11 Dynamic Routing

金力 Li Jin
li.jin@sjtu.edu.cn
上海交通大学密西根学院
Shanghai Jiao Tong University UM Joint Institute

上海交通大学
SHANGHAI JIAO TONG UNIVERSITY

Outline

- Background
  - Static & dynamic routing
  - Applications
- Single & parallel queues
  - Arrival process & service processes
  - Single queue
  - Parallel queues
- Queuing networks
  - Model
  - Bernoulli routing
  - Dynamic routing

Background

- Static & dynamic routing
- Applications

Background: static routing

- Data:
  - Demand
  - Cost function
  - Capacity
- Decision variables:
  - Link flows
- Constraints:
  - Mass conservation
  - Link capacity
- Objective:
  - Minimize total flow cost

Background: static vs. dynamic routing

Static routing

- At each diverge, traffic assigned to downstream links with time-invariant fractions.
- Only depends on model data (demand & capacity)
- Independent of real-time traffic condition (open-loop)
- Not responsive to disruptions

Dynamic routing

- At each diverge, traffic assigned with time-variant fractions.
- Depends on both model data and real-time traffic condition (closed-loop)
- Can respond to disruptions
- Requires real-time sensing capabilities

Background: dynamic routing

- Data:
  - Demand, Cost function, Capacity
  - Real-time traffic state (e.g. queue size)
- Decision variables:
  - Routing for each vehicle/customer/job
- Constraints:
  - Mass conservation
  - Link capacity
- Objective:
  - Minimize total travel cost, typically starting from current state

Vehicle routing

- A vehicle starts its trip from origin to destination
- Baidu Map or AMAP suggests multiple possible routes
- Estimated travel time on each route is predicted
- Typically vehicles select the fastest route
- Such routing is responsive to traffic congestion & traffic incidents
- Dynamic routing!

Customer routing

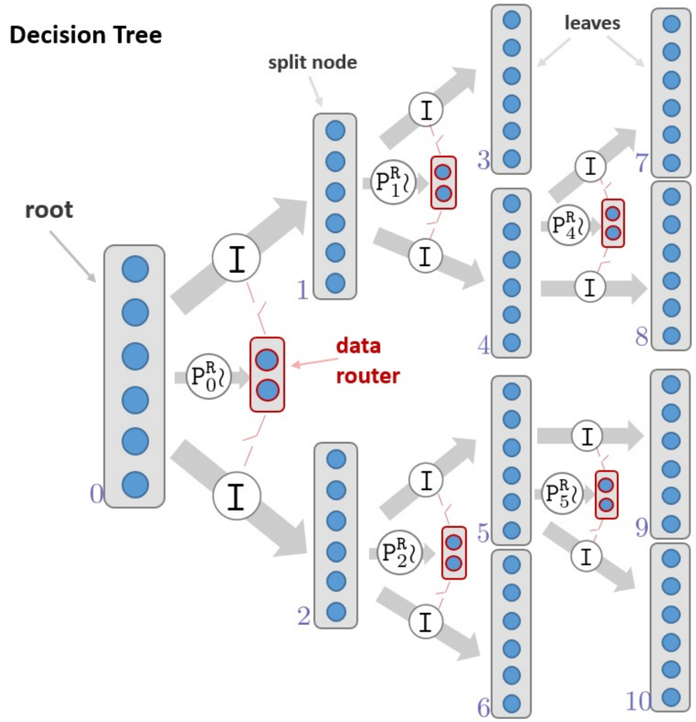
- Suppose that a supermarket has multiple cashiers
- Customers wait in separate queues for the cashiers
- When a customer arrives, he/she selects the shortest queue to join ("JSQ" policy)
- Or, joining the queue with the least items, one customer buying one week's supply vs. two customers buying two drinks

Air traffic management

- Air routes connecting airports
- Traffic flow on each route
- Traffic on different route may interact within sectors
- Sector can get congested
- Each flight needs to determine the path, i.e. sequence of sectors

Data job routing

- Jobs received by a router
- Router assigns each job to a server
- Jobs processed by a server is allocated to a further downstream server
- JSQ: a router always allocate an incoming job to the least busy server, i.e. the server with the shortest job queue



Single & Parallel Queues

- Arrival process & service processes
- Single queue
- Parallel queues

Queuing node model

- A node contains a server and a queuing space
- State $X(t)$ = # of customers waiting or being served at the node
- If $X(t) = 3$, then 2 customers are waiting and 1 being served.
- For ease of presentation, we consider discrete time

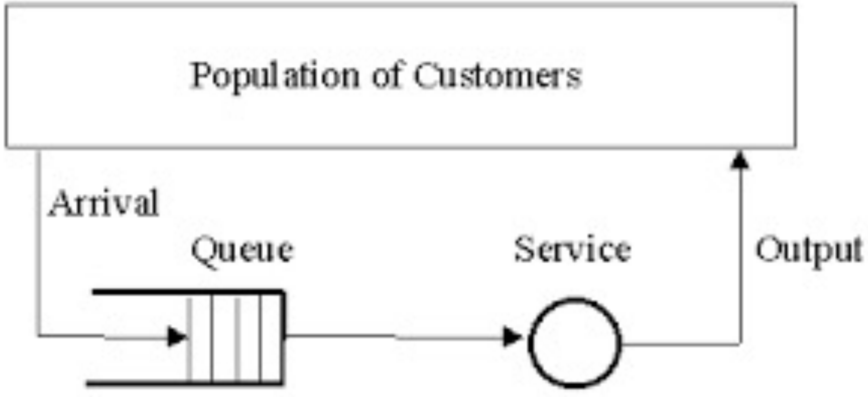


Figure 1

Arrival process

- We consider a stochastic arrival process at node i
- Bernoulli process:
  - A customer arrives at node i during one time step with probability $\lambda \in [0,1]$
  - No customers arrive at node i with probability $1-\lambda$
- Expected # of arrivals per time step = λ (demand)
- Distribution over T time steps:

$$p_N(n) = \binom{T}{n}\lambda^n(1-\lambda)^{T-n} = C_n^T\lambda^n(1-\lambda)^{T-n}$$

for $N = 0,1,\ldots,T$

- Upon arrival, a customer
  - enters the server and begins its service if the server is empty
  - joins the queue and waits otherwise

Service process

- We consider a probabilistic service process:
  - Suppose that a customer is being served at time t
  - The customer finishes service and leaves the current node at time $t + 1$ with probability μ
  - The customer stays in the node and continues its service with probability $1 - \mu$
- As soon as a customer finishes service, the next customer enters the server and begins its service
- Otherwise, the subsequent customers have to continue waiting

A single-node queuing system

- Consider an isolated node
- A single node with
  - Bernoulli arrivals with rate λ
  - Probabilistic service with rate μ
- State of the node: $X(t)$ = queue size at time t

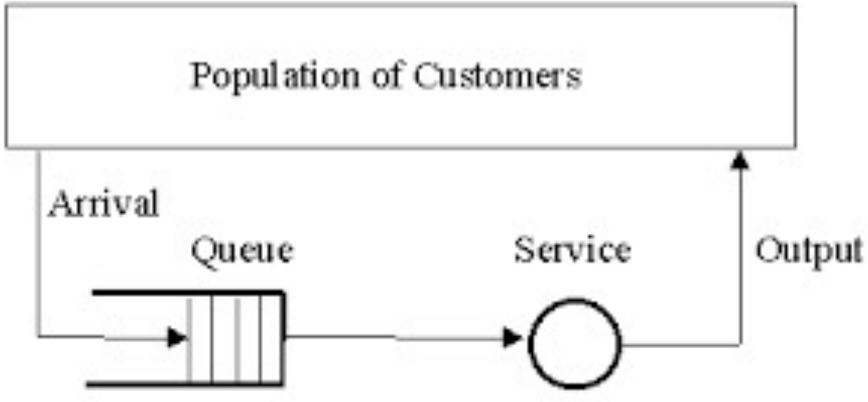


Figure 1

A single-node queuing system

- State transition probabilities (system dynamics)
- Suppose $X(t) = \xi > 0$

$$\Pr\{X(t+1) = \xi - 1\} = \mu(1-\lambda)$$
$$\Pr\{X(t+1) = \xi + 1\} = \lambda(1-\mu)$$
$$\Pr\{X(t+1) = \xi\} = (1-\lambda)(1-\mu) + \lambda\mu$$

- Interpretation of the three formulae?
- Suppose $X(t) = 0$

$$\Pr\{X(t+1) = 0\} = 1-\lambda$$
$$\Pr\{X(t+1) = 1\} = \lambda$$

- Can you draw the state transition diagram? #

HW3

Problem 1: Single queuing system with state $X(t) \in \mathbb{Z}_{\geq 0}$

a) Suppose $\lambda = 0.5$ and $\mu = 0.7$. Draw the state transition diagram. You need to indicate the numerical values for the transition probabilities.

b) Write codes to simulate the queuing system. You can assume zero initial condition. Report the time-average queue size

$$\frac{1}{T}\sum_{t=0}^{T} X(t)$$

with $T = 360$.

a) Change the value for λ from 0.1 to 0.8 while keep $\mu = 0.7$. Repeat the simulation in part b. Plot the time-average queue size vs. demand λ. Interpret the trend of the curve.

A single-node queuing system*

- Of particular interest is the steady-state behavior of the queuing system.
- Simplistically, steady-state behavior ≈ long time average. (*Ergodicity*)
- The most important performance metric for a queuing system is the steady-state or long time-average queue size

$$\bar{X} = \lim_{t\to\infty} \frac{1}{t} \sum_{s=0}^{t} X(s) = ?$$

- We say that the queuing system is
  - stable or convergent if $\bar{X} < \infty$
  - unstable if $\bar{X} = \infty$
- Can you make a scientific guess when is the queue stable?

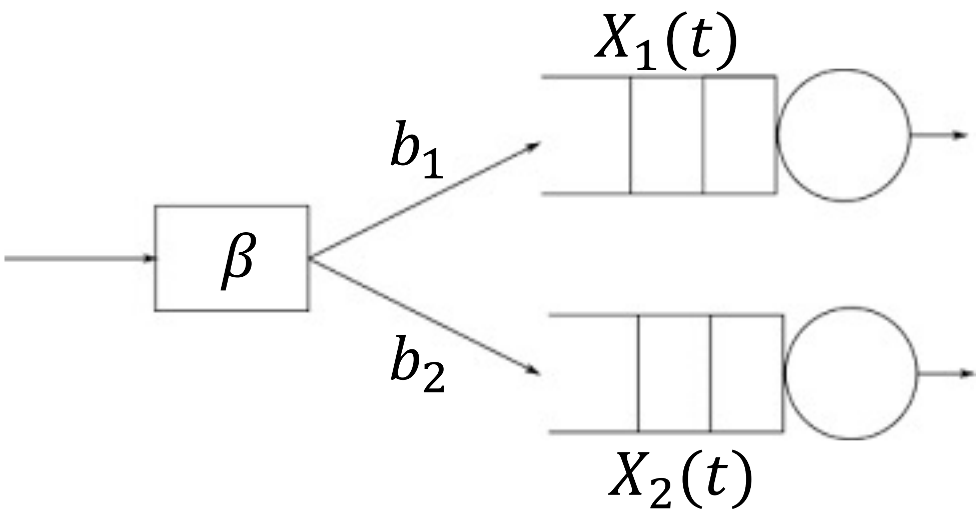
A single-node queuing system*

- Let p_{ij} be the transition probabilities, i.e.
$$p_{ij} = \Pr\{X(t+1) = j | X(t) = i\}$$
- Let π_i be the steady-state probabilities, i.e.
$$\lim_{t\to\infty} \Pr\{X(t) = i\} = \pi_i$$
- The steady-state equations are #
$$p_{01}\pi_0 = p_{10}\pi_1$$
$$\left(p_{i-1,i} + p_{i,i+1}\right)\pi_i = p_{i-1,i}\pi_{i-1} + p_{i+1,i}\pi_{i+1}$$
$$i = 1,2, \dots$$
- Then we can obtain π_i by solving the above equations
- $\bar{X} = \sum_{x=0}^{\infty} \pi_x x$

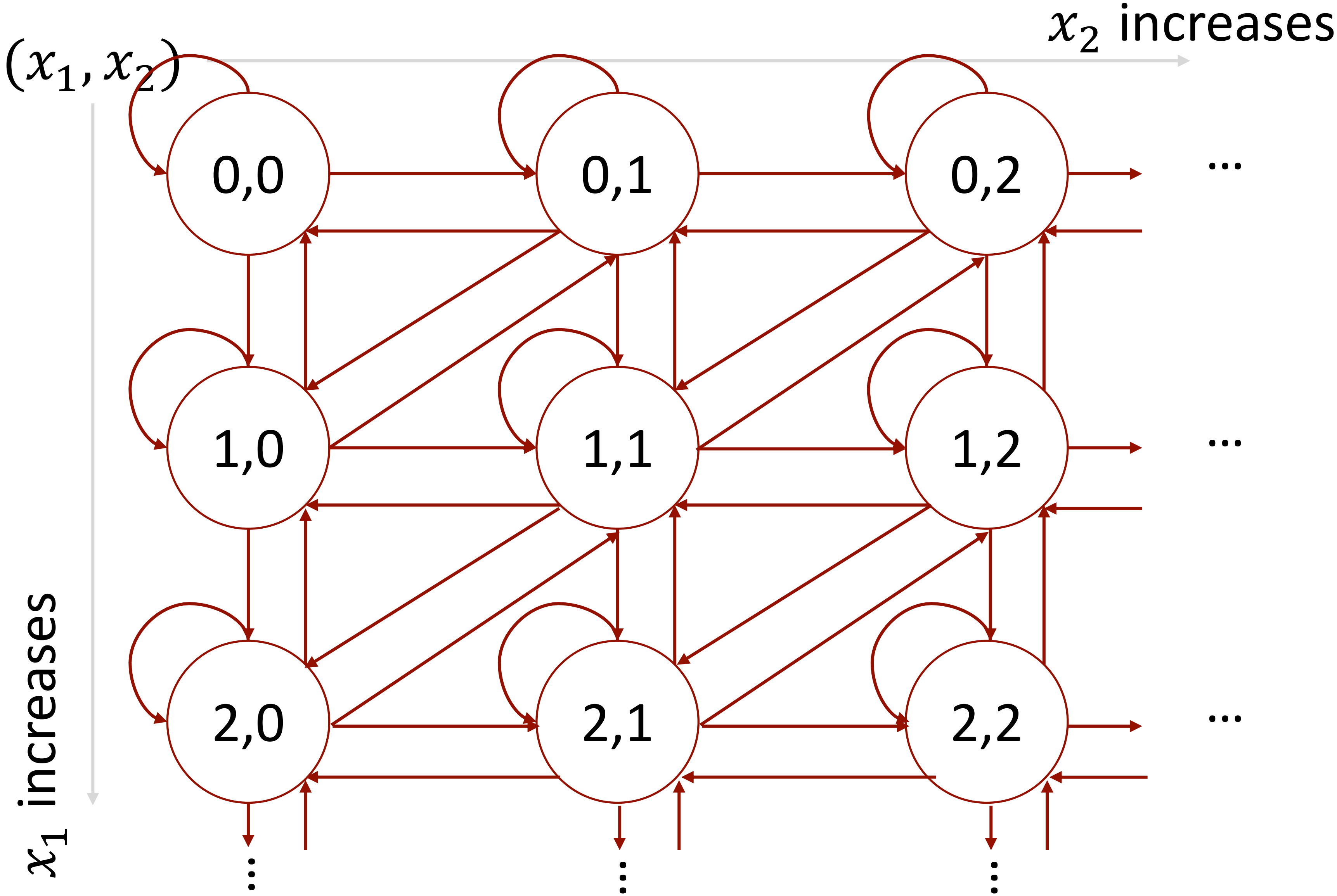
Dynamic routing: parallel queues

- Customers arrive at a router with rate λ
- Router assigns customer to one of two parallel queues
- State: $X(t) = [X_1(t)\ X_2(t)]^T \in \mathbb{Z}^2_{\geq 0}$
- Dynamic routing policy:

$$\beta: \mathbb{Z}^2_{\geq 0} \to [0,1]^2 \text{ or } \beta: \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \mapsto \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

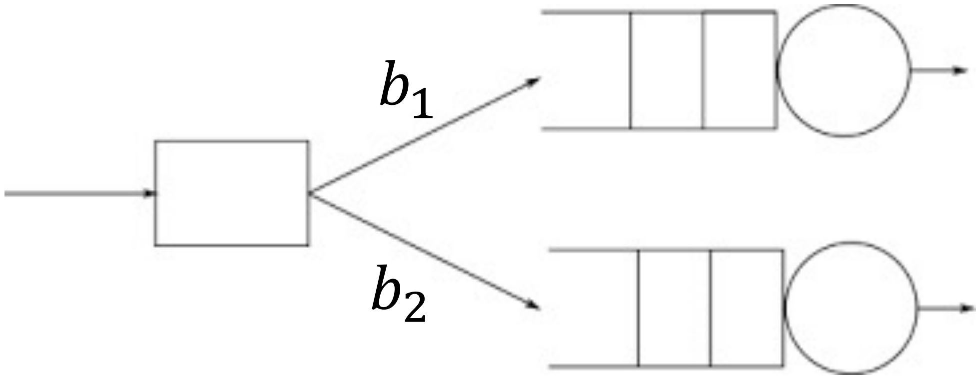


State-transition diagram



Benchmark: Bernoulli routing

- When a customer arrives, it is assigned to queue 1 with probability b_1 and queue 2 with probability b_2, respectively.
- $b_1 \in [0,1]$, $b_2 \in [0,1]$, $b_1 + b_2 = 1$
- Open-loop routing
- Independent of $X(t)$
- State transition diagram? #



HW3

Problem 2: Bernoulli routing for parallel queues.

a) Suppose that $\lambda = 0.5$, $\mu_1 = \mu_2 = 0.5$. Assume that $b_1 = 0.2$ and $b_2 = 0.8$. Draw the state transition diagram. Simulate the parallel queues with zero initial condition. Report the time-average queue size $\frac{1}{T}\sum_{t=0}^{T}\big(X_1(t) + X_2(t)\big)$.

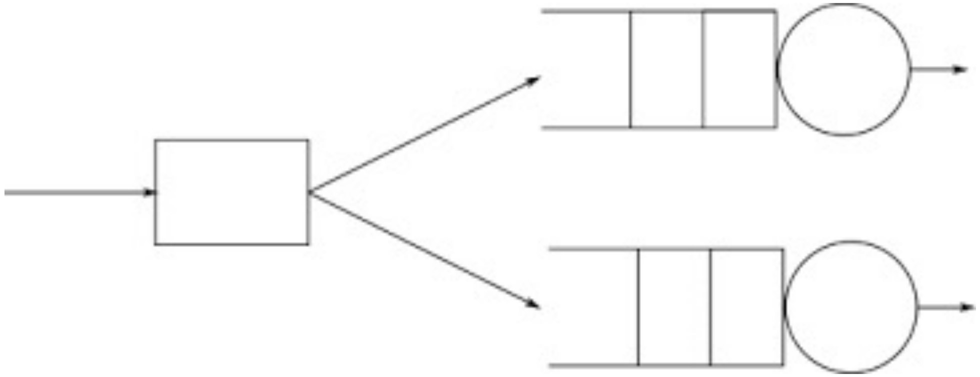
b) Repeat the above simulation for $b_1 \in \{0,0.1, \dots, 1.0\}$ and find the optimal value.

c) Suppose that $\lambda = 0.5$, $\mu_1 = 0.1$ and $\mu_2 = 0.9$. Find the optimal b_1 by simulation (i.e. trying various values for b_1).

d) Compare and interpret the results in parts b and c.

Dynamic routing: parallel queues

- Now suppose that we can route an incoming customer according to the real-time traffic state $X(t)$
- What is a good way of routing?
- Intuition:
  - If a queue is long, then do not add the customer to it.
  - If a queue is short, then it is OK to add the customer to it.
- ”Join the shortest queue” (JSQ) policy
- State transition diagram?



Dynamic routing: parallel queues

- JSQ policy:

$$\beta(x) = \begin{cases} [1\ 0]^T & x_1 < x_2 \\ [0\ 1]^T & x_1 > x_2 \\ ? & x_1 = x_2 \end{cases}$$

- Ties can be broken uniformly at random.
- That is, when $x_1 = x_2$, we set

$$\beta(x) = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

- Can adapt to randomness
- Can adapt to disruptions (e.g. server malfunction)

HW3

Problem 3: JSQ routing.

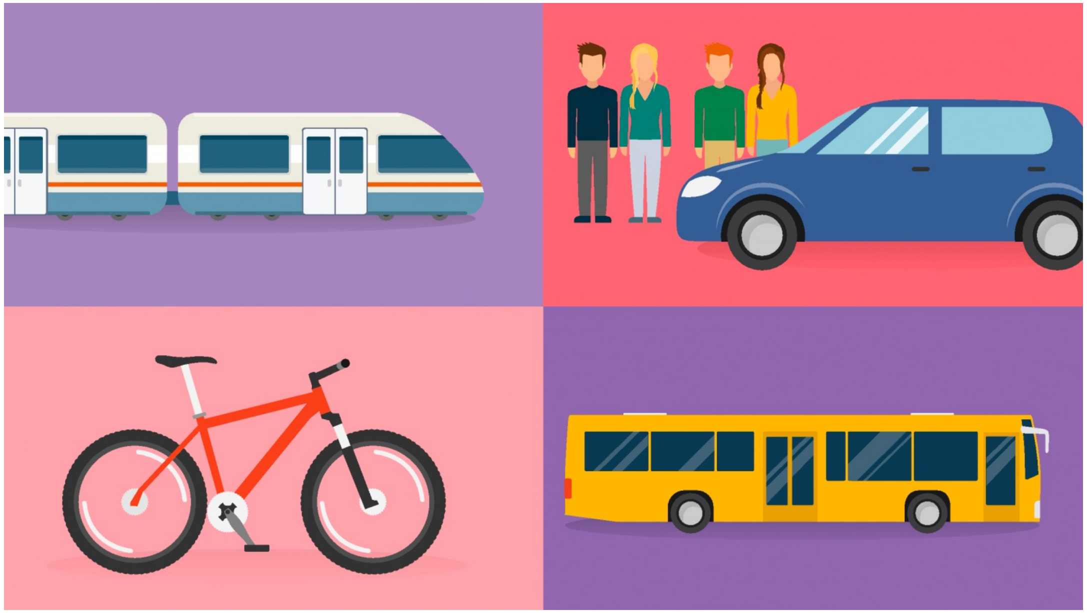
a) Repeat 2b with JSQ routing. Compare and interpret the average queue size with that in 2b.
b) Repeat 2c with JSQ routing. Compare and interpret the average queue size with that in 2c.
c) Compare the interpret JSQ’s performance in 3a and 3b.

How routing actually works in transportaion?

- There are two notions:
  1. What you want the travelers to do?
  2. How travelers would respond to your instruction?
- Usually, we are not able to force travelers to take a certain route...
- Instead, we can analyze travelers' behavior and incentivize/encourage travelers to do a favored action.
- Incentivize = pay them somehow...
- Theoretical foundation: discrete choice theory

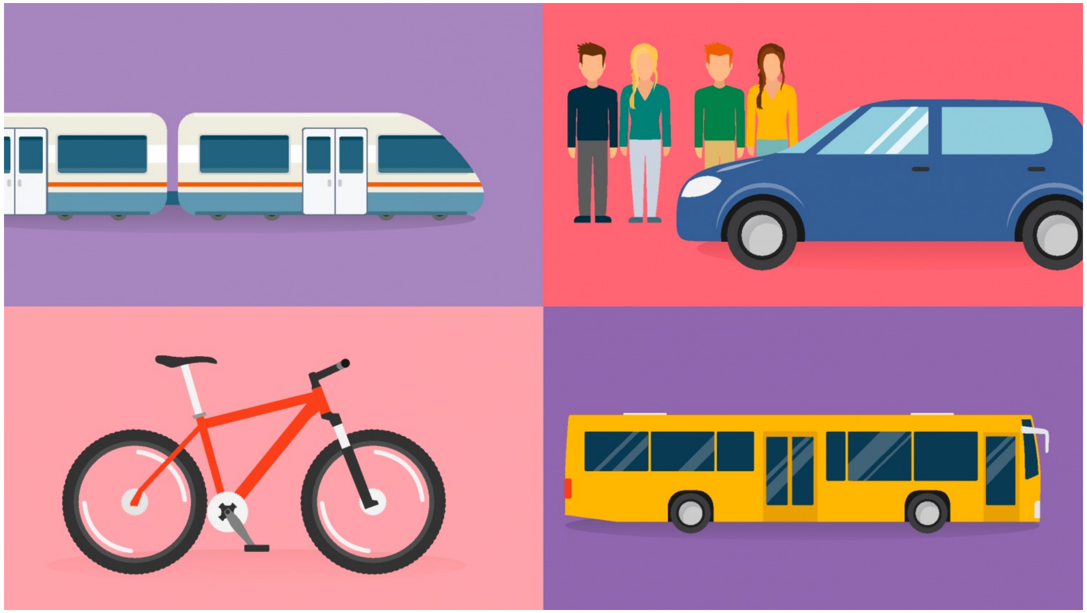
Discrete choice

- Customer chooses from K options
- A commuter chooses subway or bus
- A truck driver chooses departure time
- A driver chooses tolled or free roads



Discrete choice model

- Input: attributes of options
- Output: probability of choosing an option
- Tool: logistic regression



Logistic regression

- Logit function

$$\Pr\{G = k|X = x\} = \frac{\exp\left(\beta_{k0} + \beta_k^T x\right)}{\sum_{l=1}^{K} \exp(\beta_{l0} + \beta_l^T x)}$$

- Classification

$$G(x) = \arg\max_k \frac{\exp\left(\beta_{k0} + \beta_k^T x\right)}{\sum_{l=1}^{K} \exp(\beta_{l0} + \beta_l^T x)}$$

- Process of fitting coefficients β_{ki}: called logistic regression
- Use maximum likelihood

Basic discrete-choice model

- Utility: a quantification of customers' preferences
- Example 1: price of a product
- Example 2: price-to-quality ratio
- Example 3: travel time plus toll (value of time)
- Example 4: price plus comfort level
- Let $V_{i,n}$ be the nth customer's utility of the ith option
- The probability of choosing i is

$$P_{i,n} = \frac{\exp(V_{i,n})}{\sum_{m=1}^{K} \exp(V_{i,m})}$$

- High utility -> high probability of being chosen

Utility function

- In smart city settings, by far the most common form of utility function is linear

$$V_{i,n} = \sum_{k=1}^{K} \beta_{n,k} x_{i,k}$$

- For example, travel mode estimation

$$V_{subway} = b_0 + b_1 x(\text{travel time}) + b_2 x(\text{fare}) + b_3 x(\text{crowdness})$$

- Signs of the coefficients?

Interpretation

- Coefficients $\beta_{i,k}$: quantifies the i-th customer's preferences with respect to the k-th feature
- $\beta_{i,k} > 0$ -> ?
- $\beta_{i,k} < 0$ -> ?
- Magnitude of $\beta_{i,k}$ captures the sensitivity

$$V_{subway} = b_0 + b_1 x(\text{travel time}) + b_2 x(\text{fare}) + b_3 x(\text{crowdness})$$

Estimation

- Use maximum likelihood
- Suppose we have observation for N customers
- Both features and their actual choices are recorded
- Probability that customer i chooses option n_i

$$P_{i,n_i} = \frac{\exp(\sum_{k=1}^{K} \beta_{n_i,k} x_{i,k})}{\sum_{n=1}^{K} \exp(\sum_{k=1}^{K} \beta_{n,k} x_{i,k})}$$

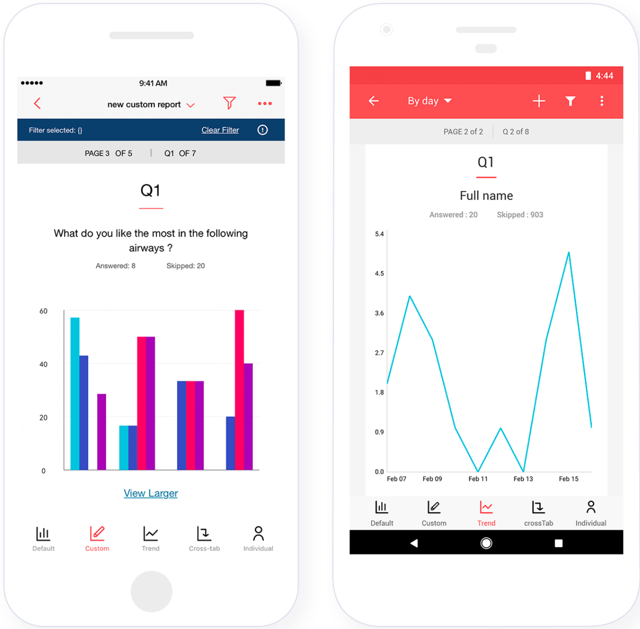
- Likelihood of their choices

$$lik(B) = \prod_{i=1}^{N} P_{i,n_i}(B)$$

- MLE: $\frac{\partial}{\partial \beta_{n,k}} lik(B) = 0$ for all n and for all k

Data

- Conventionally obtained from surveys
- Demographic information, technical and/or economical metrics, geographical information, etc.
- Modern ways of obtaining data
  - App-based surveys
  - “Big Data”
  - MTA trip records
  - Uber trip records
  - Airlines trip records…
- Important: people are increasingly concerned with efficiency vs. privacy



Mode choice

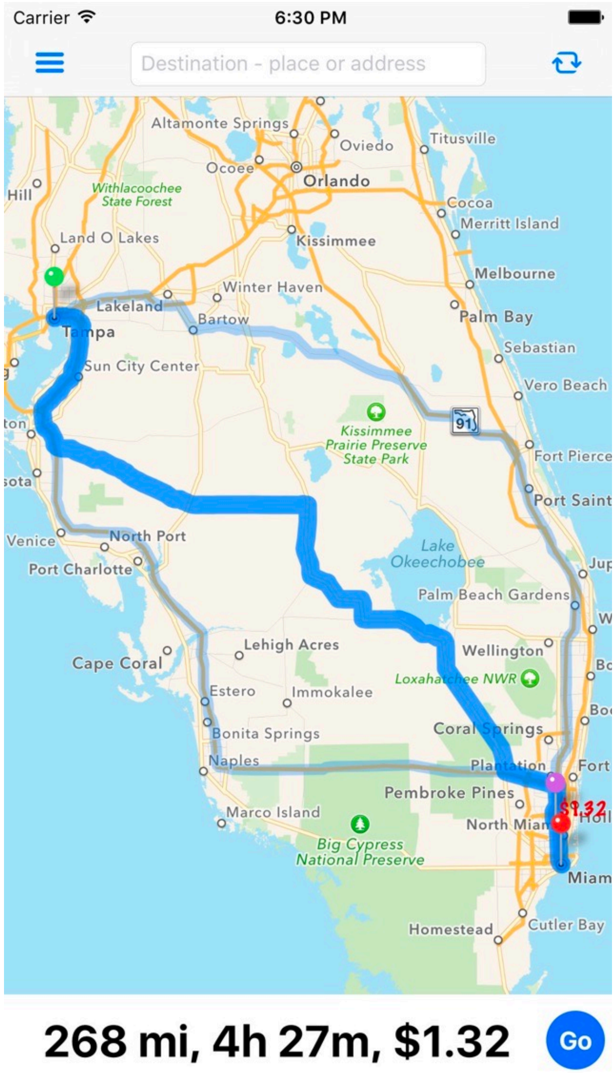
- Objective: develop a model explaining automobile ownership and commuting mode
- Application: justification for the Bay Area Rapid Transit (BART)
- Survey data
- $V=-0.0412c/w-0.0201T-0.0531T^0-0.89D^1-1.78D^3-2.15D^4$
- c=round-trip cost (\$)
- w=passenger wage rate (\$/min)
- T=in-vehicle travel time (min)
- T^0=out-of-vehicle time (min)
- D=alternative-specific dummies

Trip scheduling

- From a city manager’s perspective, we prefer to balance the trip schedules of commuters rather than concentrate them in short peak hours
- Objective: understand how people schedule trips
- V=-0.106T-0.065SDE-0.254SDL-0.58DL
- T=trip time
- SDE=schedule delay early
- SDL=schedule delay late
- DL=late dummy

Route choice

- How do people select between toll and free routes?
- Important for setting congestion pricing
- $V=-0.862D^{tag}+0.0239Inc(D^{tag})-0.766ForLang(D^{tag})-0.789D^{3}-0.357c-0.109T-0.159R+0.074Male(R)+$other terms
- D^{tag}=alternative-specific dummies
- Inc=annual income
- ForLang=foreign language
- c=toll, T=travel time
- R=reliability



Queuing Networks

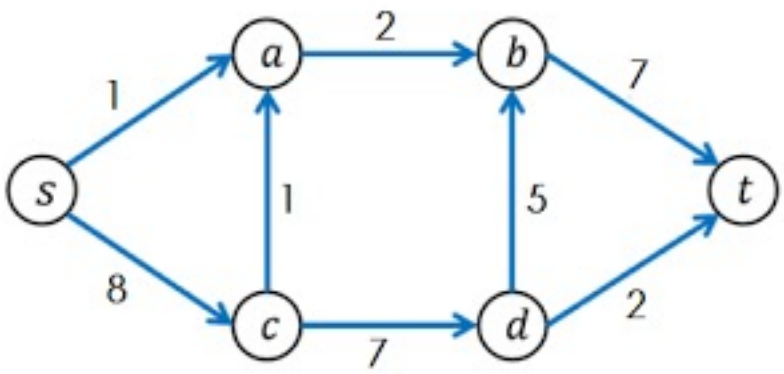
- Model
- Bernoulli routing
- Dynamic routing

Network model

- Consider a network with nodes N and links E
- We use integers to label nodes
  - Node 1, 2,…
- We use pairs of integers to label links
  - Link (1,2), (2,3),…
- Directed link (i,j)
- A “customer” (vehicle/passenger/job) enters the network via an origin node (O) and exits via a destination node (D)
- OD is predefined, but route has to be determined in real time.
- Route = a sequence of nodes

Multi-class queuing network

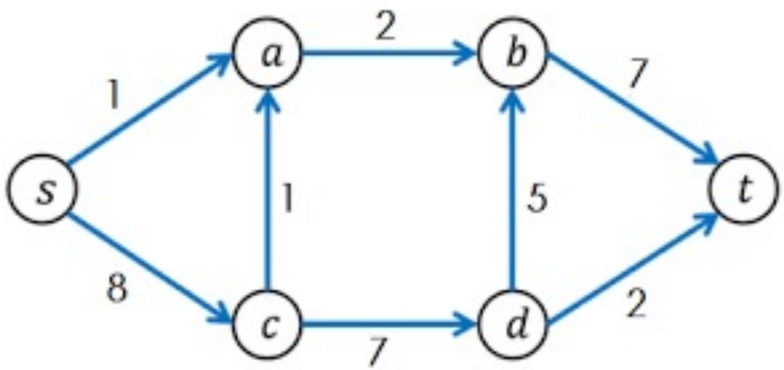
- Can we treat each customer in the same way?



- We classify customers according to their OD
- A set of classes (ODs) C
- Each class $c \in C$ has an origin o_c and a destination d_c
- Class-c arrival rate: $\lambda_c > 0$ at o_c
- Note: a node can be the origin/destination of multiple classes

Multi-class queuing network

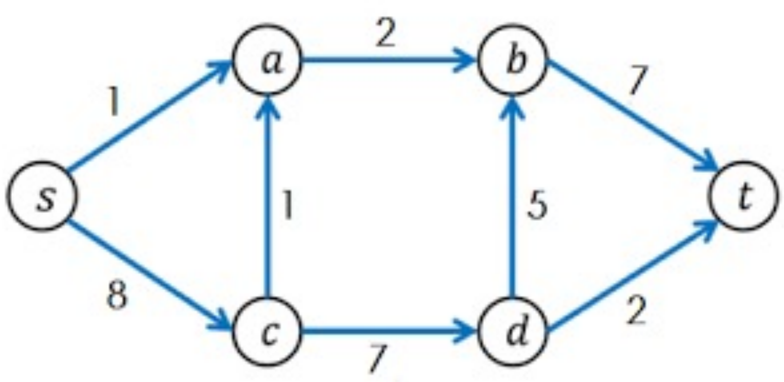
- System state: $X_{ij}^{c}(t),\ (i,j) \in E, c \in C$



- Compact notation: $X(t) = \left[X_{ij}^{c}(t)\right]_{(i,j)\in E, c\in C}$
- State space (set of states) $\mathcal{X} = \mathbb{Z}_{\geq 0}^{|E|\times|C|}$
- In general, service rate μ_i can also depend on class
- But we do not consider such complication in this lecture

Bernoulli routing

- When a customer leaves a server, it goes randomly to a downstream server with time-invariant probabilities



- For a node i with downstream nodes $Out(i;c)$ for class-c traffic, the routing probabilities are p_{ij}^c such that

$$p_{ij}^c \in [0,1] \text{ for all } j \in Out(i;c)$$

$$\sum_{j \in Out(i;c)} p_{ij}^c = 1$$

Bernoulli routing

- Data
  - Arrival rates λ_i^c & service rates μ_i
- Decision variables
  - Routing probabilities $p = \left[p_{ij}^c\right]_{(i,j),c}$
- Constraints
  - Flow conservation
- Objective
  - Minimize time-average queue sizes $\sum \bar{X}_{ij}$

Bernoulli routing

- minimize $\sum_{(i,j)\in E} \bar{X}_{ij}$

s.t. $\lambda_i^c + \sum_{j\in In(i)} f_{ji}^c = \sum_{k\in Out(i)} f_{ij}^c \quad \forall i \in N, \forall c \in C$

$\sum_{c\in C} \sum_{j\in In(i)} f_{ji}^c < \mu_i \quad \forall i \in N$

$f_{ik}^c = p_{ik}^c \sum_{j\in In(i)} f_{ji}^c \forall (i,k) \in E, \forall c \in C$

$\sum_{j\in Out(i;c)} p_{ij}^c = 1 \;\; \forall (i,j) \in E, \forall c \in C$

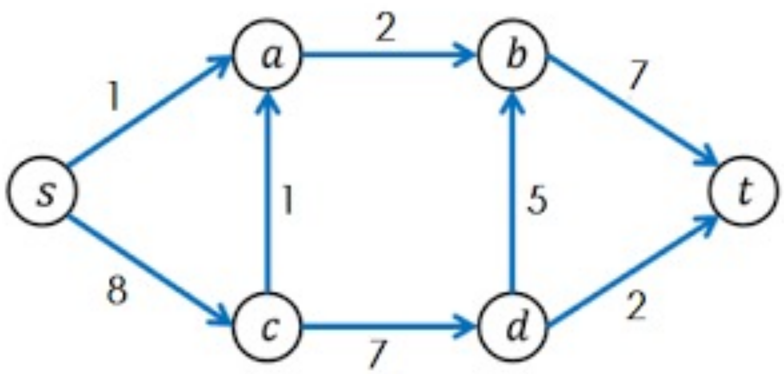
$p_{ij}^c \geq 0 \quad \forall (i,j) \in E, c \in C$

$\bar{X}_{ij} \propto \frac{\sum_{j\in In(i)} f_{ji}^c}{\mu_i - \sum_{j\in In(i)} f_{ji}^c} \quad \forall (i,j) \in E$

- Questions*
  - What is the practical and mathematical relation between p_{ij}^c and f_{ij}^c?
  - Can we simulate the system without specifying p_{ij}^c?

Dynamic routing

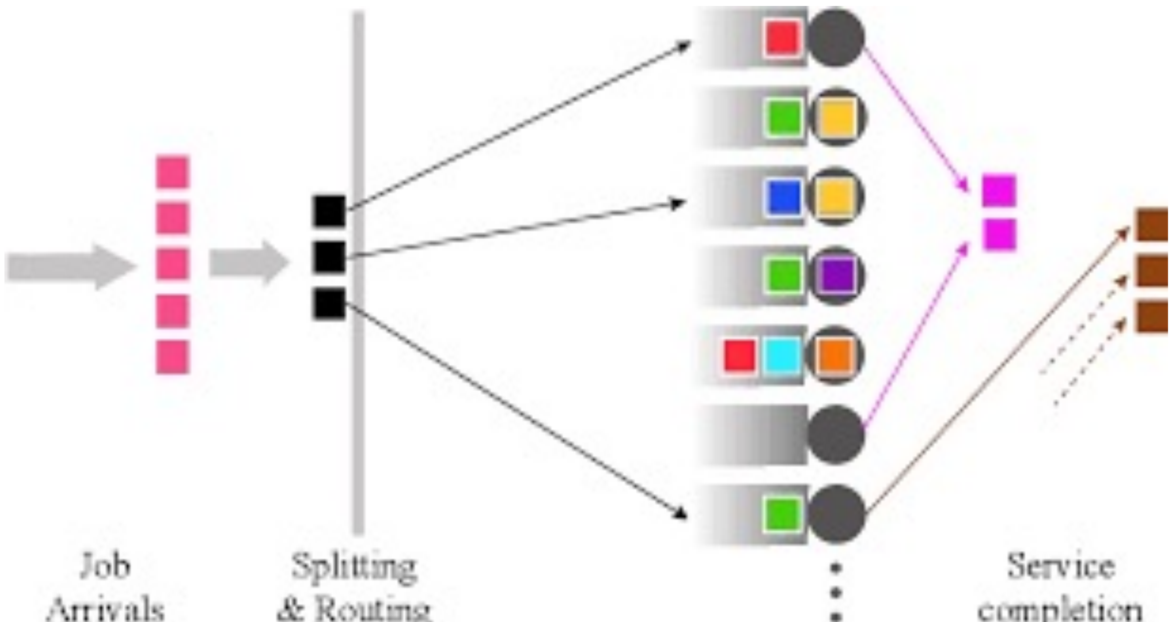
- When a customer leaves a node, its routing decision depends on real-time traffic conditions



- That is, the routing probabilities are functions of the traffic state, i.e. $\beta_{ij}^{c}: \mathcal{X} \rightarrow [0,1]$
- This is feedback control.
- Also a Markov decision process.

Does JSQ work on networks?

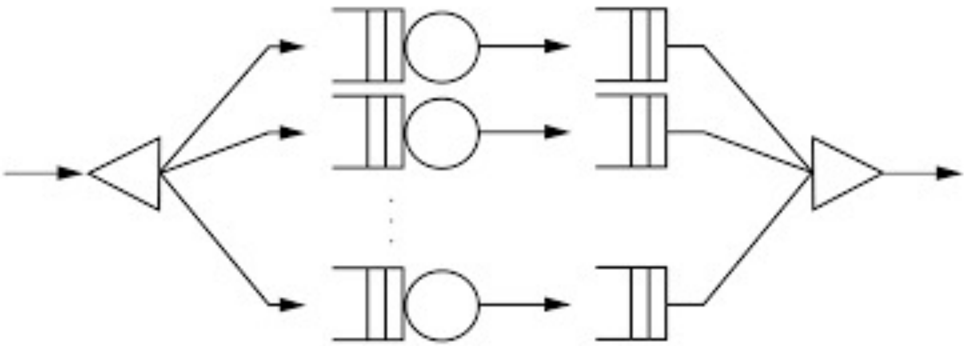
- Recall for two parallel queues, we can use the JSQ policy
- Now, suppose n parallel queues



- JSQ still works: stabilizing iff $\lambda < \sum_i \mu_i$

Does JSQ work on networks?

- However, JSQ can fail on more complex networks…



- Since JSQ is a localized routing policy, it cannot address further downstream congestions
- Fortunately, we can extend JSQ in a network setting
- "Join the shortest route" (JSR): when a customer enters the network, it selects the route with the minimal total # of customers thereon.

JSR policy*

- Suppose class-c traffic can select the route from the set R_c of routes connecting o_c and d_c
- Upon arrival, a class-c customer observes the real-time route state for each $r \in R_c$

$$Y_r(t) = \sum_{i \in r} \sum_{c \in C} X_i^c(t)$$

- Then, the incoming customer selects the route r^* such that

$$Y_{r^*}(t) = \min_{r \in R_c} Y_r(t)$$

- Alternatively, consider shortest expected service time

$$Z_r(t) = \sum_{i \in r} \sum_{c \in C} \frac{X_i^c(t)}{\mu_i}$$

Optimal routing*

- JSR ensures stability if and only if demand < capacity
- Mathematically, "demand < capacity" means that there exist f_{ij}^c such that

$$\lambda_i^c + \sum_{j \in In(i)} f_{ji}^c = \sum_{k \in Out(i)} f_{ij}^c \quad \forall i \in N, \forall c \in C$$

$$\sum_{c \in C} \sum_{j \in In(i)} f_{ji}^c < \mu_i \quad \forall i \in N$$

- However, JSR is not necessarily the optimal routing policy, in that it may not minimize expected travel time
- In general, we need reinforcement learning to optimize…

Optimal routing*

- Reinforcement learning (RL) is a class of methods to approximately solve Markov decision processes (MDPs).
- We introduced the Monte Carlo method, i.e. to simulate the model for many times and use the average result to approximate the expected performance.
- A more elegant method: temporal-difference (TD)
- Optimal routing:
  - State $X_i^c(t)$, action $b_{ij}^c(t)$, reward $\sum_i \sum_c X_i^c(t)$
  - Transition probabilities: queuing dynamics
  - Objective: find routing policy β_{ij}^c to

$$\min \mathrm{E}[\sum_{s=t}^{\infty} \gamma^{s-t} \sum_i \sum_c X_i^c(s) | X(t) = x] \ \text{for all } x$$

Optimal routing*

- To evaluate a policy β, we need to compute the value function

$$V_\beta(x) = \mathrm{E}\left[\sum_{s=t}^{\infty} \gamma^{s-t} \sum_i \sum_c X_i^c(s) \middle| X(t) = x\right]$$

- We use the subscript β to emphasize that value function depends on policy
- Exact computation of $V_\beta(x)$ is hard; approximation needed
- Solution: neural networks…
- Let's train an NN with input x and output $y = \hat{V}_\beta(x)$

Optimal routing*

- How to train the NN?
- Bellman equation

$$V_\beta(x) = r(x) + \gamma \sum_{x'} p(x'|x, \beta(x)) V_\beta(x')$$

- We start from an initial guess $V_\beta^0(x)$ (say $V_\beta^0(x) = 0$)
- $V_\beta^0(x)$ is approximated by an NN
- We simulate the network for one time step and observe the new state $X(1)$ and reward $R(1)$
- We then update the estimated values by

$$\hat{V}_\beta^1(X(1)) = R(1) + \gamma \sum_{x'} p(x'|x, \beta(x)) V_\beta^0(x')$$

Optimal routing*

- Then, we update the NN to fit the new value function

$$V_\beta^1(x) = \begin{cases} \hat{V}_\beta^1(x) & x = X(1) \\ V_\beta^0(x) & o.w. \end{cases}$$

- Subsequent iterations:

$$\hat{V}_\beta^{t+1}(X(t)) = R(1) + \gamma \sum_{x'} p(x'|x, \beta(x)) V_\beta^t(x')$$

- $V_\beta^t(x)$ is expected to converge to the true value function as $t \to \infty$
- This step is called policy evaluation, i.e. computing the value function associated with a given policy

Optimal routing*

- After evaluating a policy, we want to improve it.
- Based on the value function $V_\beta(x)$, we construct a new policy

$$\beta'(x) = \mathrm{argmin}_b \sum_{x'} p(x'|x,b) V_\beta(x')$$

- Actually, we can also use a second NN to approximate the policy $\beta'(x)$
- Two classes of NNs:
  - Value function
  - Policy
- This step is called policy improvement.
- We can repeat the above until the policy converges.

How dynamic routing addresses disruptions?

- Typical logic for dynamic routing: allocate traffic to less congested nodes…
- Demand surge: a sudden increase in demand
  - Road traffic: sports event, concert, Gaokao…
  - Data traffic: double 11 day
- Capacity drop: a sudden drop in capacity
  - Road traffic: accident, work zone
  - Data traffic: server breakdown, loss of communication
- After disruption
  - Demand surge or capacity drop leads to congestion
  - Dynamic routing responds to congestion by reducing traffic into congested links

Summary questions

- What are the demand and supply for a queuing network?
- Why queues can build up in a network?
- What are static (open-loop) routing and dynamic (closed-loop) routing?
- When is a network stable?
- What is Bernoulli routing?
- Why dynamic routing can adapt to disruptions, such as demand surge and capacity drop?

Next time

- Air traffic control
- Microscopic (trajectory) optimization
- Macroscopic (flow) optimization