Robust fluid processing networks *

Dimitris Bertsimas, † *Member, IEEE,* Ebrahim Nasrabadi, ‡ *Member, IEEE,*
and Ioannis Ch. Paschalidis,§ *Fellow, IEEE,*

***Abstract*—Fluid models provide a tractable and useful approach in approximating multiclass processing networks. However, they ignore the inherent stochasticity in arrival and service processes. To address this shortcoming, we develop a robust fluid approach to the control of processing networks. We provide insights into the mathematical structure, modeling power, tractability, and performance of the resulting model. Specifically, we show that the robust fluid model preserves the computational tractability of the classical fluid problem and retains its original structure. From the robust fluid model, we derive a (scheduling) policy that regulates how fluid from various classes is processed at the servers of the network. We present simulation results to compare the performance of our policies to several commonly used traditional methods. The results demonstrate that our robust fluid policies are near-optimal (when the optimal can be computed) and outperform policies obtained directly from the fluid model and heuristic alternatives (when it is computationally intractable to compute the optimal).**

***Index Terms*—Multiclass processing networks, fluid models, robust optimization, scheduling, optimal control.**

I. Introduction

In *multiclass processing networks*, we are concerned with serving multiple types of jobs which may differ in their arrival processes, processing times, routes through the network, and cost per unit of holding time at the various servers of the network. Such models are used in a number of application domains including manufacturing systems, multiprocessor computer systems, communication networks, data centers, and sensor networks. A fundamental control problem in these systems is that of *sequencing*. In particular, a *sequencing policy* determines at every point in time which type of job to serve at each server of the network.

Optimal sequencing decisions in a multiclass processing network are in general dynamic and state-dependent, as a decision depends on load conditions not only at the server where it is to be made but also at other servers. Naturally, uncertainties regarding the arrival and service processes further complicate the problem. As a result, this problem is both theoretically and computationally hard to solve optimally, even for problems with a few number of servers and job types. It can be formulated as a stochastic dynamic programming problem but that does not lead to tractable approaches for large instances. Thus, a number of researchers have attempted to develop tractable approximations of the optimal policy (see Bertsimas et al. [8], Chen and Mandelbaum [12], Harrison [20], Harrison and Wein [22], and Kumar [25]). This led to the study of Brownian models and fluid relaxations as approximation techniques to multiclass processing networks.

The Brownian approach was first introduced by Harrison [20] and further explored by Wein [50], [51], and other researchers, including Laws and Louth [27], Taylor and Williams [48], and Williams [53]. It approximates the processing network in a heavy-traffic regime, that is, when the workload of the system reaches its capacity limit. In several instances, a policy can be constructed which is optimal in this limiting regime. Brownian models typically make use of the mean and variance of the associated stochastic processes in deriving a simpler control problem. However, except for problems that are essentially one-dimensional, this approach is itself intractable.

On the other hand, fluid models are often tractable, but ignore the variance of the associated stochastic processes. They are deterministic, continuous approximations to stochastic, discrete networks. Research on fluid models is mainly motivated by the developments in the area of stability of multiclass processing networks using the fluid model analysis. A major breakthrough was the theory developed by Dai [15], who showed that the stability of the processing network is implied by the stability of its associated fluid model (see also [12], [45], [16], [17], [47]).

There is also a close connection between the control of processing networks and the optimal control of the corresponding fluid models. There are several examples where the solution of the fluid optimal control problem recovers significant information about the structure of an optimal policy in the original multiclass processing network (see, e.g., [3], [33], [38]). In particular, Avram et al. [3] find explicit optimal solutions for the associated fluid models of specific processing networks and derive threshold policies for the optimal (sequencing) control of these networks. They also show that the well-known $c\mu$-rule is optimal for a single-server processing network, as well as, the corresponding fluid model.

Beyond special cases, several works have developed methods and guidelines for translating policies derived for the fluid optimal control problem into an implementable control policy for the stochastic, discrete network. Related work includes [32], [4], [31], [5], [14], [34], [13]. Meyn [32] presents several numerical experiments to evaluate the performance of discrete review policies and proposes a policy based upon an affine shift of the fluid policy which gives significant improvement in his numerical experiments. A family of discrete review policies

* Research partially supported by the NSF under grants CNS-1239021 and IIS-1237022, by the ARO under grants W911NF-11-1-0227 and W911NF-12-1-0390, and by the ONR under grant N00014-10-1-0952.

† Sloan School of Management and Operations Research Center, Bldg. E40-147, Massachusetts Institute of Technology, Cambridge, MA 02139, e-mail: dbertsim@mit.edu.

‡ Operations Research Center, Massachusetts Institute of Technology, and Department of Electrical and Computer Engineering, Boston University, e-mail: nasrabad@mit.edu.

§ Department of Electrical & Computer Engineering, and Division of Systems Engineering, Boston University, 8 St. Mary's St., Boston, MA 02215, e-mail: yannisp@bu.edu, url: http://ionia.bu.edu/.

is also proposed by Maglaras [31] based on the BIGSTEP approach introduced by Harrison [21]. These policies utilize safety stocks to prevent starvation of resources in the stochastic system and are shown in [30], [31] to achieve asymptotic optimality and stability under fluid scaling. In workload models that account for work in the system for each server (see, Meyn [35], [34] for a thorough description), translation from a fluid policy to a policy for the stochastic system uses the idea of hedging-point policies adapted from the inventory control literature; these are essentially affine translations of the fluid policy that protect against the risk of potentially high cost. A comprehensive treatment of these models and policies can be found in [35] which also provides specific guidelines on selecting safety stocks and hedging points based on the parameters of the stochastic system.

A related approach to synthesizing stable policies for the stochastic system is to use tracking policies [4], [36]. Paschalidis et al. [36], in particular, propose a class of (sequencing and routing) policies that "drive" the state of the processing network towards a pre-determined target (termed, "target-pursuing" policies). An advantage of these policies is that they are amenable to distributed implementation using local state information. Using fluid model analysis, [36] shows that these policies are in fact stable.

The study of fluid models in multiclass processing networks is also motivated from the existence of very efficient optimization algorithms used to solve them. Fluid models for processing networks can be formulated as a specially structured class of continuous linear programs called *Separated Continuous Linear Programs* (SCLPs). These problems can be efficiently solved using mathematical optimization techniques, in contrast to the traditional diffusion control approach. During the last decades, significant progress has been made in solving SCLPs and their generalizations. In particular, Anderson et al. [2] characterize the extreme point solutions to SCLPs and show the existence of optimal solutions with a finite number of breakpoints in certain cases. Pullan, in a series of papers [39], [40], [41], [42], [44], extensively studies SCLPs. He develops a detailed duality theory, conditions under which an optimal solution exists with a finite number of breakpoints, as well as, a convergent algorithm for solving SCLPs. Luo and Bertsimas [29] propose a convergent numerical algorithm for a larger class of SCLPs that is able to efficiently solve problems involving hundreds of variables and constraints. Fleischer and Sethuraman [18] present polynomial-time approximation algorithms for solving SCLPs. Weiss [52] characterizes the form of optimal solutions, establishes a strong duality result and develops a solution algorithm using simplex pivot operations.

Despite extensive work on the optimal control of processing networks, this body of research still lacks a unified tractable and practical approach accommodating all salient features of the problem. While fluid models are tractable, they ignore the inherent uncertainties of the problem. This adversely impacts the performance of the policies derived from the fluid model. The majority of the approaches we reviewed earlier for translating fluid policies to the stochastic system were derived with stability being the key concern and attempt to accommodate uncertainty by appropriately modifying the optimal fluid policy. In our work we attempt to incorporate uncertainty in the fluid optimal control problem. A traditional way to handle uncertainty in optimization problems is to use stochastic optimization, where the uncertain data are modeled as random variables. However, this approach typically leads to problems that are often intractable to solve. We refer to Birge and Louveaux [10] and Shapiro [46] for more information on stochastic optimization. Another approach introduces stochasticity in the fluid model (resulting in a so-called stochastic fluid model) but it can only be used in perturbation analysis schemes, that is, producing gradient estimators of policy parameters that can be leveraged to optimize specific parametrized classes of policies (see Cassandras et al. [11]). Yet another approach is to use robust optimization, which treats the uncertainty in a deterministic manner and typically leads to tractable problems. This approach assumes that the uncertain parameters come from known sets and optimize against the worst-case realization of the parameters within the uncertainty sets. We refer to Ben-Tal et al. [6], Bertsimas et al. [7] and the references therein for a survey on robust optimization. It is this latter approach we introduce for multiclass processing networks.

Our contribution: We introduce a tractable approach that captures both dynamic and uncertain characteristics in multiclass processing networks. Our approach is to formulate the fluid control model as an SCLP and use robust optimization to deal with the uncertainty. We present insights into the modeling power, tractability, and performance of the proposed model. More specifically, our contributions are:

(i) ***Modeling power*:** We study fluid models in an uncertain environment from the viewpoint of robust optimization and introduce a *robust fluid problem*. We show that the robust fluid model still remains within the class of SCLPs. Thus, it preserves the computational tractability of the classical fluid problem, and all solution techniques for SCLPs remain applicable.

(ii) ***Insights*:** We consider a single-server processing network and derive valuable insights about properties of an optimal solution for the corresponding robust fluid problem. In particular, we use complementary slackness optimality conditions to develop a polynomial-time algorithm for solving the robust fluid problem. Our results can be seen as natural extensions to the $c\mu$-rule and the optimal priority policy for Klimov's problem [9], in the presence of parameter uncertainty.

(iii) ***Performance*:** We propose methods to translate an optimal solution for the robust fluid control problem to implementable sequencing policies for the stochastic network. Because uncertainty is handled at the robust fluid problem, our methods do not need any distributional assumption on the stochastic network. Moreover, translation of the resulting policy to the stochastic system is more direct. We report extensive simulations results to evaluate the performance of sequencing policies derived from the robust fluid model. We compare the performance of the proposed policies to several commonly used heuristic methods. Our results show that for small-

size networks, the proposed policies yield near-optimal policies (when the optimal can be computed) and for moderate to large-size networks the performance significantly outperforms the heuristic methods.

The remainder of the paper is organized as follows. In Section II, we formulate the fluid control problem of multiclass processing networks as an SCLP. In Section III, we consider uncertainty on arrival and service processes and investigate its robust counterpart. We further propose two methods to translate an optimal solution for the robust fluid control problem to implementable sequencing policies. In Section IV, we develop a polynomial-time algorithm to derive an optimal solution for the robust fluid control problem of single-server processing networks. In Section V, we report extensive simulation results to evaluate the performance of the proposed approach and compare it to other methods in the literature. Section VI contains some concluding remarks.

Notational conventions: Throughout this paper, all vectors are assumed to be column vectors and prime denotes the transpose operation. We use lower case boldface letters to denote vectors and for economy of space we write $\mathbf{x} = (x_1, \ldots, x_n)$ for the column vector $\mathbf{x}$. We use boldface upper case letters to denote matrices. We use $\mathbf{e}$ to denote the vector of all ones and $\mathbf{0}$ for the vector of all zeroes. For a set S, we write $|S|$ to denote its cardinality. We use $x(\cdot)$ to denote a function $x : [0, T] \to \mathbb{R}$ and $\mathbf{x}(\cdot)$ to denote a vector whose components are real-valued functions defined on the interval $[0, T]$. When it is clear from the context that $\mathbf{x}(\cdot)$ is a vector whose components are functions, we use $\mathbf{x}$ instead of $\mathbf{x}(\cdot)$. We use the lower case letter i to denote a job class, and use the lower case letter j to denote a server. Finally, we use $\forall t$ to refer to all $t \in [0, T]$, $\forall i$ to refer to all job classes, and $\forall j$ to refer to all servers.

II. Problem description and fluid model

In this section, we present a general framework for the fluid control of multiclass processing networks. We first describe the fluid model for a simple network considered by Harrison and Wein [22] and then describe the general problem formulation.

A. Criss-cross network

Consider the processing network in Figure 1 composed of three classes and two servers; class 1 and 2 jobs are processed at server 1 and class 3 jobs are processed at server 2. Class 1 jobs arrive at server 1 with a rate of λ_1 and class 2 jobs arrive at server 1 with a rate of λ_2. After a class 1 job completes service at server 1, it moves to server 2 and turns into a job of class 3. Once a class 3 job completes service at server 2, it exits the system. After a class 2 job completes service at server 1, it exits the system. For each class i, we let μ_i be the the service rate of these jobs; that is, the rate at which jobs are processed if the server processes class i jobs at its full capacity.

Assuming that there are jobs in the system for all three classes, the problem amounts to deciding whether server 1 should process class 1 or 2 jobs. To formulate this problem as a fluid model, we let $x_i(t)$ denote the total (fractional in general) number of class i jobs at time t and let $u_i(t)$ denote the effort that the corresponding server – denote it by $s(i)$ – spends processing class i jobs at time t. This implies that

$$\frac{u_1(t)}{\mu_1} + \frac{u_2(t)}{\mu_2} \leq 1,$$
$$\frac{u_3(t)}{\mu_3} \leq 1.$$

Assuming stability, let T be a large enough time so that the system will empty by time T. To ensure that the system reaches a state in which all of the classes are empty, it is required to have sufficient capacity to clear the arrivals. More precisely, the *traffic intensity* at both servers must be strictly smaller than one, i.e.,

$$\frac{\lambda_1}{\mu_1} + \frac{\lambda_2}{\mu_2} < 1,$$
$$\frac{\lambda_1}{\mu_3} < 1.$$

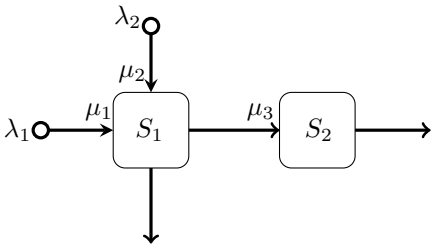


Fig. 1. Criss-cross network.

Let c_i be the cost per unit time for holding a job of class i in its corresponding buffer. The *fluid control* problem is to find a control $\mathbf{u}$ such that the total holding cost of the jobs in the system is minimized over the time interval $[0,\ T]$. This problem is formulated as follows:

$$\begin{aligned}
\min \quad & \int_0^T \mathbf{c}'\mathbf{x}(t)\, dt \\
\text{s.t.} \quad & \dot{x}_1(t) = \lambda_1 - u_1(t), \qquad \forall t, \\
& \dot{x}_2(t) = \lambda_2 - u_2(t), \qquad \forall t, \\
& \dot{x}_3(t) = u_1(t) - u_3(t), \qquad \forall t, \\
& \frac{u_1(t)}{\mu_1} + \frac{u_2(t)}{\mu_2} \leq 1, \qquad \forall t, \\
& \frac{u_3(t)}{\mu_3} \leq 1, \qquad \forall t, \\
& \mathbf{u}(t), \mathbf{x}(t) \geq \mathbf{0}, \qquad \forall t.
\end{aligned} \tag{1}$$

B. A general formulation

Consider a processing network with m servers and n different job classes. Each class i has an associated server $s(i)$ that processes jobs of class i. Jobs either leave the system or change class as they move through the network. In particular, if jobs of class i do not leave the system, they have a unique next class $r(i)$, that is, they join class $r(i)$ when they complete service at server $s(i)$. The arrivals for each class i come from other servers or from outside the system. We let λ_i be the rate

of external arrivals for class i. We set $\lambda_i = 0$ if class i has no external arrivals.

For each class i, we let the control variable $u_i(t)$ specify the effort that server $s(i)$ spends processing the jobs of class i. The state variable $x_i(t)$ denotes the number of class i jobs at time t in the system. The dynamics of the system take the form

$$\dot{x}_i(t) = \lambda_i - u_i(t) - \sum_{j \neq i} a_{ji} u_j(t), \tag{2}$$

where a_{ji} is either 0 or -1 depending on whether or not class i receives arrivals from class j. Hence, routing in the network can be represented by an $n \times n$ matrix $\mathbf{A}$, such that $a_{ii} = 1$ for $i = 1, 2, \ldots, n$, and $a_{ji} = -1$ if class i receives arrivals from class j. The dynamics of the system in matrix form can be expressed as:

$$\dot{\mathbf{x}}(t) = \boldsymbol{\lambda} - \mathbf{A}\mathbf{u}(t), \tag{3}$$

where $\boldsymbol{\lambda}$ is the vector of external arrivals. In the criss-cross network of Figure 1, we have

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}, \qquad \boldsymbol{\lambda} = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ 0 \end{bmatrix}.$$

By integrating both sides of Equation (3) with respect to t, we get the following equation:

$$\int_0^t \mathbf{A}\mathbf{u}(s)ds + \mathbf{x}(t) = \mathbf{x}(0) + \boldsymbol{\lambda} t, \tag{4}$$

where $\mathbf{x}(0)$ is the given vector of the number of jobs at time 0.

Each server may process multiple job classes, each with its own service rate. Let μ_i be the service rate of class i jobs. Then, the *service time* is given by $\tau_i := 1/\mu_i$, that is, the required time to process one unit of class i jobs. Moreover, the fraction of the effort that server $s(i)$ spends processing jobs of class i at time t is given by $\tau_i u_i(t)$. Hence, the sum of $\tau_i u_i(t)$ for all the classes processed at the same server must be less than one. This constraint can be expressed as

$$\mathbf{H}\mathbf{u}(t) \leq \mathbf{e},$$

where $\mathbf{H}$ is an $m \times n$ matrix with components

$$h_{ji} = \begin{cases} \tau_i, & \text{if } s(i) = j, \\ 0, & \text{otherwise.} \end{cases}$$

Following the above discussion, the fluid control problem can be formulated as follows:

$$\begin{aligned} \min \quad & \int_0^T \mathbf{c}'\mathbf{x}(t)\, dt \\ \text{s.t.} \quad & \int_0^t \mathbf{A}\mathbf{u}(s)ds + \mathbf{x}(t) = \mathbf{x}(0) + \boldsymbol{\lambda} t, \qquad \forall t, \\ & \mathbf{H}\mathbf{u}(t) \leq \mathbf{e}, \qquad \forall t, \\ & \mathbf{u}(t), \mathbf{x}(t) \geq \mathbf{0}, \qquad \forall t. \end{aligned} \tag{5}$$

The state variables $\mathbf{x}$ can be eliminated from the formulation of Problem (5). This can be done by substituting (4) in the objective function of Problem (5) and using integration by parts. It follows:

$$\begin{aligned} \int_0^T \mathbf{c}'\mathbf{x}(t)\, dt &= \int_0^T \mathbf{c}'\Big(\mathbf{x}(0) + \boldsymbol{\lambda} - \mathbf{A}\mathbf{u}(t)\Big)\, dt \\ &= T\mathbf{c}'\mathbf{x}(0) + \int_0^T (T-t)\mathbf{c}'(\boldsymbol{\lambda} - \mathbf{A}\mathbf{u}(t))\, dt. \end{aligned}$$

Notice that the first term is constant and does not depend on the control variables $\mathbf{u}(t)$. Thus, Problem (5) can be rewritten as

$$\begin{aligned} \min \quad & \int_0^T (T-t)\mathbf{c}'(\boldsymbol{\lambda} - \mathbf{A}\mathbf{u}(t))\, dt \\ \text{s.t.} \quad & \int_0^t \mathbf{A}\mathbf{u}(s)ds \leq \mathbf{x}(0) + \boldsymbol{\lambda} t, \qquad \forall t, \\ & \mathbf{H}\mathbf{u}(t) \leq \mathbf{e}, \qquad \forall t, \\ & \mathbf{u}(t) \geq \mathbf{0}, \qquad \forall t. \end{aligned} \tag{6}$$

We work within the space $L_\infty([0,T])$ of essentially bounded measurable functions on $[0,T]$ in which functions that differ only on a set of measure zero are identified. In particular, the components of $\mathbf{u}$ are assumed to be bounded measurable functions on $[0,T]$. We say that a control $\mathbf{u}$ is feasible if it satisfies the constraints of Problem (6) and denote the *feasible region* of Problem (6) by $\mathcal{F}$, i.e.,

$$\mathcal{F} := \{\mathbf{u} \in L_\infty^n[0,T] \mid \mathbf{u} \text{ is feasible for Problem (6)}\}.$$

Problem (6) belongs to the well studied class of SCLPs. This class of problems has been first introduced by Anderson [1] in order to model job-shop scheduling problems. Since then, a number of authors (including Pullan [39], [40], [41], [43], [44], Philpott and Craddock [37], Luo and Bertsimas [29], Fleischer and Sethuraman [18] and Weiss [52]) have studied SCLPs from different points of view. We next present some results on duality of SCLPs developed by Luo and Bertsimas [29] that we will use in Section IV.

The dual of Problem (5) is formulated as follows:

$$\begin{aligned} \max \quad & -\int_0^T (\mathbf{x}(0) + \boldsymbol{\lambda} t)'\, d\boldsymbol{\pi}(t) - \int_0^T \mathbf{e}'\boldsymbol{\eta}(t)\, dt \\ \text{s.t.} \quad & \mathbf{A}'\boldsymbol{\pi}(t) - \mathbf{H}'\boldsymbol{\eta}(t) \leq 0, \qquad \forall t, \\ & \boldsymbol{\pi}(t) \leq (T-t)\mathbf{c}, \qquad \forall t, \\ & \boldsymbol{\pi} \text{ bounded measurable with finite} \\ & \text{variation and } \boldsymbol{\pi}(T) = \mathbf{0}, \\ & \boldsymbol{\eta}(t) \geq \mathbf{0}, \qquad \forall t, \end{aligned} \tag{7}$$

where the first integral in the objective function is the Lebesgue-Stieltjes integral of the function $\mathbf{x}(0) + \boldsymbol{\lambda} t$, with respect to the function $\boldsymbol{\pi}(t)$, from 0 to T.

Suppose that $\mathbf{u}, \mathbf{x}$ is a feasible solution for Problem (5) and $\boldsymbol{\pi}$, $\boldsymbol{\eta}$ is a feasible solution for Problem (7). Then, $\mathbf{u}, \mathbf{x}$ is optimal for Problem (5) and $\boldsymbol{\pi}$, $\boldsymbol{\eta}$ is optimal for Problem (7) if the following complementary slackness conditions hold (see

[29, Corollary 1]):

$$\begin{aligned}\int_0^T \Big(-\mathbf{A}'\pi(t) + \mathbf{H}'\eta(t)\Big)' \mathbf{u}(t)\, dt = 0,\\ \int_0^T \Big(\mathbf{H}\mathbf{u}(t) - \mathbf{e}\Big)' \eta(t)\, dt = 0,\\ \int_0^T \mathbf{x}(t)'\, d\Big((T-t)\mathbf{c} - \boldsymbol{\pi}(t)\Big) = 0.\end{aligned} \tag{8}$$

III. Robust fluid model

In Problem (6), the components of the matrix $\mathbf{H}$ and vector $\boldsymbol{\lambda}$ are treated as deterministic quantities. In this section, we present a *robust* fluid model that will inject uncertainty in the fluid model. This approach assumes that the uncertain parameters come from known sets, called *uncertainty sets*. We start our discussion by modeling uncertainty sets for the fluid control problem and then investigate its robust counterpart problem.

A. Modeling the uncertainty

In practice, arrival rates and service times are not only uncertain, but also change over time. We let $\tau_i(t)$ be the actual realization of the service time and $\lambda_i(t)$ be the actual realization of the arrival rate at time t for jobs of type i. We assume that $\tau_i(t)$ can take values in the interval $[\bar{\tau}_i, \bar{\tau}_i + \tilde{\tau}_i]$ at each point in time t. We refer to $\bar{\tau}_i$ as the *nominal service time* and to $\tilde{\tau}_i$ as its *deviation*. We let $z_i(t)$ be the relative deviation from the nominal service time at time t, that is,

$$z_i(t) := \begin{cases} \frac{\tau_i(t) - \bar{\tau}_i}{\tilde{\tau}_i}, & \text{if } \tilde{\tau}_i > 0,\\ 0, & \text{if } \tilde{\tau}_i = 0.\end{cases}$$

We restrict the service times to a set of vector-valued functions $\boldsymbol{\tau}(\cdot) = (\tau_1(\cdot), \ldots, \tau_n(\cdot))$ so that

$$\tau_i(t) = \bar{\tau}_i + z_i(t)\tilde{\tau}_i, \qquad \forall i, t, \tag{9a}$$

$$\sum_{i:s(i)=j} z_i(t) \le \Gamma_j, \qquad \forall j, t, \tag{9b}$$

$$0 \le z_i(t) \le 1, \qquad \forall i, t. \tag{9c}$$

Here Γ_j is a given parameter in the interval $[0, n_j]$, where n_j is the number of job classes that are processed at server j, i.e., $n_j = |\{i \mid s(i) = j\}|$. This parameter controls the level of the uncertainty in service times. The larger Γ_j is, the more uncertain are the service times of jobs which are processed in server j.

For a given $\boldsymbol{\tau}(\cdot)$, we consider an associated $m \times n$ matrix-valued function $\mathbf{H}(\cdot)$, where

$$h_{ji}(t) = \begin{cases} \tau_i(t), & \text{if } s(i) = j,\\ 0, & \text{otherwise.}\end{cases}$$

We define the *uncertainty set* $\mathcal{U}$ to be the set of all matrix-valued functions $\mathbf{H}(\cdot)$, where $\boldsymbol{\tau}(\cdot)$ is given by (9).

In a similar way, we model the uncertainty on the arrival rates. For each class i and each point in time t, we assume that $\lambda_i(t)$ takes values in the interval $[\bar{\lambda}_i, \bar{\lambda}_i + \tilde{\lambda}_i]$. We refer to $\bar{\lambda}_i$ as the nominal arrival rate and to $\tilde{\lambda}_i$ as its *deviation*. For a given vector $\boldsymbol{\lambda}(t)$, we let

$$\zeta_i(t) := \begin{cases} \frac{\lambda_i(t) - \bar{\lambda}_i}{\tilde{\lambda}_i}, & \text{if } \tilde{\lambda}_i > 0;\\ 0, & \text{if } \tilde{\lambda}_i = 0.\end{cases}$$

We define the *uncertainty set* $\mathcal{D}$ to be the set of all vector-valued functions $\boldsymbol{\lambda}$ so that

$$\lambda_i(t) = \bar{\lambda}_i + \zeta_i(t)\tilde{\lambda}_i, \qquad \forall i, t, \tag{10a}$$

$$\sum_{i:s(i)=j} \zeta_i(t) \le \Delta_j, \qquad \forall j, t, \tag{10b}$$

$$0 \le \zeta_i(t) \le 1, \qquad \forall i, t. \tag{10c}$$

If a class i has no external arrivals, we set $\bar{\lambda}_i = \tilde{\lambda}_i = 0$. In this case, $\zeta_i(t) = 0$ for all $t \in [0, T]$, and thus, class i does not have any contribution in the summation on the left-hand side of Inequality (10b).

B. Robust counterpart problem

Having defined the uncertainty sets as above, a control $\mathbf{u}$ is called *robust* if it satisfies the constraints of Problem (6) with respect to all possible realizations of uncertain data. Let $\mathcal{S}$ denote the set of all robust controls. This means that $\mathbf{u} \in \mathcal{S}$ if and only if

$$\begin{aligned}\int_0^t \mathbf{A}\mathbf{u}(s)ds \le \mathbf{x}(0) + \boldsymbol{\lambda}(t)t, \qquad & \forall \boldsymbol{\lambda} \in \mathcal{D},\\ \mathbf{H}(t)\mathbf{u}(t) \le \mathbf{e}, \qquad & \forall \mathbf{H} \in \mathcal{U},\\ \mathbf{u}(t) \ge \mathbf{0}, & \end{aligned}$$

for all $t \in [0, T]$. We refer to a robust control with the best worst-case cost guarantee as an *optimal* robust control. The *robust counterpart* problem is to find such a control. This problem is formulated as follows:

$$\min_{\mathbf{u}\in\mathcal{S}} \max_{\boldsymbol{\lambda}\in\mathcal{D}} \int_0^T (T-t)\mathbf{c}'\Big(\boldsymbol{\lambda}(t) - \mathbf{A}\mathbf{u}(t)\Big)\, dt. \tag{11}$$

Theorem 1. *An optimal robust control can be obtained by solving the following problem:*

$$\begin{aligned}\min \quad & \int_0^T \mathbf{c}'\mathbf{x}(t)\, dt\\ \text{s.t.} \quad & \int_0^t \mathbf{A}\mathbf{u}(s)ds + \mathbf{x}(t) = \mathbf{x}(0) + \bar{\boldsymbol{\lambda}}t, \quad \forall t,\\ & \Gamma_j\beta_j(t) + \sum_{i:s(i)=j} (\bar{\tau}_i u_i(t) + \alpha_i(t)) \le 1, \quad \forall j, t,\\ & \alpha_i(t) + \beta_j(t) - u_i(t)\tilde{\tau}_i \ge 0, \quad \forall j, i \text{ with } s(i) = j, \forall t,\\ & \mathbf{u}(t), \mathbf{x}(t), \boldsymbol{\alpha}(t), \boldsymbol{\beta}(t) \ge \mathbf{0}, \quad \forall t.\end{aligned} \tag{12}$$

Proof: We first show that a control $\mathbf{u}$ is robust if and only if there are $\boldsymbol{\alpha}(\cdot), \boldsymbol{\beta}(\cdot) \ge 0$ so that

$$\int_0^t \mathbf{A}\mathbf{u}(s)ds \le \mathbf{x}(0) + \bar{\boldsymbol{\lambda}}t, \tag{13a}$$

$$\Gamma_j\beta_j(t) + \sum_{i:s(i)=j} (\bar{\tau}_i u_i(t) + \alpha_i(t)) \le 1, \quad \forall j, \tag{13b}$$

$$\alpha_i(t) + \beta_j(t) - u_i(t)\tilde{\tau}_i \geq 0, \quad \forall j, i \text{ with } s(i) = j, \quad (13c)$$

for all $t \in [0, T]$.

Given a control $\mathbf{u}$, we have

$$\int_0^t \mathbf{A}\mathbf{u}(s)ds \leq \mathbf{x}(0) + \boldsymbol{\lambda}(t)t, \qquad \forall t, \boldsymbol{\lambda} \in \mathcal{D},$$

if and only if

$$\int_0^t \mathbf{a_i}\mathbf{u}(s)ds \leq x_i(0) + \min_{\boldsymbol{\lambda}\in\mathcal{D}} \lambda_i(t)t = x_i(0) + \bar{\lambda}_i t, \quad \forall i, t,$$

where $\mathbf{a_i}$ is the ith row of the matrix $\mathbf{A}$.

In addition,

$$\mathbf{H}(t)\mathbf{u}(t) \leq \mathbf{e}, \qquad \forall t, \mathbf{H} \in \mathcal{U},$$

if and only if

$$\mathcal{Z}_j(\mathbf{u}, t) \leq 1, \qquad \forall j, t,$$

where

$$\begin{aligned} \mathcal{Z}_j(\mathbf{u}, t) := \max \quad & \sum_{i:s(i)=j} (\bar{\tau}_i + z_i(t)\tilde{\tau}_i)u_i(t) \\ \text{s.t.} \quad & \sum_{i:s(i)=j} z_i(t) \leq \Gamma_j, \\ & 0 \leq z_i(t) \leq 1, \quad \forall i : s(i) = j. \end{aligned}$$

Using strong duality for linear optimization problems, we can write:

$$\begin{aligned} \mathcal{Z}_j(\mathbf{u}, t) = \min \quad & \Gamma_j\beta_j(t) + \sum_{i:s(i)=j} (\bar{\tau}_i u_i(t) + \alpha_i(t)) \\ \text{s.t.} \quad & \alpha_i(t) + \beta_j(t) - u_i(t)\tilde{\tau}_i \geq 0, \ \forall i : s(i) = j, \\ & \alpha_i(t) \geq 0, \ \forall i : s(i) = j, \\ & \beta_j(t) \geq 0. \end{aligned}$$

This justifies constraints (13).

We now turn our attention to the objective function of Problem (11). For a given robust control $\mathbf{u} \in \mathcal{S}$, we let

$$\mathcal{Z}(\mathbf{u}) := \max_{\boldsymbol{\lambda}\in\mathcal{D}} \int_0^T (T-t)\mathbf{c}'\Big(\boldsymbol{\lambda}(t) - \mathbf{A}\mathbf{u}(t)\Big)\, dt. \quad (14)$$

It follows from the definition of $\mathcal{D}$ that

$$\begin{aligned} \mathcal{Z}(\mathbf{u}) = & \int_0^T (T-t)\mathbf{c}'\Big(\bar{\boldsymbol{\lambda}}(t) - \mathbf{A}\mathbf{u}(t)\Big)\, dt \\ & + \max \quad \int_0^T \sum_{i=1}^n (T-t)c_i\tilde{\lambda}_i\zeta_i(t)\, dt \\ & \text{s.t.} \quad \sum_{i=1}^n \zeta_i(t) \leq \Delta_j, \quad \forall j, \\ & \qquad 0 \leq z_i(t) \leq 1, \qquad \forall i, t. \end{aligned} \quad (15)$$

By taking the dual of the maximization problem, we obtain

$$\begin{aligned} \mathcal{Z}(\mathbf{u}) = & \int_0^T (T-t)\mathbf{c}'\Big(\bar{\boldsymbol{\lambda}}(t) - \mathbf{A}\mathbf{u}(t)\Big)\, dt \\ & + \min \quad \int_0^T \sum_{j=1}^m \Delta_j y_j(t)\, dt + \int_0^T \sum_{i=1}^n w_i(t)\, dt \\ & \text{s.t.} \quad y(t) + w_i(t) \geq c_i(T-t)\tilde{\lambda}_i, \quad \forall i, t, \\ & \qquad y(t), \mathbf{w} \geq 0, \qquad \forall t. \end{aligned}$$

Here, the minimization problem is independent of $\mathbf{u}$. As a result, finding an optimal robust control reduces to the following problem:

$$\begin{aligned} \min \quad & \int_0^T (T-t)\mathbf{c}'\Big(\bar{\boldsymbol{\lambda}}(t) - \mathbf{A}\mathbf{u}(t)\Big)\, dt \\ \text{s.t.} \quad & \int_0^t \mathbf{A}\mathbf{u}(s)ds \leq \mathbf{x}(0) + \bar{\boldsymbol{\lambda}}t, \qquad \forall t, \\ & \Gamma_j\beta_j(t) + \sum_{i:s(i)=j} (\bar{\tau}_i u_i(t) + \alpha_i(t)) \leq 1, \qquad \forall t, j, \\ & \alpha_i(t) + \beta_j(t) - u_i(t)\tilde{\tau}_i \geq 0, \ \forall t, i, j : s(i) = j, \\ & \mathbf{u}(t), \alpha(t), \beta(t) \geq 0, \qquad \forall t. \end{aligned} \quad (16)$$

This problem is equivalent to Problem (11) by setting

$$\mathbf{x}(t) = \mathbf{x}(0) + \bar{\boldsymbol{\lambda}}t - \int_0^t \mathbf{A}\mathbf{u}(s)ds, \qquad \forall t.$$

■

It follows from Theorem 1 that the uncertainty on arrival rates does not play any role in determining an optimal robust control and the robust counterpart problem only relies on the nominal values of the arrival rates. Hence, in the rest of the paper, the arrival rates are assumed to be deterministic and are denoted by $\boldsymbol{\lambda}$.

C. Robust policies

Having being able to solve the robust fluid problem, the next step is to translate optimal robust controls to a dynamic scheduling policy for the control of stochastic multiclass processing networks. Here, we present a simple approach, which is known as *model predictive control* in control theory and engineering practice (see, e.g., [19], [26], [28], [49]). The derived policy is similar to the discrete review policies (see, e.g., [21], [32], [31]), where the system state is reviewed at discrete points in time and at each such point control decisions are made using the optimal control policy of the associated fluid control problem. In our case, however, the impact of uncertainty has been dealt with at the fluid control level.

In model predictive control, control decisions are made at *control epochs*, i.e., at discrete points in time when the state of the system is changed due to job arrivals and departures. The main idea is to solve the robust fluid problem at every control epoch and use the first step of the optimal (fluid) control as the current sequencing decision. At the next epoch, we solve the robust fluid control problem again, and so on. Formally, to find a policy at the control epoch t, we set $x_i(0)$ to be the number of class i jobs at that epoch. We then solve Problem (11) and let $\mathbf{u}^*, \mathbf{x}^*$ be an optimal robust solution. It is known that Problem (11) has a piecewise constant optimal control and the algorithm developed by Luo and Bertsimas [29] finds such a solution. More precisely, there is a partition $\{t_0 = 0, t_1, \ldots, t_q = T\}$ of the time interval $[0, T]$ so that $\mathbf{u}^*$ is constant over $[t_{k-1}, t_k)$ for all $k = 1, \ldots, q$. This means that the control $\mathbf{u}^*(t_{k-1})$ is optimal if the *state* (that is, number of jobs in the system) is $\mathbf{x}(t_{k-1})$. In particular, $\mathbf{u}^*(0)$ is an

optimal control at the epoch t. For each class i, we let

$$p_i^* := \frac{u_i^*(0)}{\sum_{k:s(k)=s(i)} u_k^*(0)}.$$

This implies that $\sum_{i:s(i)=j} p_i^* = 1$ for each server j. We then use the following sequencing policy for the jobs at server j:

> **Robust fluid policy (RFP)**: give priority to a job class i with highest value p_i^*. If there exists more than one such job classes, break ties arbitrarily.

We note that for specific problems, we may use particular rules to break ties. With the above policy, each server will be processing at most one job at a time. The computational tractability of this model predictive control scheme depends on (a) how efficiently one can solve Problem (11) and (b) how many times one has to solve Problem (11). Regarding issue (a) we notice that Problem (11) is an instance of SCLP with $2n + m$ control variables, n state variables, and $n + m + nm$ constraints. In general, solving an SCLP is NP-hard since it includes as special case the minimum cost dynamic flow problem, which is weakly NP-hard (see [24]). However, the problem is computationally tractable in the sense that one can solve large instances. In our simulation experiments in Section V we use the algorithm of Luo and Bertsimas [29] which can handle hundreds of variables and constraints. Regarding issue (b) above, we note that in general one may need to solve a large number of SCLPs – one at each control epoch. In Section V-A we introduce a heuristic that helps reduce this number significantly. Our numerical examples in Section V will show that our approach is tractable as one can handle processing networks with tens of job classes and tens of servers.

IV. A SINGLE-SERVER SYSTEM

In this section, we show that one can find an optimal control for the robust fluid problem in polynomial time under certain conditions. We consider a single server processing network with n jobs. We let λ_i be the arrival rate, μ_i be the processing time, and c_i be the holding cost per unit of time for class i. The problem is to schedule the jobs so as the total holding cost is minimized. It is well known that an optimal policy for this problem is to give priority to a class i with highest $c_i\mu_i$ – the well known $c\mu$-rule. We show that the robust fluid control problem also yields a priority policy that can be computed in polynomial-time.

The control problem for processing the jobs on a single server is formulated as follows:

$$\begin{aligned}
\min \quad & \int_0^T \mathbf{c}'\mathbf{x}(t)\,dt \\
\text{s.t.} \quad & \int_0^t u_i(s)\,ds + x_i(t) = x_i(0) + \lambda_i t, \qquad \forall t, i, \\
& \sum_{i=1}^n \tau_i u_i(t) \le 1, \qquad \forall t, \\
& \mathbf{u}(t), \mathbf{x}(t) \ge 0, \qquad \forall t,
\end{aligned} \tag{17}$$

where $\tau_i := 1/\mu_i$ is the service time for class i jobs. We assume that service times are subject to uncertainty and fluctuate over time while the arrival rates are deterministic. For each class i and each point in time t, we let the actual realization of the service time lie in the interval $[\bar{\tau}_i, \bar{\tau}_i + \tilde{\tau}_i]$, where $\bar{\tau}_i$ is the nominal service time and $\tilde{\tau}_i$ is the deviation from its nominal value. We assume that the total relative deviation from the nominal service times is bounded by Γ. Then, by Theorem 1, the robust counterpart of Problem (17) is:

$$\begin{aligned}
\min \quad & \int_0^T \mathbf{c}'\mathbf{x}(t)\,dt \\
\text{s.t.} \quad & \int_0^t u_i(s)\,ds + x_i(t) = x_i(0) + \lambda t, \quad \forall i, t, \\
& \sum_{i=1}^n (u_i(t)\bar{\tau}_i + \alpha_i(t)) + \Gamma\beta(t) \le 1, \quad \forall t, \\
& u_i(t)\tilde{\tau}_i(t) - \alpha_i(t) - \beta(t) \le 0, \qquad \forall t, \\
& \mathbf{u}(t), \mathbf{x}(t), \boldsymbol{\alpha}(t), \beta(t), \ge \mathbf{0}, \qquad \forall t.
\end{aligned} \tag{18}$$

We next describe how to construct an optimal solution $\mathbf{u}^*, \boldsymbol{\alpha}^*, \beta^*$ for this problem by solving at most n linear optimization problems. The basic idea is that at each point in time t, the structure of $\mathbf{u}^*, \boldsymbol{\alpha}^*, \beta^*$ depends on the number of jobs in the system at time t. More specifically, we let $\mathbf{u}^*(t) := \mathbf{v}^*(t)$, $\boldsymbol{\alpha}^*(t) := \boldsymbol{\rho}^*(t)$, $\beta^*(t) := \xi^*(t)$, where $\mathbf{v}^*(t), \boldsymbol{\rho}^*(t), \xi^*(t)$ is an optimal solution for the following linear optimization problem:

$$\begin{aligned}
\max \quad & \sum_{i=1}^n (T-t)c_i v_i(t) \\
\text{s.t.} \quad & \sum_{i=1}^n \big(\bar{\tau}_i v_i(t) + \rho_i(t)\big) + \Gamma\xi(t) \le 1, \\
& v_i(t)\tilde{\tau}_i - \xi(t) - \rho_i(t) \le 0, \qquad \forall i, \\
& v_i(t) \le \lambda_i, \qquad \forall i : x_i(t) = 0, \\
& \mathbf{v}(t), \boldsymbol{\rho}(t), \xi(t) \ge \mathbf{0}.
\end{aligned} \tag{19}$$

We refer to this problem as LO(t).

Initially, we have $t = t_0 := 0$, at which point $x(0)$ is given. We solve LO(t_0) and obtain an optimal control $\mathbf{v}^*(t_0)$. We serve the jobs with this policy until a class, say class 1, is depleted, that is $x_1(t_1) = 0$ where t_1 is the depletion time of class 1. More precisely, we set $\mathbf{u}^*(t) := \mathbf{v}^*(t_0)$ for all $0 \le t < t_1$. At time $t = t_1$, a switch occurs and the policy is revised. To do that, we solve LO(t_1) to find an optimal policy $\mathbf{v}^*(t_1)$ at time t_1. We use this policy to serve jobs until another job class, say class 2, is depleted. Let t_2 be the depletion time of class 2. We then set $\mathbf{u}^*(t) := \mathbf{v}^*(t_1)$ for all $t_1 \le t < t_2$. We continue this procedure until all classes are depleted, at which point and thereafter, an optimal policy is to serve each job class i with rate λ_i and no jobs will be held in the network.

The above procedure requires solving at most n linear optimization problems and yields a piecewise-constant solution $\mathbf{u}^*, \boldsymbol{\alpha}^*, \beta^*$ for Problem (18) with breakpoints $t_0, t_1, \ldots, t_n$ so

that

$$\begin{aligned}
\mathbf{u}^*(t) &:= \begin{cases} \mathbf{v}^*(t_{k-1}), & \text{if } t_{k-1} \le t < t_k, \\ \mathbf{v}^*(t_{n-1}), & \text{if } t_n \le t \le T, \end{cases} \\
\boldsymbol{\alpha}^*(t) &:= \begin{cases} \boldsymbol{\rho}^*(t_{k-1}), & \text{if } t_{k-1} \le t < t_k, \\ \boldsymbol{\rho}^*(t_{n-1}), & \text{if } t_n \le t \le T, \end{cases} \\
\beta^*(t) &:= \begin{cases} \xi^*(t_{k-1}), & \text{if } t_{k-1} \le t < t_k, \\ \xi^*(t_{n-1}), & \text{if } t_n \le t \le T, \end{cases}
\end{aligned} \tag{20}$$

where $\mathbf{v}^*(t_{k-1})$, $\boldsymbol{\rho}^*(t_{k-1})$, $\xi^*(t_{k-1})$ is an optimal solution for LO(t_{k-1}).

Theorem 2. *The solution* $\mathbf{u}^*, \boldsymbol{\alpha}^*, \beta^*$*, given by* (20)*, is optimal for Problem* (18)*.*

Proof: It follows from the construction of $\mathbf{u}^*, \boldsymbol{\alpha}^*, \beta^*$ that it is feasible for Problem (18). To prove it is optimal, we construct a dual feasible solution for the dual of Problem (18) which satisfies optimality conditions with $\mathbf{u}^*, \boldsymbol{\alpha}^*, \beta^*$. Based on Bertsimas and Luo's [29] dual formulation (7) for SCLPs, the dual of Problem (17) is formulated as follows:

$$\begin{aligned}
\max \quad & -\int_0^T (\mathbf{x}(0) + \boldsymbol{\lambda} t)' \, d\boldsymbol{\pi}(t) - \int_0^T \eta(t)\, dt \\
\text{s.t.} \quad & \pi_i(t) - \bar{\tau}_i \eta(t) - \tilde{\tau}_i \gamma_i(t) \le 0, \qquad \forall i, t, \\
& \sum_{i=1}^n \gamma_i(t) - \Gamma \eta(t) \le 0, \\
& \gamma_i(t) - \eta(t) \le 0, \qquad \forall i, t, \\
& \pi_i(t) \le (T-t) c_i, \qquad \forall i, \\
& \boldsymbol{\pi} \text{ bounded measurable with finite} \\
& \text{ variation and } \boldsymbol{\pi}(T) = \mathbf{0}, \\
& \eta(t) \ge 0, \qquad \forall t.
\end{aligned} \tag{21}$$

Suppose that $\mathbf{u}, \boldsymbol{\alpha}, \beta$ is a feasible solution for Problem (18) and $\boldsymbol{\pi}, \eta, \gamma$ is a feasible solution for Problem (21). It follows from the complementary slackness optimality conditions (8) that $\mathbf{u}, \boldsymbol{\alpha}, \beta$ is optimal for Problem (18) and $\boldsymbol{\pi}, \eta, \gamma$ is optimal for Problem (21) if the following conditions are met:

$$\begin{aligned}
& \int_0^T \sum_{i=1}^n \Big(\pi_i(t) - \bar{\mu_i} \eta(t) - \tilde{\tau}_i \gamma_i(t) \Big) u_i(t)\, dt = 0, \\
& \int_0^T \Big(1 - \Gamma \beta(t) - \sum_{i=1}^n (\bar{\tau}_i u_i(t) + \alpha_i(t)) \Big) \eta(t)\, dt = 0, \\
& \int_0^T \Big(\Gamma \eta(t) - \sum_{i=1}^n \gamma_i(t) \Big) \beta(t)\, dt = 0, \\
& \int_0^T \sum_{i=1}^n \Big(\eta(t) - \gamma_i(t) \Big) \alpha_i(t)\, dt = 0, \\
& \int_0^T \sum_{i=1}^n x_i(t) d\Big(\pi_i(t) - (T-t) c_i \Big) = 0.
\end{aligned} \tag{22}$$

We next construct a feasible solution for Problem (21), which satisfies the above optimality conditions. To that end, we take the dual of Problem (19) and obtain the following linear optimization problem:

$$\begin{aligned}
\min \quad & \theta(t) + \boldsymbol{\lambda}' \boldsymbol{\delta}(t) \\
\text{s.t.} \quad & \delta_i(t) + \bar{\tau}_i \theta(t) + \tilde{\tau}_i q_i(t) \ge (T-t) c_i, \qquad \forall i, \\
& \Gamma \theta(t) - \sum_{i=1}^n q_i \ge 0, \\
& \theta(t) - q_i(t) \ge 0, \qquad \forall i, \\
& \boldsymbol{\delta}(t), \mathbf{q}(t), \theta \ge \mathbf{0}, \\
& \delta_i(t) = 0, \qquad \forall i : x_i(t) > 0.
\end{aligned}$$

Let $\mathbf{q}^*(t), \boldsymbol{\delta}^*(t), \theta^*(t)$ be an optimal solution for this problem. We define

$$\begin{aligned}
\boldsymbol{\pi}^*(t) &:= \begin{cases} (T-t)\mathbf{c} - \boldsymbol{\delta}^*(t_{k-1}), & \text{if } t_{k-1} \le t < t_k, \\ (T-t)\mathbf{c} - \boldsymbol{\delta}^*(t_{k-1}), & \text{if } t_{n-1} \le t \le T, \end{cases} \\
\boldsymbol{\gamma}^*(t) &:= \begin{cases} \mathbf{q}^*(t_{k-1}), & \text{if } t_{k-1} \le t < t_k, \\ \mathbf{q}^*(t_{n-1}), & \text{if } t_n \le t \le T, \end{cases} \\
\eta^*(t) &:= \begin{cases} \boldsymbol{\theta}^*(t_{k-1}), & \text{if } t_{k-1} \le t < t_k, \\ \boldsymbol{\theta}^*(t_{n-1}), & \text{if } t_n \le t \le T. \end{cases}
\end{aligned}$$

It is easy to verify that $\boldsymbol{\pi}^*, \boldsymbol{\gamma}^*, \eta^*$ is a feasible solution for Problem (21). Moreover, by the complementary slackness conditions for linear optimization, we can show that $\mathbf{u}^*$, $\boldsymbol{\rho}^*$, ξ^* and $\boldsymbol{\pi}^*, \boldsymbol{\gamma}^*, \eta^*$ satisfies the optimality conditions (22). This completes the proof. ∎

We notice that Problem (19) can be viewed as the robust counterpart of a maximization problem, where a class i with highest $c_i/\bar{\tau}_i$ must be selected. More specifically, when all service times are deterministic, that is, $\tilde{\tau}_i = 0$ for all classes i, Problem (19) is simplified as

$$\begin{aligned}
\max \quad & \sum_{i=1}^n (T-t) c_i v_i(t) \\
\text{s.t.} \quad & \sum_{i=1}^n \bar{\tau}_i v_i(t) \le 1, \\
& v_i(t) \le \lambda_i, \qquad \forall i : x_i(t) = 0, \\
& \mathbf{v}(t) \ge \mathbf{0}.
\end{aligned} \tag{23}$$

This problem gives priority to a class i with highest $c_i/\bar{\tau}_i$. Thus, our approach provides an alternative proof for the optimality of the $c\mu$-rule when all service times are deterministic. Further, the robust fluid model retains its original structure and Problem (19) can be seen as the robust version of Problem (23), thereby providing a robust generalization of the $c\mu$-rule.

A. Klimov's Problem

Next, we show that all the above results can carry over to Klimov's problem – a single server queue with probabilistic feedback (see Klimov [23], Bertsimas et al. [9], and references therein). Specifically, a p_{ij} fraction of class i jobs are fed back as jobs of class j and a p_{i0} fraction of class i jobs leave the system. As before, the problem is to schedule the jobs so as the total holding cost is minimized. The corresponding control

problem is formulated as follows:

$$\begin{aligned} \min \quad & \int_0^T \mathbf{c}'\mathbf{x}(t)\,dt \\ \text{s.t.} \quad & \int_0^t \mathbf{A}\mathbf{u}\,ds + x_i(t) = \mathbf{x}(0) + \boldsymbol{\lambda} t, \qquad \forall t, \\ & \sum_{i=1}^{n} \tau_i u_i(t) \le 1, \qquad \forall t, \\ & \mathbf{u}(t), \mathbf{x}(t) \ge \mathbf{0}, \qquad \forall t, \end{aligned} \tag{24}$$

where $\mathbf{A}$ is an $n \times n$ matrix with $a_{ii} = 1$ for $i = 1, 2, \ldots, n$, and $a_{ij} = -p_{ij}$ for $i \neq j$. One can use Gauss–Jordan elimination to verify that the matrix $\mathbf{A}$ is invertible. Then, Problem (24) can be rewritten as

$$\begin{aligned} \mathcal{Z}_{\text{CQ}} = \min \quad & \int_0^T \mathbf{c}'\mathbf{x}(t)\,dt \\ \text{s.t.} \quad & \int_0^t w_i(s)\,ds + x_i(t) = \lambda_i t, \qquad \forall i, t, \\ & \boldsymbol{\tau}'\mathbf{A}^{-1}\mathbf{w}(t) \le 1, \qquad \forall t, \\ & \mathbf{A^{-1}w(t)} \ge \mathbf{0}, \qquad \forall \mathbf{t}, \\ & \mathbf{x}(t) \ge \mathbf{0}, \qquad \forall t. \end{aligned} \tag{25}$$

This problem is the same as Problem (17), but with different matrix coefficients. Therefore, all previous techniques can be extended to Problem (25) and one can show that its robust counterpart is solvable in polynomial-time.

V. Simulation results

In this section, we provide simulation results to compare the performance of the robust fluid policy to several alternative policies in the literature. There are several motivating reasons for this simulation study. First, we wish to test how close is the performance of the proposed policy to the performance of the optimal policy in small-size networks, where the optimal can be computed. The second purpose is to test whether the robust fluid policy outperforms several heuristic policies on moderate to large-size networks. The third purpose is to examine the effect of the uncertainty set on the performance of the robust policy. More precisely, we assume that the total relative deviation of the service times from their nominal values is bounded by Γ for each server and investigate the sensitivity of the robust policy to the parameter Γ. In addition, we seek to test the performance the robust policy against heuristic alternatives under various distributions of arrival and service processes. To that end, we simulate external arrivals and service times under a hyper-exponential distribution and investigate the performance of the robust fluid policy when the *Coefficient of Variation (CoV)* increases. Finally, we are interested in the computational efficiency of our approach, which relates to how frequently we have to solve Problem (11) in order to obtain a sequencing policy for each possible network state. We comment on the latter issue in the next subsection.

A. Computational remarks

The robust fluid policy described in Section III-C requires solving Problem (11) for each state to compute an optimal control. It is computationally intractable to enumerate and evaluate the optimal control for all states on moderate to large-size networks since the number of possible states grows exponentially in the number of job classes. Instead, we propose a heuristic method to approximate optimal controls and speed up the computational time.

We first notice that some states may never appear during a simulation of the system. Therefore, we solve the robust fluid model only when the system reaches a *new* state. Let $\mathbf{n}$ be the vector representing the number of jobs in each class of the system at the current epoch. We set $\mathbf{x}(0) = \mathbf{n}$ and solve Problem (11). Let $\mathbf{u}^*, \mathbf{x}^*$ be an optimal solution, where $\mathbf{u}^*$ is piecewise constant with breakpoints $t_0 = 0, t_1, \ldots, t_q = T$. This solution does provide an optimal policy when the state is $\mathbf{x}^*(t_1), \ldots, \mathbf{x}^*(t_{q-1})$. In particular, the optimal control $\mathbf{u}^*(0)$ on the first step yields a policy to sequence the jobs at the current epoch (when the state is $\mathbf{n}$), and for $k = 1, \ldots, q-1$ the control $\mathbf{u}^*(t_k)$ is optimal if the system reaches the state $\mathbf{x}(t_k)$. We can therefore, maintain this information and use it whenever these particular states are reached, which helps to avoid re-solving Problem (11) if we already know the optimal control for a state.

In general, $\mathbf{x}^*(t_1), \ldots, \mathbf{x}^*(t_{q-1})$ are fractional, while the number of jobs in the system is integer. Suppose that the system reaches a state $\mathbf{n}$ at some later epoch and there is some k so that $|n_i - x_i^*(t_{k-1})| \le \omega$ and $n_i > 0$ if and only if $x_i^*(t_{k-1}) > 0$ for all i, where $\omega \ge 0$ is a given parameter to control the accuracy of the heuristic. Then, we apply the control $\mathbf{u}^*(t_{k-1})$ to sequence the jobs at the servers. We show in our simulation experiments that the performance of the robust policy is rather insensitive with respect to the parameter ω, while it reduces significantly the number of calls to a solver for Problem (11).

B. Network examples

We consider four different processing networks under various parameter scenarios. The problem is to determine a dynamic sequencing policy at each server so that the long-run average expected number of jobs in the system is minimized. The external arrivals are Poisson with class-dependent rates, and the service times are exponentially distributed with class-dependent rates. Let $\boldsymbol{\lambda}$ be the vector of mean arrival rates and $\boldsymbol{\tau}$ be the vector of mean services times. When solving the robust fluid problem to derive a robust policy for each state, we set the nominal values of external arrival rates to $\boldsymbol{\lambda}$ and the nominal values of service times to $\boldsymbol{\tau}$ with an allowed deviation of $0.25\boldsymbol{\tau}$ so that the total relative deviation from the nominal service times at each server is bounded by Γ. More precisely, we set $\mathbf{c} := \mathbf{e}$, $\bar{\boldsymbol{\lambda}} := \boldsymbol{\lambda}$, $\bar{\boldsymbol{\tau}} := \boldsymbol{\tau}$, $\tilde{\boldsymbol{\tau}} := 0.25\boldsymbol{\tau}$ in Problem (11).

In our simulation experiments, we report the performance of the *best* robust fluid policy, denoted by RFP, which corresponds to the best value of $\Gamma > 0$ found by doing several simulation runs. Notice that when $\Gamma = 0$, the robust fluid model reduces to the classical one. In this case, we denote

the fluid policy with FP. In order to evaluate the efficiency of our proposed approach, we calculate the percentage distance of the robust fluid policy with the best other policies in the literature and the percentage distance of the robust fluid policy with the classical fluid policy. In particular, we report

$$\mathrm{E}_1 := \frac{\text{Best other-RFP}}{\text{Best other}} \times 100\%, \quad \mathrm{E}_2 := \frac{\text{FP-RFP}}{\text{FP}} \times 100\%.$$

We first consider the criss-cross network of Figure 1, with three classes and two servers. In order to examine the effect of traffic conditions on the performance of the robust policy, we consider various traffic conditions as in [36] and list them in Table II, where the following abbreviations are used for the traffic conditions: I.L. (imbalanced light), B.L. (balanced light), I.M. (imbalanced medium), B.M. (balanced medium), I.H. (imbalanced heavy), and B.H. (balanced heavy). In this table, ρ_1 and ρ_2 are the total traffic intensities at servers 1 and 2, respectively, i.e., $\rho_1 := \lambda_1/\mu_1 + \lambda_2/\mu_2$ and $\rho_2 := \lambda_1/\mu_3$.

TABLE I
NUMERICAL RESULTS FOR THE CRISS-CROSS NETWORK OF FIGURE 1

	DP	OTP	Thr.	FP	RFP	E_1	E_2
I.L.	0.671	0.678	0.679	0.678	0.677	-1.030	0.094
B.L.	0.843	0.856	0.857	0.857	0.855	-1.450	0.305
I.M.	2.084	2.117	2.129	2.162	2.133	-2.392	1.315
B.M.	2.829	2.895	2.805	2.965	2.920	-3.217	1.517
I.H.	9.970	10.13	10.15	10.398	10.096	-1.265	2.90
B.H.	–	15.5	15.5	18.430	15.585	-2.111	12.780

TABLE II
PARAMETERS FOR THE TRAFFIC CONDITIONS OF TABLE I

	λ_1	λ_2	μ_1	μ_2	μ_3	ρ_1	ρ_2
I.L.	0.3	0.3	2	2	1.5	0.3	0.2
B.L.	0.3	0.3	2	2	1	0.3	0.3
I.M.	0.6	0.6	2	2	1.5	0.6	0.4
B.M.	0.6	0.6	2	2	1	0.6	0.6
I.H.	0.9	0.9	2	2	1.5	0.9	0.6
B.H.	0.9	0.9	2	2	1	0.9	0.9

In Table I, we report the performance of the different methods for the data shown in Table II. In the second column, we list the optimal performance obtained via dynamic programming, denoted by DP. We notice that DP is computationally intractable for the heavy traffic case (B.H.). In the third column, we report the performance of an optimized target-pursuing policy proposed in [36], denoted by OTP. In the fourth column, we list the performance of a threshold policy proposed in [22], denoted by Thr. This policy gives priority to jobs of class 1 at server 1 if the number of jobs at server 2 is below some threshold; otherwise gives priority to jobs of class 2. The results listed in the fourth column are for the best such policy (i.e., optimized over the threshold). In the fifth and sixth columns, we list the performance of FP and RFP, respectively. Finally, in the last two columns, we report E_1 and E_2.

Here are our observations from Table I. The robust policy performs better as the traffic intensity increases. More precisely, RFP performs a little bit better than FP from light to moderate traffic scenarios, and significantly better under the heavy traffic cases (in particular B.H.). In this case, we are within 2.1% of the threshold policy, which is conjectured to be asymptotically optimal in heavy traffic [22], and we outperform by more than 12.7% the fluid policy. Notice that the threshold value in the threshold policy can be interpreted as a safety stock protecting server 2 from starvation. Since RFP performs close to the threshold policy in heavy traffic, it is implied that the uncertainty incorporated in the fluid model leads to maintaining some appropriate safety stock for server 2; FP, on the other hand, is greedy and does not do that, leading to worse performance.

TABLE III
NUMERICAL RESULTS FOR THE CRISS-CROSS NETWORK OF FIGURE 1 UNDER THE TRAFFIC CASE B.H. AND DIFFERENT COEFFICIENTS OF VARIATION

CoV	Thr.	FP	RFP	E_1	E_2
1	15.370	18.430	15.585	-2.111	12.780
1.1	27.692	38.113	28.920	-7.695	20.512
1.2	37.992	48.076	37.979	-2.874	20.880
1.3	46.800	56.820	48.081	-0.777	23.007
1.4	58.213	75.980	57.627	-0.686	25.339

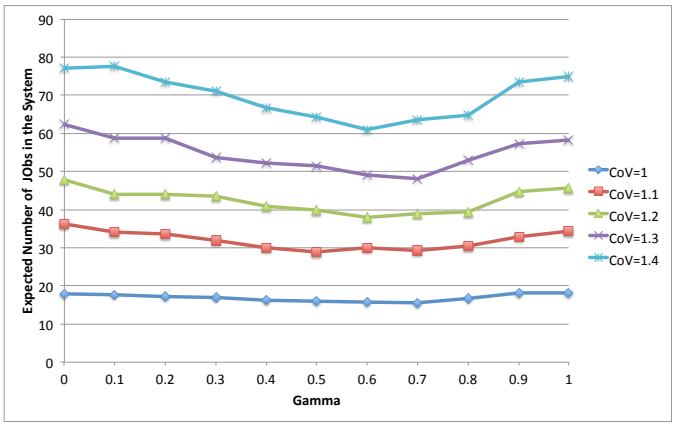
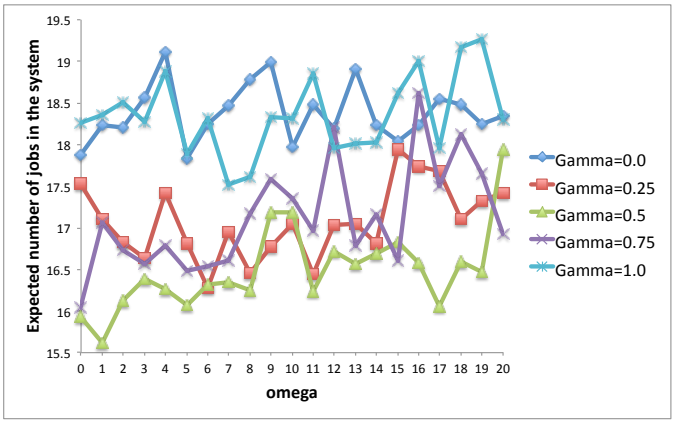


Fig. 2. Effect of Γ. Results for the criss-cross network of Figure 1 under the traffic case B.H. for different coefficients of variation and values of Γ.

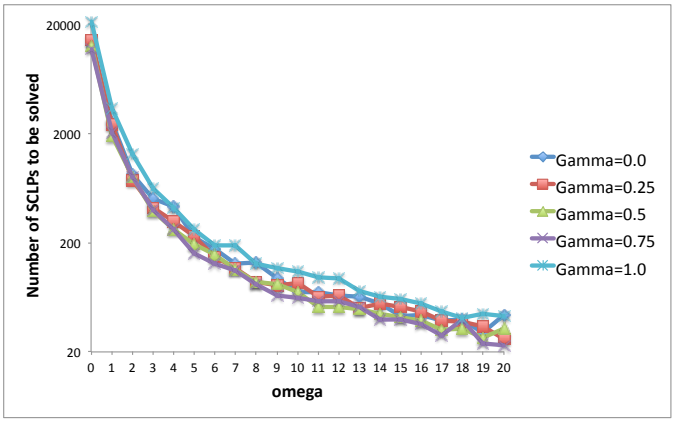
In order to test the impact of the distribution of arrival rates and service times, we simulate the external arrivals and the service times under the hyper-exponential distribution with different coefficients of variation. Tables III compares the performance of the robust policy to the fluid policy and the threshold policy under the heavy traffic case B.H. when the Coefficient of Variation (CoV) is 1.0, 1.1, 1.2, 1.3, and 1.4. We observe that the robust fluid policy performs as well as the best threshold policy and both significantly outperform the fluid policy. On average, we are within 2.33% of the threshold policy and outperform by more than 3.14% the fluid policy.

To examine the effect of Γ, in Figure 2 we report the performance of the robust policy for different values of Γ and different coefficients of variation under the heavy traffic case B.H. When CoV is close to 1.0, the performance of the robust policy is rather insensitive with respect to the parameter Γ and this makes intuitive sense. However, for larger values of CoV, injecting more uncertainty into the fluid control problem can lead to non-negligible performance improvements (about 20% in some instances) compared to the performance under $\Gamma = 0$.

In order to find out how many times we have to solve



(a) Effect of ω on the performance of RFP.



(b) Effect of ω on reducing the number of times we solve Problem (11). Note that the y-axis is in a logarithmic scale.

Fig. 3. Effect of ω. Results for the criss-cross network of Figure 1 under the traffic case B.H. and different values of Γ and Ω.

Problem (11) and how the parameter ω helps to reduce these times, we report in Figure 3 the performance of the RFP and the number of times that Problem (11) is solved for different values of Γ and ω under the heavy traffic case B.H. Here, the number of arrivals is set to $1,000,000$. We observe that the number of times that Problem (11) is solved dramatically decreases as ω increases, taking values $\omega = 0, 1, \ldots, 20$, while the performance of the RFP is not too sensitive with respect to ω. Moreover, as Figure 3 (b) highlights, the number of SCLPs we need to solve is not very sensitive to the parameter Γ which regulates the amount of uncertainty injected into the fluid model.

The second example we consider is a network with six classes and two servers as shown in Figure 4. Jobs of class 1 arrive according a Poisson process with a rate λ_1 and they visit servers 1,2,1,2, in that order, forming classes 1,2,3, and 4, respectively, and then exit the system. Jobs of class 2 arrive according to a Poisson process with a rate λ_2 and then visit servers 1 and 2, forming classes 5, and 6, respectively, and then exit the system. Servers 1 and 2 have exponentially distributed service times with rates μ_1 and μ_2, respectively.

In Table IV, we compare the performance of our robust policy with other methods for different traffic conditions as listed in Table V, where the same notation and abbreviations

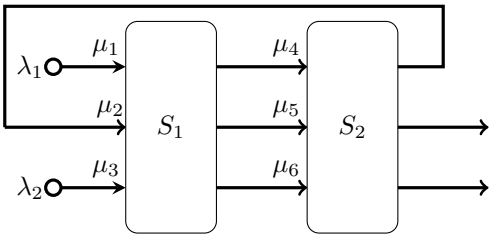


Fig. 4. A six-class network.

are used as in Table I. These parameters are taken from [36]. Both RFP and FP perform equally well from light to moderate traffic scenarios, but in heavy traffic, RFP significantly outperforms FP.

TABLE IV
NUMERICAL RESULTS FOR THE SIX-CLASS NETWORK OF FIGURE 4

	DP	OTP	FP	RFP	E_1	E_2
I.L.	0.663	0.671	0.75	0.72	-11.76	1.984
B.L.	0.798	0.803	0.923	0.912	-14.286	1.192
I.M.	1.966	2.01	2.301	2.21	-12.410	3.955
B.M.	2.56	2.59	3.024	2.952	-15.312	3.381
I.H.	–	8.32	9.435	8.926	-7.284	5.401
B.H.	–	13.6	15.670	14.081	-3.537	10.140

TABLE V
PARAMETERS FOR THE TRAFFIC CONDITIONS OF TABLE IV

	λ_1	λ_2	μ_1	μ_2	μ_3	μ_4	μ_5	μ_6
I.L.	3/140	3/140	1/8	1/2	1/4	1/4	3/14	2/3
B.L.	3/140	3/140	1/8	1/2	1/4	1/6	1/7	1
I.M.	6/140	6/140	1/8	1/2	1/4	1/4	3/14	2/3
B.M.	6/140	6/140	1/8	1/2	1/4	1/6	1/7	1
I.H.	9/140	9/140	1/8	1/2	1/4	1/4	3/14	2/3
B.H.	9/140	9/140	1/8	1/2	1/4	1/6	1/7	1

In the next two examples, we are interested to see how well the robust policy performs as the size of the network increases. We consider an extension of the six-class network in Figure 4 to a network with m servers as shown in Figure 5. There are $3 \cdot m$ classes of jobs in total and only classes 1 and 3 have external arrivals according to a Poisson process with a rate λ_1 and λ_2, respectively. We used the data in Table IV for the B.H. case. In particular, we used $\lambda_1 = \lambda_2 = 9/140$ and the service times for the odd servers $S_1, S_3, \ldots, S_{2\lfloor m/2 \rfloor + 1}$ are the same as the service times for server 1, while the service times for the even servers $S_2, S_4, \ldots, S_{2\lfloor m/2 \rfloor}$ are the same as the service times for server 2 in the six-class network. Thus, the total traffic intensity of each server is 0.9.

Table VI compares the performance of the proposed policy with other heuristic methods for $m = 2, \ldots, 10$. In this table, LBFS refers to the last-buffer first-serve policy, where a priority at a server is given to the class with highest index. FCFS refers to the first-come first-serve policy, where a priority at a server is given to jobs in order of arrival, and the $c\mu$-rule gives priority to the class i with highest $c_i\mu_i$.

We finally consider a reentrant network with m servers as shown in Figure 6. Each server processes 3 classes of jobs, and thus, there are $3 \cdot m$ classes in total. Only class 1 has external arrivals according to a Poisson process with a rate

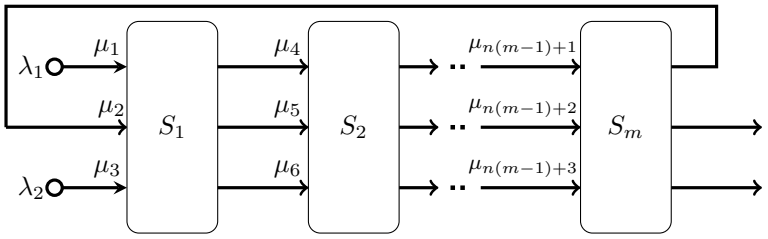


Fig. 5. An extension of the six-class network.

TABLE VI
NUMERICAL RESULTS FOR THE REENTRANT FEED-FORWARD NETWORK OF FIGURE 5

m	$c\mu$-rule	LBFS	FCFS	FP	RFP	E_1	E_2
2	18.296	15.749	40.173	15.422	15.286	3.053	1.001
3	25.425	25.257	71.518	26.140	24.917	1.350	4.673
4	35.721	34;660	114.860	38.085	36.857	-6.340	3.225
5	43.314	45.110	157.556	45.962	43.628	-0.725	5.078
6	53.801	55.724	203.418	56.857	52.980	1.527	6.818
7	60.743	65.980	251.657	64.713	59.051	2.786	8.749

λ_1. Service times are exponentially distributed with rate μ_i for class i jobs.

Table VII compares the performance of the robust policy with FP, LBFS, FCFS, and $c\mu$ policies for $m = 2, \ldots, 7$. Here, the total traffic intensity of each odd server is 0.896 and the total traffic intensity of each even server is 0.8625.

TABLE VII
NUMERICAL RESULTS FOR THE REENTRANT NETWORK OF FIGURE 6 WITH 3 CLASSES PER EACH SERVER

m	$c\mu$-rule	LBFS	FCFS	FP	RFP	E_1	E_2
2	18.199	15.911	16.341	15.422	15.663	10.085	8.663
3	26.937	30.001	27.154	25.955	24.015	10.82	7.450
4	34.934	40.167	35.633	32.014	29.925	21.241	6.525
5	42.387	57.056	47.969	40.113	36.901	14.359	8.009
6	51.945	66.042	56.996	48.781	44.261	12.942	9.265
7	59.958	87.136	71.002	54.711	48.418	14.792	11.502

We next summarize the major conclusions from our simulation study.

1. In the criss-cross network of Figure 1, the performance of our robust fluid policy is comparable to the performance of the threshold policy proposed by Harrison and Wein [22]. Moreover, the relative difference between the performance of the robust policy and the fluid policy increases as the coefficient of variation increases.
2. In both the criss-cross network of Figure 1 and the six-class network of Figure 4, the robust fluid policy outperforms the fluid policy and the efficacy of the robust policy increases with the traffic intensity.
3. In the reentrant feed-forward network of Figure 5, the performance of the robust fluid policy is close to the performance of the best other heuristic policies and is better than the performance of the fluid policy as the number of servers increases.
4. In the reentrant network of Figure 6, the performance of our robust fluid policy outperforms the performance of the heuristic ones as well as the fluid policy, and the efficacy of the robust fluid policy seems to be stable as the number of servers increases.
5. The performance of the robust policy is not very sensitive with respect to the parameter Γ for systems with low coefficient of variation (close to 1). When though the coefficient of variation is larger, accommodating uncertainty in the fluid control problem can lead to performance improvements.
6. Finally, the number of times that Problem (11) is required to be solved dramatically decreases as the parameter ω increases, while the performance of the robust fluid policy is not too sensitive with respect to ω.

VI. CONCLUSIONS

We presented a tractable approach to address uncertainty in multiclass processing networks. Unlike other approaches that make probabilistic assumptions, the proposed approach treats the uncertainty in a deterministic manner using the framework of robust optimization. It relies on modeling the fluid control problem as an SCLP and characterizing its robust counterpart. We showed that the robust problem formulation still remains within the class of SCLPs, and thus, preserves the computational complexity of the fluid control problem.

We also presented a way of translating the optimal controls from the robust fluid model to the stochastic network using ideas from model predictive control. Admittedly, we have not established stability of the class of robust fluid policies we introduced. This remains an open research question.

As our numerical results indicate, our approach leads to effective scheduling policies that perform closely against the optimal policy in small enough instances where the optimal can be computed. In other instances, where near-optimal policies can be derived in certain limiting regimes (e.g., policies based on heavy-traffic analysis), our policy performs comparable to such policies even in the traffic conditions that favor the alternative. More interestingly, in large enough problem instances where neither the optimal nor near-optimal alternatives exist, our policy clearly outperforms generic alternatives. The proposed approach scales well and can handle networks with tens of job classes and tens of servers.

REFERENCES

[1] E. J. Anderson. *A Continuous Model for Job-Shop Scheduling*. PhD thesis, University of Cambridge, 1978.
[2] E. J. Anderson, P. Nash, and A. F. Perold. Some properties of a class of continuous linear programs. *SIAM Journal on Control and Optimization*, 21:258–265, 1983.

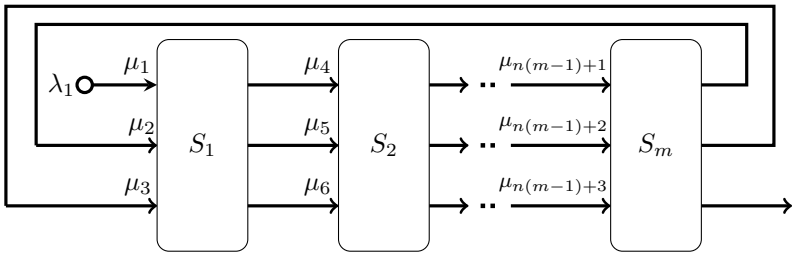


Fig. 6. A reentrant network.

[3] F. Avram, D. Bertsimas, and M. Ricard. An optimal control approach to optimization of multi-class queueing networks. In F. Kelly and R. Williams, editors, *Volume 71 of IMA volumes in Mathematics and its Applications*, pages 199–234, New York, 1995. Springer-Verlag.

[4] N. Bäuerle. Asymptotic optimality of tracking policies in stochastic networks. *Annals of Applied Probability*, pages 1065–1083, 2000.

[5] N. Bäuerle. Optimal control of queueing networks: an approach via fluid models. *Advances in Applied Probability*, 34(2):313–328, 2002.

[6] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust optimization*. Princeton University Press, 2009.

[7] D. Bertsimas, D. B. Brown, and C. Caramanis. Theory and applications of robust optimization. *SIAM Review*, 53(3):464–501, August 2011.

[8] D. Bertsimas, I. C. Paschalidis, and J. N. Tsitsiklis. Optimization of multiclass queueing networks: Polyhedral and nonlinear characterizations of achievable performance. *The Annals of Applied Probability*, pages 43–75, 1994.

[9] D. Bertsimas, I. Ch. Paschalidis, and J. N. Tsitsiklis. Branching bandits and Klimov's problem: Achievable region and side constraints. *IEEE Trans. Autom. Contr.*, 40(12):2063–2075, 1995.

[10] J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer-Verlag, 1997.

[11] C. G. Cassandras, Y. Wardi, B. Melamed, G. Sun, and C. G. Panayiotou. Perturbation analysis for online control and optimization of stochastic fluid models. *Automatic Control, IEEE Transactions on*, 47(8):1234–1248, 2002.

[12] H. Chen and A. Mandelbaum. Discrete flow networks: bottleneck analysis and fluid approximations. *Mathematics of Operations Research*, 16(2):408–446, 1991.

[13] M. Chen, I.-K. Cho, and S. P. Meyn. Reliability by design in distributed power transmission networks. *Automatica*, 42(8):1267–1281, 2006.

[14] M. Chen, R. Dubrawski, and S. P. Meyn. Management of demand-driven production systems. *IEEE Transactions on Automatic Control*, 49(5):686–698, 2004.

[15] J. G. Dai. On positive Harris recurrence of multiclass queueing networks: A unified approach via fluid limit models. *Annals of Applied Probability*, 5:49–77, 1995.

[16] J. G Dai and S. P Meyn. Stability and convergence of moments for multiclass queueing networks via fluid limit models. *IEEE Transactions on Automatic Control*, 40(11):1889–1904, 1995.

[17] J. G. Dai and G. Weiss. Stability and instability of fluid models for reentrant lines. *Mathematics of Operations Research*, 21(1):115–134, 1996.

[18] L. Fleischer and J. Sethuraman. Efficient algorithms for separated continuous linear programs: The multi-commodity fow problem with holding costs and extensions. *Mathematics of Operations Research*, 30:916–938, 2005.

[19] C. E. Garcia, D. M. Prett, and M. Morari. Model predictive control: theory and practice–a survey. *Automatica*, 25(3):335–348, 1989.

[20] J. M. Harrison. Brownian models of queueing networks with heterogeneous customer populations. In *Stochastic differential systems, stochastic control theory and applications*, pages 147–186. Springer, 1988.

[21] J. M. Harrison. The BIGSTEP approach to flow management in stochastic processing networks. *Stochastic Networks: Theory and Applications*, 4:147–186, 1996.

[22] J. M. Harrison and L. M. Wein. Scheduling networks of queues; heavy traffic analysis of a two-station closed network. *Operations Research*, 38(2):1052–1064, 1990.

[23] G.P. Klimov. Time-sharing service systems I. *Theory of Probability and its Applications*, XIX(3), 1974.

[24] B. Klinz and G. J. Woeginger. Minimum-cost dynamic flows: The series-parallel case. *Networks*, 43:153–162, 2004.

[25] S. Kumar and P. R Kumar. Performance bounds for queueing networks and scheduling policies. *IEEE Transactions on Automatic Control*, 39(8):1600–1611, 1994.

[26] M. Laumanns and E. Lefeber. Robust optimal control of material flows in demand-driven supply networks. *Physica A: Statistical Mechanics and its Applications*, 363(1):24–31, 2006.

[27] C. N. Laws and G. M. Louth. Dynamic scheduling of a four-station queueing network. *Probability in the Engineering and Informational Sciences*, 4(01):131–156, 1990.

[28] E. Lefeber, S. Lammer, and J. E. Rooda. Optimal control of a determinstic multiclass queueing system by serving several queues simultaneously. Technical report, SE Technical Report, Systems Engineering Group, The Department of Mechanical Engineering, Eindhoven University of Technology, 2008.

[29] X. Luo and D. Bertsimas. A new algorithm for state-constrained separated continuous linear programs. *SIAM Journal on Control and Optimization*, 37:177–210, 1998.

[30] C. Maglaras. Dynamic scheduling in multiclass queueing networks: Stability under discrete-review policies. *Queueing Systems*, 31(3-4):171–206, 1999.

[31] C. Maglaras. Discrete-review policies for scheduling stochastic networks: Trajectory tracking and fluid-scale asymptotic optimality. *Annals of Applied Probability*, pages 897–929, 2000.

[32] S. Meyn. Stability and optimization of queueing networks and their fluid models. *Lectures in applied mathematics-American Mathematical Society*, 33:175–200, 1997.

[33] S. P. Meyn. Sequencing and routing in multiclass queueing networks part II: Workload relaxations. *SIAM Journal on Control and Optimization*, 42(1):178–217, 2003.

[34] S. P Meyn. Workload models for stochastic networks: Value functions and performance evaluation. *IEEE Transactions on Automatic Control*, 50(8):1106–1122, 2005.

[35] S. P. Meyn. *Control techniques for complex networks*. Cambridge University Press, 2008.

[36] I. C. Paschalidis, C. Su, and M. C. Caramanis. Target-pursuing scheduling and routing policies for multiclass queueing networks. *IEEE Transactions on Automatic Control*, 49(10):1709–1722, 2004.

[37] A. B. Philpott and M. Craddock. An adaptive discretization algorithm for a class of continuous network programs. *Networks*, 26:1–11, 1995.

[38] A. Piunovskiy. Controlled jump Markov processes with local transitions and their fluid approximation. *WSEAS Transactions on Systems and Control*, 4(8):399–412, 2009.

[39] M. C. Pullan. An algorithm for a class of continuous linear programs. *SIAM Journal on Control and Optimization*, 31:1558–1577, 1993.

[40] M. C. Pullan. Forms of optimal solutions for separated continuous linear programs. *SIAM Journal on Control and Optimization*, 33:1952–1977, 1995.

[41] M. C. Pullan. A duality theory for separated continuous linear programs. *SIAM Journal on Control and Optimization*, 34:931–965, 1996.

[42] M. C. Pullan. A study of general dynamic network programs with arc time-delays. *SIAM Journal on Optimization*, 7:889–912, 1997.

[43] M. C. Pullan. Convergence of a general class of algorithms for separated continuous linear programs. *SIAM Journal on Optimization*, 10:722–731, 2000.

[44] M. C. Pullan. An extended algorithm for separated continuous linear programs. *Mathematical Programming*, 93:415–451, 2002.

[45] A. N. Rybko and A. L. Stolyar. Ergodicity of stochastic processes describing the operation of open queueing networks. *Problems of Information Transmission*, 38:3–26, 1992.

[46] A. Shapiro, D. Dentcheva, and A. Ruszczynski. *Lectures on Stochastic Programming: Modeling and Theory*. SIAM, Philadelphia, 1994.

[47] A. L. Stolyar. On the stability of multiclass queueing networks: a relaxed sufficient condition via limiting fluid processes. *Markov Processes and Related Fields*, 1(4):491–512, 1995.
[48] L. M. Taylor and R. J. Williams. Existence and uniqueness of semimartingale reflecting Brownian motions in an orthant. *Probability Theory and Related Fields*, 96(3):283–317, 1993.
[49] J. S. H. van Leeuwaarden, E. Lefeber, Y. Nazarathy, and J. E. Rooda. Model predictive control for the acquisition queue and related queueing networks. In *Proceedings of the 5th International Conference on Queueing Theory and Network Applications*, pages 193–200. ACM, 2010.
[50] L. M. Wein. Optimal control of a two-station Brownian network. *Mathematics of Operations Research*, 15:215–242, 1990.
[51] L. M. Wein. Scheduling networks of queues; heavy traffic analysis of a two station network with controllable inputs. *Operations Research*, 38:1065–1078, 1990.
[52] G. Weiss. A simplex based algorithm to solve separated continuous linear programs. *Mathematical Programming*, 115:151–198, 2008.
[53] R. J Williams. Semimartingale reflecting Brownian motions in the orthant. *IMA Volumes in Mathematics and its Applications*, 71:125–125, 1995.

Dimitris Bertsimas received the M.S. and Ph.D. degrees in Applied Mathematics and Operations Research from the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 1987 and 1988 respectively.

He has been with the MIT faculty since 1988 and he is currently the Boeing Professor of Operations Research and the co-director of the Operations Research Center. His research interests include optimization, statistics and applied probability and their applications in health care, finance, operations management and transportation. He has co-authored more than 150 scientific papers and three graduate level textbooks. He is currently department editor in Optimization for Management Science and former area editor of Operations Research in Financial Engineering. He has supervised 53 doctoral students and he is currently supervising 15 others. He is a member of the National Academy of Engineering, and he has received numerous research awards including the Morse prize (2013), the Pierskalla award (2013), the Farkas prize (2008), the Erlang prize (1996), the SIAM prize in optimization (1996), the Bodossaki prize (1998) and the Presidential Young Investigator award (1991-1996).

Ebrahim Nasrabadi received the M.S. and Ph.D. degrees in Industrial Engineering and Mathematics from Sharif University, Iran, and Technical University of Berlin, Germany, in 2003 and 2009, respectively.

He was a Postdoctoral Associate at the Sloan School of Management, Massachusetts Institute of Technology, Cambridge, MA from September 2010 to June 2013, and in the Department of Electrical and Computer Engineering at Boston University from July 2012 to June 2013. His primary areas of research include optimization and decision making under uncertainty and their applications in supply chain management and inventory planning and control.

Ioannis Ch. Paschalidis (M'96–SM'06–F'14) received the M.S. and Ph.D. degrees both in electrical engineering and computer science from the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 1993 and 1996, respectively.

In September 1996 he joined Boston University where he has been ever since. He is a Professor and Distinguished Faculty Fellow at Boston University with appointments in the Department of Electrical and Computer Engineering, the Division of Systems Engineering, and the Department of Biomedical Engineering. He is the Director of the Center for Information and Systems Engineering (CISE). He has held visiting appointments with MIT and Columbia University, New York, NY, USA. His current research interests lie in the fields of systems and control, networking, applied probability, optimization, operations research, computational biology, and medical informatics.