COLUMBIA
BUSINESS
SCHOOL

Spring 1999
Professor Costis Maglaras
B9801-33: Stochastic Processing Networks
409 Uris Hall

Homework & Solutions #2 – Fluid models and stability analysis

1. *FIFO fluid models.* Consider again the Rybko-Stolyar network in Figure 1.

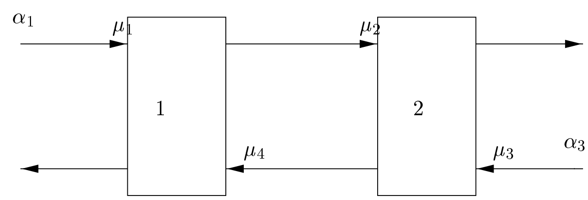

Figure 1: Rybko-Stolyar network: $\alpha_1 = \alpha_3 = 1,\ \mu_1 = \mu_3 = 6,\ \mu_2 = \mu_4 = 1.5$

   (a) Write down (in detail) the queueing network dynamics equations for the RS network operating under the FIFO discipline in both servers.

   (b) Write down the fluid model description for the RS network under FIFO.

   (c) What are the fluid model equations if server 1 is operating under FIFO and server 2 is operating under a static rule that gives priority to class 2 over class 3 jobs.

   Solution. Follows easily by specializing the equations for FIFO networks from Dai's lecture notes for the RS network.

2. *Virtual stations.* Consider now the Lu-Kumar network shown in Figure 2 (another network with a name), under the static rule that gives priority to class 4 and class 2 jobs over class 1 jobs at server 1 and class 3 jobs at server 2 respectively.

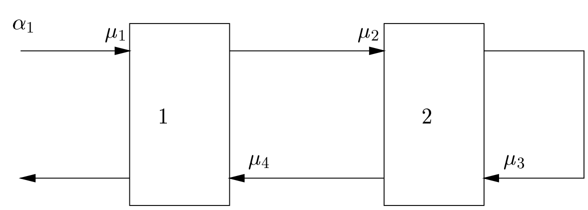

Figure 2: Lu-Kumar network: $\alpha_1 = 1,\ \mu_1 = \mu_3 = 6,\ \mu_2 = \mu_4 = 1.5$

   (a) Write down the fluid model equations.

   (b) What are the traffic intensities at each station? Is this FM stable? Provide a proof or a counterexample. (*Hint:* Compare Lu-Kumar with Rybko-Stolyar. In what ways does their behavior differ, if it does?)

   Solution. Unstable. The counterexample is virtually identical to the one reviewed in class or the one from Dai's notes.

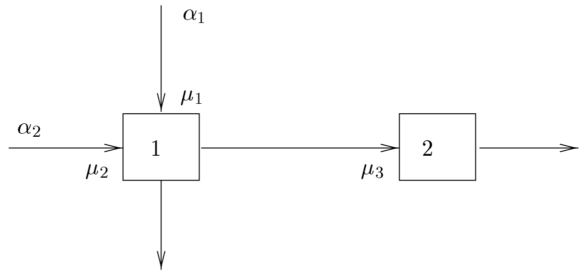

Figure 3: Criss-cross network: $\mu_1 = \mu_2 = 2,\ \mu_3 = 1$

3. *Criss-cross network.* Consider the so called "criss-cross" network shown in Figure 3.

   (a) Write down the fluid model equations. What are the traffic intensities at each station? What are the necessary conditions on α_1, α_2 that guarantee that $\rho_j < 1$ for $j = 1, 2$? (In the sequel we assume that these conditions are met.)

   (b) We are interested in minimizing expected total inventory cost $\mathbf{E}h(Z(t))$ for the queueing network, where $h(Z(t)) = Z_1(t) + Z_2(t) + Z_3(t)$. We will evaluate the "greedy" heuristic derived as follows. We first consider the fluid control policy that myopically drains cost out of the system as fast as possible; i.e., $v(t) = \text{argmin}\{dh(z(t))/dt : v \in \mathcal{V}(z(t))\}$, where $\mathcal{V}(z(t))$ is the set of admissible controls v when the state vector is $z(t)$. Characterize this fluid control policy. What is the static priority rule that emerges from this analysis? (This static rule is the "greedy" heuristic; it is part of the myopic fluid control policy that is directly implementable in the stochastic network.)

   (c) Consider the criss-cross network operating under the static priority rule derived in (b). Is the FM associated with this policy stable? Provide a proof or a counterexample. Can you deduce the stability of the queueing network? (*Hint:* Use your intuition from (b) to construct a Lyapunov function for the associated FM or to construct an unstable fluid solution).

Solution.

(a)

$$\begin{aligned}
\dot{z}_1(t) &= \alpha_1 - \mu_1 v_1(t), \\
\dot{z}_2(t) &= \alpha_2 - \mu_2 v_2(t), \\
\dot{z}_3(t) &= \mu_2 v_2(t) - \mu_3 v_3(t), \\
z(0) &= q,\ z(t) \geq 0,\ t \geq 0, \\
v_1(t) &+ v_2(t) \leq 1,\ v_3(t) \leq 1,\ t \geq 0.
\end{aligned}$$

The stability conditions are: $\rho_1 = (\alpha_1 + \alpha_2)/2 < 1$ and that $\rho_2 = \alpha_2 < 1$. (I reverted to our "usual" notation where $\alpha's$ are the arrival rates and not $\alpha's$ as stated in the question.

(b)

$$\begin{aligned}
v(t) &\in \text{argmin}\{\sum_k \dot{z}_k(t) : v_1 + v_2 \leq 1,\ v_3 \leq 1, \dot{z}_k(t) \geq 0 \text{ for all } k \ s.t.\ z_k(t) = 0\} \\
&= \text{argmax}\{\mu_1 v_1 + \mu_3 v_3 : v_1 + v_2 \leq 1,\ v_3 \leq 1, \dot{z}_k(t) \geq 0 \text{ for all } k \ s.t.\ z_k(t) = 0\}.
\end{aligned}$$

If $z(t) > 0$, then $v(t) = [1, 0, 1]'$. If $z_3(t) = 0$ and $z_1(t), z_2(t) > 0$, then $v(t) = [.5, .5, 1]'$; this will maximize the total fluid outflow from the system. If other job classes are empty the effort is again split to satisfy thevarious constraints. For example, if $z_1(t) = 0$ and $z_2(t), z_3(t) > 0$, then we need $v_1(t) = .5$ and $v_3(t) = 1$. The above equation does not specify $v_2(t)$ which we are free to choose. This will not affect the performance of the fluid model in any way. It is natural to set $v_2(t) = .5$ as well, so that the server will not incur idleness that could be prevented. This will certainly make sense in the stochastic network where pushing work forward will prevent server 2 idleness in future times. This is the myopic control law for this network.

The static priority that emerges is: class 1 has higher priority than class 2.

(c) $V(z) = z_1 + z_2 + z_3$ will serve as a Lyapunov function. Working through the different cases for z depending on what buffers are empty, one can easily show that this Lyapunov function will have a guaranteed negative drift provided that $z \neq 0$.

4. *Criss-cross (cont.).* Consider the following policy in the fluid model: if $z_3(t) > 0$, give priority to class 1 jobs, otherwise give priority to class 2 jobs. Is the FM associated with this policy stable? Again, provide a proof or a counterexample.

Solution. $V(z) = z_1 + 2z_2 + z_3$ will serve as a Lyapunov function. Some people constructed the LP described in Dai to check stability – that was, of course, correct, but did not use any specif info about the policy apart from the fact that it was non-idling.

5. *Fluid limits.* We want to construct a scheduling policy for the criss-cross network that achieves as its fluid limit the behavior considered in the question 4.

(a) Consider implementing the exact policy described in question 4; that is, give priority to class 1 unless $Z_3(t) = 0$, in which case we give priority to class 2. Explain why this policy will not achieve the desired fluid limit. You are not required to give a rigorous derivation of the fluid limit, but just a sketchy argument of how this policy fails. (*Hint:* Consider the system's behavior along the sequence of initial conditions $Z^n(0) = n[1, 1, 0]$.)

(b) How would you modify the policy in part (a) in order to achieve the desired limiting behavior? Try to be explicit and to provide some rough qualification of your answer. (Again, you don't have to derive the fluid limits under your proposed policy.)

Remark: This problem illustrates the relevant scales of magnitude in the stochastic network and in the fluid model; what appears to be a "0" in the fluid model does not literally correspond to a "0" in the queueing network.

Solution.

(a) The problem is one that has been highlighted in class repeatably. Namely, if we wait until $Z_3(t) = 0$ to switch priorities, we will end up incurring idleness while waiting for class 2 jobs to comeplete service until server 2 can start processing them. In fact, for the sequence of initial conditions $Z^n(0) = n[1, 1, 0]$, in the limit as $n \to \infty$, the cumulative idleness for server 2 will be equal to $I(t) = t/3$ for amall enough $t \geq 0$. This comes from the fact that for every job processed at server 2 that takes on average one unit of time, there was an idling period of average length 1/2 unit of time spent in waiting for server 1 to process one class 1 job. Hence, we will not achieve the desired fluid limit that would keep server 2 fully utilized by spliting server 1 effort beteween class 1 and class 2 jobs.

(b) To fix the problem one should introduce class level safety stocks in the system – as it was done in the description of discrete- and continuous-review policies. These will take the form of a threshold θ at server 2, so that server 1 switches priorities if $Z_3(t) \leq \theta$. The threshold should be roughly equal to $\log(|Q(t)|)$.

6. *Networks with routing capability.* Consider a network where every class k job can be routed upon service completion to any job class l in some set R_k^d. We assume that routing decisions are made prior to the beginning of service of each job (this is a mild assumption that simplifies notation.) Let $T_{k,l}(t)$ be the cumulative time allocated in processing class k jobs to be routed into class l jobs upon their service completion up to time t ($l \in R_k^d$). Clearly,

$$T_k(t) = \sum_{l \in R_k^d} T_{k,l}(t).$$

A control policy is the collection of cumulative allocations $T_{k,l}(\cdot)$ for all k and all $l \in R_k^d$.

(a) Write down the system dynamics equations for this class of networks.

(b) Using the notation $\nu_{k,l}(t)$ for $\dot{T}_{k,l}(t)$ ($l \in R_k^d$), what are the associated fluid model equations in differential form? (No formal derivation required.)

(c) We want to derive a condition of weak stability for the fluid model (this is equivalent to $\rho < \mathbf{1}$), such that if this condition is satisfied, then there exists at least one fluid control policy such that the nominal load at each station is less than 1. Express this condition in the form of a linear program. (Note that for this class of networks the traffic intensity at each station depends on the routing policy used. We do not require that every routing policy will be weakly stable; there could be bad routing allocations that do not "balance" the load appropriately among the servers that lead to instability.)

Remark: This is a non-trivial application of our theory. Problems of stability analysis (and, later on, optimization) of networks with routing capability are difficult to address within the realm of "traditional" queueing theory. However, within the framework of fluid models they appear to be simple extensions of the existing theory that can be readily incorporated.

Solution. We treat the case that includes input (or admission) control.

(a) Alternate routing capability arises either when a job completes service at a station and has a choice as to which buffer to join next, or upon an exogenous arrival of a job in the system that again has a choice between different buffers that it can join. We will assume that these external arrival streams or input processes can be turned off, or equivalently, that such jobs can be rejected upon arrival depending on whether such an action would be advantageous for the overall system performance. An incentive structure for accepting arriving jobs will be introduced shortly. In contrast, the models considered so far assumed that routing decisions were made in a Markovian fashion according to the transition matrix P; this corresponds to a a randomized routing policy that is *a priori* specified with no admission control capability.

In extending the multiclass model used so far it is convenient to introduce the designation of a "type" in order to identify each exogenous arrival stream that now could be routed to (or be split into) different job classes; this follows the modeling approach of Harrison in [**?**]. Types of arrivals will be indexed by j and with slight abuse of notation, the set of exogenous arrival streams will still be denoted by $\mathcal{E}$. Furthermore, it is easiest to model these input streams as being "created" or "generated" by fictitious "input servers" associated with each type of arrival. In this framework, a service time of the j^{th} input server corresponds to an interarrival time of the type j input stream drawn from the IID sequence $\{u_j(n), n \geq 1\}$, defined earlier. We denote by R_j^a the set of classes k where a type j job can be routed to upon its arrival, and by R_k^d the set of classes l where a class k job can be routed to upon its service completion. The superscripts "a" and "d" are mnemonic for arrivals and departures respectively. If $\alpha_j = 0$, that is, if there are no arrivals of type j, we set $R^a_j = \{j\}$; this is consistent with our treatment so far.

To simplify notation it will be assumed that routing decisions are made upon the beginning of service of a job by a server or creation of a job by a fictitious input server. (Note that this is not a restrictive assumption, at least in the context of our approach, and could easily be relaxed.) In this case, a class k (type j) job beginning service (creation) is already "tagged" with the destination class $l \in R^d_k$ ($l \in R^a_j$), where it will be routed upon completion of service. Let $T_{k,l}(t)$ be the cumulative time allocated up to time t in processing class k jobs that are routed into class l jobs upon their service completion ($l \in R^d_k$), $Y_{j,l}(t)$ be the cumulative time allocated up to time t in creating type j jobs routed into class l jobs ($l \in R^a_j$), and $E_{j,l}(t)$ be the cumulative number of type j jobs routed into class l jobs up to time t given the cumulative allocation processes $Y_{j,l}(t)$. Extending our earlier formulation, a control policy now takes the form of a pair of cumulative allocation processes $\{(Y(t), T(t)), t \geq 0\}$. For all classes k and any $t \geq 0$,

$$T_k(t) = \sum_{l \in R^d_k} T_{k,l}(t) \quad \text{and} \quad E_k(t) = \sum_{j: k \in R^a_j} E_{j,k}(t), \tag{1}$$

and furthermore for each $j \in \mathcal{E}$ we have that $\sum_{l \in R^a_j} Y_{j,l}(t) \leq t$, for all $t \geq 0$; the last inequality is a consequence of the input control actions up to time t.

The following notation will be useful. Let $y_{j,l}(t)$ denote the fractional effort of the fictitious input server j allocated in creating type j jobs that are routed to the class l buffer at time t, and $\nu_{k,l}(t)$ be the fractional effort of server $s(k)$ allocated in processing class k jobs that are routed to the class l buffer at time t. The notation $v_k(t)$ still denotes the fraction of effort of server $s(k)$ devoted to processing class k jobs at time t.

An incentive structure for accepting externally arriving jobs is introduced in the form of a reward rate function $r^y : \mathbf{R}^K_+ \to \mathbf{R}^{(\sum_{j \in \mathcal{E}} |R^a_j|)}$, that assigns a reward rate $r^y_{j,l}(q)$ to the activity of creating (or accepting) type j jobs that are routed to the class l buffer ($l \in R^a_j$) when the state of the system is equal to q. The resulting instantaneous reward will thus be $r^y_{j,l} y_{j,l}$.

(b) Using the standard procedure outlined in class and in Dai's notes, the fluid limit model for networks with routing and admission control capability is as follows:

$$\dot{z}(t) = \tilde{F} y(t) - \tilde{R} \nu(t), \quad q(0) = z, \tag{2}$$

$$y(t) \geq 0, \quad \nu(t) \geq 0, \quad q(t) \geq 0, \tag{3}$$

$$\tilde{A} y(t) \leq \mathbf{1}, \quad \tilde{C} \nu(t) \leq \mathbf{1}, \tag{4}$$

together with the additional policy specific equations.

(c) The issue of stability is slightly more delicate now. Assuming that there is no input control (this is the natural case to consider), one needs to first define the appropriate notion of nominal load (or traffic intensity) for networks with alternate routing capability. (Without input control the constraint $\tilde{A} y(t) \leq \mathbf{1}$ needs to be replaced by $\tilde{A} y(t) = \mathbf{1}$.) As we have seen earlier, for a system to be stable it is necessary that the nominal load at each station is smaller than its processing capacity. Though, calculating the nominal utilization level at each station is no longer straight forward, since this depends on the routing strategy employed. The following linear program, adapted from Harrison [**?**], computes a worse case bound on the traffic intensity in the network:

$$\begin{aligned} \text{minimize} \quad & \max_{1 \leq i \leq S} \rho_i \\ \text{subject to} \quad & \tilde{A} y = \mathbf{1}, \ \tilde{R} \nu = \tilde{F} y, \ \tilde{C} \nu \leq \rho, \ \nu \geq 0, \ y \geq 0. \end{aligned}$$

The pair (y, ν) describes the average rates at which jobs are processed, created, and routed through the network, and ρ_i is an upper bound on the nominal utilization level at station i. A necessary condition for fluid model stability (according to our previous definition) is that $\rho < \mathbf{1}$. It remains to establish the stability of the underlying stochastic networks, for which one first needs to extend the stability theory developed by Dai to this broader class of networks.

7. *Greedy control laws.* Proving stability has been reduced to an exercise of finding a Lyapunov function V for which $\dot{V}$ has a guaranteed negative drift. This motivates the following choice of a family of fluid control policies.

   Let $V(z(t)) = h'z(t)$ and define a fluid control policy that chooses its instantaneous resource allocation vector in order to minimize the drift of $V(z(t))$. Since, $dV(z(t))/dt = h'\dot{z}(t) = h'(\alpha - Rv(t))$, this policy translates to $v(t) \in \text{argmin}\{h'(\alpha - Rv) : v \in \mathcal{V}(z(t))\}$, which can be rewritten as

$$v(t) \in \text{argmax}\{h'Rv : v \in \mathcal{V}(z(t))\}, \tag{5}$$

   where $\mathcal{V}(z(t)) = \{v : v \geq 0,\ Cv \leq \mathbf{1},\ (\alpha - Rv)_k \geq 0 \text{ for all } k \in \mathcal{I}(z(t))\}$ and $\mathcal{I}(z(t)) = \{k : z_k(t) = 0\}$; that is $\mathcal{I}(z(t))$ is the set empty buffers at time t, and $\mathcal{V}(z(t))$ is the set of admissible controls given $z(t)$.

   The vector $R'h$ is referred to as a *reward* vector, where $(R'h)_k$ is interpreted as the reward rate for processing class k jobs. Higher reward rates can be interpreted as higher priorities among classes at the same server. The policy in (5) allocates resources for all t in order to maximize instantaneous total reward, which corresponds to minimizing the drift of $V(z(t))$.

   (a) Prove that if $R'h > 0$, then the fluid solutions associated with the policy (5) are non-idling; that is, $(Cv(t))_k = 1$ whenever $(Cz(t))_k > 0$.

   (b) Prove that if $R'h > 0$, then the fluid model under the policy (5) is stable. (*Hint:* You only need to consider the "most constrained" admissible control sets $\mathcal{V}(z(t))$ in establishing the negative drift of an appropriate Lyapunov function. You will need to justify this.)

   (c) Consider the fluid model associated with the Rybko-Stolyar network of Figure 1. Following our earlier interpretation, the reward rate vector $R'h = [1, 2, 1, 2]$ corresponds to the LBFS policy. You just proved that the greedy implementation for this reward rate vector is stable. We know, however, that LBFS is unstable for this example. What is going on?

Solution.

(a) Let $r = R'h$. For any $z(t) \neq 0$, the instantaneous allocation process in the fluid model $v(t)$, will satisfy equation (5). Suppose that there exists a server i such that $(Cz(t))_i > 0$ and $(Cv(t))_i < 1$. Let $k \in C_i$ be any job class for which $z_k(t) > 0$. Define the following instantaneous control $\hat{v} = v(t) + e_k(1 - (Cv(t))_i)$, where e_k is the k^{th} unit vector. Clearly, $\hat{v} > 0$ and $C\hat{v} = Cv(t) + Ce_k(1 - (Cv(t))_i) \leq \mathbf{1}$. Furthermore, for any job class $j \neq k$, $\alpha_j - (R\hat{v})_j \geq \alpha_j - (Rv(t))_j \geq 0$, which implies that $\hat{v}$ is a feasible instantaneous allocation. Moreover,

$$r'\hat{v} = r'v(t) + r_k(1 - (Cv(t))_i) > r'v(t),$$

which contradicts the optimality of $v(t)$. Hence, there does not exist any server i such that $(Cz(t))_i > 0$ and $(Cv(t))_i < 1$, and thus the fluid solutions satisfying equation (5) will be non-idling.

(b) We will use $V(z(t)) = h'z(t) = r'R^{-1}z(t)$. First, observe that by the definition of the matrix R, it follows that R^{-1} is componentwise non-negative. Then, since $r > 0$ it follows that $r'R^{-1} > 0$ and thus, first, $V(z(t)) > 0$ for all $z(t) \neq 0$ and second, $V(z) = 0$ only when $z = 0$. Using $V(\cdot)$ as a candidate Lyapunov function, it is sufficient to prove that for all $z(t) \neq 0$ and for some $\epsilon > 0$,

$$\frac{dV(z(t))}{dt} = \min_{v \in \mathcal{V}(z(t))} r'R^{-1}(\alpha - Rv) < -\epsilon. \tag{6}$$

Recall the definition of $\mathcal{I}(z(t)) = \{k : z_k(t) = 0\}$. The condition of equation (6) should be checked over all possible feasibility sets of the form $\mathcal{V}(z(t)) = \{v : v \geq 0,\ Cv \leq \mathbf{1},\ (Rv)_k \leq \alpha_k \text{ for all } k \in \mathcal{I}(z(t))\}$. Observe that if $\mathcal{I}(z_1) \subseteq \mathcal{I}(z_2)$, then $\mathcal{V}(z_1) \supseteq \mathcal{V}(z_2)$. This implies the drift condition need to be checked only for the extreme (most constrained) cases defined by $\mathcal{I}_k = \{1, \ldots, K\}/\{k\}$, for all job classes k. These sets correspond to the cases where all but the k^{th} job classes are empty, and the associated feasibility sets will be denoted by $\mathcal{V}_k = \{v : v \geq 0,\ Cv \leq \mathbf{1},\ R^k v \leq \alpha^k\}$, for the appropriate $(K-1) \times K$ matrix R^k and $(K-1)$-vector α^k.

In order to check the drift condition for each $\mathcal{V}_k$, consider the input $\hat{v}^k = R^{-1}\alpha + \delta_k e_k$, where $\delta_k = 1 - \rho_{s(k)} > 0$. Clearly, $\hat{v}^k \geq 0$ and also $C\hat{v}^k = CR^{-1}\alpha + \delta_k Ce_k \leq \mathbf{1}$. Finally,

$$\alpha - R\hat{v}^k = -\delta_k Re_k = \delta_k(Pe_k - e_k) \Rightarrow R^k \hat{v}^k \leq \alpha^k, \tag{7}$$

which establishes the feasibility of the instantaneous allocation $\hat{v}^k$. Furthermore, a simple calculation yields that $r'R^{-1}(\alpha - R\hat{v}^k) = -\delta_k r_k < 0$. It follows from (5) that

$$\frac{dV(z(t))}{dt} < r'R^{-1}(\alpha - R\hat{v}^k). \tag{8}$$

Hence, the linear Lyapunov function $V(z(t)) = r'R^{-1}z(t)$ satisfies the drift condition of equation (6) with $\epsilon = \min_k \delta_k r_k$. One can easily obtain a bound on the time required to empty the fluid model starting from any bounded initial condition and establish stability.

(c) For the Rybko-Stolyar network, the reward rate vector $r = R'h = [1, 2, 1, 2]$ corresponds to the LBFS policy. Although the greedy implementation of this reward rate vector into (5) will yield a stable policy, LBFS alone will be unstable. The difference is due to the fact that (5) is allowed additional flexibility when some classes $z_k(t) = 0$ that help in avoiding undesirable idleness. This difference is evident by the fact that (5) uses global information to decide on its instantaneous allocation (that is, when $z_2(t)$ is empty it switches some of the effort at server 1 to class 1 jobs thus avoiding undesirable idleness at station 2), whereas LBFS uses only local information at each station in its decision.