Queueing Models and
Service Management
Vol. 4, No. 1, page 31-53, 2021

QMSM
© PU 2018

A Stream-Processing Server with an Internal and an External Queue

Tom Cooper, Paul Ezhilchelvan, and Isi Mitrani

School of Computing, Newcastle University
NE4 5TG, United Kingdom

(Received June 2020 ; accepted October 2020)

Abstract: A stream-processing server model consisting of an external queue and an internal queue, with instantaneous or non-instantaneous transfers between the two, is analysed in the steady state. Jobs are collected into batches of fixed size prior to being transferred, but there is also a timer mechanism that may preempt such a collection. Exact and approximate solutions are obtained for both zero and non-zero transfer times. Those solutions implemented and are used in order to evaluate the trade-offs between holding costs and transfer costs. The results of several numerical experiments are presented.

Keywords: Batch transfers, cost minimization, tandem queues, two dimensional Markov processes.

1. Introduction

Stream-processing systems are designed to handle large-scale applications where continuously arriving data items are streamed into a server, or a network of servers, that evaluate user specified queries in real-time [11]. Such applications include market feed processing and eTrading in stock exchanges (e.g., OPRA – `https://www.opraplan.com/`), network access monitoring for denial of service attack patterns, fraud detection in financial services and sensor-based monitoring and automation of industrial facilities.

For a concrete example of a practical application, consider the context of Twitter wherein many thousands of tweets (small messages) are being generated by users at any moment. These tweets will constitute the stream of incoming data items to be processed by servers for analytic and statistic collection purposes. Among the metrics that are computed are the number of times a given tweet is being re-tweeted, the number of tweets referring to a given topic (or #hashtag), the most highly trending topics etc., which are continually being updated in real time and are of importance to Twitter's advertising customers. Apache Storm [13] is Twitter's first generation stream processing system and is also the one considered here. Heron [7] and Apache Flink [2] are examples of other well-known stream processing systems.

The data items, or 'jobs', arrive at a stream-processing server at the rate of many hundreds per second, and although each of them typically requires only a small amount of processing, they need to be sent to the processor with a minimum of delay. In such circumstances, the communication bandwidth between a producer of jobs and the processor that executes them becomes a critical resource. Considerable thought has gone into designing

*Corresponding author
Email : isi.mitrani]@ncl.ac.uk

servers which maximize that bandwidth; e.g., see Thompson *et al.* [12].

Existing designs for stream-processing servers contain an 'external queue', where jobs are stored, and an 'internal queue', where jobs are directly accessed and executed by the processor (in [12], it is convincingly argued that a good way to implement the internal queue is by means of a ring buffer). Such a design is used, for example, in Apache Storm.

Transfers from the external queue to the internal queue tend to incur costs. In order to minimize those costs, and also to make the transfers more efficient, jobs are grouped into batches of fixed size prior to being transferred. As it is undesirable that the processor remains idle while there are jobs present, there may also be a mechanism for transferring incomplete batches to the inner queue when the latter is empty.

We construct, analyze and solve a stochastic model of such a stream-processing server. The aim is to evaluate the trade-off between the number of jobs present (or the job response time) and the number of transfers from the external to the internal queue. One possibility is to treat the cost of a transfer as monetary (e.g., energy cost), while the transfer itself is instantaneous. Alternatively, it is possible that transfers are non-instantaneous and impose a lock on the inner queue. In that case, jobs would continue to arrive but the processor would be blocked for the duration of the transfer. Both scenarios are modeled.

As far as we have been able to determine, this interaction between the internal and external queues has not been analyzed before. Existing studies of stream-processing systems have used various approximations based on models involving a single queue. Thus, De Matteis and Mencagli [3], and Lohrmann *et al.* [8], have modelled a stream-processing station as a G/G/1 queue. When the system is heavily loaded, the average response time is approximated by applying Kingman's result [6]. This requires that the variances of the interarrival intervals and the service times are given or can be estimated. Composing these G/G/1 nodes into a network then necessitates another level of approximation.

Fu *et al.* [4], and Vakilinia *et al.* [14] have used an M/M/c queue and a G/M/c queue, respectively, to approximate a replicated stream-processing node. In [14], the authors also looked at the possibility of using G/G/c nodes, but decided instead to replace the approximation by an upper bound.

None of these studies have examined the batching of jobs, or the cost of transfers to the internal queue. Moreover, the accuracy of the chosen approximation has received very little attention.

Our model has a distant resemblance to two queues in tandem, with a constrained passage of jobs from the first to the second. There is quite a large body of literature on tandem queues (see, for example, Balsamo *et al.* [1], and Perros [10]). However, there do not appear to be any results that would apply to the present case.

The model with instantaneous transfers, and some of its properties, are introduced in section 2. The exact solution of that model is described in section 3, while section 4 presents an efficient and arbitrarily accurate approximate solution. Section 5 deals with the exact and approximate solution of the model with non-zero transfer times. Several numerical experiments evaluating the trade-offs between holding costs and transfer costs are described in section 6. Section 7 presents our conclusions.

A preliminary version of this paper was presented at the 27th IEEE International Symposium on the Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2019). It treated the model with non-zero transfer times in an incomplete fashion, omitting a number of derivations and lacking an implementation of the exact solution. All those gaps are now filled, including new proofs and numerical examples. Also, we have added a note comparing our approximaton with the matrix-analytic approach.

2. The Model with Instantaneous Transfers

Jobs arrive into the system in a Poisson stream at rate λ, and join an external queue where they build up batches of size K. An incoming job that completes such a batch causes all K jobs to be transferred instantaneously to an internal queue, where they are served one at a time by a single processor. The service times are i.i.d random variables distributed exponentially with mean $1/\mu$.

Thus, the content of the external queue never exceeds $K - 1$ jobs, while that of the internal queue is unbounded.

Whenever the internal queue becomes empty, an exponentially distributed timer is started, with mean $1/\tau$. If that timer expires before a batch of K jobs is completed in the external queue, the current content of the external queue is transferred instantaneously to the internal one; if a batch is completed before the timer expires, then the K jobs are transferred and the timer is canceled.

The system structure is illustrated in Figure 1.

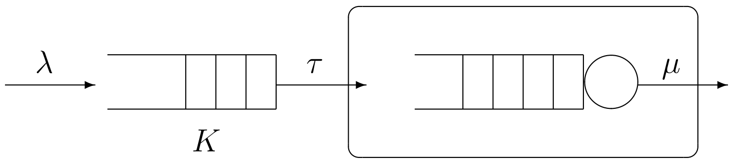

Figure 1. External and internal queues

The batching and timer mechanisms imply that the arrival process at the internal queue is not Poisson (except when $K = 1$). However, since every job arriving into the system eventually joins the internal queue, the arrival rate at that queue is λ. Therefore, the stability condition for the internal queue, and hence for the entire system, is independent of the parameters K and τ, and is simply

$$\lambda < \mu \; . \tag{1}$$

The necessity and sufficiency of that condition will also be established more formally.

For every unit of time that a job is kept in the system (in either queue), a holding cost of c_1 is incurred. Also, at every transfer from the external to the internal queue (regardless of the number of jobs transferred), a cost of c_2 is incurred. Thus, the average cost that the system incurs per unit time in the steady state is

$$C = c_1 L + c_2 T \; , \tag{2}$$

where L is the average total number of jobs present in both queues and T is the average number of transfers per unit time. In order to compute that cost function, we need to determine the joint steady-state distribution of the queue sizes.

The system state at any moment in time is described by the pair of integers (i, j), where i is the number of jobs in the internal queue and j is the number of jobs in the external queue. We wish to determine the steady-state probabilities, $\pi_{i,j}$, of states (i, j), for $i = 0, 1, \ldots$ and $j = 0, 1, \ldots, K-1$. This can be achieved by introducing the generating functions

$$g_j(z) = \sum_{i=0}^{\infty} \pi_{i,j} z^i \quad ; \quad j = 0, 1, \ldots, K-1 \ . \tag{3}$$

The probability, π_n, that there is a total of n jobs in the system, is equal to

$$\pi_n = \sum_{j=0}^{m} \pi_{n-j,j} \ , \tag{4}$$

where $m = \min(n, K-1)$. The generating function of these probabilities is given by

$$G(z) = \sum_{n=0}^{\infty} \pi_n z^n = \sum_{j=0}^{K-1} g_j(z) z^j \ . \tag{5}$$

This function satisfies the normalizing condition

$$G(1) = 1 \ . \tag{6}$$

The two components of the cost function (2), L and T, can be expressed in terms of the above probabilities and generating functions. The average total number of jobs present is obtained from

$$L = G'(1) \ . \tag{7}$$

Since transfers occur either when an arrival finds $K-1$ jobs in the external queue, or when a timer expires with the external queue non-empty and the internal one empty, the average number of transfers per unit time is given by

$$T = \lambda g_{K-1}(1) + \tau \sum_{j=1}^{K-1} \pi_{0,j} \ . \tag{8}$$

The probabilities $\pi_{i,j}$ satisfy a set of balance equations. Since state $(0, 0)$ is left only when a job arrives, and is entered only when a service completes in state $(1, 0)$, we have

$$\lambda \pi_{0,0} = \mu \pi_{1,0} \ . \tag{9}$$

When the inner queue is empty and the outer is non-empty, states are exited via an arrival or a time-out, and are entered via an arrival or a service completion. The corresponding balance equations are

$$(\lambda + \tau)\pi_{0,j} = \lambda \pi_{0,j-1} + \mu \pi_{1,j} \quad ; \quad j = 1, 2, \ldots, K-1 \ . \tag{10}$$

The other 'boundary' states, when the outer queue is empty and the inner is non-empty, are exited via an arrival or a service completion. They are entered via a service completion, a time-out that transfers an incomplete batch, or an arrival that transfers a complete batch. The resulting equations are

$$\begin{aligned}(\lambda+\mu)\pi_{i,0} &= \mu\pi_{i+1,0}+\tau\pi_{0,i} \;\;;\;\; i=1,2,\ldots,K-1 && (11)\\ (\lambda+\mu)\pi_{i,0} &= \mu\pi_{i+1,0}+\lambda\pi_{i-K,K-1} \;\;;\;\; i=K,K+1,\ldots\;. && (12)\end{aligned}$$

For all other states, both exit and entry are via an arrival or a service completion:

$$(\lambda+\mu)\pi_{i,j}=\lambda\pi_{i,j-1}+\mu\pi_{i+1,j} \;\;;\;\; i=1,2,\ldots \;\;;\;\; j=1,2,\ldots,K-1\;. \qquad (13)$$

The next step is to transform the balance equations into functional equations for the generating functions. Multiplying (9), (11) and (12) by z^i, and summing over all $i=0,1,\ldots$, yields, after dividing both sides by μ and a little manipulation, an equation of the form

$$a(z)g_0(z)=\rho z^{K+1}g_{K-1}(z)+b_0(z)\;. \qquad (14)$$

Here, $\rho=\lambda/\mu$ is the offered load at the inner queue, while

$$a(z)=\rho z+z-1\;,$$

and

$$b_0(z)=(z-1)\pi_{0,0}+\sigma z\sum_{j=1}^{K-1}z^j\pi_{0,j}\;,$$

with $\sigma=\tau/\mu$. Similarly, multiplying (10) and (13) by z^i, and summing over all $i=0,1,\ldots$, yields

$$a(z)g_j(z)=\rho z g_{j-1}(z)+b_j(z) \;\;;\;\; j=1,2,\ldots,K-1\;, \qquad (15)$$

where $a(z)$ is the same as before, and

$$b_j(z)=(z-1-\sigma z)\pi_{0,j}\;.$$

Finally, multiplying (14) and (15) by z^j, summing over $j=0,1,\ldots,K-1$ and performing cancellations, leads to a rather simple equation for the generating function of the total number of jobs in the system:

$$(1-\rho z)G(z)=\sum_{j=0}^{K-1}z^j\pi_{0,j}\;. \qquad (16)$$

Note that the parameter τ does not appear explicitly in this equation. However, it affects the probabilities $\pi_{0,j}$.

Setting $z = 1$ in (16), and remembering the normalizing condition (6), gives, as could have been expected,

$$\sum_{j=0}^{K-1} \pi_{0,j} = 1 - \rho \, . \tag{17}$$

Also, differentiating (16) with respect to z and setting $z = 1$ produces an expression for the average total number of jobs in the system:

$$L = \frac{\rho}{1-\rho} + \frac{1}{1-\rho} \sum_{j=1}^{K-1} j\pi_{0,j} \, . \tag{18}$$

So far, the quantities L and T that appear in the cost function (2) have been expressed in terms of the $K-1$ boundary probabilities $\pi_{0,j}$, for $j = 1, 2, \ldots, K-1$, and the value $g_{K-1}(1)$. The next task is to determine these unknowns.

3. Exact Solution

Equations (14) and (15) constitute a set of K simultaneous linear equations for the K generating functions $g_0(z), g_1(z), \ldots, g_{K-1}(z)$. They can be written in matrix and vector form as follows:

$$A(z)\mathbf{g}(z) = \mathbf{b}(z) \, , \tag{19}$$

where $\mathbf{g}(z)$ is the column vector of generating functions, $\mathbf{b}(z)$ is the column vector of free terms, $[b_0(z), b_1(z), \ldots, b_{K-1}(z)]$, and the $K \times K$ matrix $A(z)$ has the form

$$A(z) = \begin{bmatrix} a(z) & & & & & -\rho z^{K+1} \\ -\rho z & a(z) & & & & \\ & -\rho z & a(z) & & & \\ & & & \ddots & & \\ & & & -\rho z & a(z) & \\ & & & & -\rho z & a(z) \end{bmatrix} .$$

The solution of (19) can be expressed as

$$g_j(z) = \frac{D_j(z)}{D(z)} \; ; \; j = 0, 1, \ldots, K-1 \, , \tag{20}$$

where $D(z)$ is the determinant of the matrix $A(z)$, and $D_j(z)$ is the the determinant of the matrix, $A_j(z)$, obtained by replacing column $j+1$ of $A(z)$ with the column $\mathbf{b}(z)$ (the columns are numbered from 1 to K, rather than from 0 to $K-1$). In particular,

$$g_{K-1}(z) = \frac{D_{K-1}(z)}{D(z)} \, , \tag{21}$$

where $D_{K-1}(z)$ is the determinant of the matrix

$$A_{K-1}(z) = \begin{bmatrix} a(z) & & & & & b_0(z) \\ -\rho z & a(z) & & & & b_1(z) \\ & -\rho z & a(z) & & & b_2(z) \\ & & & \ddots & & \\ & & & -\rho z & a(z) & b_{K-2}(z) \\ & & & & -\rho z & b_{K-1}(z) \end{bmatrix} .$$

To determine the denominator in (21), expand $D(z)$ along its first row, where there are only two non-zero terms. This yields a simple expression:

$$D(z) = a(z)^K - (\rho z^2)^K \ . \tag{22}$$

Expanding $D_{K-1}(z)$ along its last column, we obtain

$$D_{K-1}(z) = \sum_{j=0}^{K-1} b_j(z) a(z)^j (\rho z)^{K-1-j} \ . \tag{23}$$

It is quite simple to verify that both $D(z)$ and $D_{K-1}(z)$ vanish at $z = 1$. Hence, the value of $g_{K-1}(1)$ is obtained by applying L'Hospital's rule:

$$g_{K-1}(1) = \frac{D'_{K-1}(1)}{D'(1)} \ . \tag{24}$$

Substituting (22) and (23) into (24), taking derivatives at $z = 1$ and performing simplifications, yields

$$g_{K-1}(1) = \frac{1}{K}\left[1 - \frac{\tau}{\lambda}\sum_{j=1}^{K-1} j\pi_{0,j}\right] \ . \tag{25}$$

It now remains to find the K probabilities $\pi_{0,j}$, for $j = 0, 1, \ldots, K-1$. One relation between them is the normalizing equation (17). To derive additional equations, note that since the power series $g_{K-1}(z)$ converges at $z = 1$, it must converge in the entire interior of the unit disc. Therefore, if the denominator in (21), $D(z)$, is equal to 0 for some z such that $|z| < 1$, then the numerator, $D_{K-1}(z)$, must also be equal to 0 at that point.

The following result provides the required number of additional equations.

Proposition 1. *The polynomial $D(z)$ has exactly $K-1$ distinct zeros, z_1, z_2, …, z_{K-1}, such that $|z_k| < 1$, $k = 1, 2, \ldots, K-1$. If K is even, then one of the zeros is real and the others are complex, forming $(K-2)/2$ complex-conjugate pairs. If K is odd, then all zeros are complex, forming $(K-1)/2$ complex-conjugate pairs.*

The proof of this proposition is in the Appendix. The zeros of $D(z)$ are obtained in closed form.

If z_k is real, then substituting it into the expression (23) for $D_{K-1}(z)$ and equating the latter to 0, provides one equation for the unknown probabilities. If z_k is complex, then equating the real and imaginary parts of $D_{K-1}(z_k)$ to 0 provides two equations. The other member of the complex-conjugate pair would not be used, as it would yield the same two equations.

Thus, Proposition 1 provides $K-1$ additional equations, which, together with (17), enable us to compute the probabilities $\pi_{0,j}$ and hence the cost function (2).

The above solution is easily implementable and is efficient. However, it does not scale well when K increases. We have observed that when K is larger than about 25, the exact solution begins to experience numerical problems. The matrix associated with the set of linear equations provided by the $K-1$ zeros of $D(z)$ becomes ill-conditioned. The solution then ceases to be reliable.

For that reason, it is desirable to develop a different solution that would be accurate and robust, and handle models with large values of K. We propose a simple approximation that satisfies these requirements.

4. Approximate Solution

The idea of the proposed approximation is to truncate the state space and devise an efficient iterative procedure for solving the resulting balance equations. An integer m is chosen appropriately, and it is assumed that all probabilities $\pi_{i,j}$, where $i > m$, are 0. Such an assumption can be justified as follows.

According to (16), the distribution of the total number of jobs in the system, n, has a geometric tail with parameter ρ. Indeed, when the inner queue is not empty, n goes up to $n+1$ and down to $n-1$ with instantaneous rates λ and μ, respectively, i.e. that number behaves like an M/M/1 queue with offered load ρ. Hence, the probabilities $\pi_{i,j}$ decrease with i roughly on the order of ρ^i. The truncation level m can be chosen so that $\rho^m < \epsilon$, for some small ϵ. Since that value can be chosen as small as desired, the approximation based on such a truncation can be considered exact for practical purposes.

The following algorithm implements an efficient solution of the truncated set of balance equations.

1. Start by making initial guesses for the values of $\pi_{0,j}$, $j = 0, 1, \dots, K-1$, and $\pi_{i,K-1}$, $i = 1, \dots, m$. For example, set $\pi_{0,j} = (\lambda/(\lambda+\tau))^j$, and $\pi_{i,K-1} = \pi_{0,K-1}(\lambda/(\lambda+\mu))^i$. Normalize those guesses so that (17) is satisfied.

2. Using equations (12) and (11), and working backwards from $i = m$ down to $i = 1$, compute the probabilities $\pi_{i,0}$. Then (9) yields a new value for $\pi_{0,0}$.

3. Using equations (13), for j increasing from $j = 1$ to $j = K-1$ and i decreasing from $i = m$ down to $i = 1$, compute the probabilities $\pi_{i,j}$. As the j'th row is completed,

(10) is used to compute a new value for $\pi_{0,j}$. By the end of this step, the initial guesses have been modified.

4. Normalize $\pi_{0,j}$ so that (17) is satisfied.

5. Iterate steps 2–4 until two consecutive estimates of the probabilities $\pi_{0,j}$, $j = 0, 1, \dots,$ $K - 1$, are sufficiently close to each other.

6. Compute the performance measures L and T according to (18), (8) and (25).

The above algorithm has the advantage that it does not involve the solution of a set of simultaneous equations. Also, the exact normalization applied at each iteration speeds up convergence. There are no problems with numerical stability, and large values of K are handled easily. Moreover, an implementation that overwrites old probabilities with new ones would need to store only one row (an array of size m) and one column (array of size K) in order to carry out the iterations.

Note. The generator of our Markov process can be represented as an upper block Hessenberg matrix of the M/G/1 type. One could therefore use a matrix-analytic algorithm (see [9]) to determine the invariant measure. Although that is, in principle, an exact solution, in practice it is an approximation because it requires the computation of a certain $(K \times K)$ matrix, G. The latter is usually evaluated iteratively as a solution to a matrix polynomial equation of order K. The $K + 1$ terms of that polynomial involve matrix multiplications whose number varies from 2 to K. Hence, the total number of multplications required to evaluate the polynomal is on the order of $O(K^2)$. Since each multiplication of $(K \times K)$ matrices has numerical complexity on the order of $O(K^3)$, the total complexity of one iteration in the computation of G is on the order of $O(K^5)$. This compares rather unfavourably with our proposed algorithm which, by exploiting the special structure of the process, has a numerical complexity on the order of $O(Km)$ per teration. We obtain high accuracy with values of m not much larger than K.

5. Model with Non-zero Transfer Times

Assume that a transfer from the outer to the inner queue takes an exponentially distributed interval of time with mean $1/\gamma$. During that interval, jobs may continue to arrive and join the outer queue, but the processor is blocked. Any preempted service is eventually resumed from the point of interruption. At the end of the transfer interval, all jobs currently in the outer queue join the inner queue.

The processor blocking means that the condition $\rho < 1$ is no longer sufficient for the stability of the system. To find the new necessary and sufficient condition we argue as follows: The fraction of time required to serve the incoming jobs is ρ; during that time, every K'th arrival causes a transfer interruption which lasts for an average of $1/\gamma$; hence, while the inner queue is not empty, the processor is blocked for a fraction $\rho\lambda/(K\gamma)$ of the time. Consequently, the stability condition is

$$\rho + \frac{\rho\lambda}{K\gamma} < 1 . \tag{26}$$

That condition will also be derived analytically.

The system state is now described by a triple, (i, j, s), where i and j are the numbers of jobs present in the inner and outer queue, and s indicates the state of the processor: $s = 0$ if the processor is blocked (i.e., a transfer is in progress), $s = 1$ if the processor is operative. Since transfers are initiated when the inner queue is empty, or when the size of the outer queue is $K - 1$, the feasible states with $s = 0$ are $(0, j, 0)$, for $j > 0$, and $(i, j, 0)$, for $j > K - 1$. The feasible states with $s = 1$ are $(i, j, 1)$, for $i \geq 0$ and $j = 0, 1, \ldots, K - 1$.

Denote the steady-state probability of $(i, j, 0)$ by $q_{i,j}$, and the steady-state probability of $(i, j, 1)$ by $\pi_{i,j}$. It is also convenient to define the probability, r_n, that there is a total of n jobs in the system *and* there is a transfer in progress. The reason for introducing those probabilities is that if a transfer completes in any state where there are n jobs present, the resulting state would be $(n, 0, 1)$.

For $n = 1, 2, \ldots, K - 1$, $r_n = q_{0,n}$ (the inner queue must then be empty), while for $n \geq K$,

$$r_n = \sum_{i=0}^{n-K} q_{i,n-i} . \tag{27}$$

A state with a total of n jobs present and a transfer in progress is exited when a job arrives or the transfer completes. It is entered when an incoming job finds $n - 1$ jobs present and a transfer in progress, or when a transfer is initiated. The corresponding balance equations are

$$\begin{aligned} (\lambda + \gamma)r_n &= \lambda r_{n-1} + \tau\pi_{0,n} \;\; ; \;\; n = 1, 2, \ldots, K - 1 & (28) \\ (\lambda + \gamma)r_n &= \lambda r_{n-1} + \lambda\pi_{n-K,K-1} \;\; ; \;\; n = K, K + 1, \ldots \;, & (29) \end{aligned}$$

with $r_0 = 0$ (there can be no transfer in an idle system).

Multiplying the above equations by z^n and summing, we obtain an equation for the generating function, $H(z)$, of the probabilities r_n:

$$[\lambda(1 - z) + \gamma]H(z) = \tau \sum_{j=1}^{K-1} \pi_{0,j} z^j + \lambda z^K g_{K-1}(z) , \tag{30}$$

where $g_{K-1}(z)$ is the generating function of the probabilities $\pi_{i,K-1}$ (defined as in (3)).

The average number of transfers per unit time can be obtained from $T = \gamma H(1)$, which reduces to the same expression as (8).

The probabilities of states where the processor is operative satisfy the following balance equations:

$$\begin{aligned}
\lambda\pi_{0,0} &= \mu\pi_{1,0} \ . & (31)\\
(\lambda+\mu)\pi_{i,0} &= \mu\pi_{i+1,0}+\gamma r_i \ ; \ i=1,2,\dots \ , & (32)\\
(\lambda+\tau)\pi_{0,j} &= \lambda\pi_{0,j-1}+\mu\pi_{1,j} \ ; \ j=1,2,\dots,K-1 \ . & (33)\\
(\lambda+\mu)\pi_{i,j} &= \lambda\pi_{i,j-1}+\mu\pi_{i+1,j} \ ; \ i=1,2,\dots \ ; \ j=1,2,\dots,K-1 \ . & (34)
\end{aligned}$$

Multiplying (31) and (32) by z^i and summing over all i yields

$$a(z)g_0(z)=(z-1)\pi_{0,0}+\frac{\gamma z}{\mu}H(z) \ , \tag{35}$$

where $a(z)=\rho z+z-1$. After substitution of (30), this becomes an equation similar to (14):

$$a(z)g_0(z)=\alpha(z)\rho z^{K+1}g_{K-1}(z)+b_0(z) \ , \tag{36}$$

where

$$\alpha(z)=\frac{\gamma}{\lambda(1-z)+\gamma} \ , \tag{37}$$

and

$$b_0(z)=(z-1)\pi_{0,0}+\alpha(z)\sigma z\sum_{j=1}^{K-1}\pi_{0,j}z^j \ , \tag{38}$$

Again, $\sigma=\tau/\mu$.

The equations satisfied by the other generating functions, $g_j(z)$, for $j=1,2,\dots,K-1$, are the same as (15). The generating function, $G(z)$, of the total number of jobs in the system when the processor is operative, is obtained by multiplying (36) and (15) by z^j and summing. This yields, after some manipulations,

$$(1-\rho z)G(z)=\xi\rho\alpha(z)z^{K+1}g_{K-1}(z)+\sum_{j=0}^{K-1}\pi_{0,j}z^j+\xi\sigma\alpha(z)\sum_{j=1}^{K-1}\pi_{0,j}z^j \ , \tag{39}$$

where $\xi=\lambda/\gamma$

The generating function of the unconditional total number of jobs in the system is $G(z)+H(z)$. This can be written as

$$(1-\rho z)[G(z)+H(z)]=\sum_{j=0}^{K-1}\pi_{0,j}z^j+\alpha(z)\left[\xi z^K g_{K-1}(z)+\eta\sum_{j=1}^{K-1}\pi_{0,j}z^j\right] \ , \tag{40}$$

where $\eta=\tau/\gamma$

Setting $z=1$ in the above, we find a new form of the normalizing equation:

$$(1-\rho)=\sum_{j=0}^{K-1}\pi_{0,j}+\xi g_{K-1}(1)+\eta\sum_{j=1}^{K-1}\pi_{0,j} \ . \tag{41}$$

This expression is quite intuitive: the first term in the right-hand side is the fraction of time that the inner queue is empty, with the processor operative, and the other two terms are the fraction of time occupied by transfers (with the processor blocked); the total is the fraction of time that the processor is not serving jobs.

Equations (8) and (41) provide a simple expression for the average number of transfers per unit time:

$$T = \gamma(1-\rho) - \gamma \sum_{j=0}^{K-1} \pi_{0,j} \; . \tag{42}$$

The total average number of jobs in the system, L, is given by $L = G'(1) + H'(1)$, and is obtained by differentiating (40) at $z = 1$. This yields

$$L = \frac{1}{1-\rho}\left[\rho + \frac{\xi}{\gamma}T + (1+\eta)\sum_{j=0}^{K-1} j\pi_{0,j} + \xi[Kg_{K-1}(1) + g'_{K-1}(1)]\right] \; . \tag{43}$$

The cost function is thus expressed in terms of the probabilities $\pi_{0,j}$, and the values of $g_{K-1}(1)$ and $g'_{K-1}(1)$. Those quantities are yet to be determined.

The exact solution of this model proceeds along similar lines to the one in in section 3. As in (21), the generating function $g_{K-1}(z)$ is expressed as the ratio of two determinants

$$g_{K-1}(z) = \frac{D_{K-1}(z)}{D(z)} \; , \tag{44}$$

where $D(z)$ now has the form

$$D(z) = \begin{vmatrix} a(z) & & & & & -\alpha(z)\rho z^{K+1} \\ -\rho z & a(z) & & & & \\ & -\rho z & a(z) & & & \\ & & & \ddots & & \\ & & & -\rho z & a(z) & \\ & & & & -\rho z & a(z) \end{vmatrix} \; , \tag{45}$$

with $a(z) = \rho z + z - 1$ and $\alpha(z)$ given by (37). The determinant $D_{K-1}(z)$ is obtained from $D(z)$ by replacing its last column with the new column vector $[b_0(z), b_1(z), \ldots, b_{K-1}(z)]$, where only $b_0(z)$ has changed. Expanding $D_{K-1}(z)$ along its last column we obtain an expression similar to (23), except that $b_0(z)$ is now given by (38).

The denominator in (44) is given by

$$D(z) = a(z)^K - \alpha(z)(\rho z^2)^K \; . \tag{46}$$

Again, both $D_{K-1}(z)$ and $D(z)$ vanish at $z = 1$, so the value of $g_{K-1}(1)$ is obtained by applying L'Hospital's rule to (44). This eventually leads to

$$g_{K-1}(1) = \frac{\gamma}{K\gamma(1-\rho) - \lambda\rho}\left[\sum_{j=0}^{K-1}\pi_{0,j} + \tau\sum_{j=1}^{K-1}\pi_{0,j}\left(\frac{\rho}{\gamma} - j\frac{1-\rho}{\lambda}\right)\right] \; . \tag{47}$$

We also need to find $g'_{K-1}(1)$. Taking derivatives in (44) gives

$$g'_{K-1}(z) = \frac{D'_{K-1}(z)D(z) - D_{K-1}(z)D'(z)}{D^2(z)} \ . \tag{48}$$

When $z = 1$, the right-hand side of (48) has an indeterminacy of order 2. Hence, two applications of L'Hospital's rule are required in order to resolve it. Differentiating the numerator and the denominator twice at $z = 1$ yields

$$g'_{K-1}(1) = \frac{D''_{K-1}(1)D'(1) - D'_{K-1}(1)D''(1)}{2D'(z)^2} \ . \tag{49}$$

Here,

$$D'(1) = \rho^{K-1}[K(1-\rho) - \xi\rho] \ , \tag{50}$$

and

$$D''(1) = K\rho^{K-2}[(K-1)(1+\rho)^2 - 2\xi\rho(1+\rho) - (4K-2)\rho^2] + 2\xi D'(1) \ . \tag{51}$$

The first and second derivatives of $D_{K-1}(x)$ at $z = 1$ are obtained by differentiating (23), using the new form of $b_0(z)$.

We are now left with K unknowns: the probabilities $\pi_{0,0}$, $\pi_{0,1}$, ..., $\pi_{0,K-1}$. A normalizing equation for those probabilities is obtained by substituting (47) into (41). We get

$$(K + \xi)\sum_{j=0}^{K-1} \pi_{0,j} + \eta \sum_{j=1}^{K-1}(K-j)\pi_{0,j} = K(1-\rho) - \xi\rho \ . \tag{52}$$

This equation proves the necessity of condition (26) for stability of the system. Indeed, if normalizeable steady-state probabilities exist, then the right-hand side of (52) must be positive, which is equivalent to (26).

The sufficiency of the condition is demonstrated by the following result:

Proposition 2. *When the inequality (26) holds, the function $D(z)$ given by (46) has exactly $K-1$ zeros, z_1, z_2, ..., z_{K-1}, such that $|z_k| < 1$, $k = 1, 2, \ldots, K-1$.*

The proof of this proposition is in the appendix. It shows that $K-1$ equations for the unknown probabilities can be obtained by equating the numerator in (44) to 0 at points z_k. Together with (52), these equations determine the remaining K unknowns $\pi_{0,0}$, $\pi_{0,1}$, ..., $\pi_{0,K-1}$.

In an implementation of the exact solution, the zeros z_k have to be computed numerically; we do not have closed-form expressions for them. That computation is not difficult, even though $D(z)$ is not a polynomial. The product $[\lambda(1-z) + \gamma]D(z)$ is a polynomial, and its zeros in the interior of the unit disc coincide with those of $D(z)$. We do not know how many of those zeros are grouped into complex-conjugate pairs and how many are real. A plausible conjecture, supported by several numerical examples, is that the situation is similar to the one described in Proposition 1: when K is even, there is one real zero and $(K-2)/2$ pairs of complex-conjugate ones; when K is odd, there are $(K-1)/2$ pairs of complex-conjugate zeros.

As with the previous model, the exact solution is fine as long as K is reasonably small, but eventually it becomes impractical. Numerical problems begin to emerge when K is larger than about 17. However, a stable, accurate and easily implementable approximate solution similar to the one in section 4 is readily available.

Truncate the state space by assuming that $\pi_{i,j} = 0$ for $i > m$, and $r_n = 0$ for $n > N$, for some m and N. That is, the inner queue does not grow beyond m during operative periods and the total number of jobs does not grow beyond N during transfer periods. Such a truncation can be made arbitrarily accurate. In view of (29), a reasonable choice for N would be $N = m + K$.

The truncated set of balance equations are solved by an iterative algorithm:

1. Start by making initial guesses for the values of $\pi_{0,j}$, $j = 0, 1, \ldots, K-1$, and $\pi_{i,K-1}$, $i = 1, \ldots, m$.

2. Using equations (28) and (29), compute the probabilities r_n, $n = 1, 2, \ldots, N$.

3. Using equations (31) and (32), and working backwards from $i = m$ down to $i = 0$, compute the probabilities $\pi_{i,0}$, including a new value for $\pi_{0,0}$.

4. Using equations (34) and (33), for j increasing from $j = 1$ to $j = K-1$ and i decreasing from $i = m$ down to $i = 0$, compute the probabilities $\pi_{i,j}$, including a new value for $\pi_{0,j}$.

5. Normalize $\pi_{0,j}$ so that (52) is satisfied.

6. Iterate steps 2–5 until two consecutive estimates of the probabilities $\pi_{0,j}$, $j = 0, 1, \ldots,$ $K-1$, are sufficiently close to each other.

7. Compute the performance measures L and T using (43), (42), (47) and (49).

Again, we have an algorithm that does not involve the solution of a set of simultaneous equations. It handles large values of K without problems, and its storage requirements are on the order of $O(m+N)$.

It should be pointed out that, since the size of the external queue is now unbounded, the matrix-analytic solution is not applicable to the model with non-instantaneous transfers.

6. Numerical Results

We have carried out several numerical experiments aimed at examining the behaviour of the system under reasonably realistic loading conditions, with zero and non-zero transfer times. The trade-offs between holding and transfer costs are evaluated, with a view to optimizing the controllable parameters, τ and K.

The first three examples concern a system with instantaneous transfer times. Jobs arrive at the rate of 750 per second, and the average service time is 1 millisecond. The offered load is thus $\rho = 0.75$.

Figure 2 illustrates a system with a batch size of $K = 25$. The unit holding and transfer costs are $c_1 = 1$ and $c_2 = 0.05$, respectively. The cost function C is plotted against the timer rate τ. In this example, the exact solution is compared with a 'cheap' approximation. We know that the approximate solution can be made arbitrarily accurate by choosing a sufficiently high truncation level. Here the aim is to show that even when that level is quite low, the results can be very acceptable. We have chosen $m = 25$, so that $\rho^m \approx 0.001$.

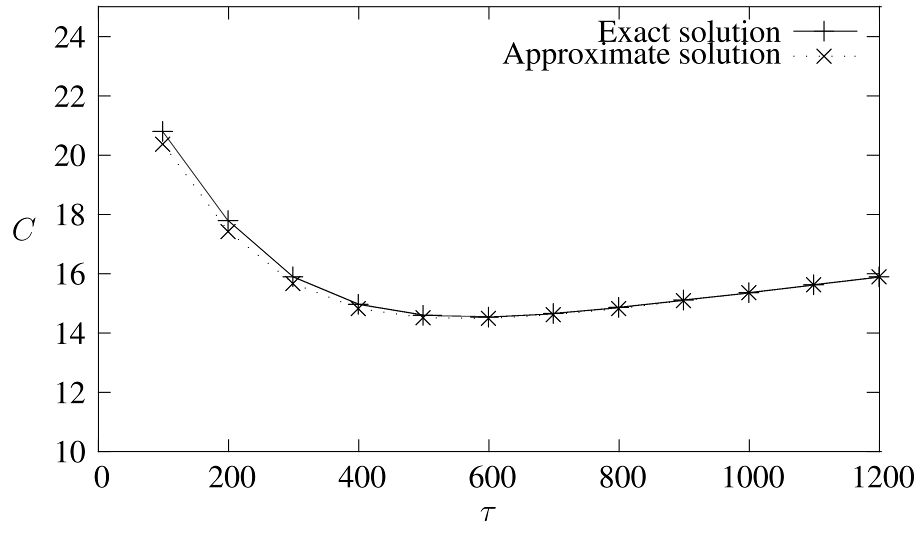

Figure 2. Instantaneous transfers: exact & approximate cost vs. τ
$K = 25$, $\lambda = 750$, $\mu = 1000$, $c_1 = 1$, $c_2 = 0.05$, $m = 25$

The optimal value of τ in this example is around 600. Below that value, the cost function is higher because of higher holding costs; above it, it is higher because of higher transfer costs. At this truncation level, the largest relative error of the approximation is about 2%. If the value of m is increased to 50, the approximate solution plot becomes indistinguishable from the exact solution.

In the second example, the batch size is increased to $K = 100$, which is the value used in the Apache Storm implementation. The arrival and service rates are as before. At this batch size, the only way of obtaining accurate results is to apply the approximate solution with a large truncation level. We have chosen $m = 200$. Given that $\rho^m < 10^{-20}$, the resulting accuracy should be more than adequate.

In Figure 3, the cost function C is plotted against τ for three different unit transfer costs: $c_2 = 0.05$, $c_2 = 0.10$ and $c_2 = 0.15$.

Intuitively, when transfers are more expensive, it is worth trying to delay them for longer. In other words, the optimal timer rate should decrease with c_2. This is indeed what is observed. As c_2 increases from 0.05 to 0.15, the optimal τ decreases from 600 to

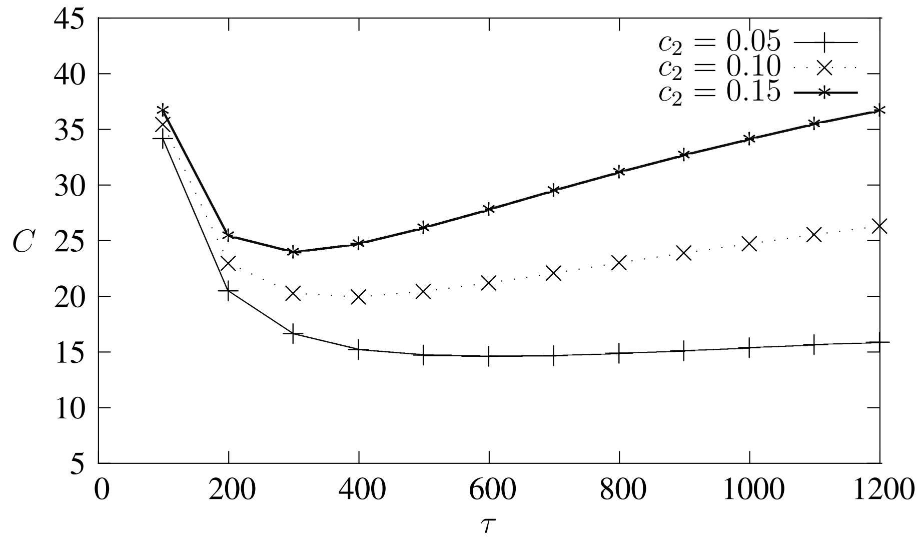

Figure 3. Larger batches: cost vs. τ for different c_2
$K = 100$, $\lambda = 750$, $\mu = 1000$, $c_1 = 1$, $m = 200$

300. In addition, when the unit transfer rate is higher, the cost function is steeper, which means that it is more important to operate close to the optimal regime.

There are also trade-offs associated with the batch size, K. Lowering the value of K reduces holding costs by 'feeding' the inner queue more frequently and hence improving the server utilization. However, the transfer costs are then increased. Moreover, the optimal value of τ depends, in general, on the value of K. One should therefore optimize with respect to τ for different batch sizes, in order to search for the globally otimal pair (K, τ).

This is done in the next experiment, illustrated in Figure 4. The batch size is varied, and the optimal value of τ is found for each value of K. The resulting *minimal* cost, C, is plotted against K, for three different unit transfer costs: $c2 = 0.05$, $c2 = 0.10$ and $c2 = 0.15$.

The results confirm the intuition that when the unit transfer cost is higher, the optimal batch size is higher (so that transfers become less frequent). The globally optimal pairs (K, τ) fot this example are: $(K = 6, \tau = 10)$ for $c_2 = 0.05$, $(K = 8, \tau = 10)$ for $c_2 = 0.10$ and $(K = 12, \tau = 10)$ for $c_2 = 0.15$.

A less intuitive observation is that the minimal cost flattens out beyond the optimal point. In other words, over-estimating the value of K, even by a large amount, would not cause a large increase in costs, as long as the corresponding value of τ is chosen optimally. What happens here is that if the batch size is large and the server is not too heavily loaded, most of the transfers occur when the inner queue is empty. The optimal τ is then large enough to ensure that the server is not kept idle for long, and small enough to prevent too many transfers. For example, the results in Figure 3 show that, when $K = 100$, the minimal costs corresponding to $c_2 = 0.05$, $c_2 = 0.10$ and $c_2 = 0.15$, achieved by $\tau = 600$, $\tau = 400$ and $\tau = 300$, are $C = 14.6$, $C = 20$ and $C = 23$, respectively. These are only a little higher than the corresponding globally minimal costs in Figure 4, $C = 13.3$, $C = 17.4$ and $C = 21$.

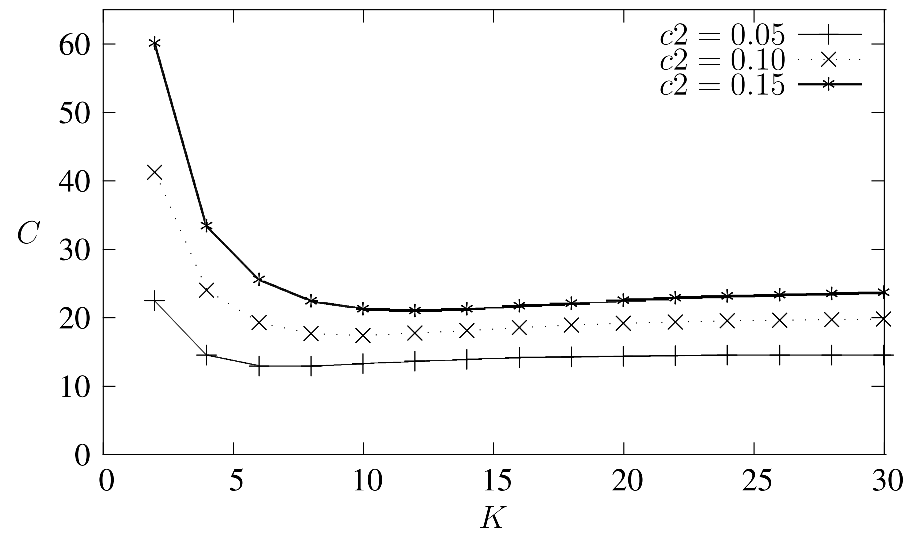

Figure 4. Optimal cost vs. K for different c_2
$\lambda = 750$, $\mu = 1000$, $c_1 = 1$

The remaining experiments concern the model with non-instantaneous transfer times. We first compare the exact and the approximate solutions. The aim is to see how low the level of truncation can be while still achieving good approximations under different loading conditions. The batch size, service rate, timer rate and transfer rate are fixed at $K = 15$, $\mu = 1000$, $\tau = 1000$, $\gamma = 600$. The holding and transfer cost coefficients are $c_1 = 1$ and $c_2 = 0.1$ respectively. The arrival rate is varied between 600 and 870 jobs per second. For this parameter set the stability condition (26) is violated when $\lambda = 941$, which means that the loading is varied between 64% and 92%.

Three versions of the cost function, plotted against λ, are shown in Figure 5. They were produced by the exact solution, the approximate solution with truncation level $m = 15$, and the approximate solution with truncation level $m = 45$, respectively.

We observe that even when the state space is truncated at the batch size, the approximation is quite good. When the truncation is at three times the batch size, the approximate solution becomes almost indistinguishable from the exact solution over the entire range of offered loads.

In Figure 6, the cost cunction is plotted against τ for three different transfer rates. The arrival rate, service rate and holding cost are as before, while the transfer cost is fixed at $c_2 = 0.1$. The batch size is $K = 25$. The approximate solution algorithm was used, but the truncation level was set at $m = 100$, which means that the results are exact for all practical purposes.

The figure shows that when γ increases, the optimal value of τ decreases. This is quite an intuitive observation. If transfers are fast, delaying them can reduce costs, but if they are slow, it is better to initiate them quickly. In fact, when $\gamma = 300$ (i.e. an average transfer is longer than three average service times), the optimal timer rate appears to be $\tau = \infty$; a transfer to an empty internal queue should be initiated as soon as a new job arrives.

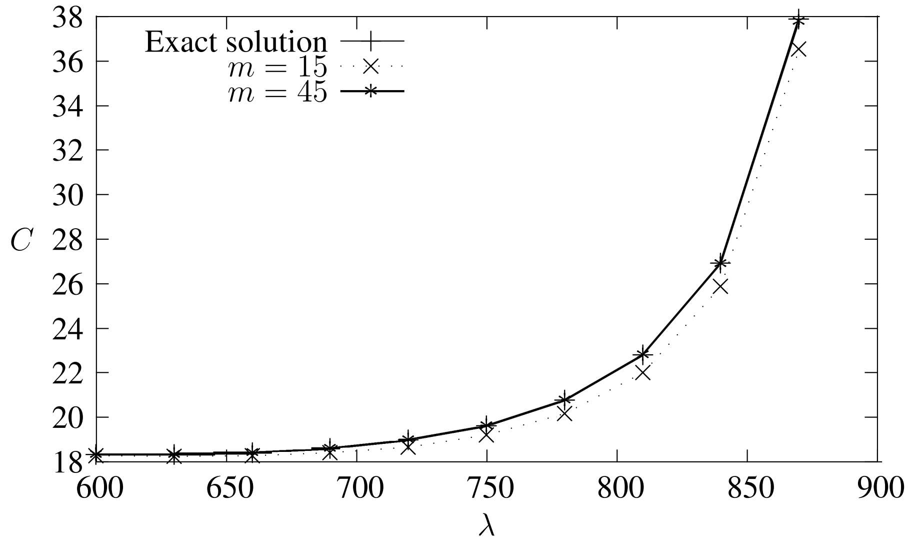

Figure 5. Non-zero transfer times: exact and approximate solutions
$K = 15$, $\mu = 1000$, $\tau = 1000$, $\gamma = 600$, $c_1 = 1$, $c_2 = 0.1$

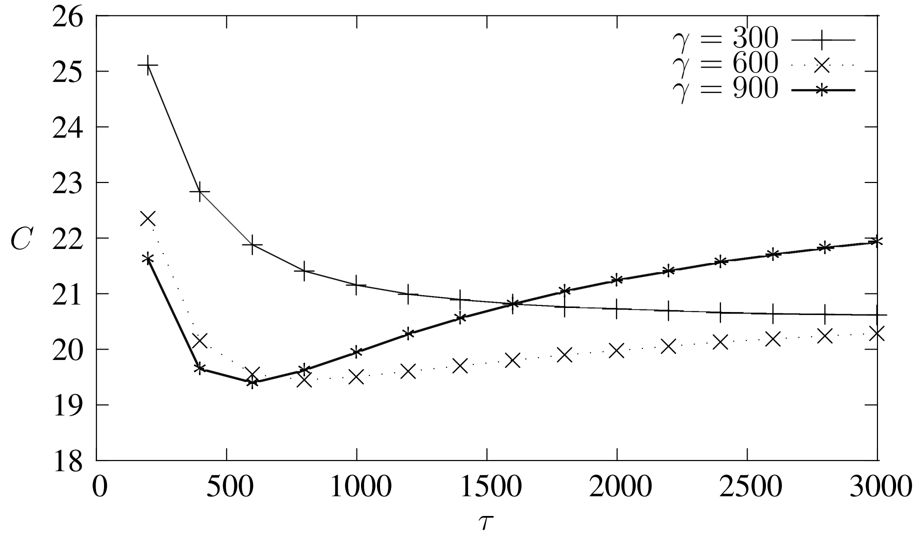

Figure 6. Non-instantaneous transfers; Cost vs. τ for different γ
$K = 25$, $\lambda = 750$, $\mu = 1000$, $c_1 = 1$, $c_2 = 0.1$, $m = 100$

Figure 7 illustrates a joint optimisation with respect to both K and τ, for the same three different transfer rates. The batch size is varied, and for each K, the optimal τ is found and the resulting cost function is plotted. Note that the stability condition (26) imposes a lower bound on K which depends on γ. When $\gamma = 900$, K cannot be lower than 4; when $\gamma = 600$, $K \geq 5$ and when $\gamma = 300$, $K \geq 9$.

The global optimum is reached for $(K = 12, \tau = 250)$ when $\gamma = 900$, for $(K = 17, \tau = 650)$ when $\gamma = 600$ and, apparently, for $(K = \infty, \tau = \infty)$ when $\gamma = 300$. This last pair of parameter values corresponds to the policy which keeps all incoming jobs in the

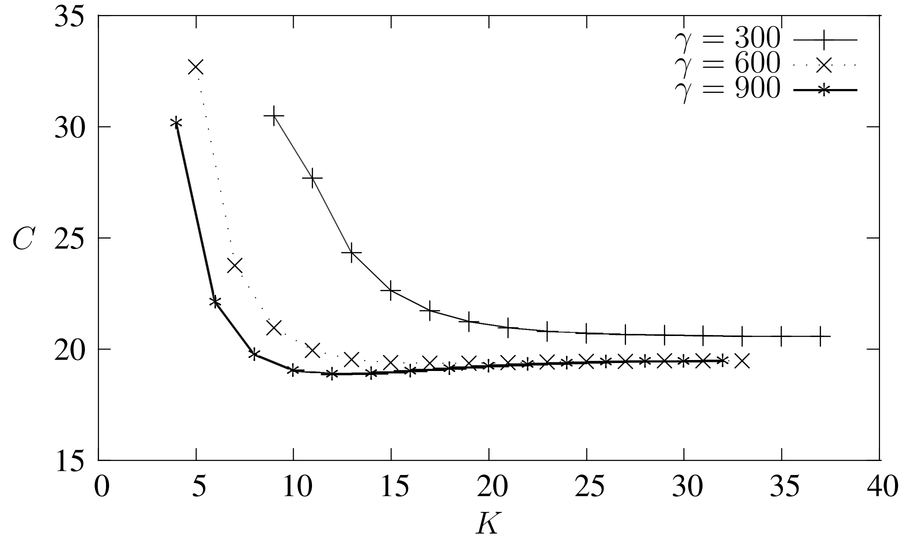

Figure 7. Non-instantaneous transfers; Optimal cost vs. K for different γ
$\lambda = 750$, $\mu = 1000$, $c_1 = 1$, $c_2 = 0.1$, $m = 100$

external queue, waiting for the internal queue to become empty; a transfer is initiated as soon as that happens, or, if the external queue is also empty at that point, as soon as the next job arrives. This might be called the 'default policy'. The above experiment, and others as well, have shown that the default policy is optimal when the transfer rate is low. Moreover, it seems that it is quite a good policy whenever the transfer rate is appreciably lower than the service rate. This is because the optimal cost function tends to flatten out when K increases, and also becomes less dependent on γ.

7. Conclusion

We have analysed a non-trivial model of a stream-processing server with an external and an internal queue. Transfers between the two may be either instantaneous or non-instantaneous. Exact and approximate solutions have been obtained, enabling the evaluation and optimization of the trade-offs between holding costs and transfer costs.

It was interesting to observe that, while *underestimating* the optimal batch size can be very expensive, *overestimating* it tends to be quite safe. This is true for both instantaneous and non-instantaneous transfers. In particular, the default policy introduced at the end of last section would be a good choice to use whenever the transfer times are non-negligible.

In some implementations of Apache Storm, the flush timer is always active, regardless of whether the internal queue is empty or not. In other words, when transfers are instantaneous, there is a transition from state (i, j) to state $(i + j, 0)$ with rate τ, for all $i \geq 0$ and $j = 1, 2, \ldots, K-1$. Appropriate modifications are in order when transfers are not instantaneous. That variation of the model can be analyzed by the methods we have employed here. The set of equations for the generating functions would be different, but one would still need to find the zeros of a denominator in order to compute the exact solution. An iterative approximate solution could be obtained by truncating the state space.

The results obtained for one server can be used to approximate the behaviour of a network of servers. One would treat each node in isolation, after determining the total arrival rate into it (external and from other nodes), and assuming that the merged process of arrivals is Poisson.

An ability to evaluate the network behaviour has significant practical implications for performance optimisation and adaptation in stream processing systems. A query in such systems is evaluated by a set of logical operators organised in a directed acyclic graph; to achieve performance targets, several instances of a logical operator may have to be working in parallel on servers such as the ones we have modeled. A model based approach to evaluating network level performance would enable one to optimize the configuration of the network to meet performance targets for a given set of input streaming rates, and to adapt them when the rates change significantly. This is something which the modern systems cannot yet do and it is our on-going work (see [5] for progress in collaboration with Twitter).

Finally, the effect of different traffic characteristics, e.g. non-exponential service times or non-Poisson arrivals, can only be evaluated by means of simulations or observations of a real-life system. That, too, would be worth pursuing further.

References

[1] Balsamo, S., de Nitto Personé, V., & Onvural, R. (2001). *Analysis of Queueing Networks with Blocking*, Kluwer Academic Publishers, Boston,

[2] Carbone, P., Katsifodimos, A., Ewen, S., Markl, V., Haridi, S., & Tzoumas, K. (2015). Apache Flink: Stream and Batch Processing in a Single Engine. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 36(4) 28–38.

[3] De Matteis, T., & Mencagli, G. (2016). Keep Calm and React with Foresight: Strategies for Low Latency and Energy Efficient Elastic Data Stream Processin. *Proceedings of 21st Acm Sigplan Symposium on Principles and Practice of Parallel Programming (PPOPP'16)*, 1–12.

[4] Fu, T. Z. J., Ding, J., Ma, R. T. B., Winslett, M., Yang, Y., & Zhang, Z. (2015). DRS: Dynamic Resource Scheduling for Real Time Analytics over Fast Streams. In *Proceedings of Int. Conf. on Distributed Computing Systems*, 411–420.

[5] Kalim, F., Cooper, T., Wu, H., Li, Y., Wang, N., Lu, N., Fu, M., Qian, X., Luo, H., Cheng, D., Wang, Y., Dai, F., Ghosh, M., & Wang, B. (2019). Caladrius: A Performance Modelling Service for Distributed Stream Processing Systems. In *Proceedings of the 35th IEEE Int. Conf. on Data Engineering (Online), 8 April 2019, Macau, SAR, China*, 1886–1897.

[6] Kingman, J. F. C. (1961). The Single Server Queue in Heavy Traffic. *Proceedings of the Cambridge Philosophical Society*, 57, 902-904.

[7] Kulkarni, S., Bhagat, N., Fu, M., Kedigehalli, V., Kellogg, C., Mittal, S., Patel, J. M., Ramasamy, K., & Taneja,S. (2015). Twitter Heron. *Proceedings of the 2015 ACM SIGMOD Int. Conf. on Management of Data*, 239-250.

[8] Lohrmann, B., Janacik, P., & Kao, O. (2015). Elastic Stream Processing with Latency Guarantees. *Proceedings of Int. Conf. on Distributed Computing Systems*, 399–410.

[9] Neuts, M. F. (1984). Matrix-analytic methods in queuing theory. *European Journal of Operational Research*, 15, 2-12.

[10] Perros, H. G. (1994). *Queueing Networks with Blocking. Exact and Approximate Solutions*. Oxford University Press, New York.

[11] Stonebraker, M., Çetintemel, U., & Zdonik, S. (2005). The 8 Requirements of Real-time Stream Processing. *ACM SIGMOD Record* , 34(4), 42-47.

[12] Thompson, M., Farley, D., Barker, M., Gee, P., & Stewart, A. (2011). DISRUPTOR: High performance alternative to bounded queues for exchanging data between concurrent threads. http://lmax-exchange.github.io/disruptor/files/Disruptor-1.0.pdf.

[13] Toshniwal, A., Donham,J., Bhagat, N., Mittal, S., Ryaboy, D., Taneja, S., Shukla, A., Ramasamy, K., Patel, J. M., Kulkarni, S., Jackson, J., Gade, K., & Fu, M. (2014). Storm@twitter. *Proceedings of 2014 ACM SIGMOD Int. Conf. on Management of Data*, 147-156.

[14] Vakilinia, S., Zhang, X., & Qiu, D. (2016). Analysis and Optimization of Big-Data Stream Processing. *Proceedings of IEEE Global Communications Conference (GLOBECOMM'16)*, 1–6.

Appendix

Proof of Proposition 1. We invoke Rouché's theorem, which states that if two holomorphic functions, $\phi(z)$ and $\psi(z)$, satisfy $|\phi(z)| > |\psi(z)|$ on a simple closed contour, then $\phi(z)$ and $\phi(z) + \psi(z)$ have the same number of zeros inside that contour. Each zero is counted according to its multiplicity.

In the case of $D(z)$, given by (22), let $\phi(z) = a(z)^K$ and $\psi(z) = -(\rho z^2)^K$. The closed contour is the unit circle. When $|z| = 1$, $|\psi(z)| = \rho^K$, while

$$|a(z)^K| = |(1+\rho)z - 1|^K \geq [|(1+\rho)z| - 1]^K = \rho^K \ , \tag{53}$$

with equality only at the point $z = 1$ (here we have used the *triangle inequality* $|v - w| \geq |v| - |w|$).

Let us modify the contour slightly, by making it pass through a point $z = 1+\epsilon$ for some positive ϵ, as illustrated in Figure 8. Since $D(1) = 0$ and $D'(1) = K\rho^{K-1}(1-\rho) > 0$, it is possible to choose ϵ so that $D(1+\epsilon) > 0$. In other words, $|\phi(1+\epsilon)| - |\psi(1+\epsilon)| > 0$.

Moreover, if ϵ is sufficiently small, the extra part of the contour will not encounter any of the exterior zeros of $D(z)$ and hence the difference $|\phi(z)| - |\psi(z)|$ will not change sign on that extra part. Then the inequality $|\phi(z)| > |\psi(z)|$ would be strict on the entire modified contour.

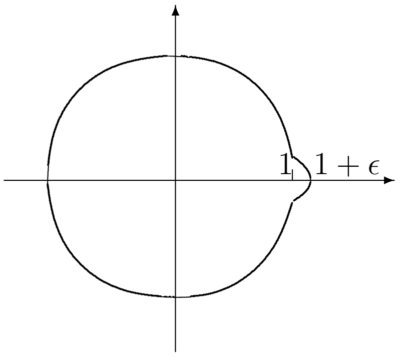

Figure 8. Modified contour

Rouché's theorem now tells us that $D(z)$ has the same number of zeros inside the modified contour as does $\phi(z) = a(z)^K$. Note that $a(z)$, which is linear in z, has a zero at $z = 1/(1+\rho)$. That is, $\phi(z)$ has a zero of order K at that point, and no other zeros. Therefore, $D(z)$ has K zeros inside the modified contour. One of these is at $z = 1$, which means that there are $K - 1$ zeros in the interior of the unit disc.

To identify the zeros of $D(z)$ explicitly, write the equation $D(z) = 0$ as

$$a(z)^K = (\rho z^2)^K \; . \tag{54}$$

Taking roots of order K at both sides, we get

$$a(z) = \rho z^2 \sqrt[K]{1} \; . \tag{55}$$

Denote the K roots of unity by e_0, e_1, ..., e_{K-1}. They are given by

$$e_k = \cos(k\theta) + i\sin(k\theta) \;\; ; \;\; k = 0, 1, \ldots, K-1 \; , \tag{56}$$

where $\theta = 2\pi/K$. Thus, $e_0 = 1$. For $k < K/2$, e_k and e_{K-k} are complex-conjugate. If K is even, then $e_{K/2} = -1$.

For each k, (55) becomes a quadratic equation

$$\rho e_k z^2 - a(z) = \rho e_k z^2 - (1+\rho)z + 1 = 0 \; . \tag{57}$$

Those equations have two roots each, $z_{k,1}$ and $z_{k,2}$, given by

$$z_{k,1} = \frac{1 + \rho + \sqrt{(1+\rho)^2 - 4\rho e_k}}{2\rho e_k} \; , \tag{58}$$

and

$$z_{k,2} = \frac{1 + \rho - \sqrt{(1+\rho)^2 - 4\rho e_k}}{2\rho e_k} \; . \tag{59}$$

The above $2K$ roots are all the zeros of $D(z)$.

Note that, for every k, $z_{k,1}$ and $z_{k,2}$ satisfy

$$|z_{k,1}||z_{k,2}| = \frac{1}{\rho} > 1 \ ; \ \ k = 0, 1, \ldots, K-1 \ . \tag{60}$$

Hence, at least one of those two roots, and in particular the one with the larger modulus, $z_{k,1}$, is outside the unit disc. Consequently, the $K-1$ zeros of $D(z)$ in the interior of the unit disc must be $z_{k,2}$, for $k = 1, 2, \ldots, K-1$. Moreover, $z_{k,2}$ and $z_{K-k,2}$ are complex-conjugate, and if K is even, $z_{K/2,2}$ is real. This completes the proof of the Proposition.

Proof of Proposition 2. Again we apply Rouché's theorem, this time to the function $D(z) = \phi(z) + \psi(z) = a(z)^K - \alpha(z)(\rho z^2)^K$, with $\alpha(z) = \gamma/[\lambda(1-z)+\gamma]$. The closed contour is the unit circle. The proof of Proposition 1 has already established that when $|z| = 1$, $|\phi(z)| \geq \rho^K$. To bound $|\psi(z)|$, note that, on the unit circle, the denominator of $\alpha(z)$ satisfies

$$|\lambda(1-z)+\gamma| \geq \lambda + \gamma - \lambda|z| = \gamma \ .$$

Therefore, $|\alpha(z)| \leq 1$, which implies $|\psi(z)| \leq \rho^K$. Thus, $|\phi(z)| \geq |\psi(z)|$. Moreover, the inequality is strict on the entire contour, except at the point $z = 1$ where $|\phi(z)| = |\psi(z)|$.

Now we note that $D'(1)$, given by (50), is positive when the stability condition (26) holds. Hence, it is possible to choose ϵ sufficiently small, so that $D(1+\epsilon) > 0$. Modifying the contour slightly, making it pass through point $1+\epsilon$ as illustrated in Figure 8, would ensure a strict inequality $|\phi(z)| > |\psi(z)|$ on the entire modified contour.

Rouché's theorem states that $D(z)$ and $\phi(z)$ have the same number of zeros in the interior of the modified contour. We saw that the latter has one zero of order K, which means that $D(z)$ has, in addition to its zero at $z = 1$, $K-1$ other zeros inside the interior of the unit disc. QED.