Northwestern University
Department of Electrical Engineering and Computer Science

ECE 454: Advanced Communication Networks

Supplemental Notes on Conservation Laws and $C\mu$-rule

In Problem 3.40, you are asked to show that in a n-class, M/G/1 non-preemptive system, the sum $\sum_{k=1}^{n} \rho_k W_k$ is independent of the priority order of classes and satisfies

$$\sum_{k=1}^{n} \rho_k W_k = \frac{R\rho}{1-\rho}, \tag{1}$$

where $\rho = \rho_1 + \cdots + \rho_n$ and R is the mean residual-time. This is known as a *conservation law*. Since R, ρ and ρ_k do not depend on the priority ordering, this means that under any non-preemptive priority policy, the feasible delays must satisfy this linear equality. In particular is says it tells us that if we decrease the delay of a certain class, then some other class must increase its delay by an appropriate amount so that (1) still holds.

This type of conservation law holds in a more general setting, which we describe next. Consider a multi-class M/G/1 system as in Sect. 3.5.3 of Bertsekas and Gallager. A general non-preemptive scheduling policy for this system is a rule π that specifies which class to serve next whenever the server finishes a job. A priority policy is a special type of scheduling policy where the rule is to always serve a packet from the highest priority class that has a packet available. In general, this policy could depend on other information, such as the number of waiting packets for each class, the time of day, etc. A policy is said to be *non-anticipative* if the scheduling decisions only depend on the past history and current state of the system (i.e. they can not anticipate the future). A policy is said to be *work conserving* if the server is never idle when there is a job waiting in the system. We will call a scheduling policy *admissible* if it is both non-anticipative and work conserving.

It can be shown that any non-preemptive admissible scheduling policy will satisfy the conservation law in (1). The basic idea here is that under any work conserving policy, the unfinished work in the system (as defined in Problem 3.40) must decrease at the same rate; this observation can be used to derive the conservation law.

In addition, to (1), the waiting time of each class k, under any non-preemptive admissible scheduling policy must satisfy

$$\rho_k W_k \geq \frac{\rho_k R}{1-\rho_k}, \tag{2}$$

or equivalently,

$$W_k \geq \frac{R}{1-\rho_k}. \tag{3}$$

Note that the right-hand side of (3) is the average queueing delay in a non-preemptive priority system, where class k is given the highest priority. More generally, given any subset $\mathcal{S} \subset \{1, 2, \ldots, n\}$, the average queueing delays within this subset will satisfy

$$\sum_{k\in\mathcal{S}} \rho_k W_k \geq \frac{\rho_{\mathcal{S}} R}{1-\rho_{\mathcal{S}}}, \tag{4}$$

where $\rho_{\mathcal{S}} = \sum_{k\in\mathcal{S}} \rho_k$. The right-hand side of (4) is the value of $\sum_{k\in\mathcal{S}} \rho_k W_k$ for any policy which gives the classes in $\mathcal{S}$ non-preemptive priority over the classes not in $\mathcal{S}$. Given a system with n classes, there will be $2^n - 2$ non-empty subsets of classes not including the whole set. For each of these subsets, there will be a corresponding inequality as in (4) which must be satisfied. For example in a system with $n = 3$ classes, in addition to the whole set, there will be the following 6 subsets : $\{1\}, \{2\}, \{3\}, \{1, 2\}, \{2, 3\}, \{1, 3\}$. Inequality (4) corresponding to the subset $\{1, 3\}$ is

$$\rho_1 W_1 + \rho_3 W_3 \geq \frac{(\rho_1 + \rho_3)R}{1 - \rho_1 - \rho_3}.$$

These $2^n - 2$ inequalities in addition to the equality in (1) determine the set of feasible waiting times. In particular they constrain these delays to lie within a $n - 1$ dimensional polyhedron (in $\mathbb{R}^n$.) Let $\mathcal{P}$ denote this polyhedron, i.e. $\mathcal{P}$ is the set of waiting time $(W_1, \ldots, W_n)$ which satisfy these inequalities and equality constraints. This polyhedron will have $n!$ corner points; each corner point can be shown to correspond to the performance (vector of average waiting times) achieved by a priority policy corresponding to one of the $n!$ orderings of the users.

There has been a long history of research on various optimal scheduling problems. In these problems, one is typically trying to find a scheduling policy which optimizes some long-term average cost. One example of this is a policy which minimizes a linear *holding cost*, i.e.,

$$\sum_{i=1}^{n} c_i N_{Q_i}$$

where N_{Q_i} is the average number of packets in the queue of class i and $c_i > 0$ is a holding cost for each class i packet. By Little's theorem, this is equivalent to a "delay" cost given by

$$\sum_{i=1}^{n} c_i \lambda_i W_i. \tag{5}$$

From our above discussion, it follows that if we are minimizing this over all admissible, non-preemptive policies, then we can simply view this problem as optimizing (5) over all $(W_1, \ldots, W_k) \in \mathcal{P}$. This is an example of a *linear programming problem* - i.e., a problem with a linear objective and linear constraints. It follows from a basic result in linear programming that a corner point of $\mathcal{P}$ must be an optimal solution, i.e. the optimal policy must be a priority policy. Furthermore for this problem, it can be shown (cf. Problem 3.40) that the optimal priority ordering is given by sorting the users in decreasing order of $c_i \mu_i$. The resulting policy is known as a $c\mu$-rule.

1 Further notes

1. Often, the above inequalities and equalities are stated in terms of the unfinished work $V_i = \rho_i W_i$. In this case, (4) can be written as

$$\sum_{k\in\mathcal{S}} V_k \leq b(\mathcal{S}),$$

where $b(\mathcal{S})$ is a *set-function* which determines the right-hand side of this constraint. In this case, the resulting polyhedron of feasible $V_1, \ldots, V_n$ has a special structure, and is referred to as a *polymatroid*. The polymatroid structure simplifies solving various optimization problems over this set.

2. In addition to the M/G/1 model with non-preemptive policies, the idea of conservation laws can be generalized to other systems (See for example Chap. 11 in H. Chen and D.Yao, "Fundamentals of Queueing Networks: Performance, Asymptotics, and Optimization," Springer, 2001.) In these cases, one needs to consider a different performance measure than the unfinished work, V_k.