COLUMBIA
BUSINESS
SCHOOL

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

Homework #2 – Fluid models and stability analysis

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

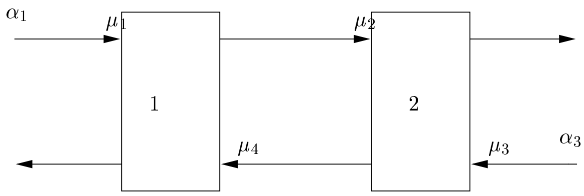

Figure 1: Rybko-Stolyar network: $\alpha_1 = \alpha_3 = 1,\ \mu_1 = \mu_3 = 6,\ \mu_2 = \mu_4 = 1.5$

   (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.

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.

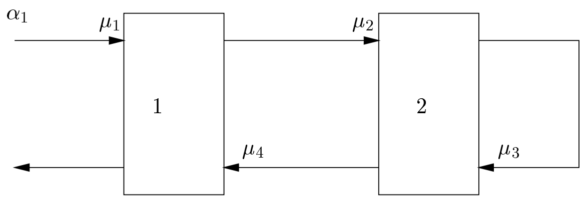

Figure 2: Lu-Kumar network: $\alpha_1 = 1,\ \mu_1 = \mu_3 = 6,\ \mu_2 = \mu_4 = 1.5$

   (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?)

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 λ_1, λ_2 that guarantee that $\rho_j < 1$ for $j = 1, 2$? (In the sequel we assume that these conditions are met.)

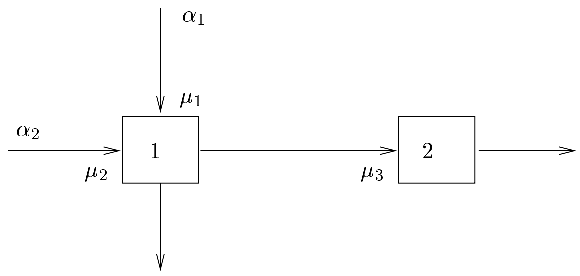

Figure 3: Criss-cross network: $\mu_1 = \mu_2 = 2,\ \mu_3 = 1$

(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).

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.

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.

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 $\dot{\bar{T}}_{k,l}(t)$ ($l \in R_k^d$), 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.

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 $v(t) \in \operatorname{argmin}\{h'(\alpha - Rv) : v \in \mathcal{V}(z(t))\}$, which can be rewritten as

   $$v(t) \in \operatorname{argmax}\{h'Rv : v \in \mathcal{V}(z(t))\}, \tag{1}$$

   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 (1) 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 (1) 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 (1) 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?