USING FLUID SOLUTIONS IN DYNAMIC SCHEDULING

Michael H. Veatch
Department of Mathematics
Gordon College
Wenham, MA 01984
veatch@gordon.edu

Abstract We review some recent work on fluid approximation of queueing network control problems and present new results. The relationship between the optimal control of a queueing network and the optimal control of the associated fluid model is investigated. A general theory, a number of examples, and a few numerical results are presented. The fluid model replaces stochastic counting processes by their mean rates. Surprisingly, although the fluid model is deterministic and transient, it often contains valuable information about the original problem. We show that, under some technical assumptions, the asymptotic slope (normal vector in higher dimensions) of the switching surfaces for large queue lengths can be found from the fluid model. The implications of the fluid policy for designing manufacturing scheduling and control policies are discussed. Unlike the queueing network control problem, which is usually an intractable dynamic program, very efficient algorithms exist to solve the fluid problem from a given initial buffer state. Fluid solutions are presented for a number of examples, some of them new.

Keywords: fluid models, dynamic scheduling, switching point controls, fluid limits

1. Introduction

Considerable attention has been paid recently to fluid models of stochastic network control problems. In the fluid model, stochastic counting processes are replaced by their mean rates. The fluid model provides a highly simplified framework for analysis. It has deterministic, continuous trajectories. For the models we consider–mostly open multiclass queueing networks with sequencing, arrival routing, and input control–the original problem has an average cost criterion but the fluid model is transient, minimizing the "cost to drain." In light of these differences, fluid models have often been viewed as being at a much higher level in the hierarchy of models. In this view, the fluid model is useful when a highly aggregated model is needed. If large time scales and queue lengths are of interest then fluid models are a natural choice because these are precisely the quantities that are scaled in fluid limit theorems. [Che94] proposes that these fluid models are suitable for long-range strategic planning, while queueing networks are appropriate for short-range, e.g., weekly, control of production operations.

This paper seeks to demonstrate another role for fluid models. Surprisingly, their solution often contains valuable information about the original problem, almost regardless of the "level of aggregation" at which decisions are to be made. Call the optimal policy for the original network the *discrete policy* and the optimal policy for the fluid model the *fluid policy.* We present an introduction to minimum cost draining of fluid models, review the existing theory relating the discrete and fluid policies, build on this theory to establish characteristics of the fluid policy which are shared by the discrete policy, and provide several examples to illustrate these connections. Two of the examples compare the fluid policy with the policy obtained from a diffusion approximation. Most of the fluid solutions are taken from earlier papers. In presenting the case for using fluid models, much of the work on fluid optimality is reviewed; however, no attempt is made to systematically survey the field and many papers are not mentioned. For a survey of fluid models and limit theorems, mostly without optimization, see [Che94]. A compilation of diffusion solutions is given by [Kel93]; [Wil96] gives a broader review of the Brownian control literature, emphasizing limit theorems.

Fluid models under a given policy are, under very general conditions, the weak limit of a sequence of systems where time and queue length are scaled by the same large factor; see [Dai95]. Thus, queue length vectors in the fluid model correspond to large vectors with the same proportions in the discrete system. Intuitively, when queue lengths are large, due to an unusual event such as a burst of arrivals, the optimal policies should be similar because of this scaling argument. We give some general results on the similarity between the discrete and fluid policy and explain the implications for designing manufacturing scheduling and control policies through a number of examples. For an important class of problems, the fluid model is (i) tractable (when the discrete system is not), (ii) insightful, in that the fluid policy provides understanding of the discrete policy, and (iii) useful, in that the fluid policy–possibly after translation or correction back to the discrete system–is stable, near optimal, and relatively simple to implement.

First, an efficient algorithm exists for a very general class of fluid models with linear costs and dynamics that are linear in the control. The algorithm of [Luo98] uses a separated continuous linear programming formulation, time discretization, and the fact that the problem admits piece-wise constant optimal controls to find them exactly. There are no restrictions on the network topology–reentrant lines and multiclass sequencing problems are included. In fact, the algorithm applies to all our examples. This algorithm solves for the (deterministic) trajectory from a given initial state. For smaller or specially structured problems, the control *policy* can be found; more examples are given in [ABR95], [Weis95], and [Set99]. Both of these methods are used in this paper. It should be noted that the fluid model is more tractable than the Brownian diffusion model, where solutions can usually be found only if a state space collapse condition is met or the problem admits pathwise optimal solutions. Flow control models, which have fluid dynamics in time intervals punctuated by Markovian changes in the dynamics (e.g., machine failures), are also not as tractable as fluid models.

The insights drawn from fluid models begin by dividing them into two types. Greedy solutions are discussed in Section 3. If costs are linear, these solutions use static priorities when all buffers are nonempty. In this case the discrete and fluid policy use the same priorities, differing only on the boundary of the state space, if at all. A single station with feedback is used to illustrate these connections. In this case the fluid model retains most or all the information needed to solve the discrete problem. Make-to-stock queues are also considered. They demonstrate a general limitation of fluid policies: They do not use safety stock.

Solutions containing switching curves are discussed in Section 4. For systems with linear costs, the fluid policy has linear switching curves (surfaces if more than two dimensions) through the origin. Meyn [Mey97], [Mey98], [Mey01a] has established that optimality of the discrete policy is preserved in the fluid limit. We use his result to show that, under some technical assumptions, the discrete switching curves (surfaces) have the same asymptotic slope (normal vector) as in the fluid. Thus, when the fluid policy switches in the interior it captures the main feature of the discrete policy. However, if the fluid policy switches on the boundary of the state space (e.g., when a buffer empties), the discrete policy will generally switch in the interior and the fluid fails to capture the essential characteristic of the discrete policy. An arrival routing example is used to numerically verify this relationship and demonstrate that the fluid policy can be near-optimal for the discrete system. Fluid solutions are also given for a series queue [ABR95] and a series make-to-stock system.

Finally, fluid trajectories are generated for the input control problem of [Wein90] and compared with his diffusion solution.

For a fluid policy to be useful, it must be translated into a discrete policy that is stable, near optimal, and reasonably easy to implement. Usually, adjustments must be made at the boundary of the state space, where the fluid policy moves material through empty buffers. Unfortunately, naive translations, such as serving the next lower priority class when a buffer is empty, might not even be stable; an example is given in Section 2. [Mag00] constructs discrete policies by tracking the fluid policy but enforcing safety stocks. These policies are stable and asymptotically optimal under fluid scaling. However, this translation introduces some artificial policy features, such as discrete review, that may not be desirable in the actual system. The nonidling property is also violated. [Bau00] shows that implementing the fluid policy in an open-loop fashion, based only on the initial state, is asymptotically optimal with discounting. This result illustrates that asymptotic optimality is a very mild criterion. It has the same fundamental limitations as the fluid model. It appears that translating a policy in a near-optimal fashion is largely an open question. See [Eng96] for work on one example and [Mey01a] for some general proposals. However, some of our examples avoid this difficulty. For the arrival routing example of Section 4.2, the fluid policy can be directly applied to the discrete system. For the examples in Section 3, a naive translation is either optimal or has little impact.

The next section introduces the discrete stochastic network control problem and its fluid model. After discussing the greedy and switching curve policies, conclusions are given in Section 5.

2. The Discrete and Fluid Models

We present the discrete stochastic network in the Markov decision process context. However, more general processes also have fluid limits; see [Dai95]. Consider a network containing m single-server stations indexed by $j = 1, \dots, m$ and n classes of jobs indexed by $i = 1, \dots, n$. Although the underlying model is continuous time, we will consider the uniformized discrete-time process. The state is $\Phi_k \in Z_+^n$, where $[\Phi_k]_i$ denotes the number of class i jobs in the system after k transitions. All transitions–arrivals and service completions–are potentially controlled. Let $\mathcal{P}$ be the routing matrix, so that after service completion at class i, a job is routed to class j with probability p_{ij}. The transition probabilities are

$$P(\Phi_{k+1} = x + e_i \mid \Phi_k = x) = \lambda_i \widetilde{v}_i(k)/\Lambda \tag{1}$$
$$P(\Phi_{k+1} = x + e_j - e_i \mid \Phi_k = x) = \mu_i p_{ij} \widetilde{u}_i(k)/\Lambda, \tag{2}$$

where e_i is the ith basis vector and we use class 0 and the convention $e_0 = 0$ to represent departures. The exogenous arrival rate of class i customers is λ_i with control $\widetilde{v}_i(k)$. The service rate for class i customers is μ_i with control $\widetilde{u}_i(k)$. The tilda denotes the discrete model. Let $\widetilde{u}(k)$ and $\widetilde{v}(k)$ be vectors defined in the natural way. The appropriate policy will be a stationary, state-feedback control, so that Φ is a Markov chain. Let $\mathcal{P}$ be its transition probability matrix, defined by (1), (2), and $\mathcal{P}_{xx} = 1 - \sum_{y \neq x} \mathcal{P}_{xy}$. The uniformization constant $\Lambda = \sum_{i=1}^{n} (\lambda_i + \mu_i)$ makes $\mathcal{P}_{xx} \geq 0$. Class i is served by station $s(i)$. The standard constraints on the controls in state $\Phi_k = x$ are

$$\sum_{i:s(i)=j} \widetilde{u}_i(k) \leq 1 \quad \text{for } j = 1, \ldots, m \tag{3}$$
$$\widetilde{u}_i(k) = 0 \ \text{ if } \ x_i = 0, \quad \widetilde{u}_i(k) \geq 0, \quad 0 \leq \widetilde{v}_i(k) \leq 1.$$

Here $\widetilde{u}_i(k)$ is the fraction of time server $s(i)$ allocates to class i. A server's allocations cannot exceed one. Other linear constraints can be added to reflect uncontrolled arrivals, arrival routing, and uncontrolled servers. (Without additional constraints, the optimal policy will have no arrivals.) The general form of constraints we will consider is

$$\begin{aligned} D\widetilde{u}(k) &\leq (\geq)\ e \\ H\widetilde{v}(k) &\leq (\geq)\ e \\ \widetilde{u}_i(k) &= 0 \ \text{ if } \ x_i = 0 \\ \widetilde{u}(k) \geq 0, &\quad \widetilde{v}(k) \geq 0. \end{aligned} \tag{4}$$

In a slight abuse of notation, let u denote the state-feedback control $(\widetilde{u}(x), \widetilde{v}(x))$. There is a linear holding cost $c'x$ in state x. The objective is to minimize average cost

$$J(x, u) = \limsup_{T \to \infty} \frac{1}{T} E_x \sum_{k=0}^{T-1} c' \Phi_k, \tag{5}$$

where E_x denotes expectation given $\Phi_0 = x$. Assuming a stable policy exists, there will typically be a solution to the average cost optimality equation

$$\eta_*/\Lambda + h(x) = \min_u c'x/\Lambda + \mathcal{P}_u h(x), \tag{6}$$

where the minimum is taken over state-feedback policies satisfying (4). In (6) the dependence of $\mathcal{P}$ on the policy has been made explicit and $\mathcal{P}h$ is function composition, i.e., $\mathcal{P}h(x) = \sum_{y \in Z_+^n} \mathcal{P}_{xy} h(y)$. The solution gives the optimal average cost, $\eta_* = \min_u J(x, u)$ for all x, and the relative value function $h(x)$. An extreme point control will achieve the minimum in (6), so the action space can be considered finite.

A fluid model is obtained when all transitions are replaced by their mean rates and a continuous state is used; $x_i(t)$ is the length of the class i queue at time t, with $x(t) \in R_+^n$. The relevant stability criterion for most fluid models is that they drain from any initial state; i.e., for each initial state we can choose a time horizon T such that $x(t) = 0$ for all $t \geq T$ for some policy. Stable models that do not drain still move to a "best state" in finite time. The fluid control problem corresponding to (4) and (6), but without arrival control, is

$$
\begin{aligned}
\text{(FNET)} \qquad \min \int_0^T & c' x(t) dt \\
\dot{x}(t) &= Bu(t) + b \\
Du(t) &\leq (\geq) \;\; e \\
x(0) &= x \\
x(t) &\geq 0, \quad u(t) \geq 0,
\end{aligned}
$$

where $b = (\lambda_1, \ldots, \lambda_n)^T$, $B = (P^T - I)M$, and $M = \text{diag}(\mu_1, \ldots, \mu_n)$. The function $x(t)$ is continuous. The optimal $u(t)$ is piece-wise constant, making $x(t)$ piece-wise linear. Let $J^*(x)$ denote the optimal fluid cost from initial state x.

The feasible u in state $x > 0$ in Z_+^n are the same as the feasible $\widetilde{u}$. If $x_i(t) = 0$, however, (FNET) requires

$$(Bu(t) + b)_i \geq 0. \tag{7}$$

If (7) does not imply $u_i(t) = 0$, then the fluid constraints are weaker than the discrete constraints (4) and we say there is a *translation problem.* We will use only simple translations that enforce $u_i(t) = 0$. However, there are many examples of non-acyclic networks where these translations are unstable. The network of [Ryb92] shown in Figure 1 is a simple example.

Consider the data $\lambda_1 = \lambda_3 = 1$, $\mu_1 = \mu_3 = 6$, $\mu_2 = \mu_4 = 1.5$, and $c = 1$. The fluid policy is last-buffer-first-served (LBFS), i.e., classes 2 and 4 have priority, except that the server is split to avoid starvation whenever the exiting class at the other server is empty. Any translation that only makes adjustments on the boundary is inadequate for the discrete system; see [Mag00]. The difficulty is that the servers wait too

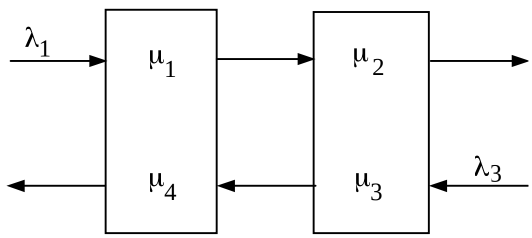

Figure 1. The Rybko-Stolyar network.

long to switch from the exiting to the entering class, causing undesirable idleness. The translation that waits the longest to switch, pure LBFS, is actually unstable.

Two variations of (FNET) will occur in the examples. First, arrival or input control can be added by replacing b with $bV(t)$, where $V(t) = \text{diag}(v_1(t), \dots, v_n(t))$ and placing appropriate constraints on $V(t)$. However, an equivalent formulation is to simply redefine b and D. Second, make-to-stock systems are modeled by removing the nonnegativity constraints on certain classes. Other extensions, such as controlled routing, can also be accommodated.

3. Greedy Solutions

For some well-known examples the greedy policy is discrete optimal and for others it provides an attractive approximate policy. After some general observations about the simplicity of greedy policies, we give an example in Section 3.1 where it is discrete optimal and in 3.2 where it is only fluid optimal. Section 4.1 gives an example where the greedy policy is fluid optimal in sufficiently heavy traffic. Interestingly, there seems to be a general connection between heavy traffic and effectiveness of the greedy policy; it is explored in [Mey01b].

Similar greedy solutions can be found using either the fluid or the discrete system. For the fluid, the greedy control in state x minimizes the time derivative of the cost rate. It is a solution to the LP

$$\text{(LPNET)} \quad \begin{aligned} &\min c' Bu \\ &\dot{x} = Bu + b \\ &Du \le e \\ &x_i \dot{x}_i \ge 0, \quad i = 1, \dots, n \\ &u \ge 0. \end{aligned}$$

Chen and Yao (1993) give a procedure to generate a trajectory $x(t)$ for the greedy policy: Solve (LPNET) with $x = x(0)$. Apply this u until some $x_i(t)$ hits 0, then resolve. If there is no translation problem, greedy policies for the fluid problem translate directly into greedy policies for the discrete problem:

$$\widetilde{u}(x) = u(x) \quad \text{for all } x \in Z_+^n. \tag{8}$$

To see this, replace the relative cost function in (6) with the cost rate $c'x$. Setting $v_i = 1$ to eliminate arrival control, the function to be minimized is

$$\begin{aligned} \mathcal{P}_u c'x &= c'x + \sum_{i=1}^{n} c_i \lambda_i / \Lambda + \sum_{i=1}^{n} \sum_{j=1}^{n} (c_j - c_i) \mu_i p_{ij} u_i / \Lambda \\ &= c'(x + \dot{x}/\Lambda). \end{aligned} \tag{9}$$

Minimizing (9) subject to (4) is equivalent to (LPNET). If there is a translation problem, then the greedy control for the discrete problem will generally differ on the boundary.

Because of the linear objective function, the fluid policy is greedy in the interior of the state space ($x > 0$) exactly when it is static priority. Clearly, a greedy policy can always be chosen to be static priority because the constraints in (LPNET) depend only on which $x_i = 0$. For $x > 0$, minimizing (9) is accomplished by serving at each station the class with minimal index

$$f_i = \sum_{j=1}^{n} (c_j - c_i) \mu_i p_{ij}, \tag{10}$$

where we use class 0 and the convention $c_0 = 0$ to represent departures. If all indices are positive, a station idles. The indices (10) are static and we say the greedy policy has static priorities on the interior of the state space. [ABR95] shows that the fluid policy in general has dynamic indices. For these indices to be static, it can be shown that they must agree with (10). (Here we use the fact that the state space is unbounded.) Thus, if the fluid policy is static priority on $x > 0$, then it is also greedy on $x > 0$.

3.1 Single Station with Feedback

When there is only one station, the discrete control problem is tractable. Still, it is interesting to note that the fluid and discrete policies are essentially identical. Although there is a translation problem, i.e., they differ

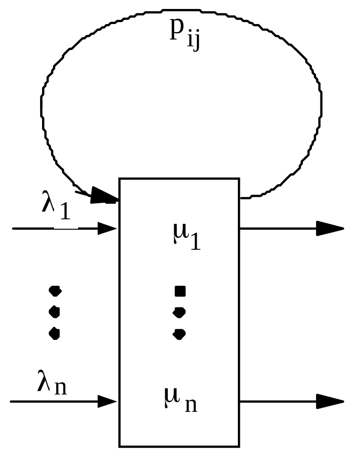

Figure 2. Klimov's problem.

on the boundary of the state space, the naive translation gives the correct discrete policy. For a single station without feedback, the familiar $c\mu$ rule is optimal for both. A single station with feedback is known as Klimov's problem (Figure 2). A static priority, greedy policy is optimal for both. [Che93] gives a simple algorithm for computing the priorities based on (LPNET) and uses duality to show that this greedy solution is also optimal for the fluid model. [Luo95] gives a dual algorithm to solve (FNET) directly, obtaining a static priority policy. He proves that the fluid priorities are consistent with the discrete priorities by showing that his dual algorithm also solves the discrete problem. Here we illustrate the simplicity of finding the priorities.

For $x > 0$, (LPNET) can be written

$$\begin{aligned} \min \ c'Bu \\ \sum_{i=1}^{n} u_i \leq 1 \\ u \geq 0, \end{aligned}$$

the solution of which is to serve the class with smallest

$$\left[c'B\right]_i = \left(\sum_{j=1}^{n} p_{ij}c_j - c_i\right)\mu_i.$$

If $x_i = 0$ for $i \in R$ and $x_i > 0$ for $i \in N$, and all classes in R have higher priorities than all classes in N, then it can be shown that the solution to (LPNET) has $\dot{x}_i = 0$ for $i \in R$, i.e., buffers do not refill once they are emptied. Thus, (LPNET) only needs to be solved for R containing the k highest priority classes, $k = 0, \ldots, n-1$. Using this fact, (LPNET) can be written

$$\begin{aligned} \min \ & c'Bu \\ [Bu + b]_R &= 0 \\ \sum_{i=1}^{n} u_i &\leq 1 \\ u &\geq 0. \end{aligned} \tag{11}$$

For $i \in N$, let $\overline{u}_i$ be the maximum feasible value of u_i (the rest of the server's time is allocated to R to keep $x_R = 0$). [Luo95] proves that $\overline{u}_i$ is well-defined. Then the solution to (11) allocates $\overline{u}_i$ to class $i \in N$ with minimal

$$\left(\sum_{j=1}^{n} p_{ij} c_j - c_i \right) \mu_i \overline{u}_i. \tag{12}$$

The priorities can be found by repeatedly applying (12), adding the next priority class to R each time. The references above give explicit formulas for (12).

3.2 Make-to-Stock Queues: Fluid and Diffusion Models

Now consider a single-station production system that operates in make-to-stock fashion. Class i items are produced from an unlimited supply of raw material and placed in a finished goods inventory, x_i. This inventory is depleted by demand at rate λ_i. Unmet demand is backordered and recorded as negative inventory. The fluid model is

$$\begin{aligned} \min \int_0^T & \sum_{i=1}^{n} \left(c_i^+ x_i^+(t) + c_i^- x_i^-(t) \right) dt \\ & \dot{x}_i(t) = \mu_i u_i(t) - \lambda_i \\ & \sum_{i=1}^{n} u_i(t) \leq 1 \\ & x_i(t) = x_i^+(t) - x_i^-(t) \\ & x(0) = x \\ & x^+(t) \geq 0, \quad x^-(t) \geq 0, \quad u(t) \geq 0. \end{aligned}$$

Decomposing x into x^+ (inventory) and x^- (backorders) allows holding and backorder cost to be written as a linear function of the state variables.

This example illustrates a fundamental limitation of fluid models: They do not use safety stock. The discrete policy, described in [deV00], consists of switching surfaces and a hedging point at which the server idles. When $x(t) \leq 0$ a $c\mu$ rule is used. In contrast, the fluid policy from initial $x \leq 0$ is the same as for the single-station queue, namely, the $c\mu$ rule. It operates the system without safety stock, ignoring the make-to-stock capability and the data c^+. Generally, not holding finished goods inventory in virtual buffers is suboptimal. In contrast, the fluid policy's lack of safety stock in real buffers can be unstable, as discussed at the beginning of Section 3.

Diffusion models, with their recognition of randomness, do considerably better than fluid models for this problem. They set reasonable hedging points and recover the $c\mu$ rule; see [Wein92] and [Vea96]. In the single-class case, the diffusion model gives a hedging point (and, hence, a policy) which is asymptotically optimal as $\rho \to 1$. The fluid policy is only optimal when ρ is sufficiently small and holding costs are large. Similar conditions for optimality of the fluid policy hold for two classes; see [Vea01b].

4. Fluid Limits and Asymptotic Switching Curves

The solution to many stochastic network control problems has a threshold form–below some switching surface (switching curve if $n = 2$), one extreme point is used and above it another is used. Based on scaling arguments and numerical results, it has been conjectured that the fluid policy gives the correct asymptotic orientation of these surfaces. In other words, the fluid tells us what to do when queues are large. This section presents a general theory, culminating in a corollary that establishes this connection between the fluid and discrete switching surfaces. Several examples are given in the following subsections. To simplify the exposition, consider the sequencing problem (4) without arrival control and when the extreme points have $\widetilde{u}_i = 0$ or 1. In this section we refer to $\Phi(t)$ and $\widetilde{u}(t)$ in continuous time. They can be constructed from the discrete-time versions as piece-wise constant, right-continuous left-limit functions or can be thought of as the continuous-time MDP before uniformization.

The most common justification for using fluid models is the existence of fluid limits when time and queue length are scaled by the same factor. The following result, called a functional strong law of large numbers, is proven in more general form in [Dai95]. Consider a sequence of discrete systems indexed by N such that $\Phi^N(0) = \lfloor Nx(0) \rfloor$, $x(0) \in R^n_+$. All use the same policy. Introduce

$$\widetilde{T}_i(t) = \int_0^t \widetilde{u}_i(s)ds$$

to represent the cumulative time that server i has devoted to class i customers by time t.

Theorem 1 (Dai) *For almost all sample paths there is a subsequence* $\{N_r\}$ *such that, as* $N_r \to \infty$,

$$\begin{aligned} & \frac{1}{N_r}(\Phi^{N_r}(N_r t), \widetilde{T}^{N_r}(N_r t)) \to (x(t), T(t)), \\ \textit{where} \quad & x(t) = x(0) + BT(t) + bt \\ & T(t) = \int_0^t u(s)ds \qquad (13) \\ & Du(t) \le e \\ & x(t) \ge 0, \quad u(t) \ge 0. \end{aligned}$$

The limit exists uniformly on compact sets. The processes $(x(t), T(t))$ are called the *fluid limit.* Note that each initial state and policy has at least one fluid limit and that they satisfy the same constraints as the fluid model. Thus, the fluid model includes all fluid limits and possibly more trajectories.

Theorem 1 does not guarantee a *unique* fluid limit for a given initial state and policy. Different sample paths might have different fluid limits; even for one sample path, different subsequences might have different fluid limits. Because the theory of unique fluid limits is incomplete, we will simply *assume* that the policies we consider have unique fluid limits from all initial states, except possibly a set of lower dimension. We conjecture that this is not a restrictive assumption. The fact that these fluid limits satisfy the fluid model constraints (13) and that they are all generated by the same discrete policy severely limits how nonuniqueness can arise. A trivial nonuniqueness occurs if the discrete policy is not unique and ties are not broken in a consistent fashion. This difficulty can be avoided by appropriate selection from among the optimal policies. The only examples of reasonable state-feedback policies giving non-unique fluid limits that we are aware of are policies such as "shortest queue first" that make the fluid trajectory map discontinuous on switching surfaces. We also note that [Mag00] makes a similar assumption, namely, that the trajectory map is continuous. However, some skepticism about our assumption may be warranted in light of the nonunique fluid limits and other unexpected behavior encountered in unstable systems.

Theorem 1 also does not give a method for computing fluid limits. Again the theory is incomplete–in fact, constructing the fluid limit is known to be complicated in some cases–but we will proceed in a pragmatic manner. At all differentiable points of $T(\cdot)$, (13) defines a fluid control $u(t)$ in state $x(t)$ and $\dot{x}(t)$ is the velocity in that state. Under mild conditions (see the semigroup property of [Mag00]), the collection of fluid limits $x(\cdot)$ from all initial states, called the trajectory map, defines a fluid policy $u(x)$ (except possibly on a set of lower dimension). Call this policy the *fluid limit policy* of $\widetilde{u}$.

In order to establish a connection between the discrete and fluid policies, we make the following additional assumption about the discrete policy, which we call a *scalable policy.* Define the *scaled policy* associated with $\widetilde{u}$ as $\overline{u}(x) = \lim_{N\to\infty} \widetilde{u}(\lfloor Nx_1 \rfloor, \ldots, \lfloor Nx_n \rfloor)$ when the limit exists. Assume that $\overline{u}(x)$ exists except on a set of lower dimension and consists only of extreme points of (4). Let $\overline{u}^1, \ldots, \overline{u}^K$ be the values taken on by $\overline{u}(\cdot)$ and $S_k = \{x : \overline{u}(x) = \overline{u}^k\}$, $k = 1, \ldots, K$. Call the S_k control switching sets (CSSs). By construction, the CSSs are scale-invariant: If $x \in S_k$ then $\alpha x \in S_k$ for $\alpha > 0$.

To illustrate the general result to follow, consider a small example. The scaled policy associated with the following scalable policy for two queues in series (Section 4.2) with $\mu_1 < \mu_2$ is shown in Figure 3. Always serve at the downstream queue: $\widetilde{u}_2(x) = 1$ if $x_2 > 0$ and 0 otherwise. Use a switching curve at the upstream queue: $\widetilde{u}_1(x) = 1$ if $x_2 < s(x_1)$ and 0 otherwise. Let $\gamma = \lim_{x\to\infty} s(x)/x$. We will show that the fluid limit policy has the switching curve $x_2 = \gamma x_1$ and that if $\widetilde{u}$ is discrete optimal then its fluid limit policy is fluid optimal. If $\gamma > 0$, the CSSs are $S_0 = \{0\}$, $S_1 = \{x : x_1 > 0,\ x_2 = 0\}$, $S_2 = \{x : x_1 = 0,\ x_2 > 0\}$, $S_3 = \{x : 0 < x_2 < \gamma x_1\}$ and $S_4 = \{x : x_2 > \gamma x_1\}$. Two adjustments to $\overline{u}$ are needed to obtain the fluid limit policy. The limit $\overline{u}(1, \gamma)$ may not exist. However, the switching curve is *deflective*, i.e., trajectories do not remain on the curve, and we can arbitrarily set $u(1, \gamma) = \overline{u}^3 = (1, 1)$. In contrast, S_1 is *attractive* (trajectories remain in S_1) and the scaled control $\overline{u}^1 = (1, 0)$ needs to be adjusted by "time averaging" over the time the discrete system spends in S_1 and S_3 when $x_1 \gg 0$. On S_1, the fluid limit control is the convex combination of $\overline{u}^1$ and $\overline{u}^3$ that remains in S_1: $u(1, 0) = \alpha\overline{u}^1 + (1 - \alpha)\overline{u}^3$ and $\dot{x}_2(t) = 0$, leading to $u(1, 0) = (1, \mu_1/\mu_2)$. For $x \in S_k$, $k \neq 1$, set $u(x) = \overline{u}^k$. Theorem 2 below shows that u is the fluid limit policy associated with $\widetilde{u}$ in S_3 and S_4. The fluid

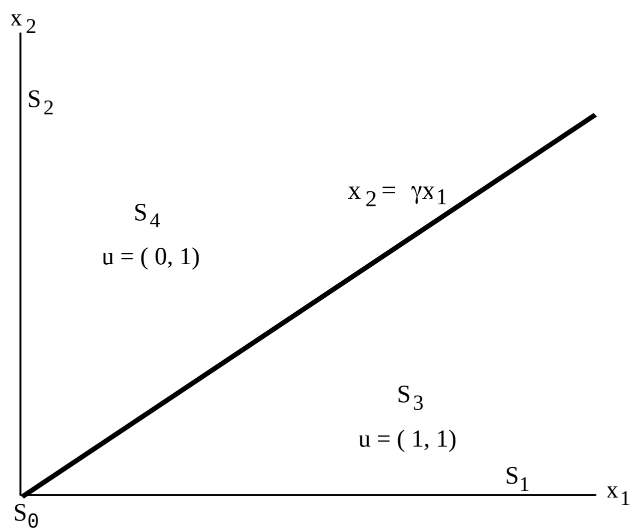

Figure 3. Control switching sets for two queues in series.

limit is not defined by (13) on the transient S_2 and the switching curve (nondifferentiable points), and we have defined u on S_1 in the only way to construct trajectories that are smooth there and are consistent with the nonidling nature of $\widetilde{u}$. As u is the only fluid policy that satisfies (13) and is consistent with $\widetilde{u}$, it must contain the (unique) fluid limit policy of $\widetilde{u}$.

The following theorem establishes that all switching surfaces in the interior of the state space are the same for the fluid limit policy and the scaled policy.

Theorem 2 *At z in the interior of a CSS S_k of full dimension, the fluid limit policy matches the scaled policy, $u(z) = \overline{u}^k$.*

Proof. We proceed as in, e.g., [Dai95, proof of Theorem 7.1]. Let $B_i = \{\alpha x \in R^n_+ : 0 \leq \alpha \leq 1 \text{ and } \widetilde{u}_i(x) > 0\}$ and define

$$f_i(x) = \min\{\min_{y \in B_i} |x - y| \, , 1\},$$

where $|x| \equiv \sum_i |x_i|$. For all x with $\widetilde{u}_i(x) > 0$, $f_i(x) = f_i(x/N) = 0$ so that

$$\int_0^t f_i(\frac{\Phi^N(s)}{N}) d\widetilde{T}_i^N(s) = 0. \tag{14}$$

Because f_i is bounded and continuous, Lemma 4.4 of [Dai95] applies and (14) passes to the fluid limit:

$$\int_0^t f_i(\frac{\Phi^N(s)}{N})d\widetilde{T}_i^N(s) \to \int_0^t f_i(x(s))dT_i(s). \tag{15}$$

Here $x(\cdot)$ is the fluid limit and the convergence is uniformly on compact sets. Now, (14) and (15) imply that $f_i(x(t)) = 0$ or $u_i(t) = 0$ for almost every t along a trajectory. We can select a fluid limit policy for which this is true at all t, since doing so will not change $T(\cdot)$. Let z be in the interior of a CSS S_k of full dimension with $\overline{u}_i^k = 0$ so that $f_i(x) > 0$. Consider the fluid trajectory with $x(0) = z$. It remains in S_k for a nonzero time; hence, $u_i(z) = 0 = \overline{u}_i^k$.

A similar argument applies to CSSs where $\overline{u}_i^k = 1$ By assumption, $\overline{u}_i^k = 0$ or 1, so these are the only two cases. □

The key theoretical result is that optimality is preserved in the fluid limit. It is proven in [Mey01a, Theorem 6] using the "fluid scale asymptotic optimality" result of [Mey97, Theorem 7.2]. Here is the part of his theorem that we will use.

Theorem 3 (Meyn) *If a stable policy exists for the discrete system, then there exists a discrete optimal policy whose fluid limit is optimal for the fluid model, in the sense that, for any x, almost every sample path from x has a fluid limit that achieves the optimal fluid cost $J^*(x)$.*

The following corollary establishes a relationship between the discrete policy (through the scaled policy $\overline{u}$) and the fluid policy.

Corollary 1 *If a stable policy exists for the discrete system, then there exists a discrete optimal policy $\widetilde{u}$ whose scaled policy $\overline{u}$ matches some fluid optimal policy in the interior of CSSs of full dimension.*

Proof. Consider a discrete optimal policy that satisfies Theorem 3. The assumption that the fluid limit is unique implies that its fluid limit policy u matches some fluid optimal policy (except on a set of lower dimension). In particular, $u(x)$ is a fluid optimal control for x in the interior of CSSs of full dimension because, for any fluid limit with $x(t) = x$, $T(\cdot)$ is differentiable at t. Theorem 2 guarantees that $\overline{u}$ matches u in these interiors. □

The fluid policy is known to consist of CSSs bounded by linear switching surfaces through the origin. By the corollary, some discrete policy, when scaled, has the same switching surfaces as some fluid policy. If we avoid the particular parameter values that lead to ties, these switching

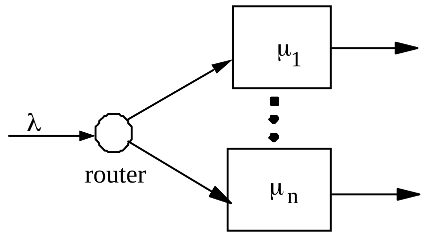

Figure 4. Arrival routing.

surfaces will be unique. Also, it appears that Meyn's proof applies to *any* discrete policy. Thus, we can say that any (scaled) discrete and fluid policy have the same switching surfaces. More precisely, the asymptotic slopes (unit normal vectors for $n > 2$ dimensions) of the discrete switching surfaces as $|x| \to \infty$ are the same as the slopes of the corresponding fluid switching surfaces.

4.1 Arrival Routing

The arrival routing problem analyzed in [Haj84] (Figure 4) provides a good illustration of how switching curves are related in the discrete and fluid models. The fluid solution is presented in more detail in [Vea01a]. Customers arrive at rate λ and are routed to one of n servers to minimize holding cost. Server i has service rate μ_i and holding cost rate $c_i > 0$. Its fluid model can be written in the form of (FNET):

$$\begin{aligned} J(x) = \min \int_0^T & c' x(t) dt \\ & \dot{x}_i(t) = \lambda v_i(t) - \mu_i u_i(t) \\ & \sum_{i=1}^n v_i(t) = 1 \\ & x(0) = x \\ x(t) \geq 0, \quad & v(t) \geq 0, \quad 0 \leq u(t) \leq 1. \end{aligned} \tag{16}$$

If $v_i(t) = 1$, arrivals are routed to class i at t. The control u_i has been added to enforce idling at empty queues. It is optimal to set $u_i(t) = 1$ if $x_i(t) > 0$ and $u_i(t) = \lambda v_i(t)/\mu_i$ otherwise. The greedy policy will avoid starving servers whenever possible and then route to the server with smallest holding cost. In light of this, we order the classes $c_1 \leq \ldots \leq c_n$.

For $n = 2$ servers, we consider two cases. If $\lambda \geq \mu_2$, the fluid policy is greedy. It routes to 1 whenever $x_2 > 0$. When $x_2 = 0$, it avoids starving server 2 by setting

$$v_1 = 1 - \mu_2/\lambda \quad \text{and } v_2 = \mu_2/\lambda. \tag{17}$$

The second case, $\lambda < \mu_2$, has a switching curve: Route to server 2 if

$$x_2 \leq \gamma x_1 = \frac{c_1}{c_2}\left(\frac{\mu_2 - \lambda}{\mu_1}\right) x_1, \tag{18}$$

otherwise route to server 1. Note that this policy makes a tradeoff between the higher short term cost of routing to server 2 and postponing the starvation of server 2. The optimality of (17) and (18) is shown in [Vea01a].

The fluid policy for n servers makes similar tradeoffs. Let M be the integer satisfying

$$\sum_{i=M}^{n} \mu_i > \lambda \geq \sum_{i=M+1}^{n} \mu_i.$$

The fluid policy avoids starving whenever possible and can be chosen so that servers $M+1, \ldots, n$ never starve. Let $E(x) = \{i : x_i = 0\}$. Avoiding starving requires routing $\lambda_0(t) = \sum_{i \in E(x(t))} \mu_i$ to empty buffers. The remaining decision is how to route $\lambda_1(t) = \lambda - \lambda_0(t)$. We will describe policies by how they route $\lambda_1(t)$; essentially, this is the routing to nonempty buffers. We consider only policies that do not split $\lambda_1(t)$ between servers, since splitting is not needed to achieve optimality. This part of the problem ends at the starvation time $\tau_S = \min\{t : \lambda_1(t) < 0\}$. No servers are starved before τ_S. Let $x(t)$ be an optimal trajectory and $t_i = \liminf \{t > 0 : x_i(t) = 0\}$. Note that if $x_i(0) = 0$ but the initial routing makes $\dot{x}_i(0) > 0$, then t_i is the *next* time buffer i empties.

The fluid policy has the following properties, proven in [Vea01a].

1 Never route to servers $M+1, \ldots, n$.

2 Never switch to an empty buffer at $t > 0$.

3 Only switch to higher-cost servers (from i to j, $j > i$).

4 If $t_i < \tau_S$, then never route to servers i or higher.

It is optimal to route to the server with the minimal index $k_i(x) = D_{e_i} J(x)$. This directional derivative will exist for all x and can be thought of as the incremental cost per unit of fluid initially in buffer i. Furthermore, [ABR95] shows that $k_i(x)$ is continuous along a trajectory. Thus, if the optimal routing switches from server i to server j at x, then

$$k_i(x) = k_j(x). \tag{19}$$

Consider a small amount of additional fluid initially in buffer i. Relative to $x(t)$, the additional fluid stays in buffer i until t_i. If $t_i \geq \tau_S$ then the additional fluid leaves the system at t_i. If $t_i < \tau_S$ and routing is to l at t_i, then the additional fluid is in buffer l from t_i until t_l. This shifting continues until τ_S, giving a sequence of cost terms

$$k_i(x) = c_i t_i + c_l(t_l - t_i) + \dots \quad . \tag{20}$$

A consequence of the properties above is that trajectories are acyclic, in the sense that once a buffer empties it remains empty. If a trajectory switches from server i to j at x, then, by Property 4, $t_i \geq \tau_S$. Hence, (20) contains only the first term and (19) simplifies to

$$c_i t_i = c_j t_j. \tag{21}$$

The optimal policy never routes to i again, so $t_i = x_i/\mu_i$. Because t_j depends on the policy, we do not have a simple algorithm for computing an optimal trajectory. Let τ be the time of the next switch after switching to j at time 0. If there are no more switches, set $\tau = t_j$. Again using Property 3,

$$t_j = \frac{x_j + \int_0^\tau \lambda_1(s)ds}{\mu_j}. \tag{22}$$

Note that τ and $\lambda_1(\cdot)$ depend on x. These conditions are consistent with those for two servers, as can be seen by setting $i = 1$, $j = 2$, $\tau = t_2$ and $\lambda_1(s) = \lambda$. For $i < j$, (21) has a positive solution if and only if $j \leq M$. The parameter M specifies which servers have switching surfaces in the interior of the state space.

In principle, the following approach can be used to compute the switching surfaces. However, the number of cases to be computed is exponential in n. First, find the i, M switching surfaces. Setting $j = M$, there are no more switches, so $\tau = t_j$. Knowing the future control, find $\lambda_1(s)$ in terms of x and solve (22) and (21). Next, find the $i, M-1$ switching surfaces (set $j = M - 1$). To find τ, consider the trajectory that continues routing to $M - 1$. If it intersects the $M - 1, M$ switching surface, the intersection is at τ. If not, set $\tau = t_j$. Again, the future routing is known and we can find $\lambda_1(s)$. Continue in this fashion, decreasing j to find all of the switching surfaces.

Figure 5 compares the fluid and discrete policies for an example with two servers and a switching curve. The discrete policy was found using dynamic programming value iteration on a truncated state space. The policies were compared by computing their average cost in the discrete model. The fluid policy was translated to the discrete model by applying

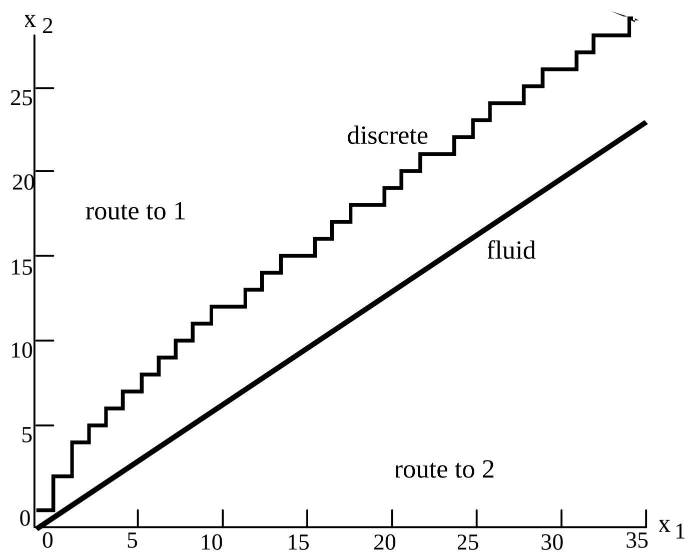

Figure 5. Fluid and discrete switching curves for arrival routing, $\lambda = 1$, $\mu = (0.5, 1.5)$, $c = (1, 1.5)$.

(18) directly except at the origin, where routing is to the server with minimum index c_i/μ_i. This index gives the optimal control in light traffic and matches the discrete policy. Routing is to server 1 above the curve and server 2 below it. Average cost is 1.18 for the discrete policy and 1.36 for the fluid policy (15% suboptimal). This and other examples suggest that the fluid policy is similar to the discrete policy when there are switching surfaces on the interior of the state space, but far from optimal when it switches on the boundary.

4.2 Series Systems

Consider a system of n stations in series, each serving a single class, with arrivals at rate λ to the first class. The fluid model is

$$\begin{aligned}
&\min \int_0^T c' x(t)dt \\
&\quad \dot{x}_1(t) = \lambda - \mu_1 u_1(t) \\
&\quad \dot{x}_i(t) = \mu_{i-1} u_{i-1}(t) - \mu_i u_i(t), \quad i = 2, \ldots, n \\
&\quad x(0) = x \\
&\quad x(t) \geq 0, \quad 0 \leq u(t) \leq 1.
\end{aligned}$$

If $c_i \geq c_{i+1}$ (or $i = n$), it is optimal to serve at the maximum possible rate at station i. We restrict our attention to the more interesting case where value is added, $c_1 \leq \ldots \leq c_n$. If $\mu_1 \geq \ldots \geq \mu_n$, starvation can be avoided until $x(t) = 0$ and the fluid policy simply avoids starving station n. More specifically, $u_i(t) = \mu_n/\mu_i$ if $x_j(t) = 0$, $j = i+1, \ldots, n$ and zero otherwise.

For the two-station system, [ABR95] show that the remaining case, $c_1 < c_2$ and $\mu_1 < \mu_2$, has a switching curve: idle station 1 if

$$x_2 \geq \frac{c_1}{c_2}\left(\frac{\mu_2 - \mu_1}{\mu_1 - \lambda}\right) x_1, \tag{23}$$

otherwise $u_1 = 1$. For n stations, [Luo95] establishes that there exist optimal fluid solutions with the following properties.

1 Once a station starts working, it works at the maximum rate such that all empty queues will remain empty.

2 $u(t)$ is piece-wise constant with at most $2n + 2$ pieces. It changes only when a station starts working or a queue empties.

The small number of break points suggests that there is an efficient algorithm to find the fluid policy for this model. See [Set99] for progress on this problem.

The discrete problem has only been solved using iterative dynamic programming methods on small systems. Most of the work on this problem assumes a finite buffer control, such as a kanban mechanism, then evaluates performance and optimizes within this class of controls. Several aspects of the fluid solution are reasonable for the discrete problem. First, the discrete policy is nonidling exactly when the fluid policy is, namely, when $c_1 \geq \ldots \geq c_n$. Also, replacing "empty" in property 1) with "a small buffer" we obtain, roughly, a kanban blocking mechanism upstream of a bottleneck.

As in Section 3.2, the series system can be extended to a make-to-stock setting with backordering as shown in Figure 6. Backorders, with cost c_{n+1}^-, are only allowed in class $n + 1$, representing finished goods. The state x_1 can be eliminated, since μ_1 functions as arrival control. The fluid model dynamics are unchanged for $i = 2, \ldots, n$ and

$$\dot{x}_{n+1}(t) = \mu_n u_n(t) - \lambda.$$

The fluid policy again neglects the finished goods hedging point. However, it does produce ahead of demand at the upstream stations while

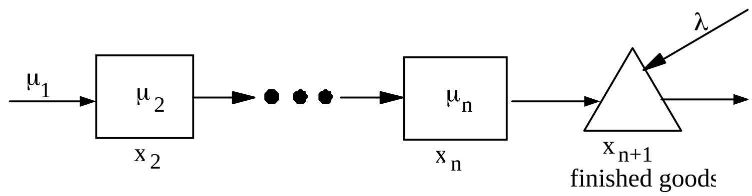

Figure 6. Series make-to-stock system.

there are backorders. For two stations, station 2 works if and only if $x_3 \leq 0$. Station 1 has a switching curve in the case $\mu_1 < \mu_2$: Idle station 1 if

$$x_3 \geq -\left[\frac{(c_2/c_3^-)(\mu_1 - \lambda) + \mu_2 - \lambda}{\mu_2 - \mu_1}\right] x_2, \tag{24}$$

otherwise $u_1 = 1$. Note that to have $u_1 = 1$ there must be backorders ($x_3 < 0$). It is interesting to compare the fluid policy with base stock and kanban policies. Stability requires the slope in (24) to be strictly less than -1. Base stock policies have a slope of -1; hence, they cannot be optimal or even asymptotically optimal under fluid scaling. Kanban, or any finite buffer policies, have an asymptotic slope of $-\infty$, which can only be optimal when $\mu_1 \geq \mu_2$. Similarly, (23) shows that kanban type policies can only be optimal for a series queue when $\mu_1 \geq \mu_2$. The discrete policy is computed numerically in [Vea94]. For the data $\lambda = 1$, $\mu_1 = 1.2$, $\mu_2 = 2$, $c_2 = 1$, and $c_3^- = 4$, (24) gives a slope (dx_3/dx_2) of -1.31 for the station 1 switching curve, while their policy (case 3) has a slope of about -1.05. The discrepancy is small enough that it could be due to state space truncation. The zero slope of the station 2 switching curve matches the computed discrete policy even more closely.

Returning to make-to-stock systems with n stations, [Per94] and [Luo95] give the following properties of fluid solutions.

1 Once a station starts working, it works at the maximum rate such that all empty queues will remain empty until backorders are removed.

2 Once backorders are removed, there will be no future backorders.

3 Once backorders are removed and a station starts working, it works at the maximum rate such that all empty queues will remain empty.

4 $u(t)$ is piece-wise constant with at most $4n + 4$ pieces. It changes only when a station starts working, a queue empties, or backorders are removed.

4.3 Input Control

Next we consider a two-station, six-class network with controllable inputs studied in [Wein90]. We are particularly interested in comparing his diffusion solution with the fluid solution. This example illustrates that the fluid model can give useful information that is in some ways better than the diffusion model. The timing of arrivals is controlled but not the mix. In Wein's formulation, the time of each arrival is specified by the control. An approximate MDP formulation is to use a scalar arrival control $0 \le \tilde{v}(k) \le 1$ and make λ_i large in (1). Average cost is minimized subject to the average throughput constraint

$$\lim \sup_{T \to \infty} \frac{1}{T} E_x \sum_{k=0}^{T-1} \sum_{i=1}^{n} \lambda_i \tilde{v}(k) \ge \overline{\lambda}. \tag{25}$$

The dynamic programming formulation (6) does not apply to this problem because of (25).

It would not be appropriate to use a time-averaged throughput constraint in a fluid model, since the fluid model is transient. Instead, we apply the more stringent constraint that the output rate must be at least $\overline{\lambda}$ at every t. Then the fluid input control problem corresponding to (FNET) is

$$\begin{aligned}
\text{(FIC)} \qquad \min \int_0^T c' x(t) dt& \\
\dot{x}(t) &= Bu(t) + bv(t) \\
Du(t) &\le e \\
a'u(t) &\ge \overline{\lambda} \\
x(0) &= x \\
x(t) &\ge 0, \;\; u(t) \ge 0, \;\; 0 \le v(t) \le 1,
\end{aligned}$$

where $a_i = \mu_i p_{i0}$ so that $a'u(t)$ is the flow out of the network. [Ric95] analyzes (FIC) and proves that, given the optimal sequencing control, the optimal input control is the minimal $v(t)$ that maintains $x(t^+) \ge 0$.

The network is shown in Figure 7. The data are $\mu = (1/4,\ 1,\ 1/8,\ 1/6,\ 1/2,\ 1/7)$, $c = (1, 1, 1, 1, 1, 1)$, $\overline{\lambda} = 0.127$, and $\lambda_1 = \lambda_3$ (input is equally divided between classes 1 and 3). The utilization required to meet (25) is $\rho_1 = \rho_2 = 0.9$. To describe the input control based on the diffusion model, let M_{ji} equal the expected time that station j must devote to a class i customer before it exits the network. Define the two-dimensional workload process $w(t) = Mx(t)$ and interpret $w_j(t)$ as the expected amount of work for station j represented by customers in the

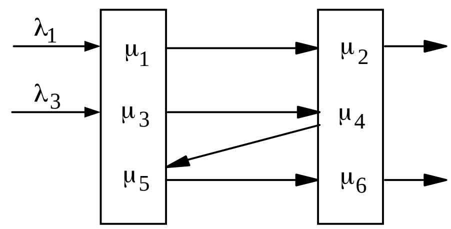

Figure 7. Wein's input control example.

network at time t. The diffusion control only accepts input if

$$w_1 < 19 \text{ and } w_2 - \frac{1}{4}w_1 \le 0 \text{ or} \tag{26}$$

$$w_2 < 62 \text{ and } w_1 - \frac{2}{13}w_2 \le 0, \tag{27}$$

where

$$w_1 = 4x_1 + 10x_3 + 2x_4 + 2x_5$$
$$w_2 = x_1 + x_2 + 13x_3 + 13x_4 + 7x_5 + 7x_6.$$

We have not found the complete fluid policy. However, fluid trajectories can be used to compare the fluid and diffusion input controls in the workload space (Figure 8). The constraints $w_1 < 19$ and $w_2 < 62$ are not shown. They are based on variance and cannot be captured by the fluid model. When only $x_i > 0$, the workload lies on a line through the origin with slope M_{2i}/M_{1i}. The diffusion control (26) and (27) accepts input when on or above the line for class 4, or when on the line for class 1. In contrast, the computed trajectories and analysis suggest that the fluid control accepts input if and only if $x_1 = x_2 = x_5 = x_6 = 0$. Thus, input *may* occur between the lines for class 3 and 4. For example, the optimal fluid trajectory from $x = (0, 0, 1, 1, 0, 0)$ accepts input when $x_3 > 0$, disagreeing with the diffusion control, while the trajectory from $x = (1, 1, 1, 1, 1, 1)$ does not.

It is peculiar that the diffusion control accepts input *above* the line for class 4, since there is no need to accept input when x_2 or x_6 is large. The difficulty arises from translating the diffusion policy to the discrete model. Under heavy traffic scaling, the workload can be instantaneously shifted between classes. Let $\widehat{w}$ denote the diffusion scaled version of w. Input control allows $\widehat{w}_1$ and $\widehat{w}_2$ to be instantaneously *simultaneously* reduced, but not rebalanced. The *workload imbalance,* $\widehat{w}_1(t) - \widehat{w}_2(t)$, is

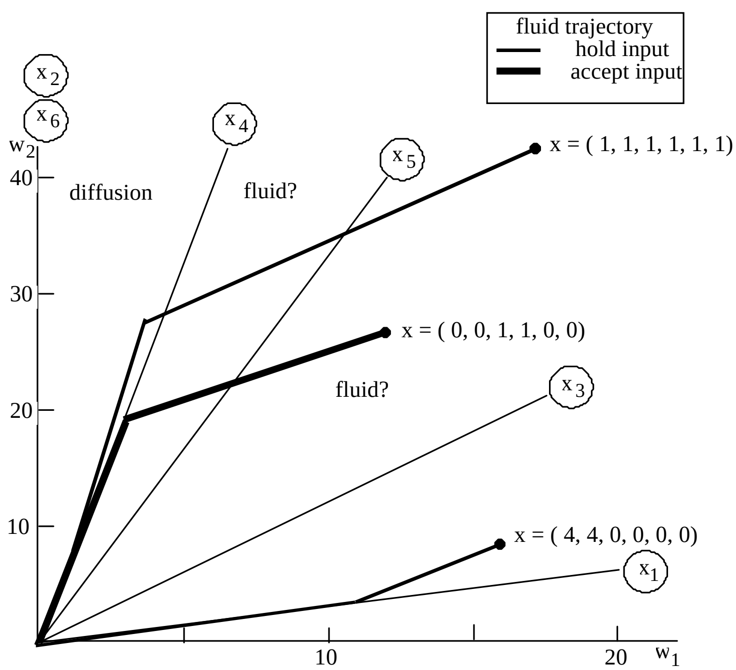

Figure 8. Regions where input is accepted for Wein's example.

the only independent quantity that cannot be immediately eliminated. All workload is held in class 1 (by giving it the lowest priority) when station 1 is behind and in class 4 when station 2 is behind. Hence, $x_2 = x_6 = 0$ is maintained and input is accepted *on* the line for class 4. Wein interprets this as *on or above* the line for the discrete model. The example suggests that it should be interpreted as *on* the line, with equality replacing less than or equal to in (27) (or within some ϵ of the line to allow safety stock).

Sequencing priorities are compared in Table 1. The diffusion priorities are one of many that are consistent with the diffusion solution; they reverse when station 1 has more work in the network. The fluid sequencing at station 2 begins serving each class in the same order as the diffusion, but splits the server between classes. In the second time interval, classes 6 and 2 are served. In the last time interval, class 4 is the only non-zero class.

Table 1. Priorities (highest to lowest) for Wein's input control example

	Diffusion(DRC)*	Greedy	Fluid from $x = (1, 1, 1, 1, 1, 1)$
Station 1	1, 3, 5	1, 5, 3	1, 5, 3
Station 2	2, 6, 4	2, 6, 4	2, 6/2, 6/4/2, 4

* When there is more work in the network for station 2

To summarize, the policies differ significantly in their input control and sequencing. The diffusion accepts input when the ratio w_1/w_2 of workloads for the two stations is far from one. The fluid accepts input only when needed for feasibility; roughly, when classes that have small M_{ji} (the "downstream" queues) are empty. We find the fluid input control, with its recognition of the importance of small or empty queues, to contain more potentially useful information. Much of the difference between the sequencing policies appears to be due to the different throughput constraint in the fluid model. The constraint *at each time* in (FIC) causes the output to be spread over time so that input can be avoided for as long as possible. The server splitting shown in Table 1 accomplishes this. In light of the formulation difference, it would be even more surprising if, as suggested by our qualitative analysis, the fluid policy performs better than the diffusion in the discrete model.

5. Conclusion

Numerous examples have demonstrated that the fluid model can contain valuable information about the optimal control of a stochastic network, including sequencing, scheduling, arrival routing, and work release decisions. The first connection between the fluid policy and the discrete policy (the optimal policy for the stochastic network) is their *priorities*. For problems that admit greedy solutions, the discrete and fluid policy use the same priorities when all buffers are nonempty. This correspondence suggests that the fluid priorities may be useful for other, more difficult problems.

When policies exhibit switching surfaces, a second, more powerful connection holds. The asymptotic slopes (normal vectors in higher dimensions) of the switching surfaces can be obtained from the fluid policy. We have provided a general theory for this equivalence, albeit with some difficult to check assumptions. Our numerical examples, combined with other numerical studies, suggest that the equivalence is robust.

We have also demonstrated that the fluid policy can often be found analytically; if not, the optimal trajectory from an initial state can be efficiently computed using the algorithm of [Luo98]. In fact, fluid solutions can be obtained in many more cases than diffusion solutions. Establishing a connection between the fluid and discrete models through limit theorems is also more straightforward than establishing a heavy traffic limit theorem. In light of this simplicity, it is appealing to use the fluid approximation whenever it gives comparable or better information than the diffusion approximation. The make-to-stock example illustrates that for systems where safety stock is important, the diffusion policy is better–though it may be possible to synthesize a good policy from the fluid policy and some other stochastic method. The input control example suggests that the fluid policy can be better than the diffusion, particularly when there are difficulties translating the diffusion policy back to the discrete model.

When considering the relationship between the discrete and fluid policy, it is important to distinguish between three cases.

1 *Scalable interior switching.* If the fluid policy switches in the interior of the state space, then the discrete policy does also with the same asymptotic slopes.

2 *Nonscalable interior switching.* If the discrete policy switches in the interior of the state space, but scales to a boundary switching policy (because the asymptotic slopes are zero), then the fluid policy switches on the boundary.

3 *No interior switching.* If the discrete policy does not switch in the interior of the state space (it either switches on the boundary or has static priorities), then neither does the fluid policy.

In cases 1) and 3), the fluid policy is in some sense close to the discrete policy, while in case 2) the fluid policy fails to capture a primary feature of the discrete policy. It would be of interest to classify problems as belonging to 2) or 3), since solving the fluid model does not distinguish between these cases. It would also be of interest to know, as has been conjectured, whether whenever the fluid policy is static priority the discrete policy is also static priority.

Finally, fluid policies need to be adjusted near the boundaries, or *translated* back to the original problem. A major question for future research is whether general methods can be developed to adjust the fluid policy to give a good policy for the original problem (stable for all problems and robustly near optimal). However, our numerical tests on the arrival routing problem demonstrate that, even without any adjustments, the fluid switching curve policy can perform well. For some problems, then, the fluid switching surfaces are useful without any adjustments.

6. Acknowledgments

I would like to thank Dimitris Bertsimas and Costis Maglaras for their many helpful suggestions and Jay Sethuraman and Michael Yee for assisting me in performing the numerical work. I would also like to recognize the work of Michael Ricard and Andy Luo in their theses, which I have drawn on heavily.

References

[ABR95] Avram F., Bertsimas D., Ricard, M. "Fluid models of sequencing problems in open queueing networks: An optimal control approach." In *Stochastic Networks,* F. Kelly and R. Williams, eds., vol. 71 of the Proceedings of the IMA, pp. 199-234. New York: Springer-Verlag, 1995.

[Bau00] Bäuerle, N. Asymptotic optimality of tracking policies in stochastic networks. *Ann. Appl. Prob.* 2000; 10:1065-1083.

[Che93] Chen H. and Yao D.D. Dynamic scheduling of multiclass fluid networks. *Oper. Res.* 1993; 41:1104-1115.

[Che94] Chen H., Mandelbaum A. "Hierarchical modeling of stochastic networks; Part I: Fluid models." In *Stochastic modeling and analysis of manufacturing systems,* D. D. Yao, ed. New York: Springer-Verlag, pp. 47-105, 1994.

[Dai95] Dai, J.G. On the positive Harris recurrence for multiclass queueing networks: A unified approach via fluid limit models. *Ann. Appl. Prob.* 1995; 5:49-77.

[deV00] de Véricourt F., Karaesman F., Dallery Y. Dynamic scheduling in a make-to-stock system: A partial characterization of optimal policies *Oper. Res.* 2000; 48:811-819.

[Eng96] Eng D., Humphrey J., Meyn S.P. Fluid network models: Linear programs for control and performance bounds. In *Proceedings of the 13th IFAC World Congress.*, J. Cruz, J. Gertler and M. Peshkin, eds., vol. B, pp. 19-24, San Francisco, CA, 1996.

[Haj84] Hajek B. Optimal control of two interacting service stations. *IEEE Trans. Automat. Control* 1984; AC-29:491-499.

[Kel93] Kelly F.P., Laws, C.N. Dynamic routing in open queueing models: Brownian models, cut constraints and resource pooling. *Queueing Systems* 1993; 13:47-86.

[Luo95] Luo X.D. Continuous linear programming: Theory, algorithms, and applications. Ph.D. thesis, MIT Operations Research Center, Cambridge MA, 1995.

[Luo98] Luo X.D., Bertsimas D. A new algorithm for state-constrained separated continuous linear programming. *SIAM J. Control and Optim.* 1998; 37:177-210.

[Mag00] Maglaras C. Discrete-review policies for scheduling stochastic networks: Fluid-scale asymptotic optimality. *Adv. Appl. Prob.* 2000; vol. 10, no. 3.

[Mey97] Meyn S.P. The policy improvement algorithm for Markov decision processes with general state space. *IEEE Trans. Automat. Control* 1997; AC-42:191-197.

[Mey98] ______ Stability and optimization of multiclass queueing networks and their fluid models. Vol. 33 of *Lectures in Applied Mathematics*, pp. 175-17; American Mathematical Society, 1998.

[Mey01a] ______ Sequencing and routing in multiclass queueing networks. Part I: Feedback regulation. 2000 IEEE International Symposium on Information Theory, Sorrento, Italy, June 25 - June 30, 2000, and *SIAM J. Control and Optimization*, 2001; 40:741-776.

[Mey01b] ______ Sequencing and routing in multiclass queueing networks. Part II: Workload relaxations. To appear, *SIAM J. Control and Optimization*, 2001.

[Per94] Perkins J.R., Kumar P.R. Optimal control of pull manufacturing systems. Technical report University of Illinois, Urbana, IL, 1994.

[Ric95] Ricard M.J. Optimization of queueing networks: An optimal control approach. Ph.D. thesis, MIT Operations Research Center, Cambridge MA, 1995.

[Ryb92] Rybko A.N., Stolyar A.N. Ergodicity of stochastic processes describing the operations of open queueing networks. *Problems of Information Transmission,* 1992; 28:199-220.

[Set99] Sethuraman J. Scheduling job shops and multiclass queueing networks using fluid and semidefinite relaxations. Ph.D. thesis, MIT Operations Research Center, Cambridge MA, 1999.

[Vea01a] Veatch M.H. Fluid analysis of arrival routing. *IEEE Trans. Automat. Contr.* 2001; 46:1254-1257.

[Vea01b] Veatch M.H., de Véricourt F. Zero-inventory conditions for a two part-type make-to-stock production system. Submitted for publication, 2001.

[Vea94] Veatch M.H., Wein L.M. Optimal control of a two-station tandem production/inventory system. *Oper. Res.* 1994; 42:337-350.

[Vea96] Veatch M.H., Wein L.M. Scheduling a make-to-stock queue: Index policies and hedging points. *Oper. Res.* 1996; 44:634-647.

[Wein90] Wein L.M. Scheduling networks of queues: Heavy traffic analysis of a two-station network with controllable input. *Oper. Res.* 1990; 38:1065-1078.

[Wein92] ______ Dynamic scheduling of a multiclass make-to-stock queue. *Oper. Res.* 1992; 40:724-735.

[Weis95] Weiss G. On optimal draining of re-entrant fluid lines. In *Stochastic Networks,* F. Kelly and R. Williams, eds., vol. 71 of the Proceedings of the IMA, pp. 91-103. New York: Springer-Verlag, 1995.

[Wil96] Williams R.J. On the approximation of queueing networks in heavy traffic. In *Stochastic Networks: Theory and Applications,* F. Kelly, S. Zachary, and I. Ziedeins, eds., Oxford University Press, pp. 35-56, 1996.