Spatially Distributed Queues II

M/G/1
2 Servers
N servers: Hypercube Queueing Model Approximations
Setup: Hypercube Queueing Model

- Region comprised of geographical atoms or nodes
- Each node $j$ is an independent Poisson generator, with rate $\lambda_j$
- Travel times: $\tau_{ij} =$ travel time from node $i$ to node $j$
- $N$ servers
- Server locations are random: $l_{nj}$
Setup: Hypercube Queueing Model - con't.

- Server assignment: one assigned
- State dependent dispatching
- Service times: mean = \(1/\mu_n\), negative exponential density
- Service time dependence on travel time
- We allow a queue (FCFS, infinite capacity)
Fixed Preference Dispatch Policies for the Model

- Idea: for each atom, say Atom 12, there exists a vector of length $N$ that is the preference-ordered list of servers to assign to a customer from that atom.

- Example: $\{3,1,7,5,6,4,2\}$, for $N=7$.

- Dispatcher always will assign the most preferred available server to the customer.

- Usually order this list in terms of some travel time criterion.
Example Dispatch Policies

- **SCM: Strict Center of Mass**
  - Place server at its center of mass
  - Place customer at its center of mass
  - Estimate travel times: center of mass to center of mass

- **MCM: Modified Center of Mass**
  - Place server at its center of mass
  - Keep customer at centroid of atom
  - Estimate travel times: center of mass to centroid of atom
Example Dispatch Policies

- **EMCM**: Expected Modified Center of Mass
  - Do the conditional expected travel time calculation correctly, conditioned on the centroid of the atom containing the customer
Are fixed preference policies optimal?

AVL: Automatic Vehicle Location: dispatch the real time nearest server

This can be incorporated into the Hypercube framework, but very carefully!

Consider two servers:

\[ \text{Customer} \]
\[ \text{RA2} \quad X \quad \text{RA1} \]
Customer in square marked X. Place an asterisk in each square that could have the closest server. Assume each server is available and is located 'somewhere' in his/her square "police beat."
What to know about the Hypercube Queueing Model

- Know the 2-server setup
- Be able to work with a 3-server model
  - Read in the text the formulas to apply
- Forget the cases for \(N>3\) servers.
- Know Hypercube Approximation Procedure (still to come -- fasten your seat belts!)
Hypercube Approximation
Procedure: A General Technique

- Want to reduce dramatically the number of simultaneous equations to solve.
- The procedure reduces the number of equations from $2^N$ simultaneous linear equations to $N$ simultaneous nonlinear equations.
- We look at only those performance measures we need, not at micro-structure of the binary state space.
Hypercube Approximation Procedure
A General Technique

Theory: Sampling Servers Without Replacement from $M/M/N$ Queue

From $M/M/N/\infty$ we know the aggregate state probabilities:

\[ P\{S_k\} \equiv P_k = \frac{N^k \rho^k P_0}{k!} \quad k = 0, 1, 2, \ldots, N - 1 \]

\[ P\{S_N\} \equiv P_N = \frac{N^N \rho^N P_0}{(N! [1 - \rho])} \]

\[ P\{S_0\} \equiv P_0 = \left[ \sum_{i=0}^{N-1} N^i \rho^i / i! + N^N \rho^N / (N! [1 - \rho]) \right]^{-1} \]
The Hypercube Model, when the state space is compressed from its cube in $N$ dimensions to a 'line' birth and death process, always reduces to an $M/M/N$ queue (assuming service times are not server-specific)
For our applications, we do not need to know the fine grained binary state probabilities. Rather we need dispatch probabilities and server workloads.

What about 'B-' probability reasoning? "Flips coins" until first Heads is obtained:

\[
P\{B_1, B_2, ..., B_j, F_{j+1}\} \approx \begin{cases} 
\rho^j (1 - \rho) & j = 0,1,2,...,N-1 \\
\rho^N & j = N
\end{cases}
\]

Incompatible with known state probability \(P_N\)

Doesn't include biases.

Let's "Divide and conquer":

\[
P\{B_1, B_2, ..., B_j, F_{j+1}\} = \sum_{k=0}^{k=N} P\{B_1, B_2, ..., B_j, F_{j+1} \mid S_k\} P_k \quad (*)
\]
Working carefully and slowly to find the state-conditioned dispatch probabilities:

\[
P\{B_1, B_2, \ldots, B_j, F_{j+1} \mid S_k\} = P\{F_{j+1} \mid B_1, B_2, \ldots, B_j, S_k\} \ldots P\{B_2 \mid B_1 S_k\} P\{B_1 \mid S_k\}
\]

\[
P\{B_1, B_2, \ldots, B_j, F_{j+1} \mid S_k\} = \frac{N - k}{N - j} \ldots \frac{k - 1}{N - 1} \ldots \frac{k}{N} \quad (**)
\]

Can plug (**) back into (*) and obtain an exact expression. Manipulate it to obtain a convenient form as "B-" probability reasoning with an 'A+' correction term:

\[
P\{B_1, B_2, \ldots, B_j, F_{j+1}\} = Q(N, \rho, j) \rho^j (1 - \rho) \quad (***)
\]

"Correction factor"
Explore properties of Correction Factor

The desired dispatch probabilities can be written as a telescopied expression:

\[
P\{B_1, B_2, \ldots, B_j, F_{j+1}\} = P\{F_{j+1} \mid B_1 B_2 \ldots B_j\} P\{B_j \mid B_1 B_2 \ldots B_{j-1}\} \ldots P\{B_1\}
\]

Use above in Eq.(*** ) to obtain:

\[
Q(N, r, j) = \left[\frac{P\{F_{j+1} \mid B_1 \ldots B_j\}}{1 - \rho}\right] \left[\frac{P\{B_j \mid B_1 \ldots B_{j-1}\}}{\rho}\right] \ldots \left[\frac{P\{B_1\}}{\rho}\right]
\]

\[
\leq 1 \quad \geq 1 \quad = 1
\]
\( G^k_n \equiv \) set of geographical atoms for which unit \( n \) is the \( k^{th} \) preferred dispatch alternative

\( n_{lj} \equiv \) id \# of the \( j^{th} \) preferred unit for atom \( l \)

Set \( \mu = 1 \)

\[
\rho_n = \sum_{j \in G^1_n} \lambda_j P\{F_n\} + \sum_{j \in G^2_n} \lambda_j P\{B_{nj_1} F_n\} + \sum_{j \in G^3_n} \lambda_j P\{B_{nj_1} B_{nj_2} F_n\} + ... + \lambda P_N / N
\]

\[
\rho_n = \sum_{j \in G^1_n} \lambda_j (1 - \rho_n) + \sum_{j \in G^2_n} \lambda_j Q(N, \rho, 1) \rho_{nj_1} (1 - \rho_n) + \sum_{j \in G^3_n} \lambda_j Q(N, \rho, 2) \rho_{nj_1} \rho_{nj_2} (1 - \rho_n) + ... + \lambda P_N / N
\]
The last equation gives $N$ nonlinear simultaneous equations in
the unknown workloads, $\rho_n$, subject to the constraint that

$$\sum_{n=1}^{N} \rho_n = \lambda \quad "normalization"$$

Typically converges in 3 to 5 iterations, within 1 to 2% of 'exact Hypercube' results

Response patterns:

$$f_{nkj} = \frac{\lambda_k}{\lambda} Q(N, \rho, j - 1) \left\{ \prod_{l=1}^{j-1} \rho_{nkl} \right\} (1 - \rho_{nkj})$$

- $\text{id} \# \text{ of } j^{th} \text{ preferred unit for atom } k$
- $j-1 \text{ more preferred units}$
- $j^{th} \text{ preferred unit}$
Square Root Laws (approximations)

In Chapter 3 we found

\[ E[D] = C \sqrt{\frac{A}{N_0}} \]

- Area of service region
- Number of mobile servers
- depends on distance metric and location strategy

Assumes all \( N_0 \) servers are available or free (not busy)
Now consider $N$ to be a R.V.

Might we expect the following to be true?

$$E[D \mid N = k] = C \sqrt{\frac{A}{k}} \quad k = 1, 2, \ldots, N_0$$

What if the locations of servers were determined by a homogenous spatial Poisson process, with busy servers selected by "random erasers"?
Getting to Expected Travel Distance

\[ E[D] = P_0 D_0 + \sum_{k=1}^{N_0} P_k C \sqrt{\frac{A}{k}} \]

From \( M/M/N_0 \) queueing model

where \( P_k \) = Probability of \( k \) servers available \( (M/M/N_0) \)
Moving to $E[D]$

Since $P_0 \approx 0$, we can write

$$E[D] \approx C \sqrt{A E_{\text{states of } M / M / N_0}} \left[1 / \sqrt{N} \right]$$

We now apply "B-" probability reasoning, to get

$$E[D] \approx C \sqrt{\frac{A}{E[N]}}$$

(Jensen's Inequality shows that this Eq. is a lower bound to true $E[D]$.)
Finishing

\[ E[N] \approx N_0 - N_0 \rho = N_0 (1 - \rho) \]

\[ E[D] \approx C \sqrt{\frac{A}{N_0 (1 - \rho)}} \]

\[ E[T] \approx \frac{C}{v} \sqrt{\frac{A}{N_0 (1 - \rho)}} + \frac{v}{a} \]

Acceleration term

Great results in practice
Jensen's Inequality

If $g(X)$ is a convex function over the region of non-zero probability, then

$$E[g(X)] \geq g(E[X])$$

(Problem 5.5 explores this further.)
Jensen's Inequality

\[ \frac{1}{\sqrt{N}} \]
E[1/\sqrt{N}] = 0.5*(1) + 0.5*(10)^{-0.5} = 0.5*(1 + 0.316) = 0.658

1/E[N]^{-0.5} = 1/(1*0.5 + 10*0.5)^{-0.5} = 1/(5.5)^{-0.5} = 1/2.345 = 0.426

Suppose 50% of probability mass here

"B-" mean (0.426)

And suppose 50% of probability mass here

True mean (0.658)