Predicting Queueing Delays in Delay Tolerant Networks with Application in Space

Nikolaos Bezirgiannidis and Vassilis Tsaoussidis

Space Internetworking Center,
Democritus University of Thrace, Greece
{nbezirgi,vtsaousi}@ee.duth.gr

Abstract. In this paper we present a study of queueing delays experienced in Delay Tolerant Networks with topology based on deterministic contact plan schedules. We examine a generic scenario and propose a sampling procedure that extracts measurements of queueing rates and queue lengths. Sampling queueing information is transmitted to network nodes, which then form time series and can be used to forecast future queueing rates. Through simulations we show that the introduced method can be useful for DTNs with predetermined contact schedules, such as the Interplanetary Internet, providing accurate end-to-end delivery delay predictions.

Keywords: Delay Tolerant Networking, Queueing Delay, Interplanetary Internet, Space Communications, Contact Graph Routing

1 Introduction

Although transmission rates in space communications increase, space application data increase even faster: high-quality images and vast volumes of telemetry data are expected to be delivered daily. Therefore, data volumes transmitted may increase disproportionally to the number of launched space assets. Beyond that, space assets cannot be upgraded to match the evolution of network bandwidth capacity. Therefore, queueing delay, which is in essence the waiting time until all data ahead of the current data item is forwarded, becomes significant in space, for three main reasons: first, it can become a considerable part of the total delay in planetary networks where propagation delay is not prohibitive; second, in networks with common disruptions, as in space environments, even a small queueing delay may lead to the loss of a transmission opportunity, thus postponing the data transmission for a significant amount of time until line-of-sight and transmission opportunity have returned; third, by observing queueing lengths and delays we can foresee when the space asset resources (e.g., buffer space) may be exhausted, which could be a potential disaster for some space applications.

In this work, we attempt to predict queueing delay in Delay-Tolerant Networks (DTNs). DTN [11] has been designated as the technology of choice for

inter-agency cross-support operations, Solar System Internet (SSI) [1] and deep-space missions. In [2] we have made a preliminary step towards predicting the delivery time of a bundle (i.e., DTN Protocol Data Unit [8]) by presenting a method to forecast propagation and transmission delay in data transmissions, assuming high data priority and therefore zero queueing delay. The latter, however, can be an important component of the total delay and its calculation may be really challenging, especially in complex topologies where a number of nodes are transmitting data concurrently.

In this context, we examine a generic scenario and study the outbound queue of a network node that receives unicast data simultaneously and/or successively from a number of nodes and enqueues it in an outbound queue for transmission to the next node. Even though the production and delivery rates of data cannot be foreseen, past measurements include valuable information that can assist in estimation of the corresponding future rates via time-series forecasting. The rationale for this argument is that, in space environments, data transmission flows follow a time-dependent scheme, since: a) mission data availability follows a time-dependent (rather than random) pattern, b) periodicity is imposed by planet rotations, satellite orbital movements and occasional high or low data rate passes [14], and c) linear dependency is inflicted by spaceship movements, as well as linearly evolving space weather phenomena. In a similar way, the authors in [5] use a time series to accurately model the requests received by a www server.

We present a simple method in which all nodes extract queueing rate measurements in a per-contact (i.e., per-transmission opportunity) granularity. Extracted measurements are then disseminated to all neighbours, and are stored in each node's contact plan, composing different time series between each pair of network nodes. The available time series information are then used to forecast future queueing rates and the predictions are combined with the contact plan schedules to estimate the queueing delay for the bundles to be transmitted. We evaluate the proposed method through simulations with different sets of parameters and the results show that it provides accurate information of the queueing delay, thus improving the overall precision of the estimated end-to-end bundle delivery delay.

The introduced method is, to the best of our knowledge, the first work to attempt a forecast of queueing delays in deterministic DTNs (including Space DTNs) with topologies based on contact plans. Although it was designed to apply in space internetworks, it may also cover other contact-plan-based DTNs in a similar way. Our technique can be used as a standalone tool to provide accurate information on the end-to-end delivery delays in transport- or service-layer protocols of the DTN stack (e.g. [16], [17]). It can also be incorporated into the administrative Bundle Delivery Time Estimation tool [2], to enhance the prediction of the total delay required for a bundle to reach its destination.

The rest of the paper is structured as follows: In Section 2 we discuss a number of related studies on queueing delay, both in earlier internetworks and in DTNs. In Section 3 we briefly present the space internetworking background

which is used as the base for our analysis, and in section 4 we present our approach, including the generic scenario that we target, and the methods we use for measurements extraction, information dissemination, and forecasting. In section 5 we describe the simulation methodology that we used, and in section 6 we present the simulation results. In section 7 we discuss the possible ways to extend our method, and, finally, we conclude the paper in Section 8.

2 Related Work

Packet queueing delay in computer networks has been concerning the scientists since the early stages of Internet, and numerous research papers have been published on the topic. In as early as 1985, Takagi and Kleinrock [20] study a CSMA-CD system and analytically calculate the average queueing delay of packets. In 1989, Demers et al. [9] suggest the use of the average queueing delay as a metric to control traffic in datagram networks, as opposed to flow control algorithms. End-to-end delay of Internet packets has also been thoroughly studied in the past 30 years. Bolot [6] analyses end-to-end packet delay using a probing process and discusses, among others, the queueing delay distribution. In the same context, Karam and Tobagi [15] discuss voice traffic over the Internet and emphasize the queueing component of the delay, as the only source of jitter, whereas Garetto and Towsley [12] study TCP traffic generated by file transmissions and its significant impact on queueing delays in the Internet.

In the DTN paradigm, on the other hand, queueing delay modelling and analysis differ from Internet-based internetworks. The main motivation for scientists to study queue lengths and the corresponding delays in DTNs has been their impact in routing efficiency. In [13], the authors present and compare different source routing algorithms based on the amount of knowledge that is available at the transmission initiation node. They exalt the knowledge of queueing occupancies in network nodes and state that, amongst all "oracles" that provide different types of information, (e.g., contact plan, buffer/queueing occupancies, traffic demand), the "queueing oracle" is the most difficult to realize, in order to achieve a complete knowledge of the queue occupancies in network nodes. In [10], a DTN-based link-state algorithm is applied on wireless networks in developing regions. The used link information includes queue occupancy, among others, and routing decisions incorporate the queueing delay that is calculated based on the most recent cached copy of the link information. Queueing delay has also been used as part of a performance metric in [19]; Seligman et al. propose a DTN routing scheme with push and pull functions and measure its congestion control effectiveness with a time-weighted network storage metric. This metric is the product of the storage used by all queued messages and the amount of time they remain queued.

In the typical routing algorithm that targets Interplanetary Internet, namely Contact Graph Routing (CGR - [7]), the arrival time computation does not take into account the queueing component of the delivery latency. In [3] we have proposed a modification of the CGR algorithm with the use of Earliest

Transmission Opportunity (ETO) parameter, which incorporates the available information on already enqueued data in the computation of bundle delivery delay. Queue information for the local node's outducts (based only on already queued bundles) are incorporated in the delivery delay estimation and can be disseminated to other network nodes via Contact Plan Update Protocol (CPUP). However, queue length increase will generate a CPUP message only when it exceeds some predefined threshold and the message is transmitted only when there are available contact opportunities before information gets obsolete. Therefore, ETO updates with CPUP might sometimes prove inadequate for timely updates and, thus, queue length information through the path to destination might be inaccurate. In this work we exploit the queue length information in the local outducts as introduced in [3] and we move a step forward: besides measuring queue lengths, we attempt to forecast future queueing rates and delays. In section 6, the reactive computation of queueing delays with the use of CPUP is compared to its proactive equivalent with the use of forecasting, introduced in this work.

3 Background

The future SSI Architecture, which is a fundamental part of our analysis, involves cooperation of different agencies under a common framework. The overlay layer protocol that attempts to unify different internetworks and infrastructures (e.g., Interplanetary Network, Near-Earth Network, Deep-Space Network, satellite communications, planetary surface networks including the Internet, ad-hoc or opportunistic networks, etc.) is the Bundle Protocol (BP) [18]. A bundle, the BP's protocol data unit of operation, is created whenever an application initiates a data unit transmission the next neighbour is selected by the routing algorithm to forward it. The bundle is then enqueued in the outbound queue that corresponds to the selected neighbouring node and to the bundle's priority, until the underlying convergence layer protocol [18] initiate its transmission. Consecutively received bundles that are routed to the same neighbour are being enqueued in the same outbound queue, provided they have the same priority. BP comprises three different priority classes, namely bulk, normal and expedited, which characterize bundles according to the application service requirements. These classes suggest a relative prioritization that forwarding policies take into consideration to decide on the bundle to be forwarded.

In space environments, communication is not always possible between any two nodes. It requires line-of-sight between the space assets, as well as adequate provisioning by the owner agency, since for example ground stations are responsible for receiving data from multiple space missions and the distinct reception intervals for each mission have to be configured in advance. The transmission opportunities among the network nodes are also referred to as *contacts* and the complete list of future contacts form the *contact plan*. A typical space contact plan follows a periodic pattern due to the periodic nature of space asset movements.

Routing decisions for next neighbour has not been standardized yet in the framework of SSI operations. However, basic routing functionality involves the exploitation of the detailed knowledge of contacts amongst space assets well in advance [7]. A space internetwork topology is thus deterministic and configured using a contact plan, and BP routing decisions are based on the predefined contact schedules. Our method covers all networks that exhibit such predictability and base routing decisions on contact plan configurations; in this context, it may apply to other forms of disruptive internetworks as well.

The sampling procedures and queue information extraction can be part of the DTN management framework, in the context of instrumentation statistics that are periodically taken from all network nodes in order to examine the health of space assets and avoid system malfunctions. The dissemination of the extracted information can be achieved either via DTNMP [4] or with CPUP [3], both protocols compatible with the DTN architecture.

4 Queueing Delay Estimation Method

4.1 Generic Scenario

We study the queues and queueing delays in a BP outduct queue by considering a generic scenario with topology as depicted in Fig. 1. In this topology, N sender nodes are transmitting data to destination node D via node A. Thus all data sent from nodes $1, 2, ..., N$ to D or beyond need to be stored in the relay node A and then forwarded to D. This store-and-forward procedure inevitably imposes extra waiting time for any bundle enqueued in the outduct from A to D, until all previously enqueued bundles are forwarded. The corresponding generic contact plan is illustrated in Fig. 2, where a single period of transmission opportunities is depicted. The period starts from the end of the previous $A - D$ contact and ends at the next $A - D$ contact. Note that nodes $1 - N$ may have more than one communication opportunity with A during a cycle. Our primary interest is in the bundle queueing delay and, consequently, in the total end-to-end bundle delivery latency, from bundle creation time until arrival at destination D. We initially consider a simple case where all bundles are transmitted with equal priority and, thus, there is a single outbound queue for the A-D outduct.

4.2 Queueing Rate Measurements and Dissemination

In order to study the queue length and all queueing rates through time, we apply a sampling process in a per-contact granularity. When a contact from node k to A ends, the number of bytes that arrived over this contact and were enqueued for delivery to D are counted. This amount is then divided by the contact duration to obtain the average queueing rate r_{kAD} that node k imposes into outduct $A - D$. Note that this queueing rate typically differs from the $k - A$ transmission rate, due to transmission and retransmission overhead and since some of the delivered bundles may not be forwarded to this outduct to D. Furthermore, upon the end

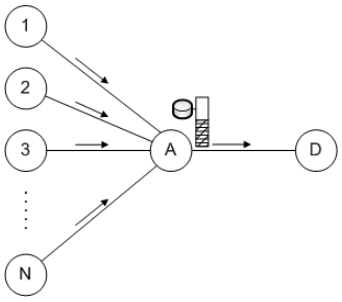


Fig. 1: Scenario Topology

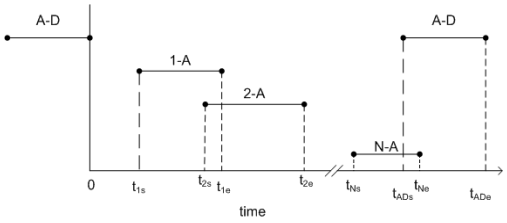


Fig. 2: Contact Plan

of the $A - D$ contact, node A calculates the remaining queue length $Qrem_{AD}$ at the specific outduct. Information about the extracted queueing rates and the remaining queue is transmitted to all neighbouring nodes other than the outduct destination (i.e. D in this example) at the next available opportunity, via CPUP. The dissemination mechanism of CPUP is responsible to relay the information PDUs to all network nodes. In our two-hop scenario, PDUs are transmitted from A to nodes $1..N$ and no further transmissions occur.

Measurement granularity for the queueing rates could be improved, if sampling occurred in a number of time intervals during each contact. This would impose serious overhead, however, and would increase the complexity of historic rates management.

4.3 Prediction of Future Queueing Rates

Following the dissemination of the measurements, all network nodes have received past values of queueing rates and remaining queue lengths. The past rate values comprise a time series for each distinct pair of neighbouring links. For example, for links $k \rightarrow A$ and $A \rightarrow D$, the time series contains past average values of r_{kAD}, i.e. data transmitted during contacts $k \rightarrow A$ and enqueued in the outduct $A \rightarrow D$.

Due to the mainly deterministic and periodic nature of Space communications, we argue that the past observations can be useful to predict future values such as queueing rates, with some accuracy. Time series may incorporate periodicity and/or a linear trend. In this context, a number of different forecasting techniques might apply in our model. The procedure used in [2], for example, utilizes a triple exponential smoothing method, which incorporates possible trends and/or periodicities in the BER time series under study. In this initial work we do not focus on the optimization of the time series forecasting method, but make a first step into proving the applicability of our proposal. Therefore we choose a simple exponential moving average (EMA) forecasting method for low complexity and low processing overhead purposes. For any contact j, the EMA S_j is calculated recursively, by computing $S_j = \alpha r_j + (1 - \alpha)S_{j-1}$, where r_j is the measured rate for contact j, and α the constant smoothing parameter,

$0 < \alpha \leq 1$. The forecast rate is set equal to the EMA of the previous time step (i.e., contact): $\hat{r}_{j+1} = S_j$. In our simulations we examine different values for the smoothing parameter α, including $\alpha = 1$, which is equivalent to a random walk model. When the time series include missing values, due to delays in the arrival of information updates, the last computed EMA is reused.

Queue length at the end of contacts $A - D$, noted as $Qrem_{AD}$, also form a time series and a similar forecasting procedure applies.

4.4 Bundle Delivery Delay Calculation

The introduced method applies on the output of any contact-plan-based routing algorithm, that is the path to destination, and calculates the total delay from bundle creation time to the arrival at destination. In our generic scenario, a bundle is created in node k and the routing algorithm selects path $k \rightarrow A \rightarrow D$. The transmission from k to A comprises the following components: i) propagation delay $d_{pr.k-A}$, ii) transmission delay $d_{tr.k-A} = (bundle\_size + overhead)/Bandwidth_{k-A}$, iii) processing delay, iv) queuing delay $d_{q.k-A}$, and v) total waiting time $d_{w.k-A}$ until transmission opportunity is available. Queueing delay for the first transmission hop is calculated based on the queue information that is available for the local outduct and may exceed the duration of a single contact. Waiting time is extracted from the contact plan and may also span across more than one time periods, if the data ahead have filled the capacity of the next contact(s). In contact plans where contacts are not often, the waiting time can be the most significant part of the total delay. Based on the aforementioned delay components and assuming trivial processing delays, expected arrival time at node A is calculated as follows, if t_{cr} the bundle creation time:

$$t_{arr.A} = t_{cr} + d_{w.k-A} + d_{q.k-A} + d_{pr.k-A} + d_{tr.k-A} \tag{1}$$

For the next transmission hop, the total delay has the same, aforementioned components, with starting time equal to $t_{arr.A}$. Calculation of the queueing delay $d_{q.A-D}$ exploits the contact plan information, the queueing rates $\hat{r}_{iAD}, i = 1..N, i \neq k$, as well as the remaining data in-queue $\hat{Q}rem_{AD}$ at the end time t_0 of the most recent $A - D$ contact before $t_{arr.A}$. All these values can either be the actual measurements, if the corresponding information has already arrived at k, or the values predicted using the proposed forecasting procedure. Queueing delay for the bundle in outduct $A - D$ is computed as follows:

$$d_{q.AD} = \frac{\hat{Q}rem_{AD} + \sum_i \hat{r}_{iAD} \times \tau_i}{Bandwidth_{A-D}} \tag{2}$$

, where i represents all contacts that may cause backlog, or, in other words contacts that are active during the time interval from the end time t_0 of the last contact A-D, until the expected bundle arrival time at node A, $t_{arr.A}$, and τ_i the contacts' duration. The waiting time $d_{w.A-D}$ for the bundle is the interval

between the arrival time $t_{arr.A}$ and the next available contact $A - D$, plus all intervals between consecutive $A - D$ contacts that the bundle waits in queue. Using these calculations and eq. 2, bundle arrival time at destination node D, which is the output of our method, becomes:

$$t_{arr.D} = t_{arr.A} + d_{w.A-D} + d_{q.A-D} + d_{pr.A-D} + d_{tr.A-D} \tag{3}$$

5 Simulation Methodology

For the evaluation of our prediction method, we have used a Java discrete-event simulator designed for Space-based scenarios, used initially in [3]. Our simulator utilizes the BP functionality and different routing algorithms apply on the contact plan simulation input.

We conducted a variety of simulations with different sets of parameters and with periodic contact plans with period equal to half day and total duration equal to one week. Contacts were randomly put during this time period and followed a periodic pattern afterwards. For each set of parameters we performed 100 repetitions to have a statistically adequate sample. The topology used is the one depicted in Fig. 2, with different number of input nodes and varying parameters displayed in Table 1.

We define λ as the ratio of the sum of all first-hop ($1..N - A$) contact volume capacities divided by the sum of all second-hop ($A - D$) contact volume capacities:

$$\lambda = \frac{N \times r_1 \times \tau_1}{r_2 \times \tau_2}, \tag{4}$$

where r_1 is the transmission rate of the first-hop links, r_2 is the transmission rate of the second-hop links, τ_1 is the duration of contacts $1..N - A$ and τ_2 is the duration of contacts $A - D$. The value of λ is practically the ratio of the capacities of the two transmission hops. When $\lambda > 1$, the queueing system is unstable and can potentially lead to storage exhaustion and node failures. In our simulations we have used three different values of λ, 0.1, 0.5, and 0.9 and we set $\tau_1 = 600s$. The respective durations of the second-hop contacts are calculated using (4).

We have also examined different data production levels, with respect to the maximum amount of data that each of the first N nodes can transmit during the total simulation time. Bundle creation times are uniform for the total simulation period.

In order to evaluate the prediction accuracy of our method, we measure the *Bundle Delivery Delay Prediction Error*, both as an absolute time unit ($BDDPredErr$), and as a percentage ($BDDPredErr$) of the *Bundle Delivery Delay* (BDD):

$$BDDPredErr = BDD - BundleDeliveryDelayEstimation \tag{5}$$

$$NormalizedBDDPredErr = \frac{BDD - BundleDeliveryDelayEstimation}{BDD} \quad (6)$$

Furthermore, in each simulation we calculate *RelativeOverhead*, which is the total number of bytes of the measurement information messages, divided by the total number of data payload bytes.

In the next section we provide comparisons of four different prediction methods: i)the BDD estimation implemented in CGR [7], mentioned also as No Forecasting ii)the delivery time estimation method that reactively exploits the queue data based on CPUP update messages presented in [3] and mentioned here as Reactive Estimation with CPUP, iii) the prediction method proposed in this work, mentioned as Forecasting with Exponential Smoothing, and iv) a prediction method similar to ours, but where there are no network updates on the queueing rates and future lengths and rates are not forecast but, instead, are set equal to the nominal link transmission rates. The latter is mentioned as Forecasting with Nominal Rate.

$$RelativeOverhead = \frac{\text{Total Overhead Bytes}}{\text{Total Data Payload Bytes}} \quad (7)$$

Table 1: Simulation Parameters

Parameter	**Value(s)**
Number of Producing Nodes N	2, 5, 10, 20
Bundle Size N	64 Kbytes
Capacities Ratio λ	0.1, 0.5, 0.9
Transmission Rate $1..N-A$	64 Kbps
Transmission Rate $A-D$	512 Kbps
Propagation Delay $1..N-A$	0.01s
Propagation Delay $A-D$	1s
Contact Duration $1..N-A$	600s

6 Simulation Results

An initial observation of the simulation results was that the occurrence of the contacts during the time period had significant impact on the total bundle delivery delay. The reason for this is the fact that the most significant portion of the total bundle delivery delay was the waiting time, in the order of tens of thousands of seconds, since contacts occur twice per day. So, when a bundle arrives at A and there is enough backlog ahead, it may be queued for a period of time longer than the contact duration, and thus it has to wait for the next transmission opportunity (i.e., half a day in our simulations). We observed that, depending on the contact occurrences, the simulations were divided into two

groups. In the first and most common one, all bundles were transmitted during the contacts predicted by CGR; in other words, there were no queueing delays large enough to cause any bundles to miss the transmission opportunity and wait for a total transmission cycle. In these simulations, to which we will refer from now on as *Case 1* simulations, the $BDDPredErr$ (i.e. the error in bundle delivery delay prediction) does not exceed the duration of a contact, and comprises a small percentage of the total delivery delay. In the second observed group of simulations (referred to from now on as *Case 2* simulations), on the other hand, queueing delays caused loss of transmission opportunities for a portion of the transmitted bundles, resulting in a significant $BDDPredErr$.

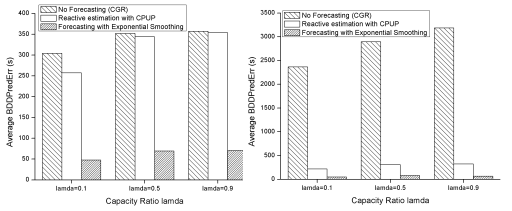
The percentage of the *Case 2* simulations depends heavily on the number of network nodes and the contact plan. Table 2 shows the percentage of *Case 2* simulations for different number of nodes, and the corresponding average percentage of bundles (in these simulations) that miss the contact opportunities due to queueing delays.

Table 2: *Case 2* simulations as a percentage of total simulations and average percentage of bundles that missed contact opportunities, in *Case 2* simulations

N	***Case 2* Simulations (%)**	**Bundles that missed transmission opportunity (%)**
2	3.33	24.75
5	4	7.6
10	12	6.66
20	23.67	2.42

For example, when $N = 10$, 11.33% of the conducted simulations were *Case 2*, and an average of 6.66% of the bundles in each simulation was actually transmitted during a different contact than the one that CGR predicted. Even though this percentage of bundles seems small, $BDDPredErr$ calculated by CGR approaches the time period, i.e. half day. This may have significant impact on the performance of the application or service layers residing on top of BP, such as unnecessary retransmissions due to timeout expirations and delayed in-order delivery, when the Delay-Tolerant Payload Conditioning protocol [16] is used. In Fig. 3 we present the average $BDDPredErr$ for different values of capacity ratio λ, with $N = 10$ producing nodes. The bundles that have lost a transmission window are reflected in the significant error observed in Fig. 3b, when no forecasting is used. In our simulations we have observed that both the reactive queue estimation method with CPUP, and the proactive forecasting method are able to predict this deviation for all bundles, that is 100% of the bundles for all set of parameters, resulting in a major $BDDPredErr$ decrease and resolving the aforementioned misbehaviour.

Nevertheless, due to the large fluctuation in the bundle delivery delay prediction, average values is not the most appropriate statistical function. In order to capture the whole range of prediction errors we use the $NormalizedBDDPredErr$



(a) *Case 1* Simulations (b) *Case 2* Simulations

Fig. 3: Average $BDDPredErr$ versus the capacity ratio λ, with $N = 10$.

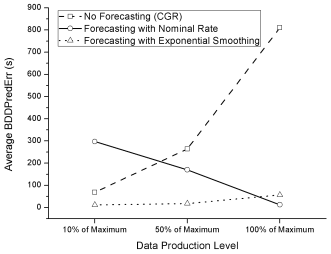
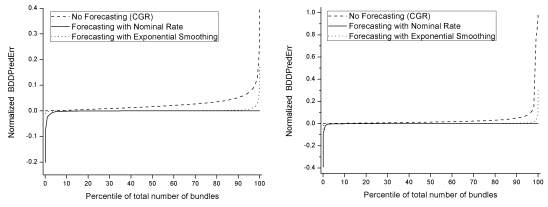


Fig. 4: Average $BDDPredErr$ versus the data production level

percentiles: all bundle delivery delay prediction errors are sorted in an ascending order and the k-th percentile corresponds to the $NormalizedBDDPredErr$ that is greater than the $k\%$ of all bundle delivery delay prediction errors. In Figures 5 and 7 we depict the $NormalizedBDDPredErr$ percentiles for sample simulations of different parameter sets, for $N = 20$ and $N = 2$, respectively. In the former (Fig. 5) we compare our method with CGR and a forecasting method with the use of nominal transmission rates, rather than predictions. In the latter (Fig. 7) we compare our exponential forecasting method with CGR, and with the reactive estimation with the use of CPUP. In Fig. 5 all algorithms achieve small prediction errors for the majority of bundles; there is, however, a 2%-3% of the bundles that all algorithms err. The CGR prediction error reaches 90% of the bundle delivery delay, for the *Case 2* simulation, and 40% of the bundle delivery delay, for the *Case 1* simulation. For our exponential smoothing method, the respective errors are less than 20%, whereas the forecasting with nominal rates provides an overall good prediction, leaving a tail of overestimation for 4 % of the bundles. For $N = 2$ (Fig. 7), $NormalizedBDDPredErr$ is significantly improved for a larger percentage of all bundles, with both reactive CPUP estimation and our forecasting method. Fig. 7a shows that in a *Case 1* simulation, the prediction accuracy can be improved with the exponential smoothing forecasting method for all bundles. However, since the queueing component is a tiny portion of the total end-to-end delivery delay, $NormalizedBDDPredErr$ does not exceed the amount of 0.4%.

So far, we have used a uniform data production rate, equal to the maximum rate that the network can serve. The prediction method with the use of nominal transmission rates provides good accuracy, as depicted in 5. However, in cases where network nodes produce less data than the network can serve, its performance degrades. In Fig. 4 we measure the Average $NormalizedBDDPredErr$ for different production levels, presented as a percentage of the maximum amount of data that can be served. Although forecasting with nominal rates outperforms the other algorithms for large data rate productions, its results for 10% of the maximum production rate become even worse than with CGR. In our method, despite the fact that network nodes have no prior knowledge of the production

rates of other nodes, they achieve a good estimation for all production rates, due to the past queueing values obtained through update messages. Note that in Fig. 4, the average $BDDPredErr$ represents the mean of absolute values, whereas in the percentiles figures we also provided the negative, overestimated values.



(a) *Case 1* Simulation (b) *Case 2* Simulation

Fig. 5: $NormalizedBDDPredErr$ versus the percentiles of total number of bundles for sample simulations with $N = 20$ and λ=0.9.

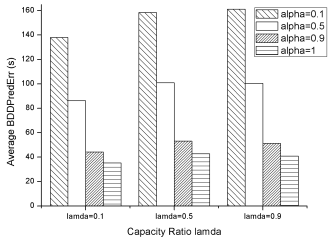
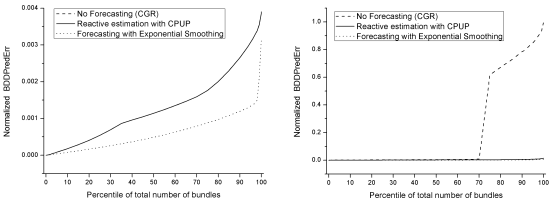


Fig. 6: Average $BDDPredErr$ for different values of the smoothing parameter α, with $N = 5$.



(a) *Case 1* Simulation (b) *Case 2* Simulation

Fig. 7: $NormalizedBDDPredErr$ versus the percentiles of total number of bundles for sample simulations with $N = 2$ and λ=0.9.

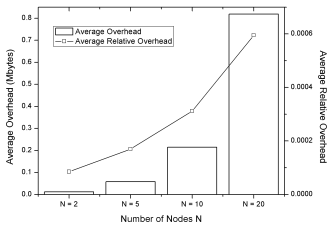


Fig. 8: Total Overhead versus the number of nodes N

In Fig. 8 we illustrate the overall overhead caused by the update messages in relevance to the transmitted amounts of data payloads. The amount of overhead bytes span from 11.7 Kbytes for simulations with data transmissions of 137 Mbytes (N=2), to 818 Kbytes for simulations with data transmissions of 1.37 Gbytes (N=20).

Finally, we study the impact of the exponential smoothing parameter by using the different values $\alpha = 0.1, 0.5, 0.9, 1$. Figure 6 illustrates that the predictions are more accurate for values of α near 1, (i.e., more sensitive to changes), which shows larger dependency on the recent values than on the history observations. This behaviour is justified by the use of a uniform production rate in our simulations: the resulting transmission rates increase gradually from zero to the steady-state rate, stay there till the end of bundle productions and decrease gradually to zero again. Different production rates than the uniform we used in this work might require less sensitivity to fluctuations and more weight on the history values.

7 Future Work

As mentioned in the description of our method, in this initial work we do not focus on the optimization of the queueing rates forecasting procedure. Instead, we describe the proof-of-concept and attempt an initial evaluation with a simple exponential smoothing forecasting method. A potential future expansion of our work is the analysis of different time series forecasting methods, such as triple exponential smoothing or ARMA/ARIMA, and the assessment of the trade-off between the practicality of the prediction accuracy and the computational overhead that time series calculation will impose on the energy-sensitive Space assets.

We also intend to study the performance of our method when there are bundles with different priority levels, such as the three priority classes defined in [18] (i.e., bulk, normal and expedited), For that, the proposed method should be modified with respect to the forwarding policy that is applied on the DTN nodes. With a typical routing scheme (Contact Graph Routing [7]) that defines three different outbound queues for each neighbouring node depending on bundle priorities, our method can be easily adjusted: queue lengths and queueing rates will be stored, disseminated and forecast for each of the three outbound queues separately.

Furthermore, the proposed forecasting method has not been incorporated into any contact-plan-based routing algorithm. It is merely a tool to estimate end-to-end delivery delay on the bundle route extracted by the routing algorithm. It is in our future plans to study the applicability of an optimized version of the proposed forecasting method in routing algorithm decisions, and to examine the complexity inflicted by this incorporation.

8 Conclusion

In this paper, we introduced a novel method to predict queueing rates and queueing delays in contact-plan-based DTNs with application in Space communications. Queue length statistics are extracted in a per-contact granularity and is disseminated to the network nodes. These historical data are then used to predict future queueing rates via time series forecasting and, ultimately, improve the estimation of bundle queueing delays en route to destination. Through extensive simulations we showed that it outperforms both the calculation of end-to-end delays provided in CGR and the reactive updates of queue lengths with the use of update messages, without inflicting any significant transmission overhead.

Our method can assist the configuration of higher layer protocols and services, providing a more accurate end-to-end delivery delay estimate (e.g. configuration of retransmission timers, etc.). It can also be used as an administrative tool to analyse queue length distributions and queueing delays in DTNs with deterministic contact schedules.

References

1. Sisg operations concept for ssi - final version. Interagency Operations Advisory Group, Space Internetworking Strategy Group (January 2011)
2. Bezirgiannidis, N., Burleigh, S., Tsaoussidis, V.: Delivery time estimation for space bundles. IEEE Transactions on Aerospace and Electronic Systems 49(3), 1897–1910 (2013)
3. Bezirgiannidis, N., Tsapeli, F., Diamantopoulos, S., Tsaoussidis, V.: Towards flexibility and accuracy in space dtn communications. In: ACM MobiCom CHANTS 2013. pp. 43–48. ACM, New York, NY, USA (2013)
4. Birrane, E., Ramachandran, V.: Delay tolerant network management protocol. Tech. rep., Delay-Tolerant Networking Research Group (2013)
5. Bolot, J.C., Hoschka, P.: Performance engineering of the world wide web: application to dimensioning and cache design. Comput. Netw. ISDN Syst. 28(7-11), 1397–1405 (May 1996)
6. Bolot, J.C.: End-to-end packet delay and loss behavior in the internet. In: SIGCOMM 1993. pp. 289–298. ACM, New York, NY, USA (1993)
7. Burleigh, S.: Contact graph routing. Tech. Rep. draft-burleigh-dtnrg-cgr-01, Network Working Group (Jul 2010)
8. Cerf, V., Burleigh, S., Hooke, A., Torgerson, L., Durst, R., Scott, K., Fall, K., Weiss, H.: Delay-Tolerant Networking Architecture. RFC 4838 (Informational) (Apr 2007), `http://www.ietf.org/rfc/rfc4838.txt`
9. Demers, A., Keshav, S., Shenker, S.: Analysis and simulation of a fair queueing algorithm. SIGCOMM Comput. Commun. Rev. 19(4), 1–12 (Aug 1989)
10. Demmer, M., Fall, K.: Dtlsr: delay tolerant routing for developing regions. In: NSDR 2007. pp. 5:1–5:6. ACM, New York, NY, USA (2007)
11. Fall, K.: A delay-tolerant network architecture for challenged internets. In: ACM SIGCOMM 2003. pp. 27–34. New York, NY, USA (2003)
12. Garetto, M., Towsley, D.: Modeling, simulation and measurements of queuing delay under long-tail internet traffic. In: ACM SIGMETRICS 2003. pp. 47–57. SIGMETRICS '03, ACM, New York, NY, USA (2003)
13. Jain, S., Fall, K., Patra, R.: Routing in a delay tolerant network. SIGCOMM Comput. Commun. Rev. 34(4), 145–158 (Aug 2004)
14. Jennings, E., Segui, J., Gao, J., Clare, L., Abraham, D.: The impact of traffic prioritization on deep space network mission traffic. In: Aerospace Conference, 2011 IEEE. pp. 1–6 (2011)
15. Karam, M., Tobagi, F.: Analysis of the delay and jitter of voice traffic over the internet. In: Proceedings of IEEE INFOCOM 2001. vol. 2, pp. 824–833 vol.2 (2001)
16. Papastergiou, G., Alexiadis, I., Burleigh, S., Tsaoussidis, V.: Delay tolerant payload conditioning protocol. Computer Networks 59(0), 244 – 263 (2014)
17. Papastergiou, G., Bezirgiannidis, N., Tsaoussidis, V.: On the performance of erasure coding over space dtns. In: Wired/Wireless Internet Communication 2012, Lecture Notes in Computer Science, vol. 7277, pp. 269–281 (2012)
18. Scott, K., Burleigh, S.: Bundle Protocol Specification. RFC 5050 (Experimental) (Nov 2007), `http://www.ietf.org/rfc/rfc5050.txt`
19. Seligman, M., Fall, K., Mundur, P.: Storage routing for dtn congestion control. Wireless Communications and Mobile Computing 7(10), 1183–1196 (2007)
20. Takagi, H., Kleinrock, L.: Mean packet queueing delay in a buffered two-user csma/cd system. Communications, IEEE Transactions on 33(10), 1136–1139 (1985)