Tutorial: Optimal Control of Queueing Networks

Mike Veatch

Presented at INFORMS Austin

November 7, 2010
Overview

- Network models
- MDP formulations: features, efficient formulations
- Software
- Greedy policies
- Fluid model policies
- Monotone control: switching curve policies
Answer questions such as
How does the length of a production line affect optimal work-in process?
How does the amount of cross-training of workers affect waiting time?
How does the choice of which queue to serve depend on the queue lengths in a reentrant flow system?

Moderate size networks

Often interested in optimal policy or its structural properties

Model the features that matter for the parameter range of interest. Check by simulating if needed.

Understand greedy actions and the tradeoff that makes other actions optimal

Understand the deterministic (fluid) equivalent
Series line  Decisions: idle/busy at upstream servers \((c_1 < c_2 \ldots < c_n)\)

Make-to-stock manufacturing system: finished goods are held to meet demand. Decisions: idle/busy; Veatch and Wein (1994, 1996)

Decisions: class served by each server (or server idles)

**Multiple part types**

Part type 1

\[ \ldots \]

Part type \( k \)

Machine 1 Machine 2 \[ \ldots \] Machine \( m \)

**Reentrant flow**

\[ \alpha_1 \]

\[ \mu_1 \]

\[ \mu_2 \]

\[ \mu_3 \]
- Multiple part types, reentrant flow, probabilistic routing

Taken from: S. Meyn, *Controlling Complex Networks*
Parallel flexible servers
Harrison (1998)
Bell and Williams (2001)
- Arbitrary mapping from servers to classes

Series line with flexible servers
Andradottir, Ayhan, and Down (2003)
Andradottir Ayhan (2005)

Additional features Harrison, A broader view of Brownian networks (2003)
- Activities can use multiple servers, multiple job classes, and can produce multiple jobs (join and split)
Overview

- Network models
- MDP formulations: features, efficient formulations
- Software
- Greedy policies
- Fluid model policies
- Monotone control: switching curve policies
MDP Modeling Issues

- Describing states and transitions in $\mathbb{Z}^n_+$ gives a common language (not just manufacturing, communications, call centers, etc.)

**Choosing the objective**

- Time horizon?
  - Infinite-horizon average cost is often realistic and avoids having to choose a discount rate
  - Discounted has better theory

- Queue length or throughput/service level?
  - This talk uses average weighted queue length
  - Also practical to minimize queue length subject to a service level constraint, giving a constrained MDP. Sennott (2001), Altman (1999)
queue lengths \( x = (x_1, \ldots, x_n) \) \( n \) classes

server \( \sigma(i) \) serves class \( i \)

exponential service/interarrival times,

rates \( \mu_i \) and \( \lambda_i \),

control \( u_i = 1 \) if serve class \( i \), \( 0 \) otherwise \( \sum_i (\lambda_i + \mu_i) = 1 \)

feasible actions: \( u_i \leq x_i \) and \( \sum_{i: \sigma(i)=\sigma} u_i \leq 1 \) for all machines \( \sigma \)

Convert to discrete time

uniformized transition probability matrix under policy \( u \) \( P_u \)

one-stage cost \( c^T x \)

average cost \( J \)

differential cost of starting in state \( x \) relative to state 0: \( h(x) \)

Bellman’s equation

\[
J + h(x) = c^T x + \min_{u \in A(x)} (P_u h)(x), \quad x \in \mathbb{Z}_+^n
\]

Solutions are optimal average cost \( J^* \) and differential cost \( h^*(x) \)
To simplify notation: no routing control, splitting/merging or simultaneously serving more than one customer with the same resource

$A_{kj}$ units of resource $k$ required to perform activity $j$

$q_k$ units of resource $k$

$R_{ij} = 1$ if activity $j$ serves class $i$ and 0 otherwise

$p_{ij}$ = probability that a customer finishing service at class $i$ will be routed to class $j$

$p_{i0}$ = probability that a customer finishing service at class $i$ will depart

Feasible actions: $Au \leq q$, $Ru \geq x$, $u \geq 0$

Stability

effective arrival rate $\lambda$: $\lambda = \alpha + P^T \lambda$

$M = \text{diag}(\mu_j)$

static allocation LP must have optimal solution $\rho < 1$

Possible transitions: arrivals, routing from $i$ to $j$, departure

$$\begin{align*}
\min & \quad \rho \\
\text{s.t.} & \quad Au \leq \rho q \\
& \quad RMu = \lambda \\
& \quad u \geq 0
\end{align*}$$
State Space Truncations

- Construct a sequence of MDPs with finite state spaces $S_N$ that increase to the whole state space, $S = \bigcup_{N=1}^{\infty} S_N$
- Transitions out of $S_N$ are turned off or mapped to some state in $S_N$

**Theorem** (Sennott 1999) If solutions $h$ to Bellman’s equation are bounded above in $N$ (and bounded below in $x$), then the sequence of finite state MDPs converges to the countable state MDP in average cost and policy

- Motivates truncating the space
- Naïve truncation: $x_i \leq N$
- Better: $x_i \leq N_i$ with $N_i$ larger for classes with arrivals
- Searching for accurate $\{N_i\}$ requires many runs
- The optimal policy on a truncated space may also limit queue lengths in classes without arrivals
- Can determine these limits from the policy, at least for series lines Veatch (2006)
- Other models have clearer truncations, e.g., max inventory level
Non-exponential Distributions

Method of stages
- Service time is sum of exponentials (smaller coefficient of variation)
- Adding $S$ stages multiplies number of states by $S$ for each server
- Can uniformize to obtain discrete time Markov chain as before

Embed DTMC at service completions (one non-exponential distr.)
- Number of arrivals in a period ~ Poisson. Increases number of transitions (nonzeros in $P$)

Discretization: Sample at discrete times
- Number of potential events in a period ~ Poisson.

Deterministic service times
- Machines often have nearly constant processing times plus failures
- Discretize. If service time = time step, no additional states are needed.
  Service times must be equal
- If service times are small integer ratios, service time = $k$(time step)
  Must keep track of “age,” multiplying number of states by $k$
Machine Failures

- $\alpha_i = 1$ if machine $i$ is operational, 0 if failed

- Each unreliable machine doubles the number of states

- Approximate model: limit the number of failed states and adjust failure rate Veatch (2006)

- Use with deterministic service times: one source of variability
Speed-ups

- Avoid self-transitions when uniformizing, e.g., use maximum service rate of a server, not the sum of its service rates
- Initialize $h$ from previous runs or a simulated policy
- Apply penalty at queue length boundary: If $x_i = N_i$ and a transition increases $x_i$, project the new state back to the boundary and apply a penalty computed from a previous run
- Asynchronous value iteration
- Numerical results for a make-to-stock series line with 5 unreliable machines. Runs with 12 million states took 7 hours

<table>
<thead>
<tr>
<th>Enhancements</th>
<th>Iterations</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>3300</td>
<td>357</td>
</tr>
<tr>
<td>Asynchronous VI</td>
<td>2900</td>
<td>258</td>
</tr>
<tr>
<td>Initialization</td>
<td>1100</td>
<td>93.6</td>
</tr>
<tr>
<td>Boundary penalty</td>
<td>800</td>
<td>72.8</td>
</tr>
<tr>
<td>Limited number of failures</td>
<td>800</td>
<td>36.9</td>
</tr>
</tbody>
</table>
Overview

- Network models
- MDP formulations: features, efficient formulations
- **Software**
- Greedy policies
- Fluid model policies
- Monotone control: switching curve policies
- Stochastic processing network structure is an input!
  - Number of classes determines number of nested loops: creates n-dimensional counters. Outer loops use counter, inner loop just increments an integer
  - Loop over feasible actions
  - Arbitrary routing within the network: compute $(P_u h)(x)$
Software: Usability

- Commented input file contains all data
  - network structure: server-to-class mapping, routing
  - arrival and service rates
  - algorithm parameters, including queue length truncations
- Includes documentation
- Extensively tested and used at Gordon College; some testing by other researchers
- Access: Download and compile C++ code
  - [http://www.math-cs.gordon.edu/~senning/software/qnetdp-1.1.1.tar.gz](http://www.math-cs.gordon.edu/~senning/software/qnetdp-1.1.1.tar.gz)
  (look on [http://www.math-cs.gordon.edu/~senning/qnetdp](http://www.math-cs.gordon.edu/~senning/qnetdp) to get the latest version).
Software: Speed

- Solved 6-class example with traffic intensity of 0.6 with truncation (40, 2, 40, 10, 4, 4) in ~1 hour
- Truncation has 1.4 million states, appears accurate to within 0.1%
\( c = (1,3), \mu = (1, .5, 1), \text{traffic intensity} = .9 \)

- As arrival rate \( \lambda_1 \) increases, more “helping” is used

\( \lambda_1 = 0.9 \)
\( \lambda_1 = 1 \)
\( \lambda_1 = 1.1 \)

○ serve class 2
○ serve class 1 (help)
Optimal Policy-: 3 x 3 Floater

Server 3:
- serve class 1
- serve class 2
- serve class 3

- Resembles Longest Queue policy, particularly in states with small $x_i$
Server 1 gives nearly static priority to class 1 (highest cost)
Server 3 often serves class 2 (middle cost), especially when $x_1$ is large
Explanation: server 2 is serving class 1 in these states
Overview

- Network models
- MDP formulations: features, efficient formulations
- Software
- **Greedy policies**
- Fluid model policies
- Monotone control: switching curve policies
Greedy Policies

**Greedy policy**

\[ \tilde{u}(x) = \arg \min_{u \in A(x)} E_u \left( c^T x(t + 1) \mid x(t) = x \right) \]

- A network admits a pathwise optimal solution if, from any initial state, there is a policy that minimizes the cost rate at all times \( t \) with probability one
  - Implies that greedy is optimal
- For linear cost \( c^T x \), greedy action depends only on which buffers are empty, so greedy is static priority for \( x > 0 \)
- Examples
  - Multiclass queue: \( c\mu \) rule
  - Klimov’s problem: single server with rework
  - Other systems with certain parameter values
Stochastic coupling

- Walrand (1988) route to shortest queue
- Liu, Nain, Towsley (1995) sample path methods
- de Vericourt and Veatch (2003) make-to-stock queue
- Trajectories under two different policies are compared
- Show that one trajectory has smaller (or equal) cost rate at all times; may eventually merge with probability one

Preservation by DP operator

- If the greedy policy is also $h^*$-greedy, it is optimal

$$u^*(x) = \arg \min_{u \in A(x)} E_u \left( h(x(t+1)) \mid x(t) = x \right)$$

- Show that this property is preserved by the DP operator. Then greedy is optimal for all time horizons.
A $c\mu$ Rule for the “N” System

**Theorem.** If $c_2\mu_{22} \geq c_1\mu_{21}$ then the $c\mu$ rule is optimal.

- The $c\mu$ rule sets Server 2’s priorities. When $x_1 = 1$, resolve competition for class 1 customer using greedy.

- For the case in the theorem:
  - Server 2 gives priority to class 2: “fixed before help”
  - Uses the faster server when $x = (1, 0)$
A \( c\mu \) Rule for \( m \times m \) Two-tiered

Theorem. If servers can cooperate and \( c_i \mu_{ii} \geq c_1 \mu_{i1}, \ i = 2, \ldots, m \) then the \( c\mu \) rule is optimal.

The \( c\mu \) rule for this case:

- Server 2: priority to class 2
- Server 3: priority to class 3
- etc.
**cμ Rules—Parallel Flexible Servers**

**Multiclass queue:** Buyukkoc, Varaiya, and Walrand (1985)
- Time interchange argument
- Uses discretization (or discrete time model) so that arrivals are independent of service

\[ c_2 \mu_{22} \geq c_1 \mu_{21} \]

**“N” system with “fixed before help”**
- Down and Lewis (2008) Policy is preserved by DP operator
- Veatch (2008 w.p.) Stochastic coupling: processes merge, but policy is not just a time interchange

**“W” system with “fixed before help” and failures**
- Saghafian, Van Oyen, and Kolfal (2009 w.p.) Policy is preserved by DP operator

**m × m Two-tiered system with “fixed before help”**
- Stochastic coupling: processes do not (always) merge, but expected future cost is equal
When Are Greedy Policies Not Optimal?

- There must be a *tradeoff* between short-term and future costs.
- Future costs are incurred by the greedy policy, relative to an optimal policy, only when a queue empties under the greedy policy, forcing the action to change—e.g., idling a server.
- The optimal policy uses safety stock, or buffering, to prevent future idleness.
- The queue may empty because a slower server can’t keep up or because of random service times.

\[
\alpha = 1, \mu = (0.5, 1.5), c = (1, 1.5)
\]
When to Consider Switching Curves

- First identify a \textit{tradeoff} between short-term and future costs

- Does the future cost occur with sufficient probability to be significant?

- Sensitivities are near boundaries of the state space (small queues)—the exact location of the switching curve usually doesn’t matter
  
  Chen, Pandit and Meyn (2003) In search of sensitivity…

- The queue may empty because a slower server can’t keep up or because of random service times
Overview

- Network models
- MDP formulations: features, efficient formulations
- Software
- Greedy policies
- Fluid model policies
- Monotone control: switching curve policies
Fluid Model

- Replace processes by their mean rates. Continuous, deterministic, transient

\[ V^*(x) = \min_u \int_0^T c'q(t)dt \]

where

\[ \dot{q}(t) = Bu(t) + \alpha \]

\[ q(0) = x \]

<table>
<thead>
<tr>
<th>Discrete feasibility (DFEAS)</th>
<th>Fluid feasibility (FFEAS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C\tilde{u}(t) \leq e )</td>
<td>( Cu(t) \leq 1 )</td>
</tr>
<tr>
<td>( \tilde{u}(t) \geq 0 )</td>
<td>( u(t) \geq 0 )</td>
</tr>
<tr>
<td>boundary:</td>
<td>boundary:</td>
</tr>
<tr>
<td>( \tilde{u}(t) = 0 ) if ( x_i(t) = 0 )</td>
<td>([Bu(t) + \lambda v(t)]_i \geq 0 ) if ( q_i(t) = 0 )</td>
</tr>
</tbody>
</table>

- (FFEAS) does not imply (DFEAS)
  - Greedy for fluid differs from greedy for discrete only on boundaries
  - There may be a translation feasibility problem
Fluid Limits and Asymptotic Optimality

For each work-conserving policy and $x \in \mathbb{R}^n_+$, consider a sequence of processes $x^N$ with initial states $N x$
cumulative allocation: $\tilde{T}_i(t) = \int_0^t \tilde{u}_i(s)ds$

Existence of fluid limit Dai (1995) For a.e. sample path, there is a subsequence of
$\frac{1}{N} \left( x^N(Nt), \tilde{T}^N(Nt) \right)$ that converges u.o.c. to $(q(t), T(t))$ satisfying

$(\text{FFEAS})$, the fluid dynamics, and $T_i(t) = \int_0^t u_i(s)ds$.

Asymptotic optimality Meyn (2000) If a stable policy exists, then there exists
an optimal policy for the MDP whose fluid limits are optimal for the fluid.
Further, $\lim_{\theta \to \infty} \frac{h^*(\theta x)}{V^*(\theta x)} = 1$

- The fluid limit may not be unique or easily constructed
Scaling the Policy

**scaled policy:** \( \bar{u}(x) = \lim_{N \to \infty} \tilde{u}(\lfloor Nx_1 \rfloor, \ldots, \lfloor Nx_n \rfloor) \) when limit exists

**fluid limit policy:** Under mild conditions, the collection of fluid limit trajectories under a policy defines a fluid policy \( u(x) \) except at \( x \) where

- the control changes (the allocation \( T(t) \) is not differentiable) or
- the fluid limit is not unique

… Asssume:

1) Unique fluid limits from all initial states (except a set of lower dim.)
2) **Scalable:** scaled policy exists and consists only of extreme points (""")

**control switching sets (CSSs):** region where a certain action is used

**Theorem** In the interior of a CSS of full dimension, the fluid limit policy matches the scaled policy.

**Corollary** If a stable policy exists for the MDP, then there exists a discrete optimal policy whose scaled policy matches some fluid optimal policy in the interior of CSSs of full dimension.

\[
\text{Asymptotic slopes of switching curves agree for some discrete optimal policy and fluid optimal policy.}
\]
Arrival Routing: Fluid and MDP Policy

Case 1. $\alpha \geq \mu_2$
$\alpha = 1$, $\mu = (0.65, 0.65)$, $c = (1, 2)$

Case 2. $\alpha < \mu_2$
$\alpha = 1$, $\mu = (0.5, 1.5)$, $c = (1, 1.5)$
Fluid Switching Curves—Summary

- Holds for other problems and higher dimensions: For linear cost sequencing and routing problems, the fluid policy defines asymptotic slopes of switching surfaces
- Fluid policies can be computed for small or simple problems
- Distinguish three cases:

  1) *Neither is greedy*. Both have interior switching; fluid policy is “close.” Non-greedy policy because a server will fall behind.
  2) *Fluid is greedy*. MDP has interior switching and is non-greedy to buffer against randomness.
  3) *Both are greedy*. Fluid policy matches MDP policy except on boundary
Overview

- Network models
- MDP formulations: features, efficient formulations
- Software
- Greedy policies
- Fluid model policies
- **Monotone control: switching curve policies**
Monotone Control

- Optimal policies have switching curve form in most models: serve class $i$ when $x_k \leq s(x)$ and class $j$ otherwise ($x_i, x_j > 0$)
- Generalizes to models with costs of service: optimal class $i$ service rate is increasing (decreasing) in $x_j$, known as monotone control

Geometry of transitions

- Serve departing class $i$: transition $x \rightarrow x - e_i$ will be decreasing in $x_j$ if $h$ is submodular: $h(x + e_i) - h(x) \geq h(x + e_i + e_j) - h(x + e_j)$
- Interpretation: a departure from queue $j$ can turn class $i$ service on but not off (increase service rate). More-more relationship
- Supermodular: $h(x + e_i) - h(x) \leq h(x + e_i + e_j) - h(x + e_j)$
  A departure from queue $j$ can turn class $i$ service off but not on (increase service rate). More-less relationship
Submodularity-type Inequalities

- Can extend submodularity to other transitions $d_i$ and $d_j$
  \[ h(x + d_i) - h(x) \geq h(x + d_i + d_j) - h(x + d_j) \quad \text{Veatch and Wein (1992)} \]
- Altman et al. (2003) extensively study the most common transitions, i.e., $d = e_i - e_j$ for a customer moving from one class to another
- Convexity is often included with submodularity. It can be interpreted as: a transition occurring can turn that transition off but not on.

Implications for switching curves
- Submodularity and convexity imply that the switching curve for a class $i$ departures is monotonic in $x_j$
- Similar limitations on switching curves for other transitions; see Veatch and Wein (1992)
Establishing Submodularity-type Inequalities

- Show that the inequalities (generally including convexity) hold for the cost rate and are preserved by the DP operator.
- Many papers do this in an ad hoc fashion for one model.
- Most difficulties occur at the state space boundaries; some general results in Weber and Stidham (1987) and Veatch and Wein (1992).
- The score space approach of Glasserman and Yao (1994) converts all transitions to $e_i$ but increases the dimension.
- Koole (2007) provides a general framework using event-based DP: the DP operator is decomposed into simple operators, each of which is shown to preserve certain inequalities.
- Induction on $x_i$ has been used on some problems Wu, Lewis, and Veatch (2005).
Conclusion

- Computing power and careful truncation, etc. make it possible to compute optimal control for interesting networks with several classes and additional features.
- General software makes it convenient (“standard” stochastic processing networks).
- Look for tradeoffs and main features of a model before numerically optimizing.
- Policy visualization software makes it easier to look for policy insights.
- Fluid analysis can give more information about the optimal policy: asymptotic slopes of switching curves.
- Switching curve policies are pervasive and can be proven for some networks.