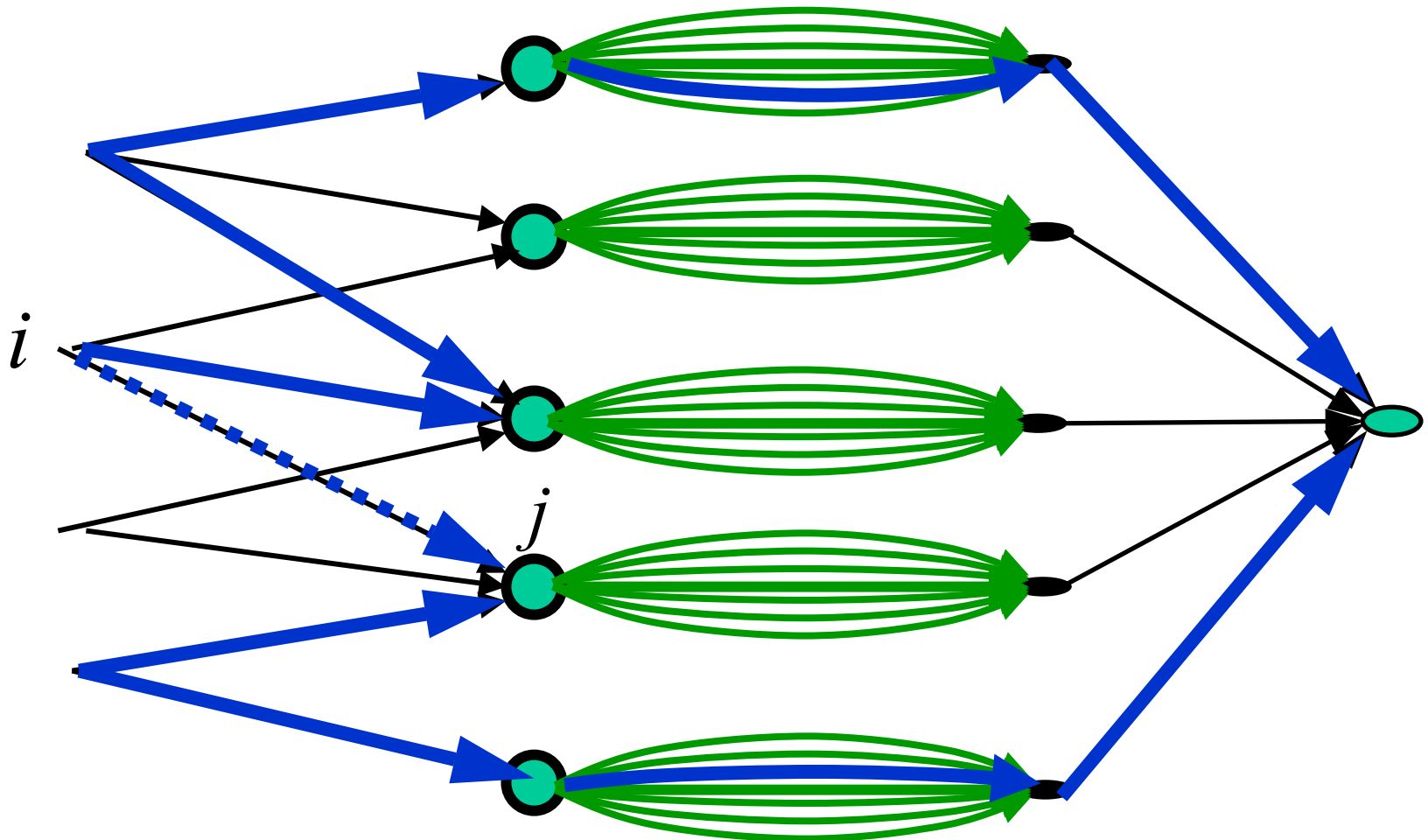
Two-stage experiments

- We are solving problems with the structure:



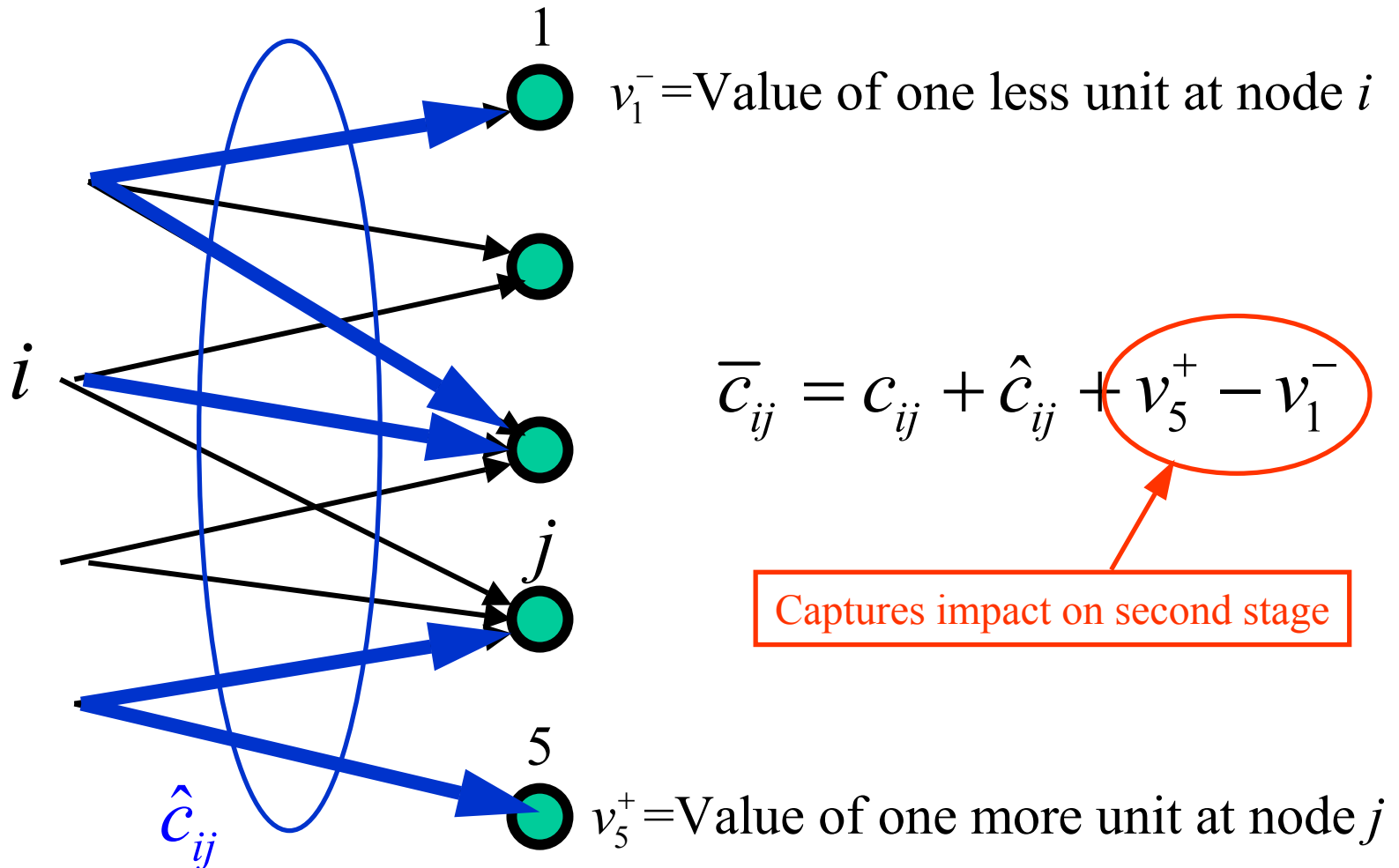
Two-stage experiments

- Duals reflect flow augmenting cycles through the network:



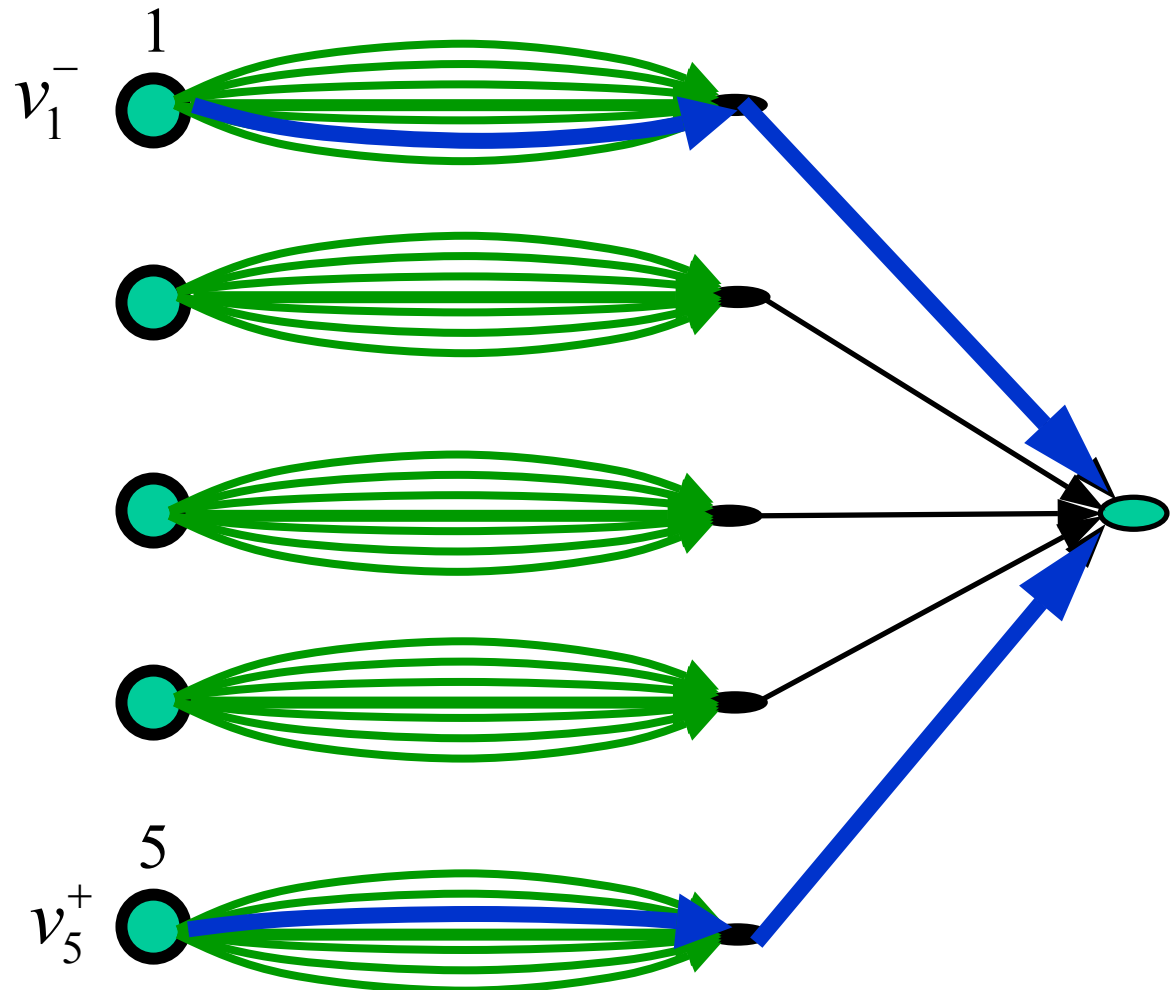
Two-stage experiments

- Duals reflect flow augmenting cycles through the network:



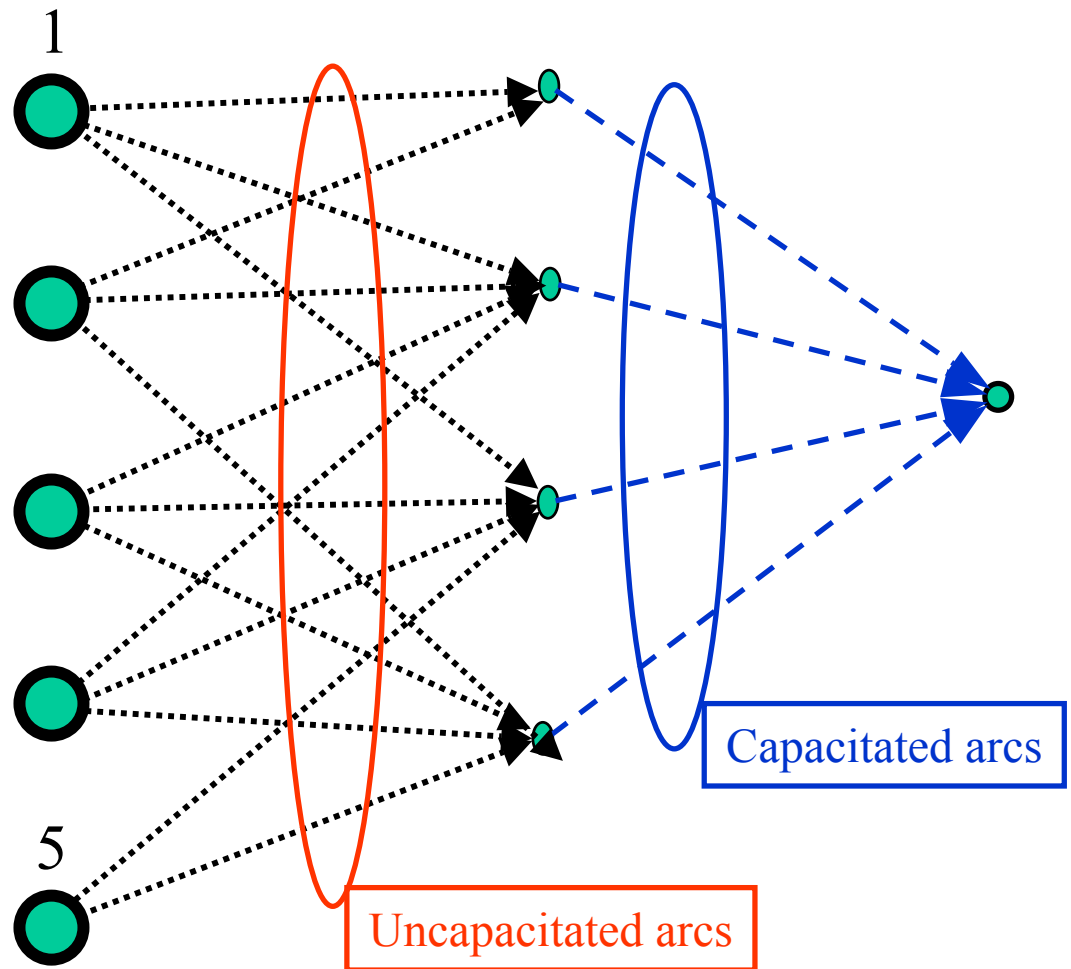
Two-stage experiments

- Duals reflect flow augmenting cycles through the network:



Two-stage experiments

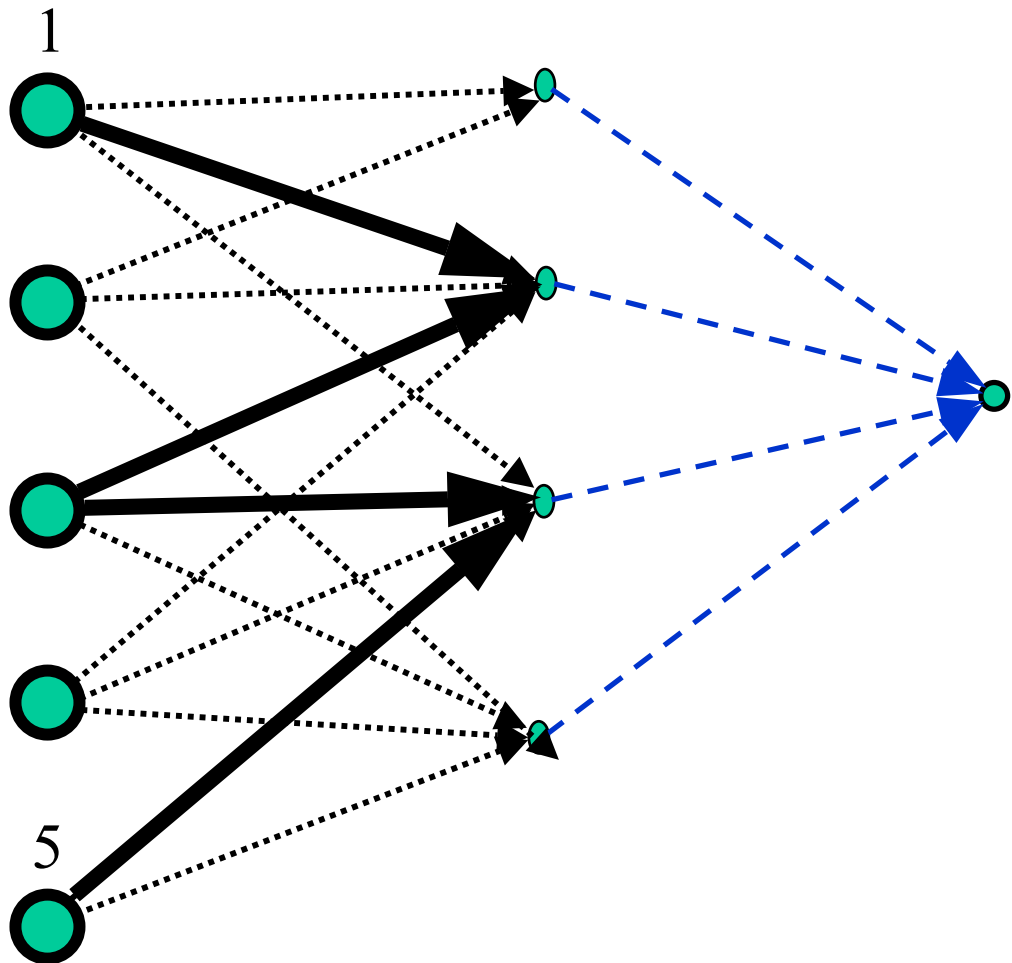
- We solve a sample realization of the second stage:



Two-stage experiments

- (Sample) flow augmenting path from 5 to 1:

$v_5 - v_1$ is a flow augmenting path from node 5 back to node 1.

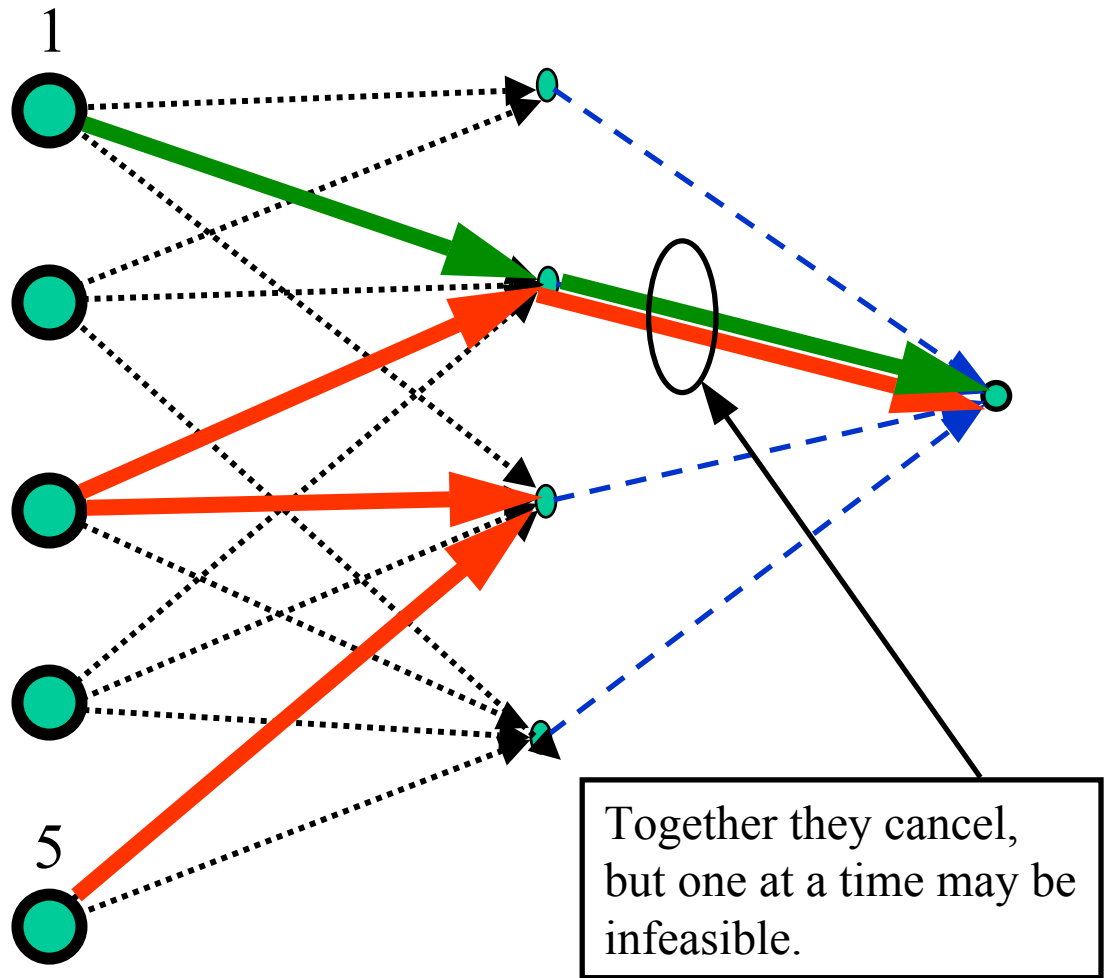


Two-stage experiments

- (Sample) flow augmenting path from 5 to 1:

We compute v_5 as the flow augmenting path from 5 to the supersink.

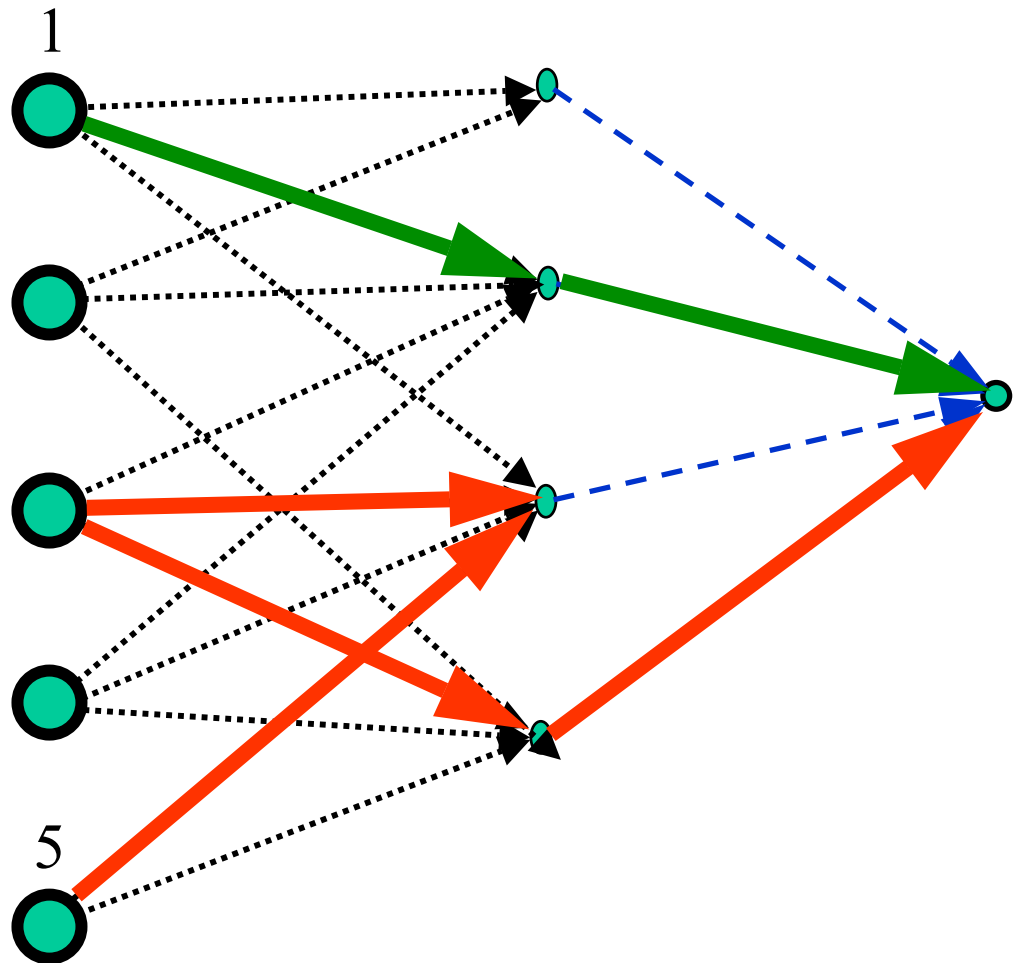
v_1 is a flow augmenting path from the supersink to node 1.



Two-stage experiments

- (Sample) flow augmenting path from 5 to 1:

The flow augmenting path from 1 to 5 through the supersink can be more costly than the path from 1 to 5.

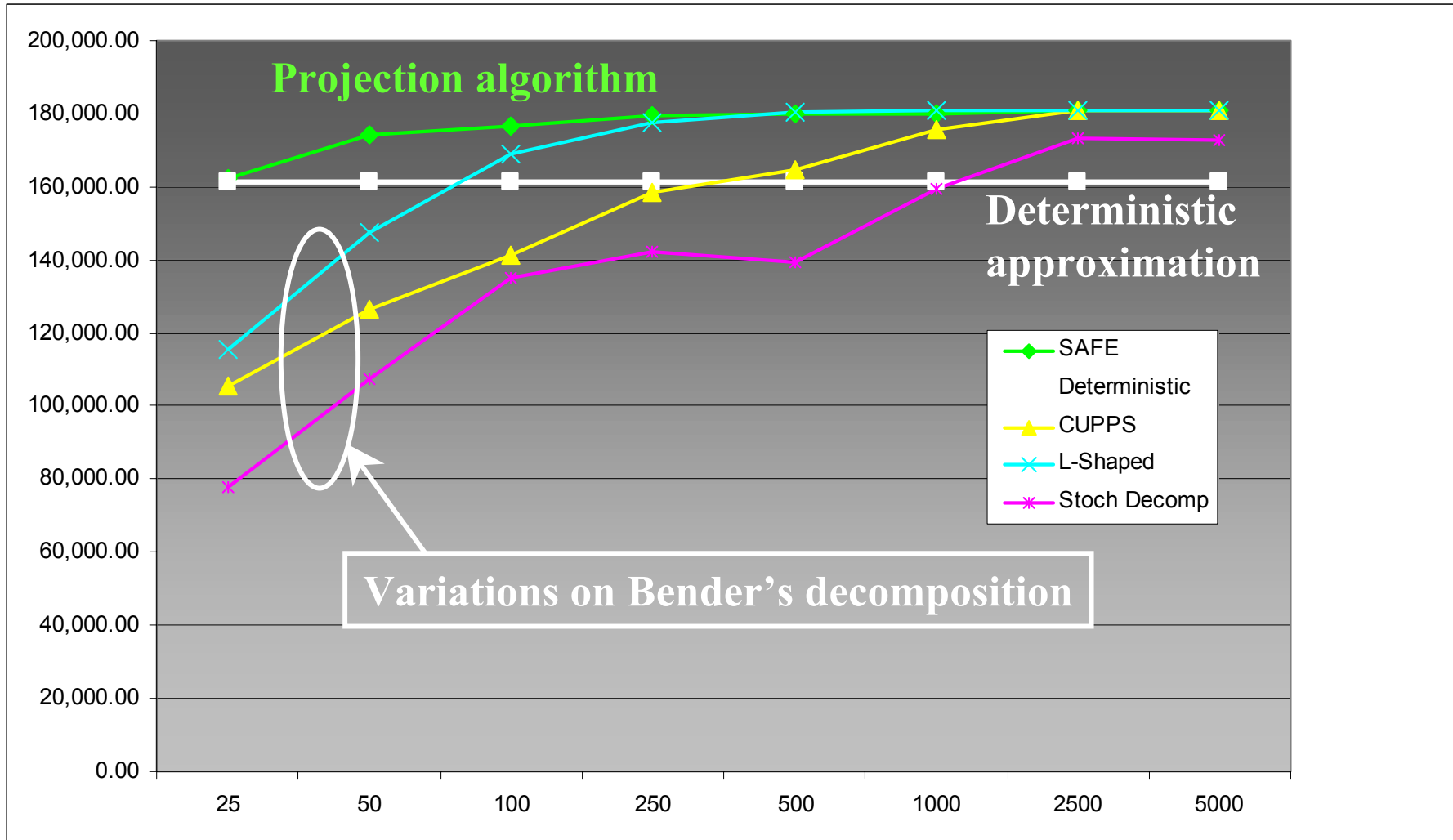


Two-stage experiments

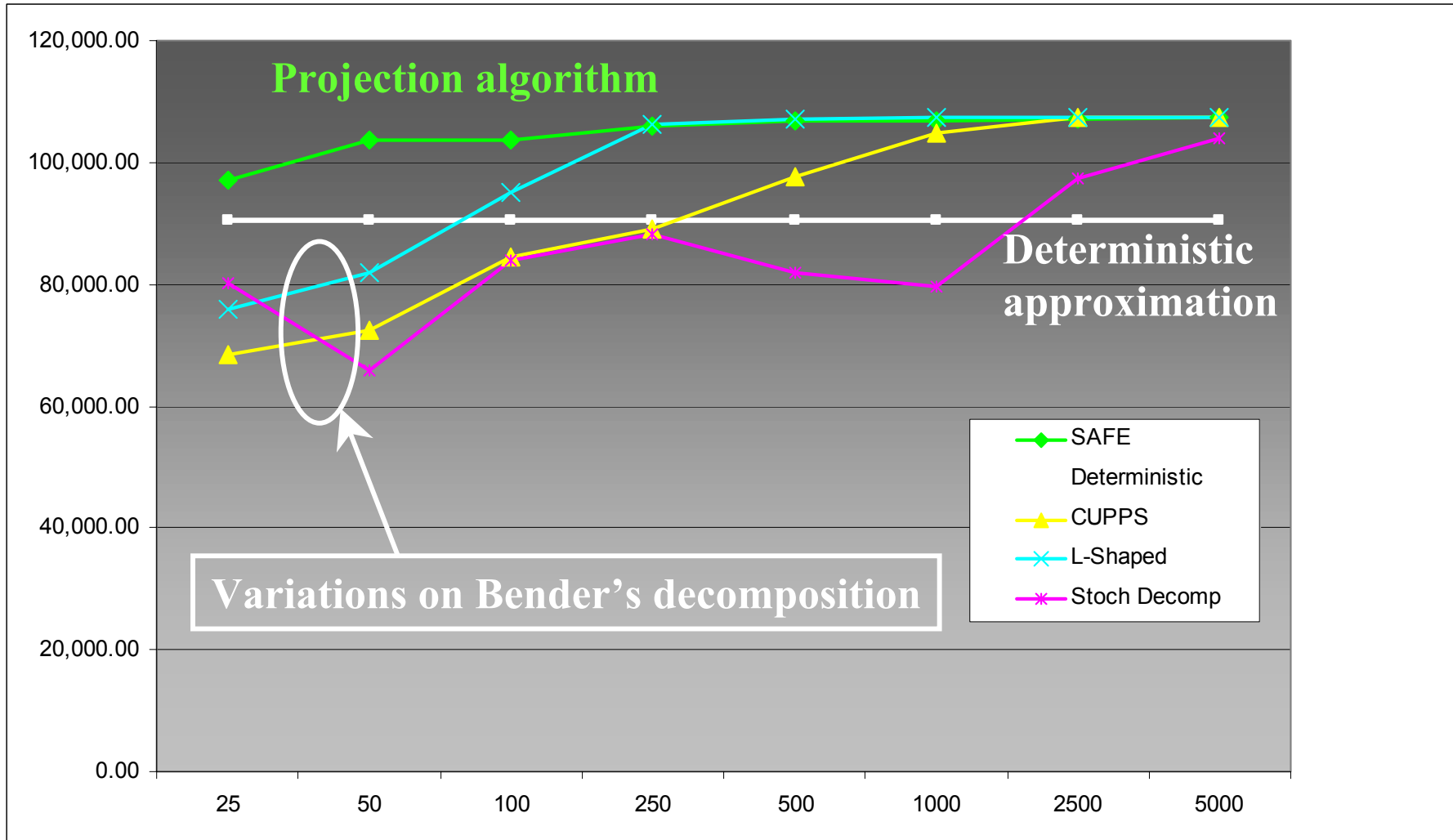
■ Questions:

- » How good is our approximation?
- » What is the speed of convergence?
- » How do we compare against deterministic approximations?

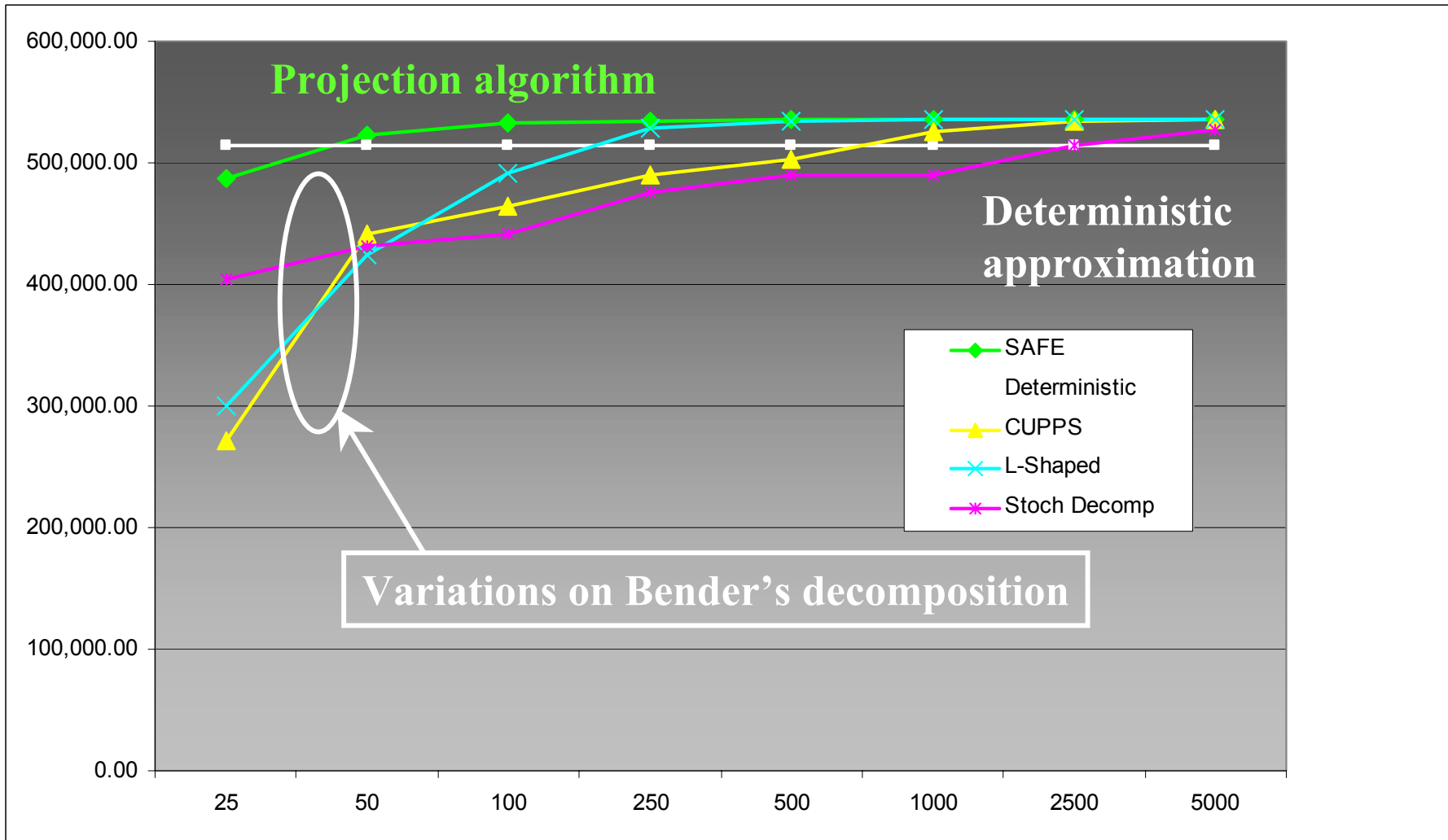
Two-stage experiments



Two-stage experiments

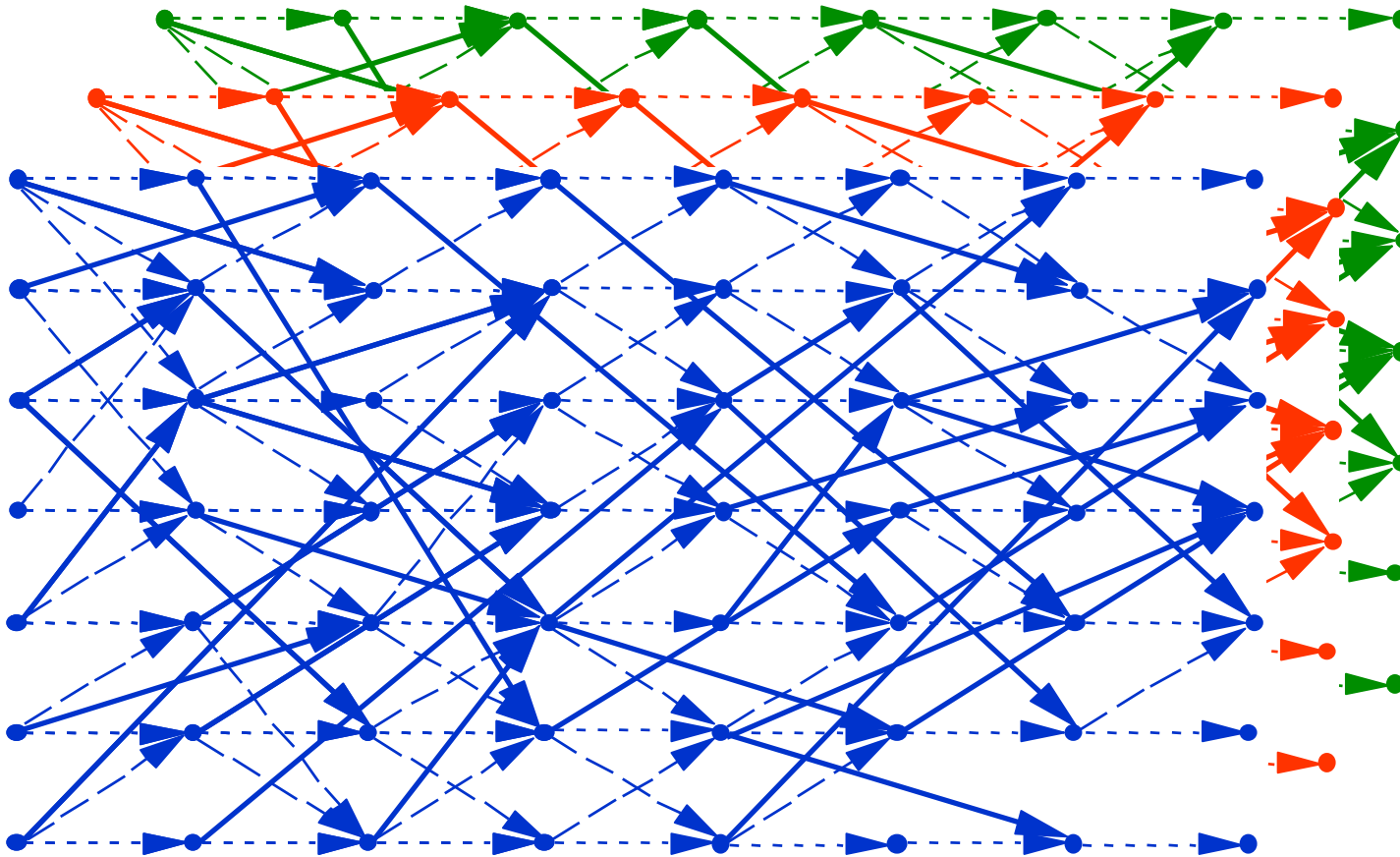


Two-stage experiments

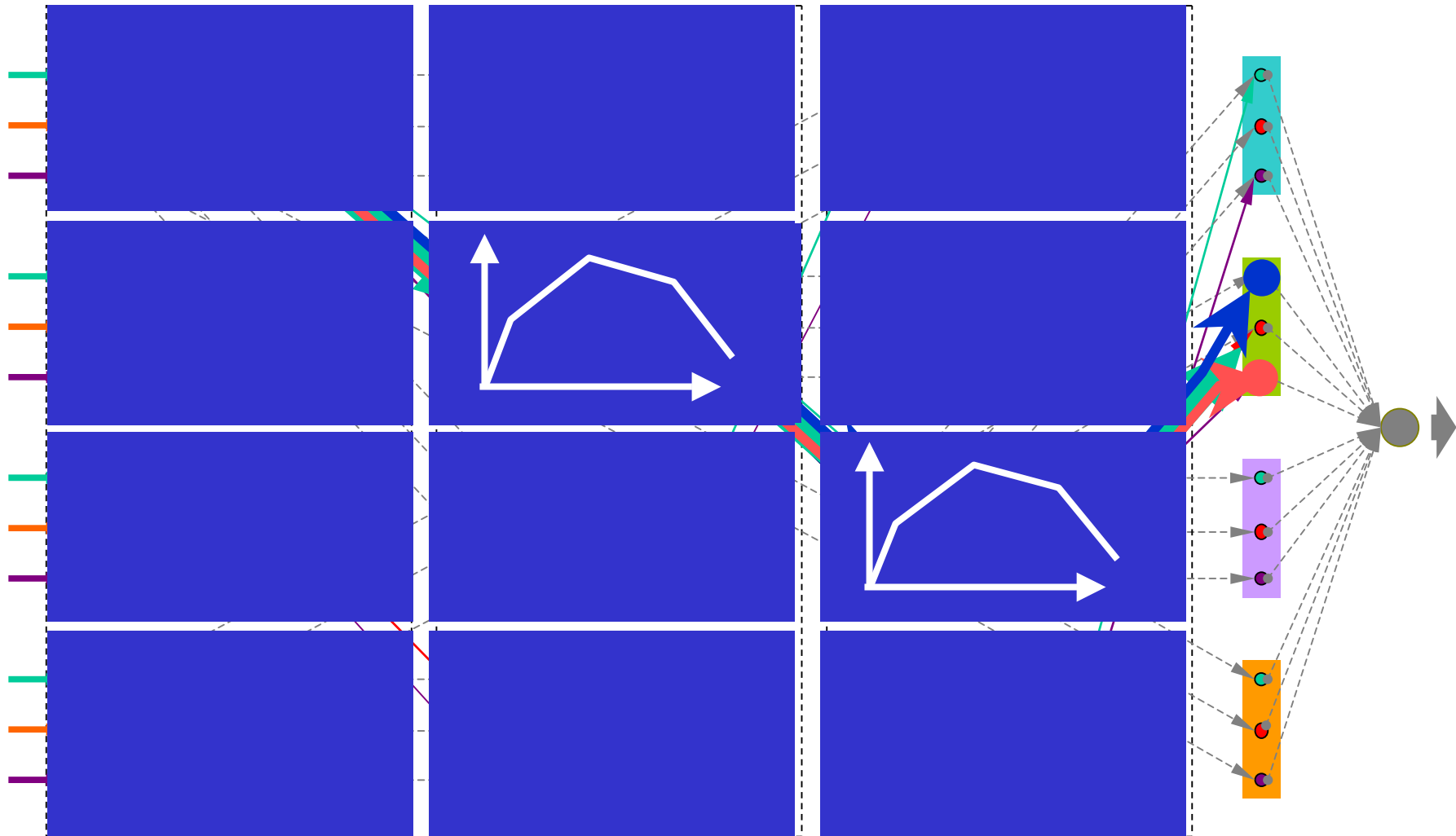


Multistage experiments

■ Multicommodity flow formulation

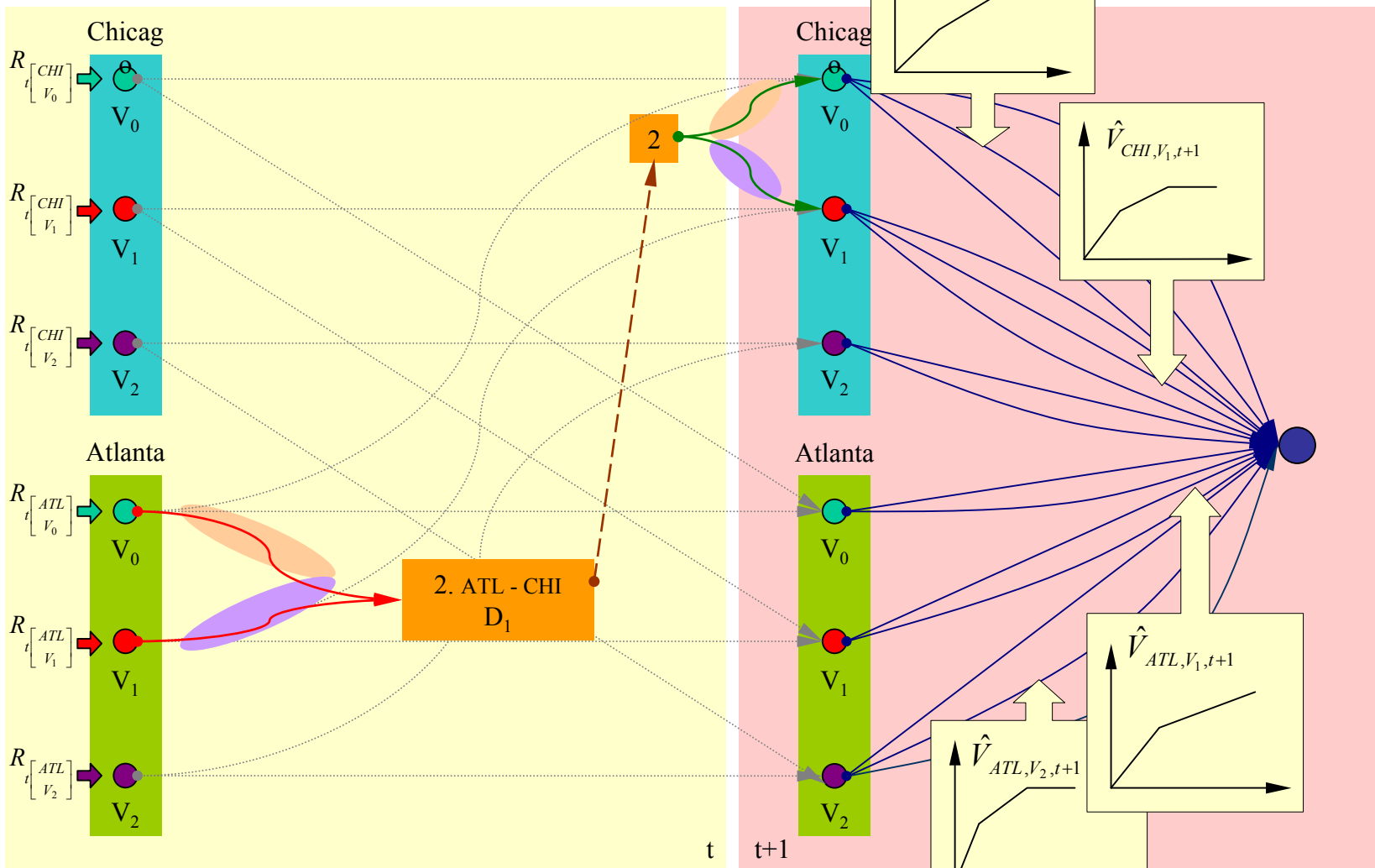


Dynamic resource allocation

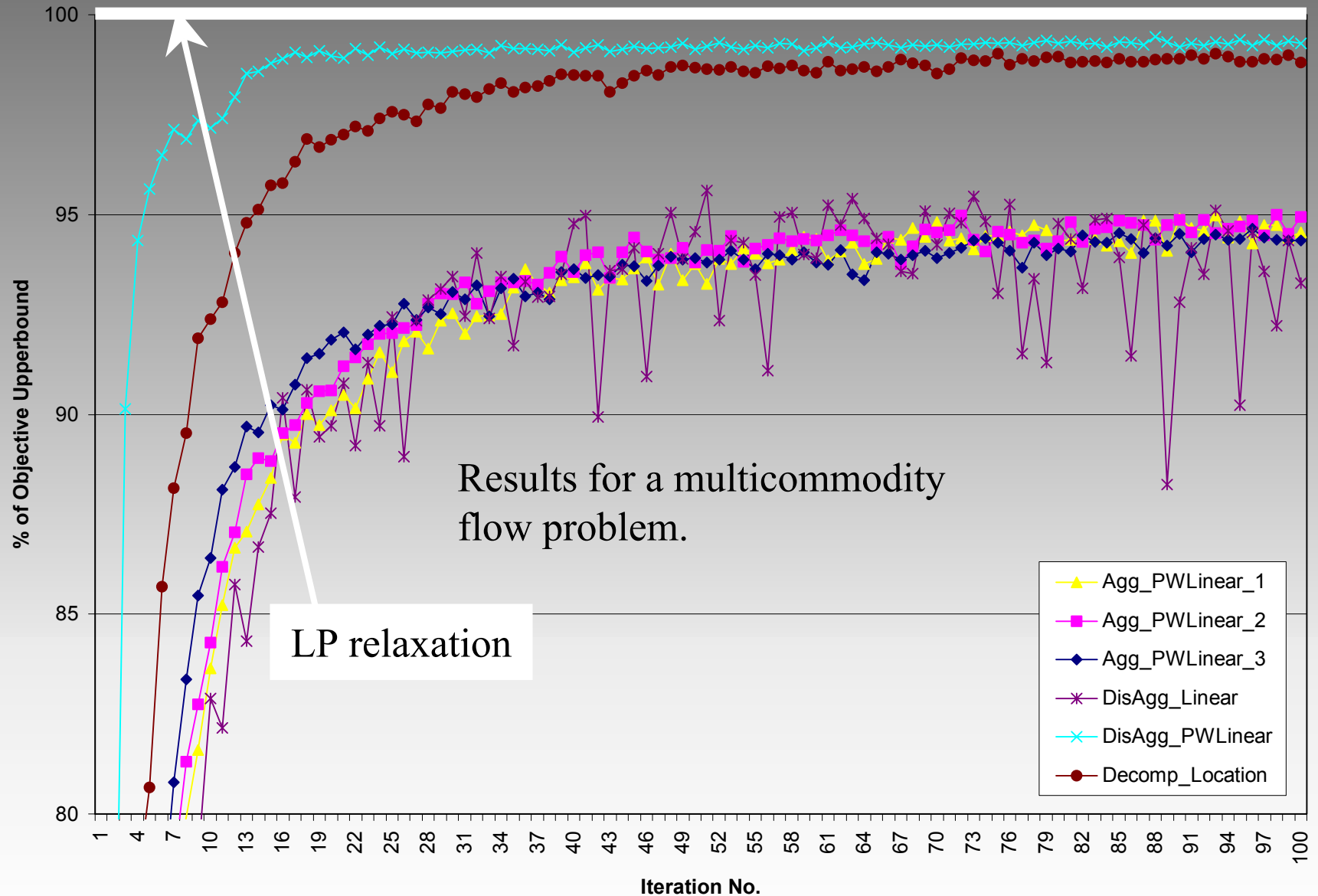


Dynamic resource allocation

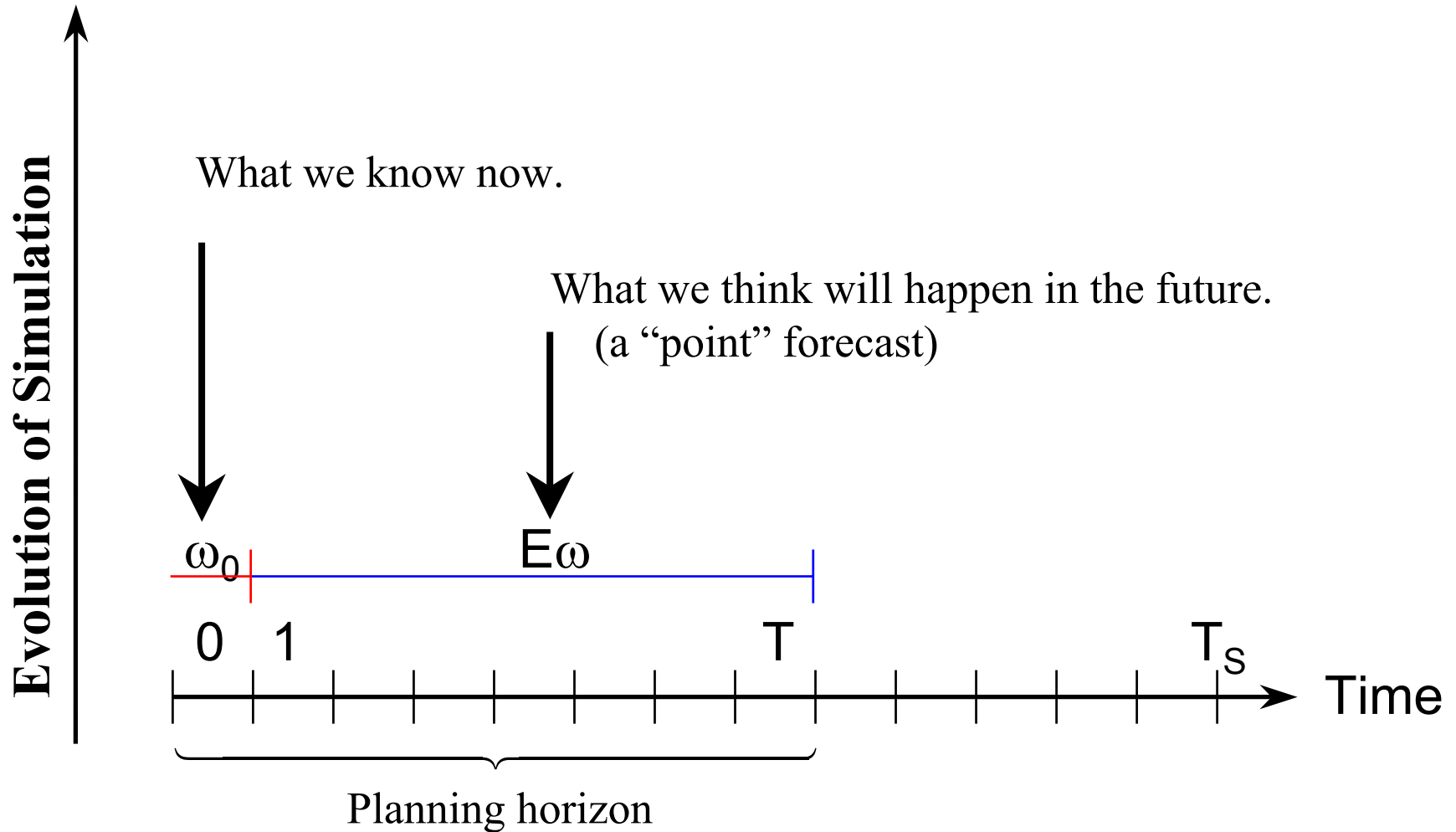
$$\max_{x_t \in X_t} c_t(x_t) + \hat{V}_{t+1}(R_{t+1})$$



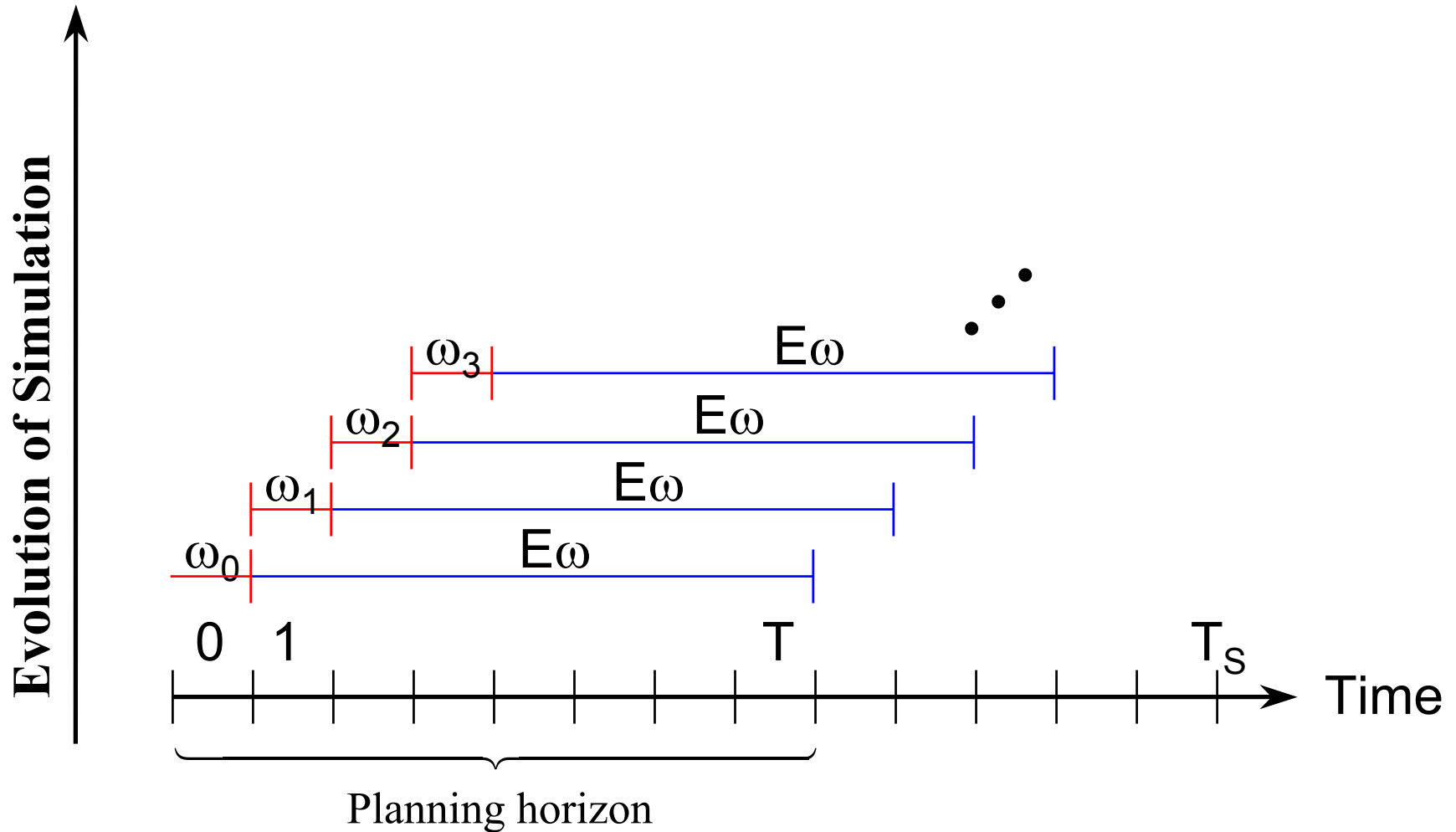
Multistage experiments



Multistage experiments

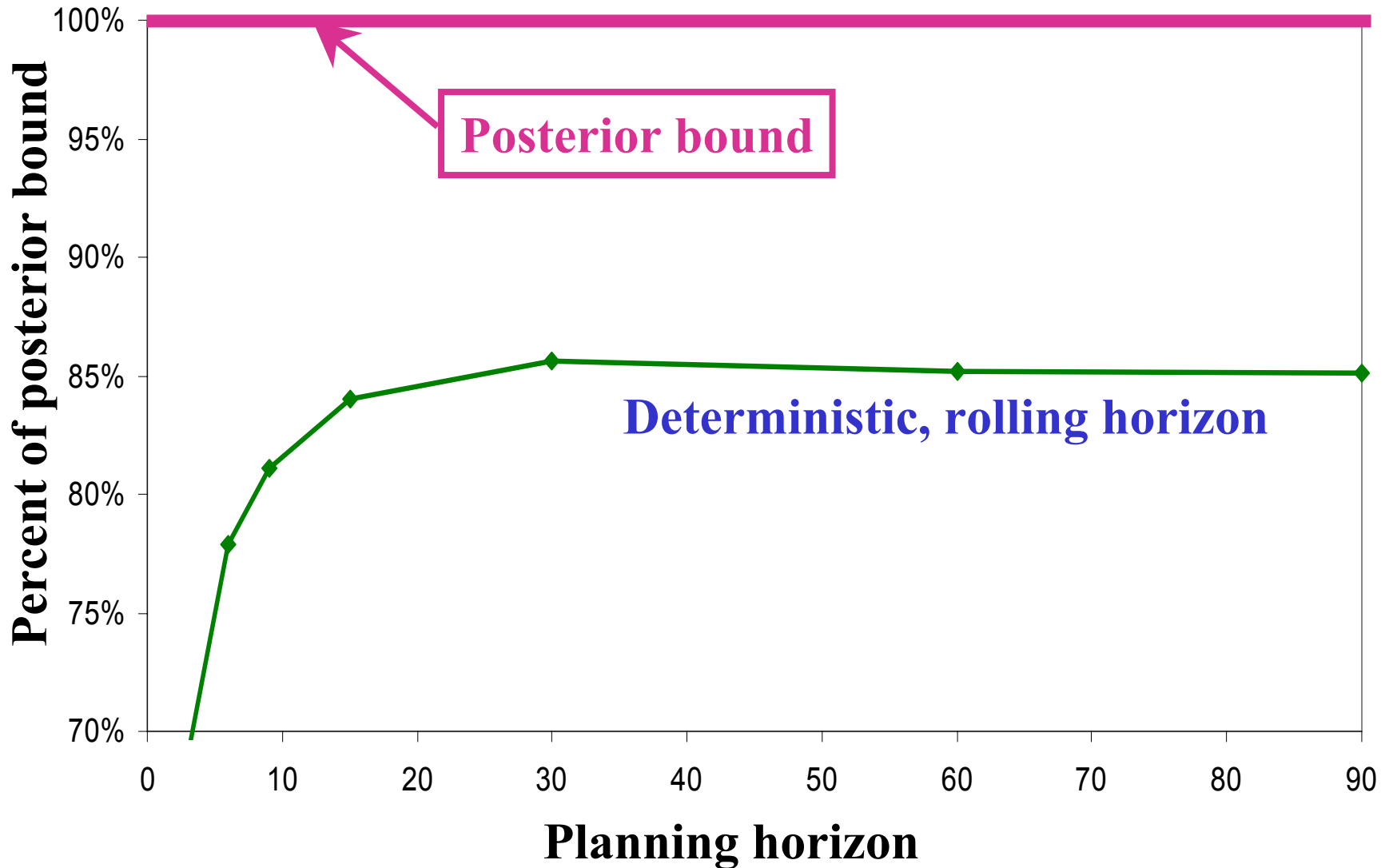


Multistage experiments

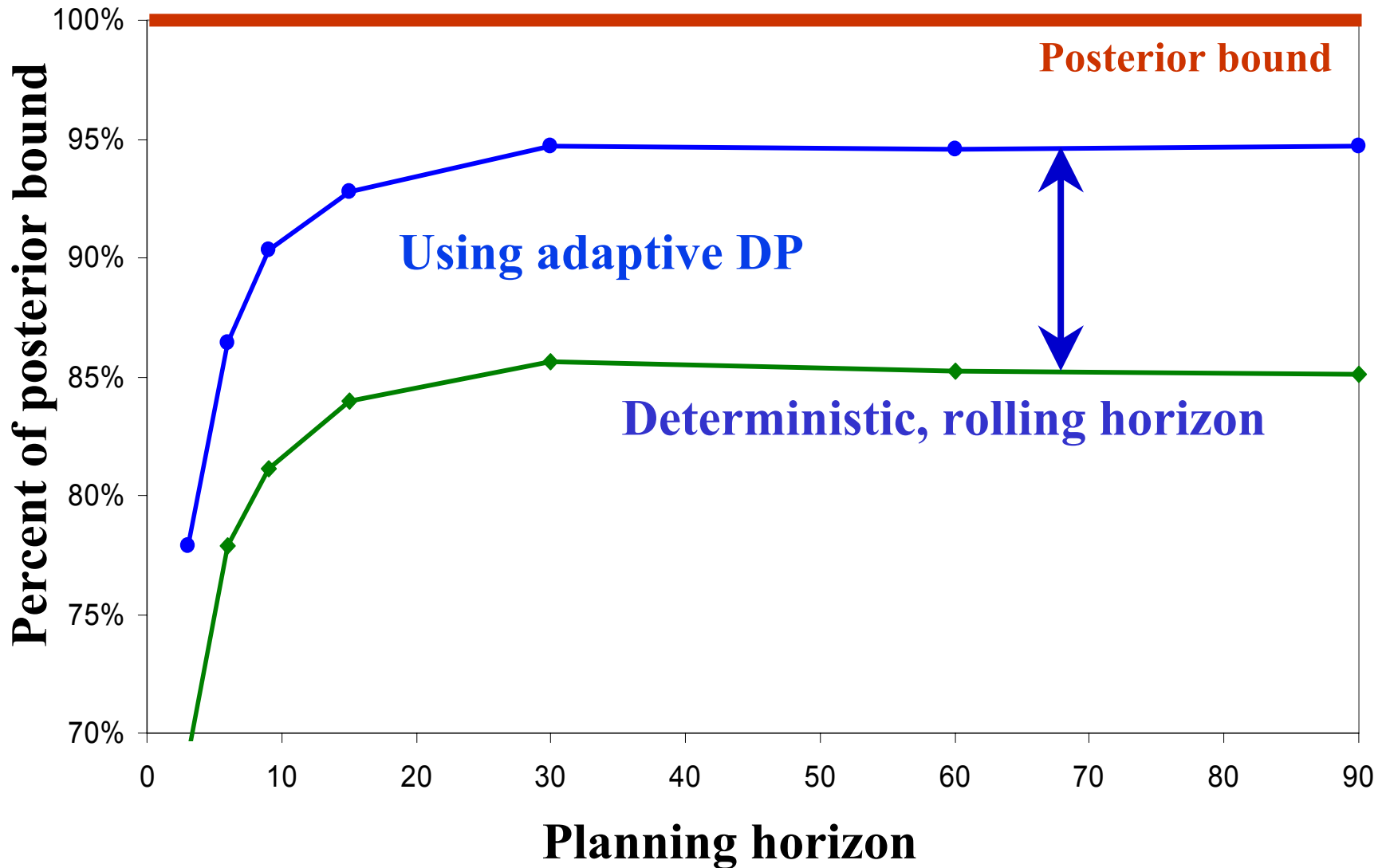


Multistage experiments

Network

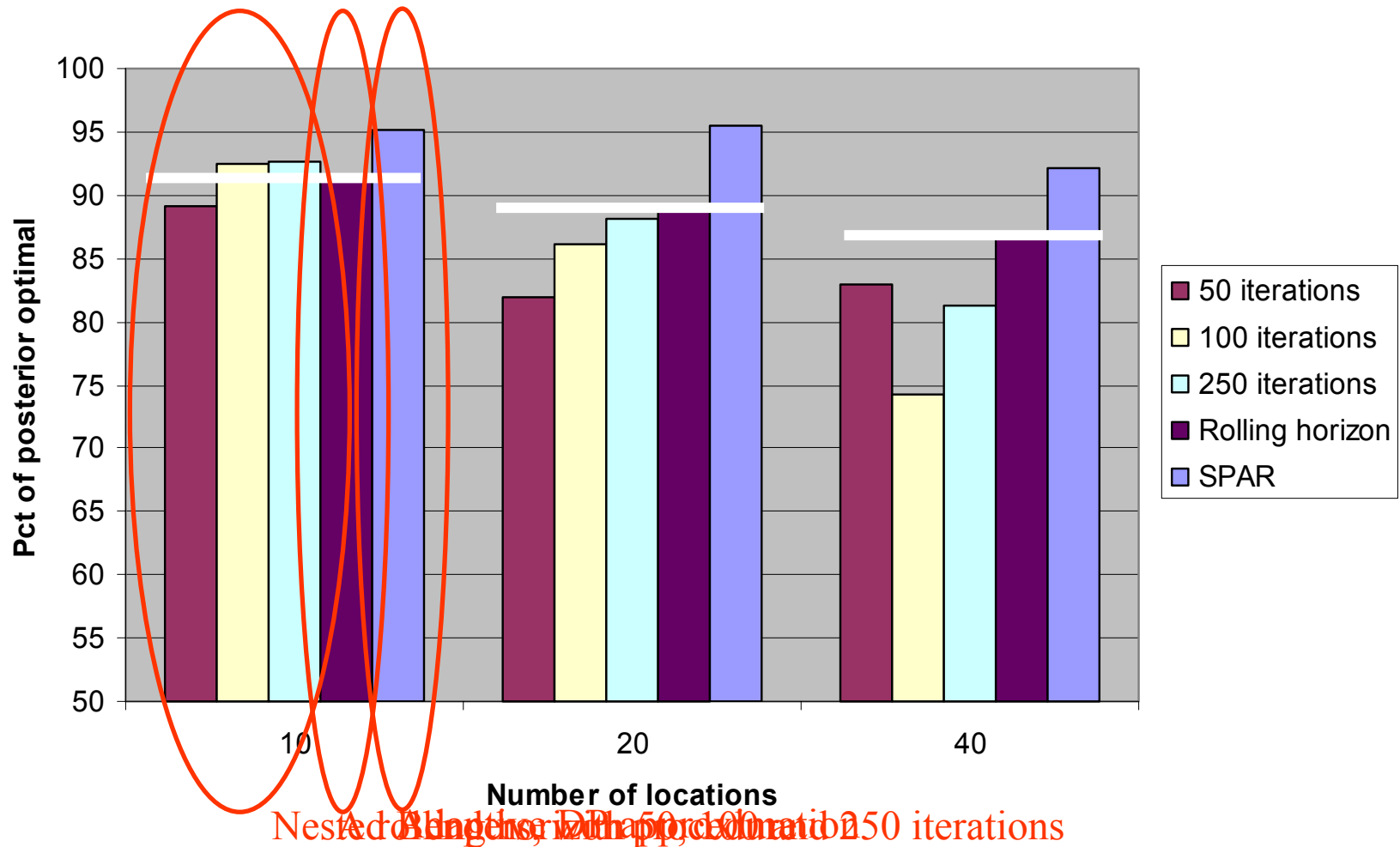


Multistage experiments

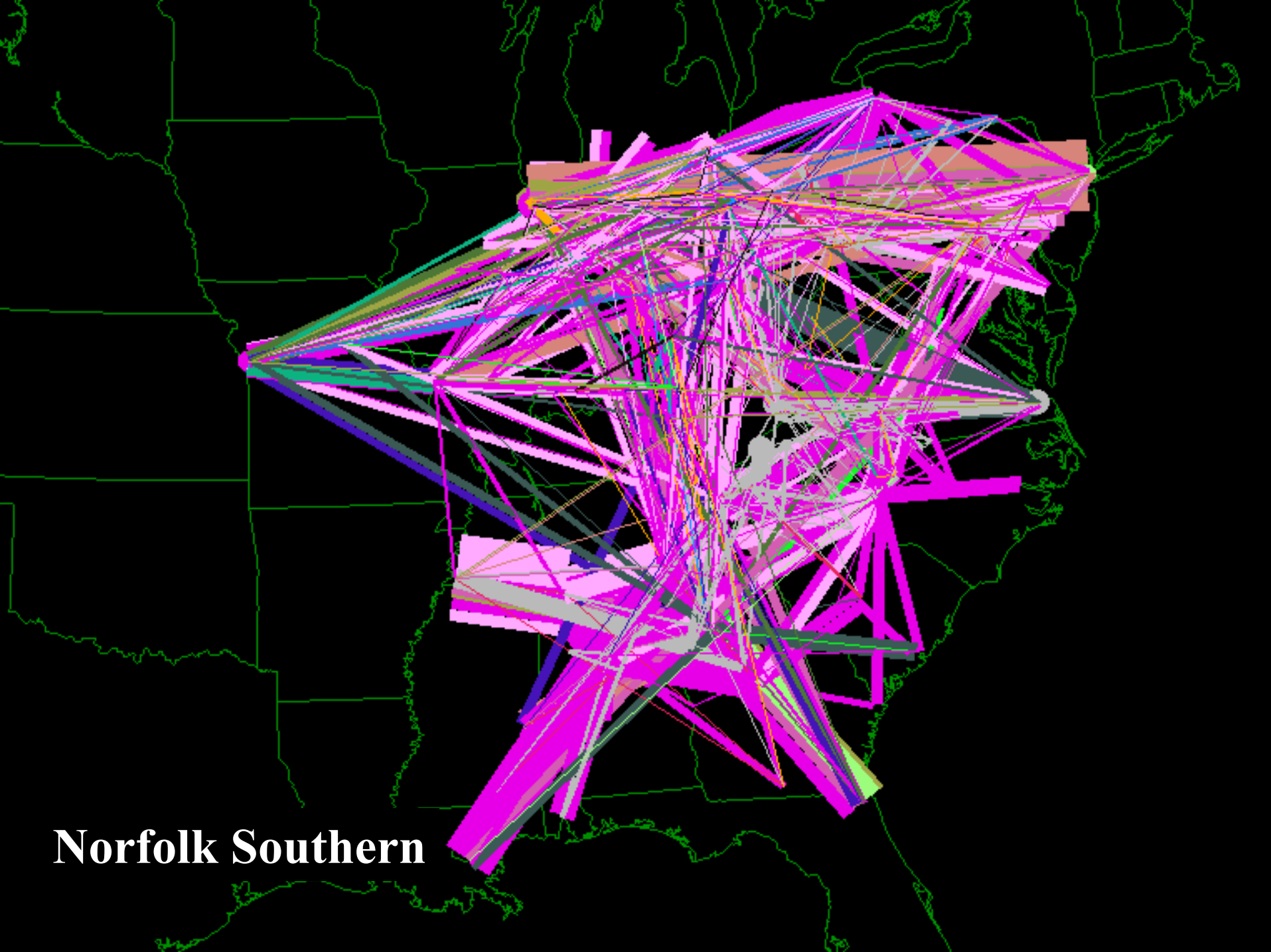


Multistage experiments

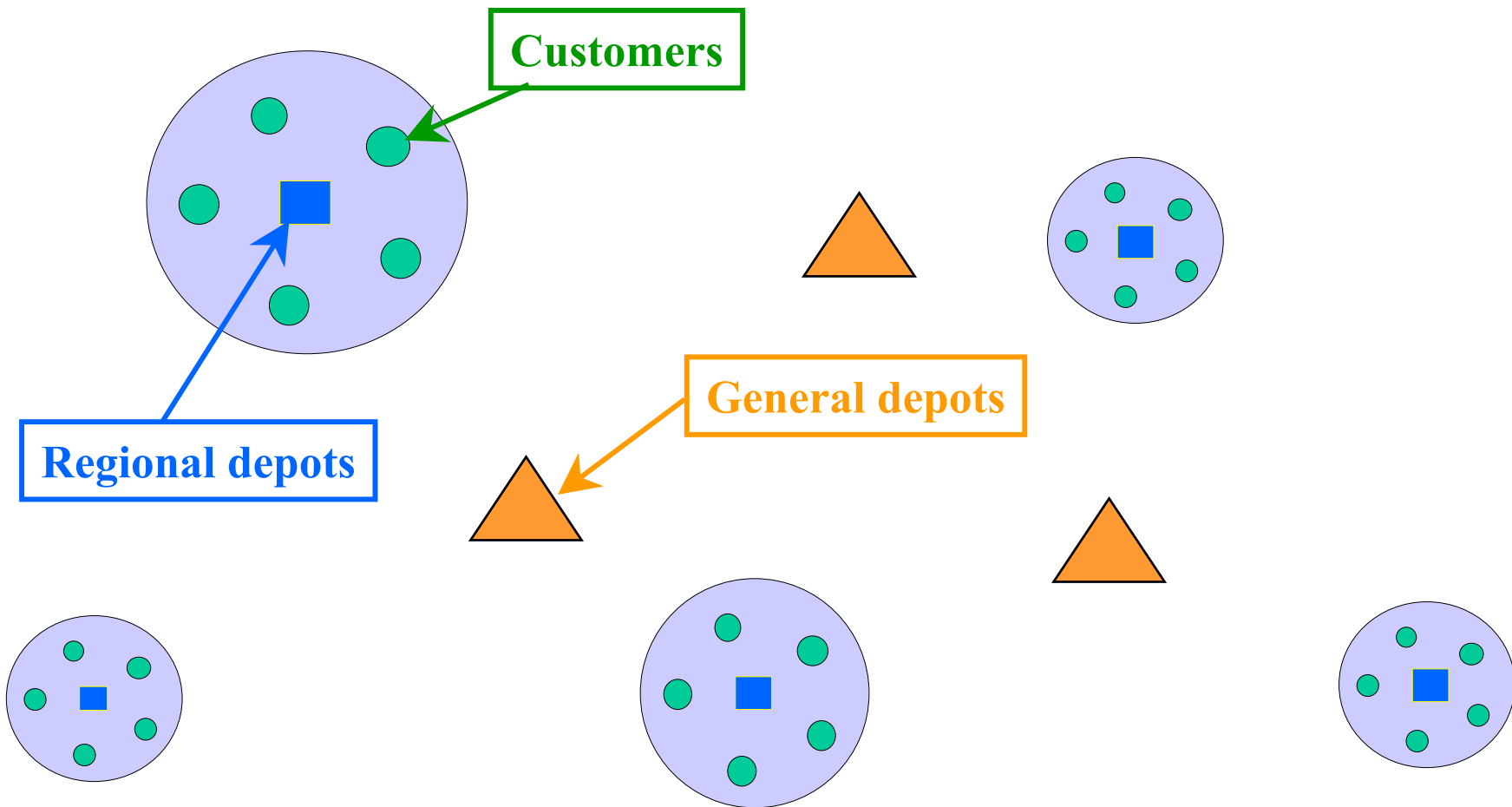
Multistage nested Benders vs. rolling horizon



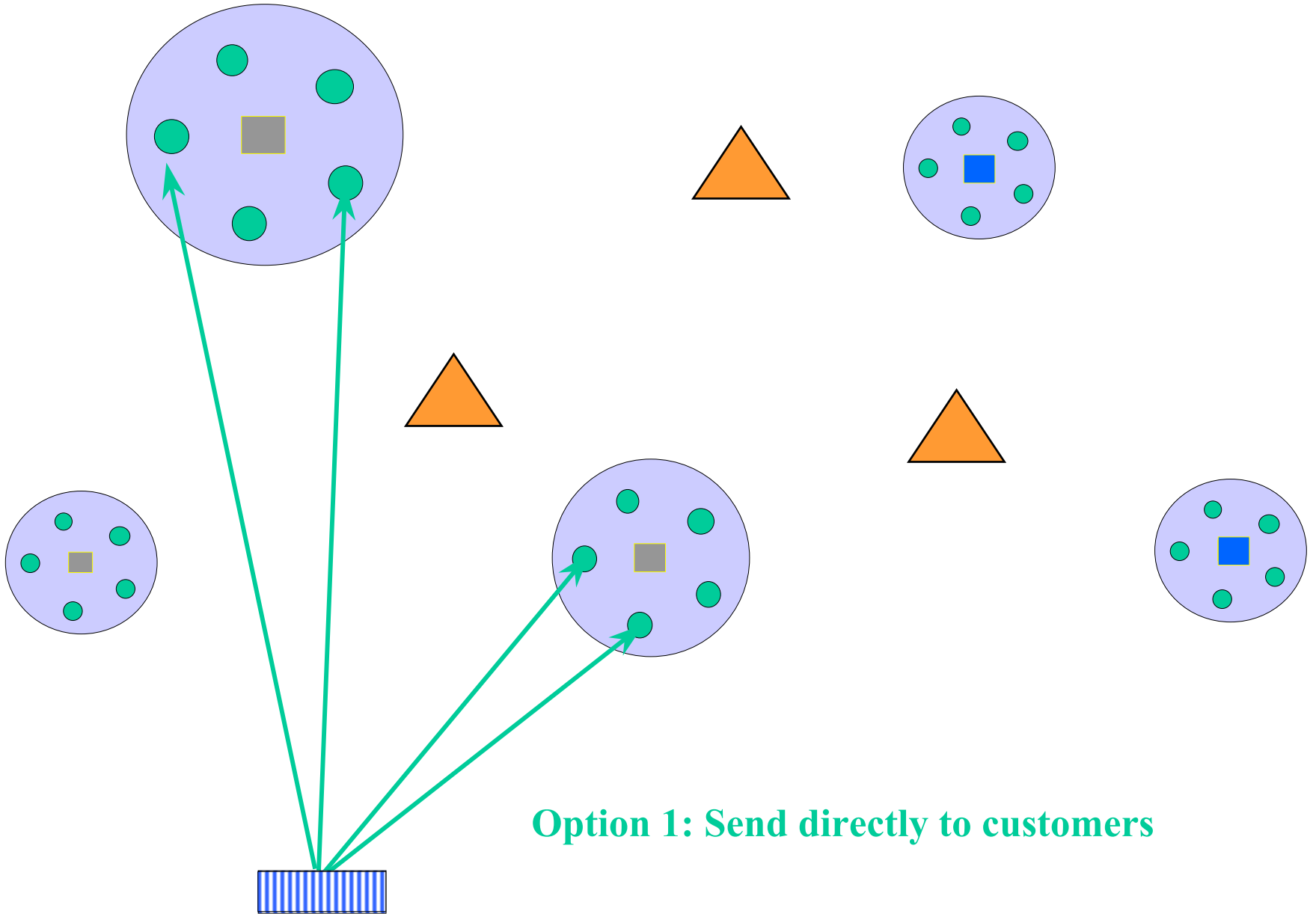
Nested Benders with 50 iterations
Nested Benders with 100 iterations
Nested Benders with 250 iterations



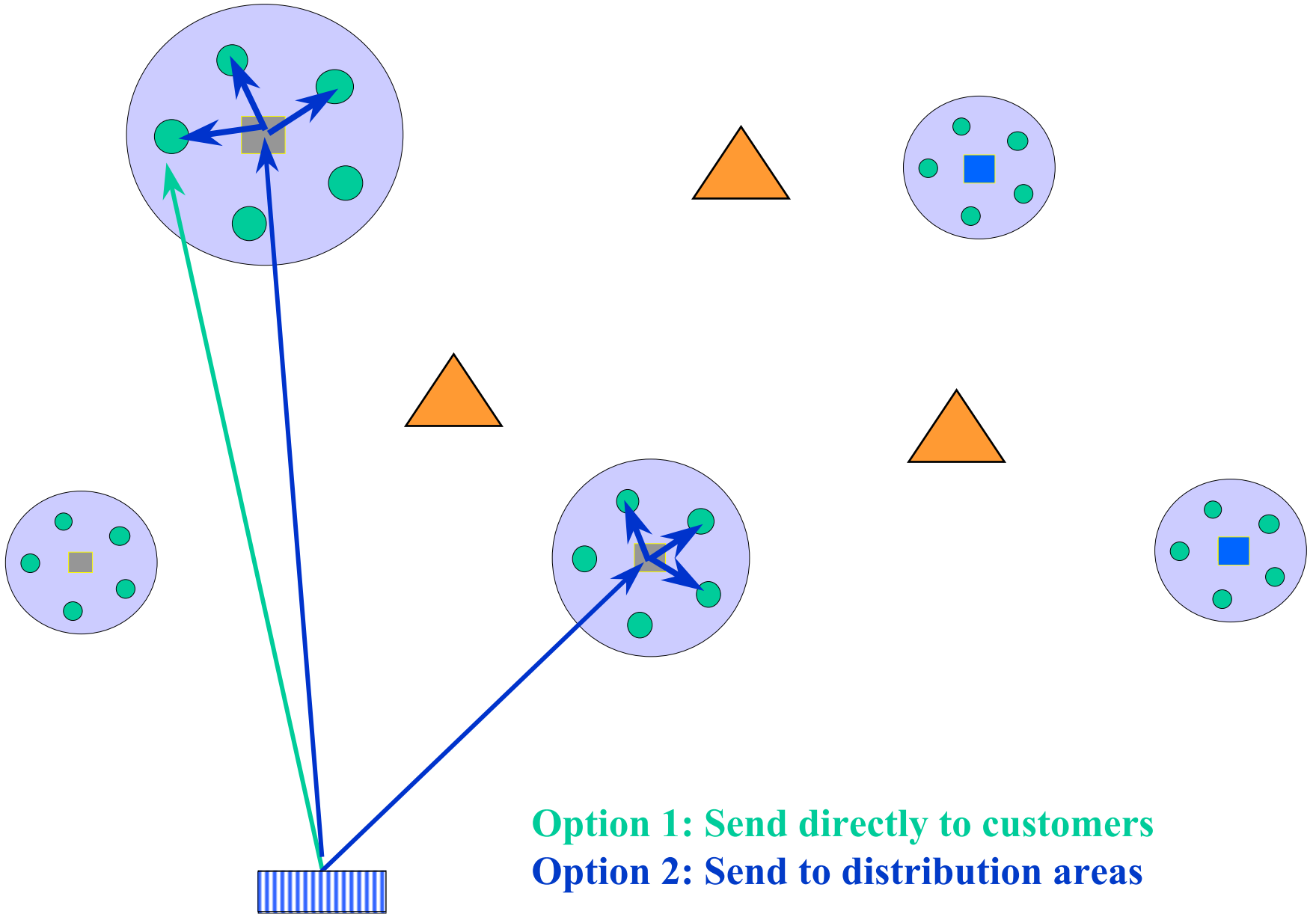
Norfolk Southern



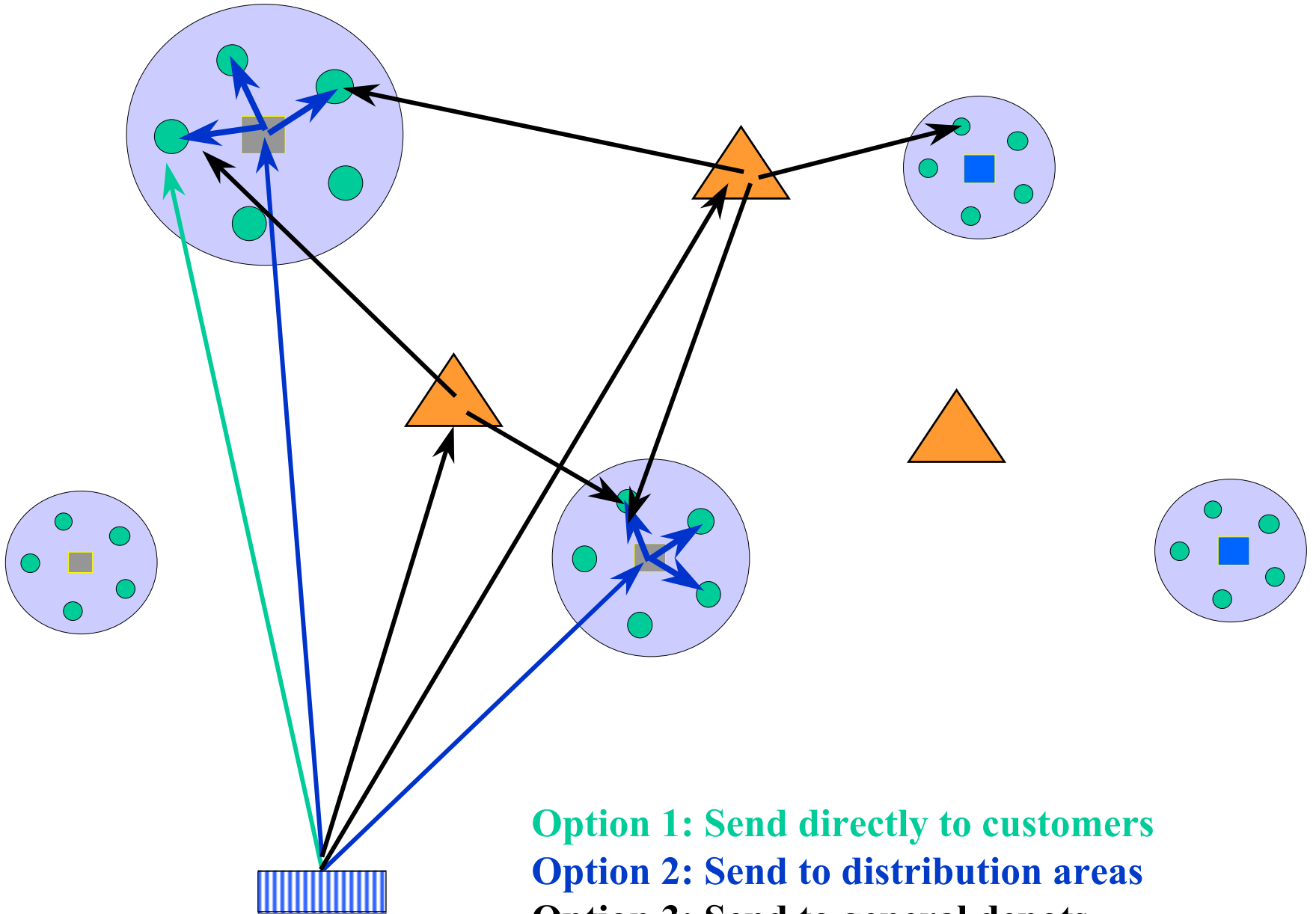
When a boxcar becomes empty, we have three options:



Option 1: Send directly to customers



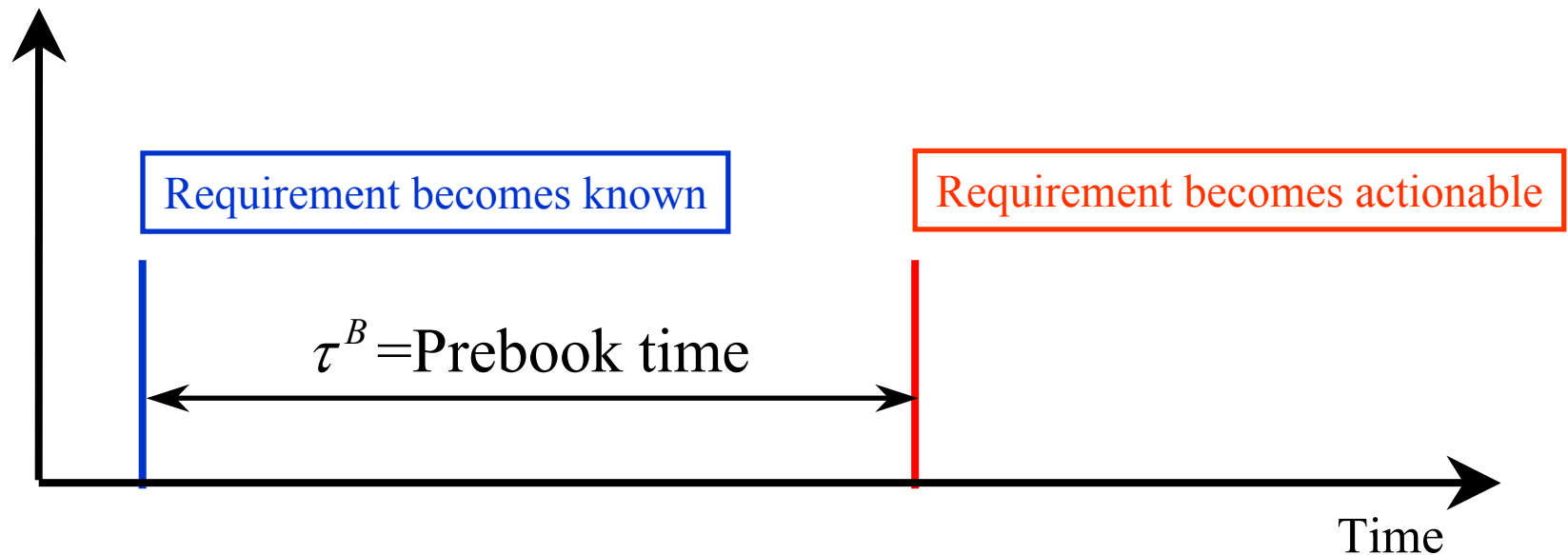
Option 1: Send directly to customers
Option 2: Send to distribution areas



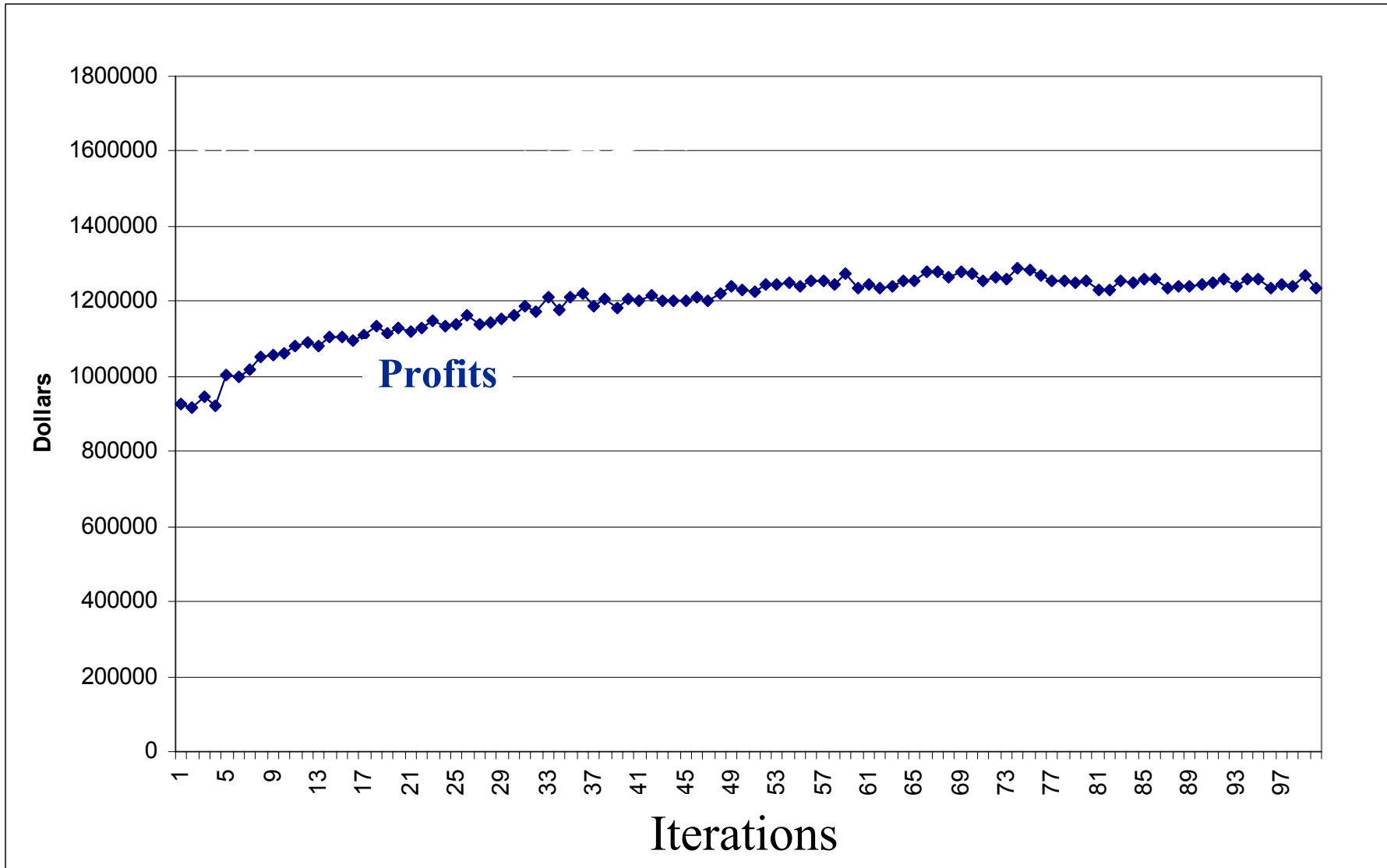
Option 1: Send directly to customers
Option 2: Send to distribution areas
Option 3: Send to general depots

A car distribution problem

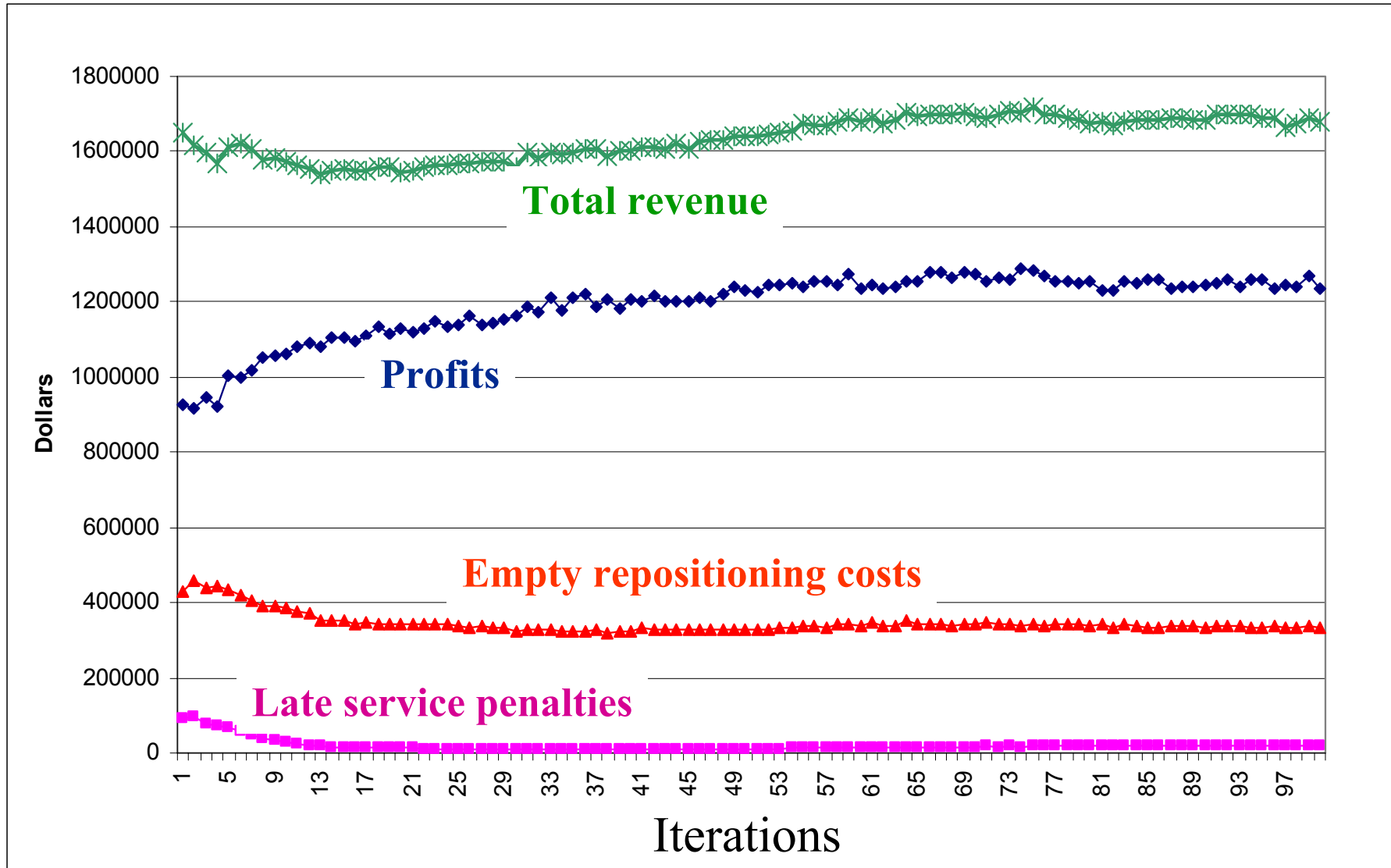
- For railroads, customers call in orders the week before:



A car distribution problem

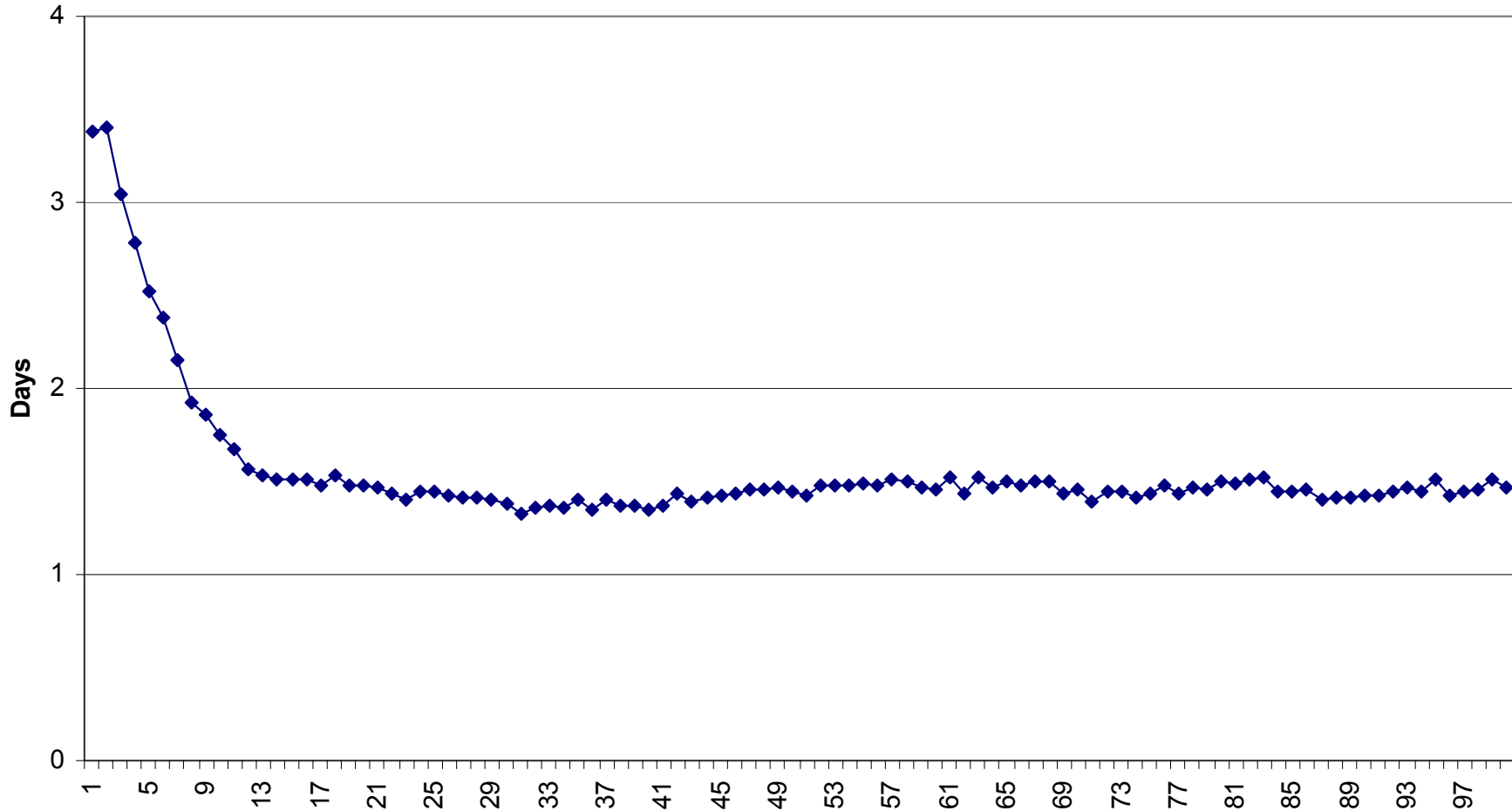


A car distribution problem



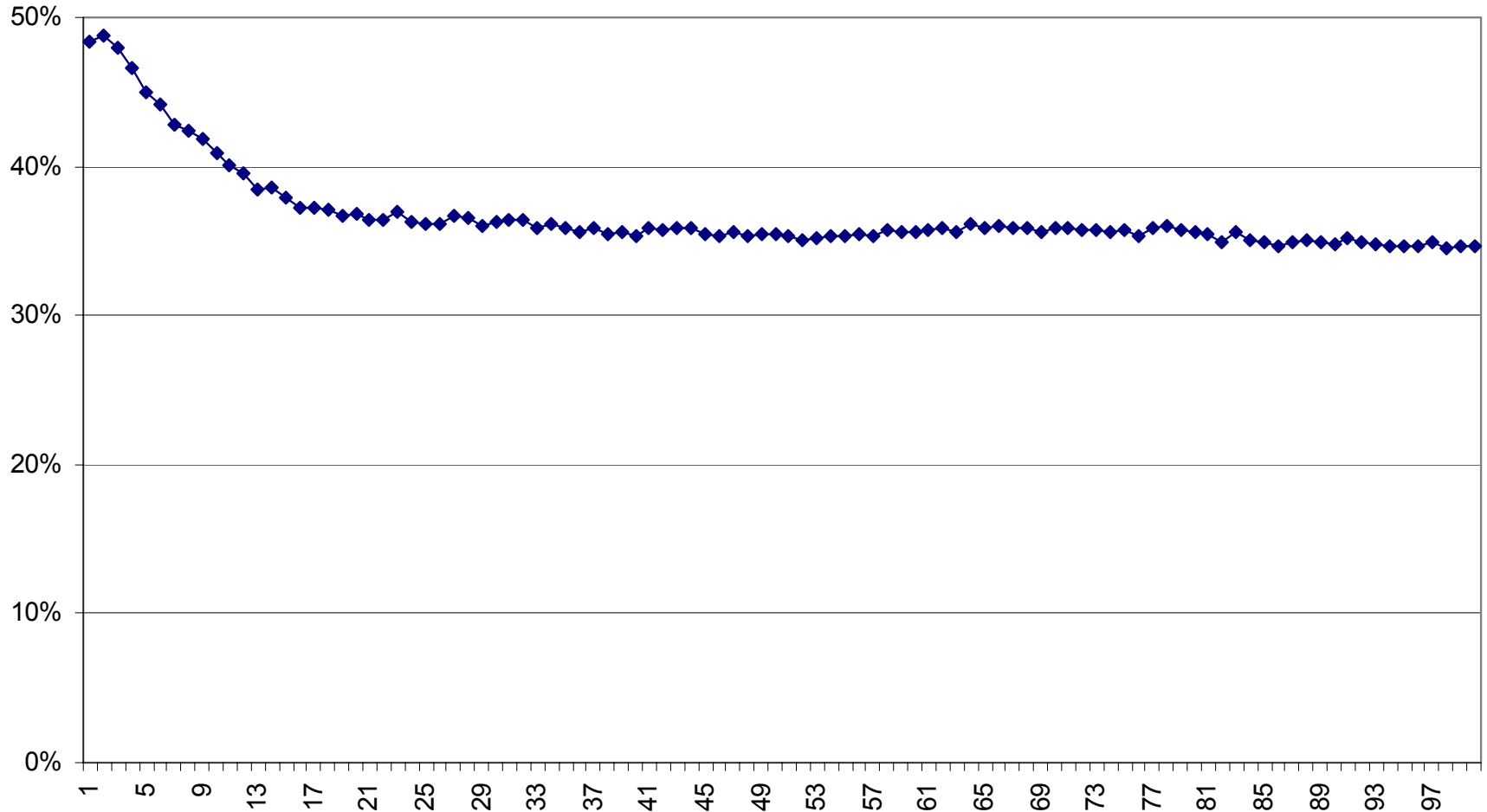
A car distribution problem

Average delay (days) per filled order



A car distribution problem

Empty miles as a percent of total miles





■ Other cool topics:

» Multiattribute resources:

- We have worked on single and multicommodity problems, but always assumed that the number of types of resources was not too large. In many applications, the attributes of a resource are fairly complex, producing attribute spaces that are too large to enumerate.

» Multi-layer problems

- A fractional jet company has to manage the movement of both aircraft and pilots simultaneously.

» Costing and pricing

- What is the impact of a change in price on expected flows?

» Value of information

- What is the value of having more advance information?

Questions???