

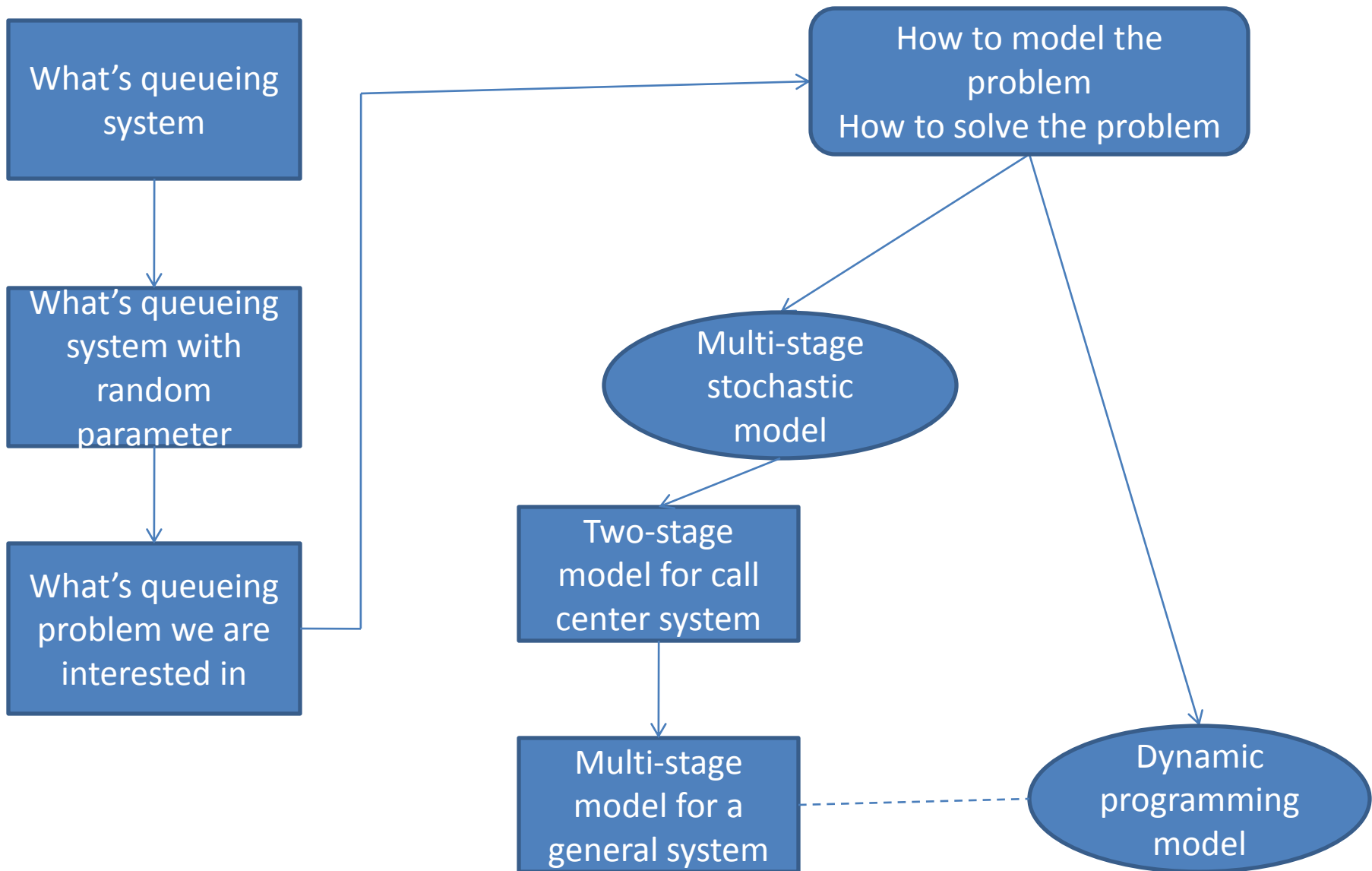
Queueing System with Random Parameters

Jing Zan

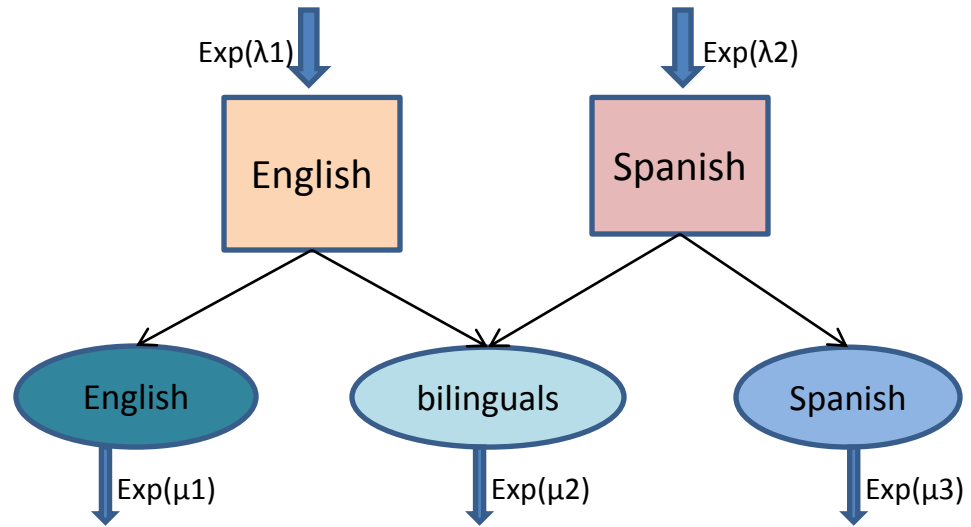
Graduate Program

OR/IE

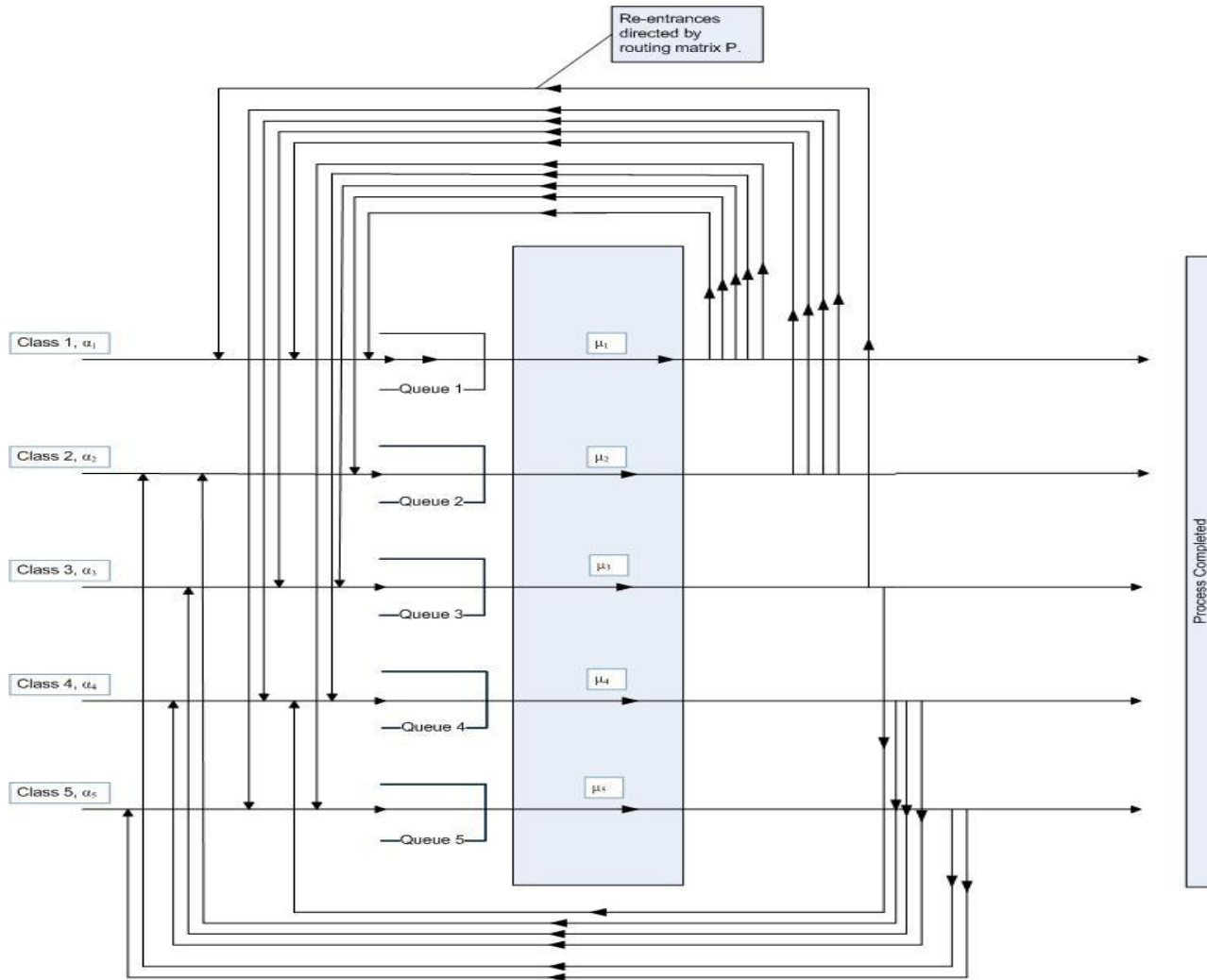
The University of Texas at Austin



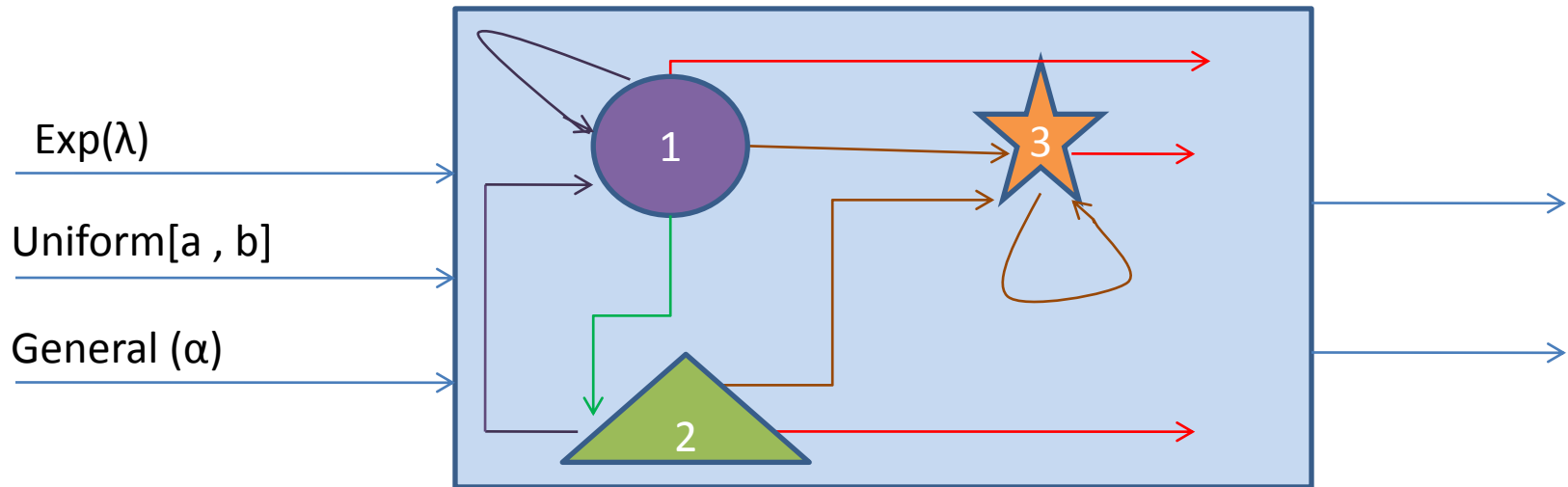
Call Center Model



A general queueing system



Queue system with Random Parameters

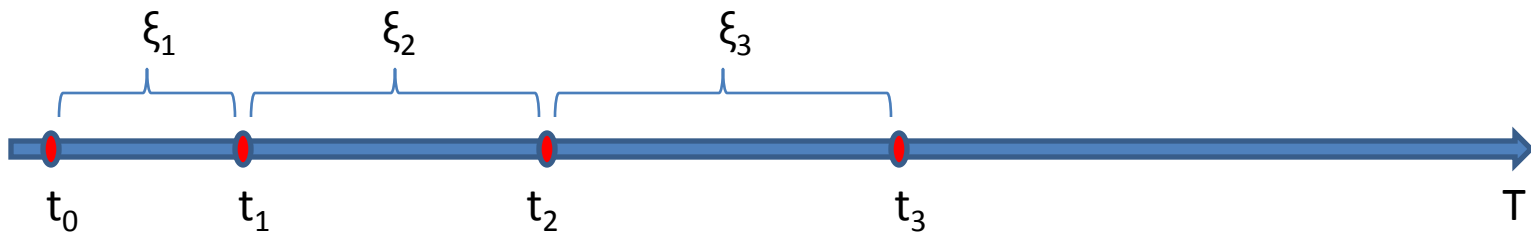


- Process randomness: random arrival of service requests
- Process randomness: random service
- Parameter randomness: random parameters (rates, demands, capacity, etc.)

- **Input:**
 - Distribution of the inter-arrival of customers ($\exp(\lambda)$)
 - Distribution of service time ($\exp(\mu)$)
 - Distribution of all the random parameters (λ, μ)
 - Cost parameters
- **Decision:**
 - Number of servers
 - Allocation of servers
- **Objective:**
 - Save money
 - Save time
 - Satisfy service quality constraints

Problems we are interested in...

- Random rates
- Big decisions
- Small decisions



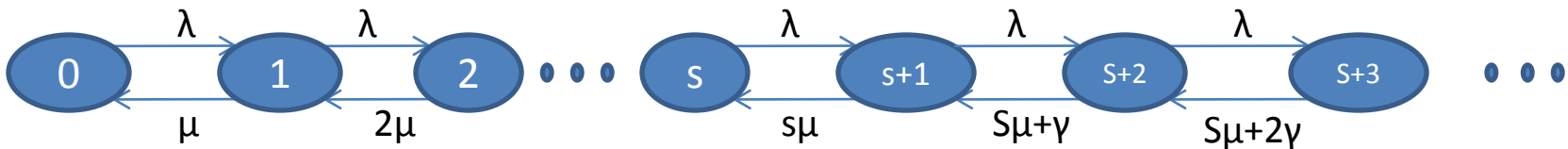
Does the place
where
randomness
shows up matter?
 λ random? μ
random?

It may affect the
convexity of the
problem sometime

Difficulty in Building a Queueing Model

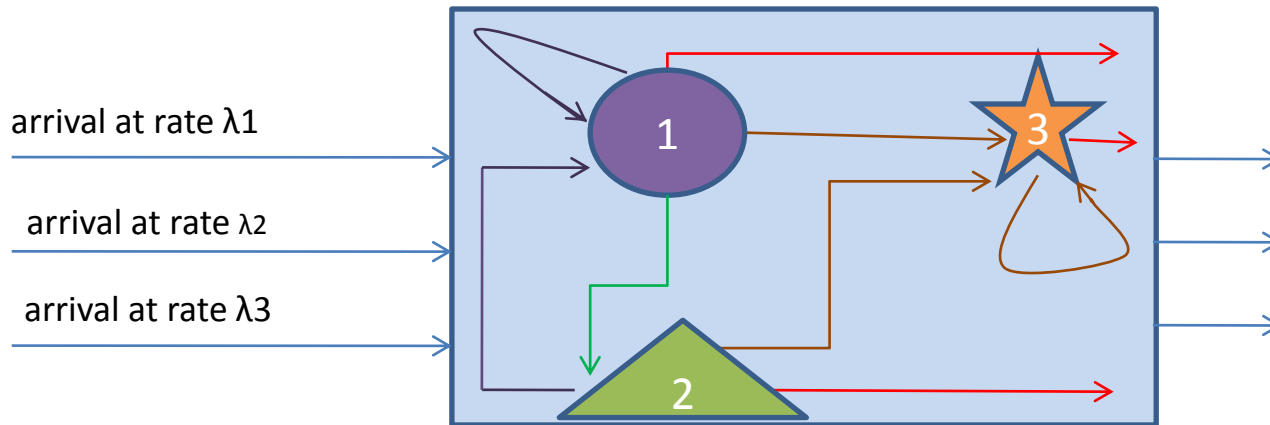
- With the Markovian assumption
- Without all the Markovian assumption

Example: M/M/S+M
queueing model



$$\pi_0 = \left[\sum_{j=0}^s \frac{(\lambda/\mu)^j}{j!} + \frac{(\lambda/\mu)^s}{s!} \sum_{j=s+1}^{\infty} \prod_{k=s+1}^j \left(\frac{\lambda}{s\mu + (k-s)\gamma} \right) \right]^{-1}$$

Fluid Model



- arrival at fixed rate
- service at fixed rate
- **get rid of process randomness**
- the rates can still be random

Why fluid relaxation becomes so interesting ?

Stability of multiclass queueing networks is implied by stability of their deterministic fluid counterparts

Queueing Model VS Fluid Model



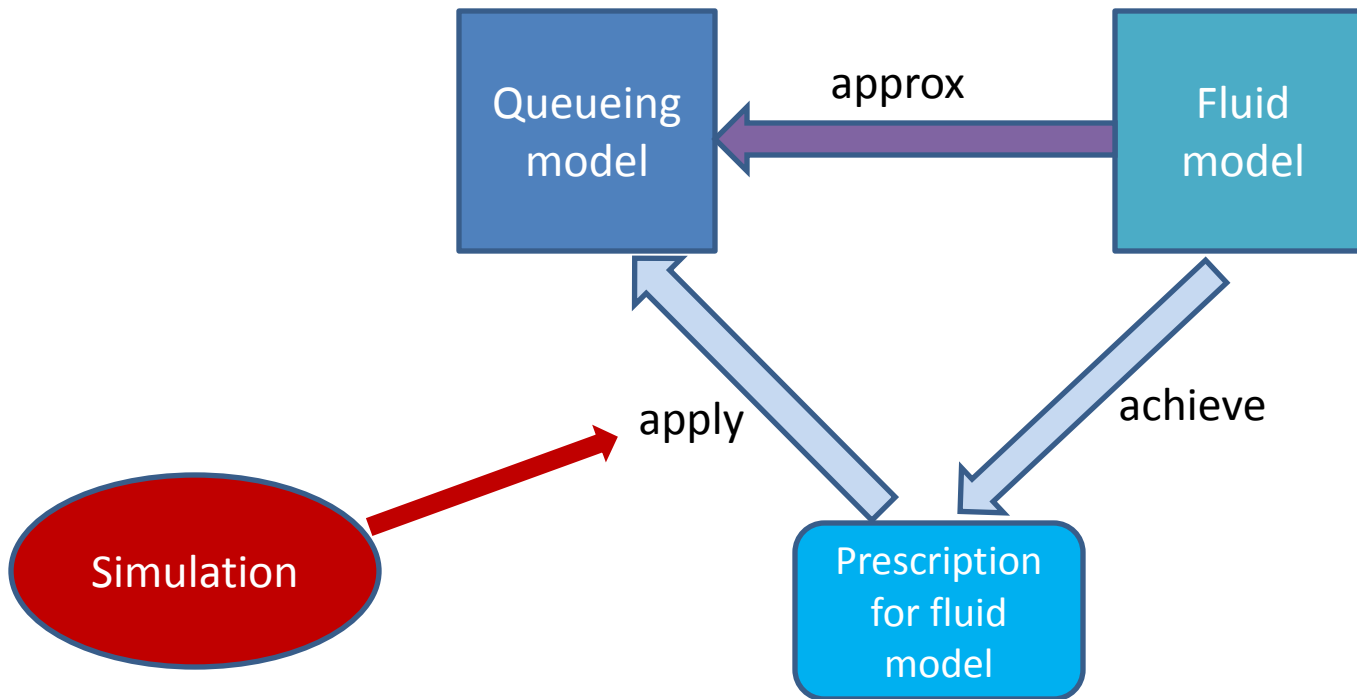
VS



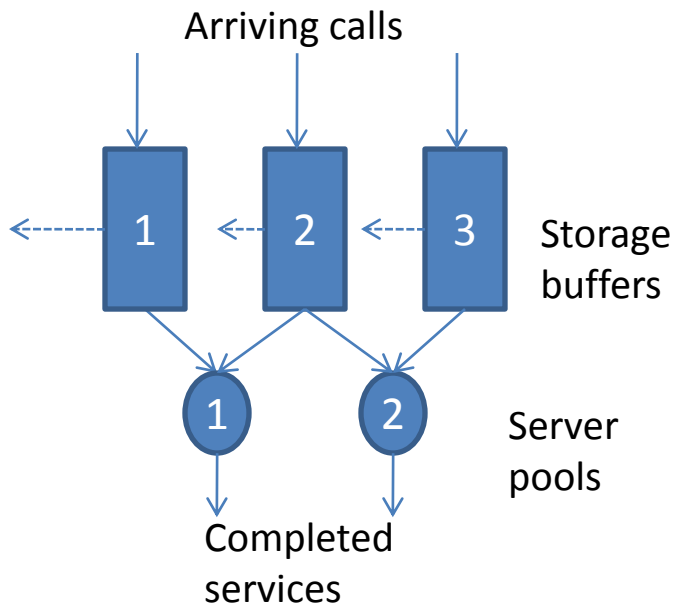
- Discrete
- Has process randomness
- long-run average measures

- Continuous
- Has no process randomness
- Drain the initial queue or cost ASAP

Fluid Model Approx Queueing Model



Multi-Class Multi Pools Call Center Model



- Stochastic Arrival rates
- Abandon
- Staffing costs + abandonment penalties
- Big decision-staffing problem
- Small decision-dynamic routing problem
- Apply fluid model

Two-Stage LP with recourse

$$\min_b cb + E_{w \in \Omega} \{g(b, x)\}$$

$$\lambda^w, w \in \Omega$$

$$g(b, x^w) = \min_x p(\lambda^w - Rx)$$

$$\text{s. t. } Rx \leq \lambda^w$$

$$Ax \leq b$$

$$x \geq 0$$

Choose b

Choose x



See the realization of randomness

First stage: choose capacity b and incurs cost cb

Random demand λ is observed

Second stage: given the observation, choose allocation x

How to Solve?

- Turn into a LP:

$$\min_b cb + p^1[p(\lambda^{w_1} - Rx^{w_1})] + p^2[p(\lambda^{w_2} - Rx^{w_2})] + p^3[p(\lambda^{w_s} - Rx^{w_s})]$$

$$s. t. \quad Rx^{w_1} \leq \lambda^{w_1}$$

$$Rx^{w_2} \leq \lambda^{w_2}$$

$$Rx^{w_s} \leq \lambda^{w_s}$$

$$| \quad Ax^{w_1} \leq b$$

$$Ax^{w_2} \leq b$$

$$Ax^{w_s} \leq b$$

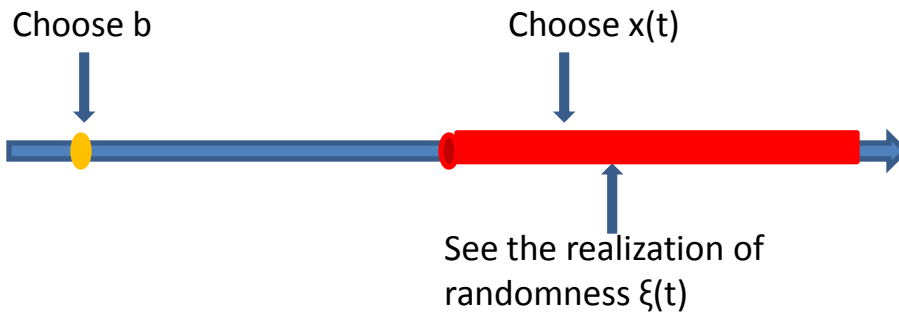
$$x^{w_1}, x^{w_2}, x^{w_s} \geq 0$$

- Simulation
- Check if the problem is still convex

Allocation decision can be changed continuously

$$\min_{b \geq 0} cb + E\left[\int_0^T g(\Lambda(t), b, x) dt\right]$$

- The above problem is convex in b
- The gradient-descent method can be used

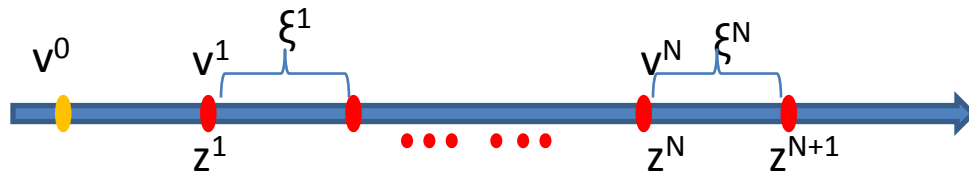


DP Model

$$\min_{v^0 \in V^0} E f_0(z^1, \xi^1)$$

$$s. t. \quad z^{n+1} = g(z^n, v^n, \xi^n), \quad n = 1, \dots, N$$

$$f_n(z^{n+1}, \xi^{n+1}) = C(v^n, z^{n+1}) + \min_{v^{n+1} \in V^{n+1}(z^{n+1}, \xi^{n+1})} E_{(\xi^{n+2} | \xi^{n+1})} f_{n+1}(z^{n+2}, \xi^{n+2})$$



Dynamics of the system

Set V^{n+1} is the set of feasible decisions at time point $n+1$

• Define set : $y(z^n, \xi^n) = \{(v^n, z^{n+1}) : v^n \in V^n, z^{n+1} = g(z^n, v^n, \xi^n)\}$

• By the principal of optimality (Bellman 1957), the above DP can be solved by solving:

$$f_{n-1}(z^n, \xi^n) = E \left\{ \min_{(v^n, z^{n+1}) \in y(z^n, \xi^n)} C(v^n, z^{n+1}) + f_n(z^{n+1}, \xi^{n+1}) \mid z^n \right\}$$

- Start from the last stage
- Evaluate all the possible values of function f for all the value of Z
- Take care of the expectation value
- Solve the whole thing backward

Drawbacks

- computing the expectation may be intractable
- The number of possible values of z may be very large (what happen when z takes continuous values)



- Use Monte Carlo simulation to get rid of the expectation
- Define and update a approximate value function of f to avoid evaluate all the possible value of function $f(v, z, \xi)$

Approximate DP

$$f_{n-1}(z^n, \xi^n) = E \left\{ \min_{(v^n, z^{n+1}) \in \mathcal{Y}(z^n, \xi^n)} C(v^n, z^{n+1}) + f_n(z^{n+1}, \xi^{n+1}) \mid z^n \right\}$$

Draw Monte Carlo Sample of
 $\{\xi^n: n=1, \dots, N\}$

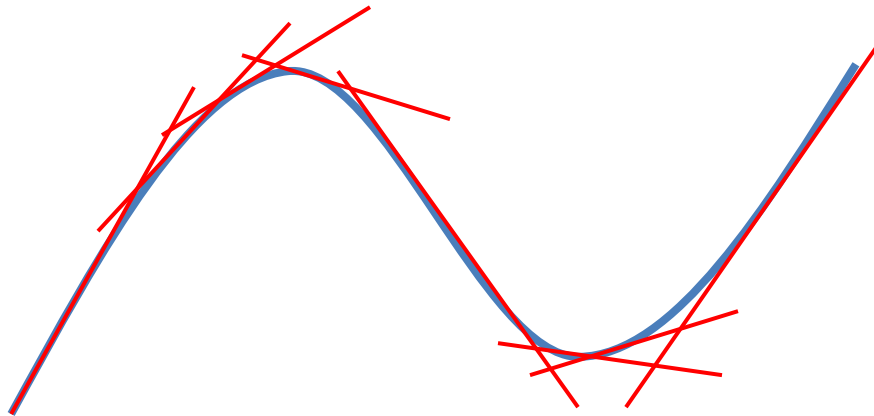
$$f_{n-1}(z^n, \xi^n) = \min_{(v^n, z^{n+1}) \in \mathcal{Y}(z^n, \xi^n)} C(v^n, z^{n+1}) + f_n(z^{n+1}, \xi^{n+1})$$

Replace value function f_n by
its approximate function \hat{f}_n

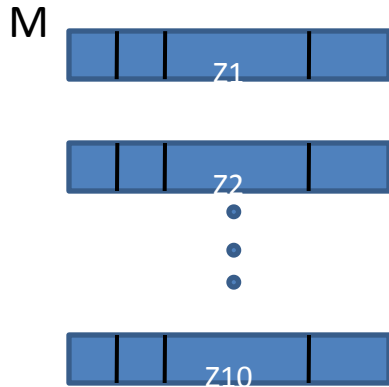
$$\hat{f}_{n-1}(z^n, \xi^n) = \min_{(v^n, z^{n+1}) \in \mathcal{Y}(z^n, \xi^n)} C(v^n, z^{n+1}) + \hat{f}_n(z^{n+1}, \xi^{n+1})$$

How to Choose Approx Cost Function?

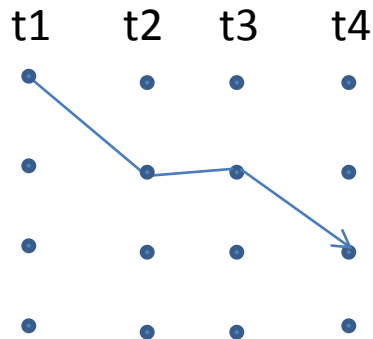
- Piece-wise linear function
- Linearize objective function



Why Need Approx Cost Function?

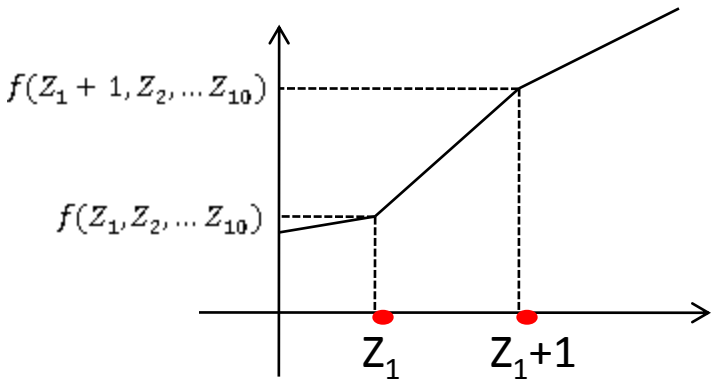
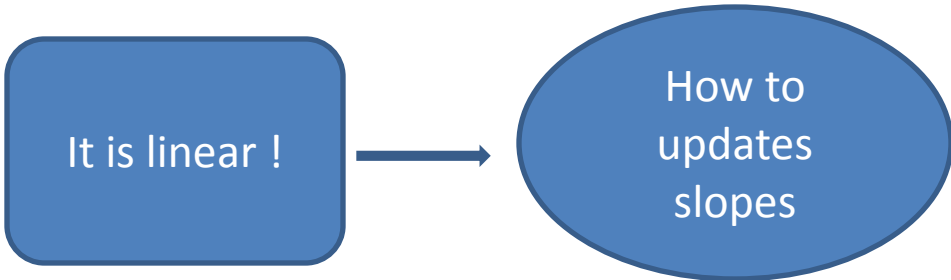


- state $s=(z_1,z_2,\dots,z_{10})$
- # of states $|S|: M^{10}$
- $f: S \rightarrow R$
- $T \cdot M^{10}$



- $\hat{f} = \sum C_i \cdot Z_i$
- $\hat{f} : S \rightarrow R$
- # of computations we need to take: 10
- $T \cdot 10$

How to Update Approx Cost Function



$$\frac{f(Z_1 + 1, Z_2, \dots, Z_{10}) - f(Z_1, Z_2, \dots, Z_{10})}{(Z_1 + 1) - Z_1}$$

↑

$$\frac{\hat{f}(Z_1 + 1, Z_2, \dots, Z_{10}) - \hat{f}(Z_1, Z_2, \dots, Z_{10})}{(Z_1 + 1) - Z_1}$$

Approx DP Alg

- Step 0: Initialization: Choose an approximation \hat{f}_n for f_n for all n. Set iteration counter k=1.
- Step 1: Forward Pass:
 - Step 1.0: Initialize the forward pass: Initialize Z^1 to the initial queueing length. Obtain a sample realization of $\{\xi^n : \text{for all } n\}$, say $\{\hat{\xi}^n : \text{for all } n\}$. Set n=1.
 - Step 1.1 Solve $\hat{f}_{n-1}(z^n, \hat{\xi}^n) = \min_{(v^n, z^{n+1}) \in \mathcal{Y}(z^n, \hat{\xi}^n)} C(v^n, z^{n+1}) + \hat{f}_n(z^{n+1}, \hat{\xi}^{n+1})$ to get v^n .
 - Step 1.2 Apply the system dynamics to get Z^{n+1} : $z^{n+1} = g(z^n, v^n, \hat{\xi}^n)$
 - Step 1.3 Set n=n+1. go to step 1.1.
- Step 2 Approximate value function \hat{f}_n update for all n
- Step 3 Set n=n+1. Go to step 1.

Things We are working on...

Have built the fluid model
with random parameter
for our problem

Have
implemented
the multi-stage
stochastic
model

Need to
implement the
DP model

Compare the
results from the
two models