RESEARCH ARTICLE

Blocking Analysis of Persistent Resource Allocations for M2M Applications in Wireless Systems

Jason Brown, Nusrat Afrin and Jamil Y Khan*

School of Electrical Engineering and Computer Science, The University of Newcastle, Callaghan, NSW 2308, AUSTRALIA

ABSTRACT

Wide area wireless systems conventionally employ dynamic scheduling for stochastic or bursty applications and persistent resource allocations of a given period for deterministic applications such as voice. When considering persistent resource allocations for machine-to-machine (M2M) applications from different markets, a wide range of allocation periods may be required to fully support the diversity of applications. The set of periods supported by the wireless system is a compromise between efficient use of the available resources and supporting as many M2M applications as possible. We consider two schemes: a simply periodic system which offers a limited set of periods with very efficient use of resources, and a complex periodic system which offers a wider range of periods at the cost of lower efficiency. We derive formulae for the blocking probability of these two systems by considering different resource sharing policies of the Erlang Multirate Loss Model (EMLM) and the concepts of packing (when a new persistent allocation is admitted to the system) and repacking (when an existing persistent allocation leaves the system). The theoretical models are verified using a discrete event simulation with variable offered traffic loads. The concepts discussed in this paper are generic, but may find particular application in Long Term Evolution (LTE) and Worldwide Interoperability for Microwave Access (WiMAX) networks for the purposes of system configuration (particularly in terms of the set of periods supported for persistent allocations), resource dimensioning and system performance characterisation.

∗Correspondence
Email: jamil.khan@newcastle.edu.au

1 INTRODUCTION

There is considerable interest in deploying M2M applications from such sectors as utilities, transportation and agriculture over wide area wireless systems such as 3GPP LTE and IEEE 802.16/WiMAX networks [1][2][3]. This is due in part to the relatively low device and airtime costs, wide area coverage, low latency and flexible data rates. However, the introduction of M2M applications into systems originally designed for human-to-human (H2H) and human-to-machine (H2M) communications results in a number of design, implementation and operational challenges which are being investigated by the standards bodies [4][5]. For example, a key concern has been overload of the random access channel caused by the relatively large number of M2M devices that are expected to reside in each cell attempting to send data near simultaneously. A significant amount of research has been devoted to this particular topic and many different solutions proposed [6][7].

Once a device which wishes to send data has successfully negotiated the random access procedure, the base station scheduler must schedule resources for that device. Wide area wireless systems typically use at least two resource allocation strategies for H2H and H2M transmission as illustrated in Figure 1(a) and (b). For bursty data, dynamic packet-by-packet scheduling is employed in which the scheduler explicitly allocates resources to a device by sending an uplink grant on the downlink control channel to the device prior to each and every transmission (see Figure 1(a)). The duration between grants and the volume of resources allocated at each grant for an individual device is generally variable. This facilitates maximum flexibility for the scheduler, allowing it to satisfy one or more system objectives such as maximising spectral efficiency, ensuring fairness between devices and/or meeting Quality of Service (QoS) targets of users. For deterministic applications of which interactive voice is the primary example, a static persistent resource allocation in which the device is allocated a fixed resource with a fixed period to match the characteristics of the underling application is instead employed (see Figure 1(b)). The advantage of this persistent scheme is that the resource allocation occurs explicitly only once (i.e. at the setup of the persistent resource allocation) rather than on an ongoing basis which frees up control channel resources. In LTE, the static persistent resource allocation scheme for voice is known as Semi-Persistent Scheduling (SPS) [8].

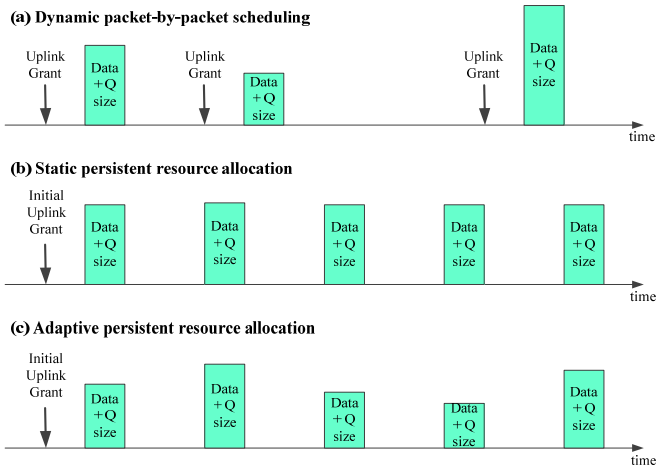


Figure 1. Comparison of Resource Allocation Schemes (from the perspective of one device)

For M2M transmission, the two resource allocation schemes are appropriate for different applications. For example, an isolated sensor which under normal conditions sends data packets intermittently is best scheduled using dynamic packet-by-packet scheduling. However, in the case of a Wide Area Measurement System (WAMS) for the Smart Grid in which Phasor Measurement Units (PMUs) or synchrophasors generate periodic packets of a fixed size which need to be sent to a WAMS server with very low latency to afford control and protection of the transmission and distribution electricity networks [3], a static persistent resource allocation scheme is clearly more appropriate. Some research on the application of static persistent resource allocation schemes to M2M applications can be found in [9-14].

There are reasons to consider persistent resource allocations with a fixed transmission period for M2M devices with a stream of packets to send even when the packet arrival process is not deterministic and/or the packet size is not fixed. This is particularly true for highly delay sensitive M2M applications such as monitoring and control applications because persistent resource allocations effectively provide a guaranteed exclusive resource to a single device. Secondly, a device with a persistent resource allocation has knowledge of its transmission opportunities in advance and can in principle disable its transceiver between such opportunities, thereby more effectively managing power; this is extremely important for battery operated M2M devices in the field that cannot be recharged easily or frequently. Thirdly, the widespread use of M2M gateways [3] to aggregate traffic from multiple M2M devices in certain M2M applications before transmission over the wide area wireless network means that even when the individual device data is highly bursty, the aggregated traffic from the M2M gateway is much less so (assuming independence between individual devices) and is more appropriate to be sent using a persistent resource allocation.

When a static persistent resource allocation as illustrated in Figure 1(b) is employed to carry data from a device with a stochastic data source, the fixed resources assigned periodically will sometimes be insufficient and sometimes excessive to serve the pending data at the device. An improvement in this case is adaptive persistent resource allocation [15] as illustrated in Figure 1(c). In this scheme, the amount of resources allocated to a device can vary from one transmission opportunity to the next based upon shared knowledge at the device and base station about the instantaneous device queue size (which is piggybacked with uplink data packets). The device and base station employ the most recent queue size data available as input to a pre-agreed adaptation function to calculate the amount of uplink resources required for the next transmission opportunity (up to some maximum negotiated when the persistent allocation is initially setup) without any need to signal updated uplink grants on the downlink control channel. If less than the maximum volume of resources is required, the resources saved can be re-used by the base station scheduler for dynamic packet-by-packet scheduling [15].

On the one hand, the base station scheduler should be flexible and grant a persistent resource allocation of the desired period requested by the M2M application. However, supporting arbitrary periods can lead to inefficiencies due to potential time domain collisions between devices which implies allocations must be separated by some other mechanism e.g. frequency domain partitioning. For example, consider Figure 2 in which two persistent resource allocations with periods $T_1 \in \mathbb{N}^+$ and $T_2 \in \mathbb{N}^+$ slots where $T_2 > T_1$ are initially offset by c slots ($1 \leq c \leq T_1 - 1$).

The condition for no time domain collisions is that there are no integer values of x and y which result in a solution to Eq. (1):

$$xT_1 = c + yT_2 \quad (1)$$

Eq. (1) can be re-written as:

$$xT_1 - yT_2 = c \quad (2)$$

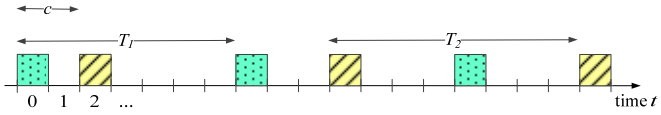


Figure 2. Persistent Resource Allocations

Eq. (2) is a linear Diophantine equation and as such a solution exists if and only if c is an integer multiple of $\gcd(T_1, T_2)$ [16] where $\gcd(\cdot)$ is the greatest common divisor function. If T_1 and T_2 are coprime i.e. $\gcd(T_1, T_2) = 1$, any value of c is an integer multiple of $\gcd(T_1, T_2)$ and the two allocations will collide in the time domain with period $\mathrm{lcm}(T_1, T_2) = T_1 T_2$ slots where $\mathrm{lcm}(\cdot)$ is the least common multiple function. Conversely, if T_2 is an integer multiple of T_1, $\gcd(T_1, T_2) = T_1$ and given that $1 \leq c \leq T_1 - 1$, there is no solution to Eq. (2) and no time domain collisions occur irrespective of the value of c.

For this reason, we concentrate on systems which offer a limited set of periods which are integer multiples of a fixed base period T. We first consider a simply periodic system [17][18] in which each supported period is an integer multiple of each smaller supported period in the system (and ultimately of the base period T). Such a system is very efficient in terms of its use of resources because there is no opportunity for time domain collisions between distinct allocations provided they are offset appropriately from each other in terms of time. However, the number of supported periods in such a system is usually insufficient to support a wide range of M2M applications appropriately. Consequently, we also introduce a new specific type of complex periodic system in which the set of supported periods is the union of two or more subsets where the periods in each subset follow the simply periodic rules. This allows for a more granular set of supported periods at the expense of less efficient resource allocation because allocations from different subsets cannot be multiplexed together on the same logical time slot (to be defined later) without causing time domain collisions.

The main focus of the paper is on the analysis of blocking probability for each supported persistent resource allocation type (i.e. of a given period) in simply periodic and complex periodic systems. Persistent resource allocation types with smaller periods are generally more difficult to accommodate than those with larger periods because they require more resources per unit time, therefore the blocking probability is generally higher. We make use of and expand the Erlang Multirate Loss Model (EMLM) [19][20] in this analysis since different allocation periods can be viewed in terms of different data rates. The EMLM supports a flexible resource sharing policy that specifies which request types can use which resources at which times. Therefore we define the resource sharing policies for simply periodic and complex periodic systems as part of the analysis. This depends upon the new concepts of packing (when a new persistent resource allocation is admitted to the system) and repacking (when an existing persistent resource allocation leaves the system) which we introduce in this paper. For example, two persistent resource allocations of period 2T can be packed to use exactly the same physical resources as a single persistent resource allocation of period T would do, or alternatively they can be allocated in an unpacked manner as illustrated in Figure 3. The importance of packing and repacking is that it affects which persistent resource allocation requests can be accepted by the system when the available resources are in short supply (as will be shown

later in the paper), and therefore ultimately affects the blocking probability.

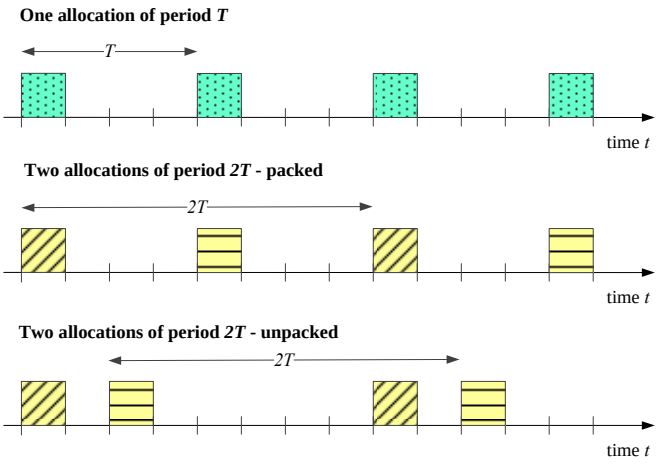


Figure 3. Packed and Unpacked Persistent Resource Allocations

The utility of the blocking probability models is that they allow accurate assessment of the split between the M2M offered traffic of each type that can be carried over persistent resource allocations and that which is blocked from receiving a persistent resource allocation and must therefore be carried using conventional dynamic packet-by-packet scheduling. This facilitates accurate prediction of system performance.

The contributions of this paper are as follows:

- introduction of the concepts of simply periodic and complex periodic systems in relation to the set of periods which are supported for persistent resource allocations in M2M systems.
- introduction of the concepts of packing and repacking of persistent resource allocations.
- analysis of the blocking probability for different types of persistent resource allocations using and expanding the EMLM with resource usage policies which are specific to simply periodic and complex periodic systems.
- validation of the blocking probability models using a discrete event simulation to produce numerical results which can be compared with the theoretical results.

The recent work of [21] introduces a tree based resource allocation algorithm for persistent resource allocations in M2M systems. This algorithm attempts to minimise the number of channels required to carry every member of a set of persistent resource allocations without assuming a specific discipline (e.g. as in simply periodic or complex periodic) for the set of supported periods. However, it does not address the analysis of blocking probability for the different resource allocation types when the number of channels is fixed as this paper does.

The remainder of this paper is organised as follows. Section 2 presents the system model. This includes introducing the concepts of simply periodic and complex periodic systems, and packing and repacking of persistent resource allocations. The blocking probability for persistent resource allocations of different periods is derived in Section 3 with reference to the EMLM and resource usage policies which are specific to simply periodic and complex periodic systems. Section 4 validates the theoretical models for blocking probability via a discrete event simulation. Finally, we discuss conclusions and present ideas for future research in Section 5.

2 System Model

2.1 Introduction

We consider a time slotted system in which a scheduler allocates resources to individual devices via persistent and/or dynamic scheduling. The system supports persistent allocations of different periods to reflect the diverse application requirements associated with different devices. However, all supported periods are integer multiples of a fixed base period per the discussion in Section 1. In the example of Figure 4, four persistent allocations are active, one with base period T physical time slots, two with period 2T physical time slots and one with period 3T physical time slots.

We define a logical time slot as repeated instances of a physical time slot across the base period for persistent allocations (i.e. T physical time slots in this example). Thus the persistent allocation with period T physical time slots exclusively occupies the first logical time slot. The two persistent allocations with period 2T physical time slots (corresponding to devices 2a and 2b) share the third logical time slot (although they could in principle have been scheduled on different logical time slots). The persistent allocation with period 3T physical time slots partially occupies the fourth logical time slot, and there is spare capacity on this logical time slot for two more persistent allocations with period 3T physical time slots.

Devices specify the required period when requesting a persistent allocation, and the scheduler either grants/admits or denies/blocks the request depending upon whether sufficient spare capacity exists in logical time slots to accommodate the request at the time it is made. We assume throughout that devices request a period which is sufficient to transfer packets to satisfy the prescribed delay budget of the associated application.

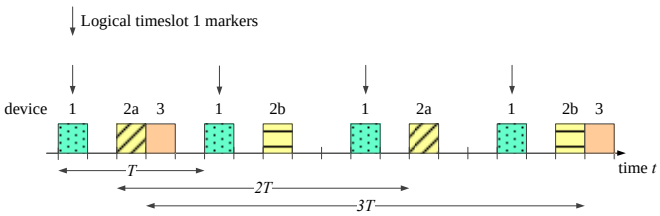


Figure 4. Persistent Allocations with Different Periods

The scheduler can also dynamically assign resources to devices, either on a physical time slot which is not currently associated with a persistent allocation or from the pool of remaining resources on a physical time slot for which a persistent allocation is currently active. For example, in an Orthogonal Frequency Division Multiple Access (OFDMA) based system such as LTE, resources are allocated both in the frequency and time domain so both persistent and dynamic resource allocations can be accommodated in the same physical time slot.

2.2 Assumptions

The following assumptions are made for the analysis presented in Section 3. These are justified in the following text.

i. Persistent allocations take precedence over dynamic allocations (see Section 2.3).
ii. A persistent allocation request that cannot be admitted because there are insufficient persistent resources available is blocked and served instead on a dynamic basis without any retries.
iii. A maximum of one persistent allocation is allowed per physical time slot (see Section 2.4).
iv. A maximum of one persistent allocation is allowed per device (see Section 2.4).
v. The set of persistent allocation periods supported by the system conforms to one of two disciplines: a *simply periodic* system or a *complex periodic* system (see Section 2.5).
vi. Newly admitted persistent allocations are packed as tightly as possible with existing persistent allocations (see Section 2.6).
vii. When a persistent allocation terminates, the remaining persistent allocations are re-packed as tightly as possible, if necessary (see Section 2.6).
viii. Persistent allocation requests of a specific period have a Poisson arrival process.

2.3 Precedence

The scheduler gives precedence to persistent allocations over dynamic allocations because they are reserved resource allocations. Therefore resources are not explicitly reserved for dynamic allocations in this model; the scheduler uses the resources remaining after persistent allocations have been made to make dynamic allocations.

2.4 Number of Persistent Allocations

In this model, only a single persistent allocation can occupy a physical time slot. This may be appropriate for systems in which the volume of resources (e.g. in the frequency domain) assigned to a persistent allocation can change on a period by period basis according to an automatic adaption algorithm, or when it is necessary to ensure that sufficient resources will be available for dynamic allocations.

A device can hold only a single persistent allocation at any one time. In particular, this precludes the situation in which a device makes multiple requests for persistent allocations (possibly with different periods) simultaneously or near simultaneously, which is difficult to model analytically because of the dependence between requests.

2.5 Set of Periods Supported

In the field of processor task scheduling, a set of k task periods $\Psi = \{T_1, \dots, T_k\}$ is referred to as simply periodic if for each pair of task periods T_i and T_j such that $T_j > T_i$, T_j is an integer multiple of T_i [17][18]. This implies:

$$T_j = \left(\prod_{z=i}^{j-1} \alpha_z\right) T_i = c_{i,j} T_i \quad \forall\, i,j: 1 \le i < j \le k \tag{3}$$

where $\{\alpha_z\}$ and $\{c_{i,j}\}$ are positive integers.

We adopt this definition for one set of possible persistent allocation periods that the system supports. The motivation behind such a structure is efficiency: if $T_j/T_i = c_{i,j}$ where $c_{i,j}$ is a positive integer, then exactly $c_{i,j}$ persistent allocations with period T_j can be time multiplexed or packed with the same logical time slot utilisation as a single persistent allocation of period T_i and with no wasted capacity.

Note that the example system in Figure 4 is not simply periodic because there are a pair of periods (i.e. $T_2 = 2T$ and $T_3 = 3T$) for which the longer period is not an integer multiple of the shorter period. This is important because persistent allocations with such periods cannot be time multiplexed onto the same logical time slot without time domain collisions, so this scheme is generally less efficient than the simply periodic scheme. However, a simply periodic system may not offer a convenient or appropriate set of periods for a given application, so there is still interest in complex periodic systems.

In addition to a simply periodic system, we will also analyse a complex periodic system in which the supported set of persistent allocation periods is the union of two or more subsets each of which are individually simply periodic and which have the same base period T. More formally, if we consider s simply periodic subsets where the p^{th} subset is $\Psi^{<p>} = \{T_1^{<p>}, \dots, T_{k_p}^{<p>}\}$ and the complete set of supported periods $\Psi = \bigcup_{p=1}^{s} \Psi^{<p>}$, then with reference to Eq. (3):

$$T_j^{<p>} = \left(\prod_{z=i}^{j-1} \alpha_z^{<p>}\right) T_i^{<p>} = c_{i,j}^{<p>} T_i^{<p>} \quad \forall\, i,j: 1 \le i < j \le k_p \tag{4}$$

where $\{\alpha_z^{<p>}\}$ and $\{c_{i,j}^{<p>}\}$ are positive integers.

We further constrain any pairwise combination of the coefficients $c_{i,j}^{<p>}$ and $c_{l,m}^{<q>}$ from different subsets (i.e. $p \neq q$) to be coprime so that persistent allocations with periods from $\Psi^{<p>}$ cannot be time multiplexed in the same logical time slot as those with periods from $\Psi^{<q>}$ and vice versa. Therefore:

$$\gcd\left(c_{i,j}^{<p>}, c_{l,m}^{<q>}\right) = 1 \quad \begin{array}{l} \forall\ i,j: 1 \le i < j \le k_p, \\ \forall\ l,m: 1 \le l < m \le k_q, \\ \forall\ p,q: 1 \le p < q \le s \end{array} \tag{5}$$

Finally the base or minimum supported period of all s simply periodic subsets is assumed to be identical so:

$$T_1^{<p>} = T_1^{<q>} = T \quad \forall\, p,q: 1 \le p < q \le s \tag{6}$$

For part of the analysis, we may still refer to the complete set of supported periods in a complex periodic system, $\Psi = \bigcup_{p=1}^{s} \Psi^{<p>}$, using the same notation as for simply periodic systems i.e. $\Psi = \{T_1, \dots, T_k\}$. This is for convenience when the analysis applies equally to both types of systems.

A persistent resource allocation with period T_i physical time slots from a simply periodic subset is referred to as a Type i persistent resource allocation in the remainder of this paper.

2.6 Packing and Re-packing

When the scheduler admits a new persistent resource allocation to the system, there is often more than one choice of logical time slot (and offset within that logical time slot) in which to accommodate the allocation. The choice is important because it potentially impacts whether future persistent resource allocation requests can be admitted or must be blocked. In particular, the scheduler requires a completely free logical time slot to accommodate a future Type 1 persistent resource allocation request (i.e. a request for a persistent resource allocation with the base period). Therefore the scheduler should pack a new admitted persistent resource allocation with pre-existing active

persistent resource allocations so as to maximize the potential to accept any type of future persistent resource allocation request.

Figure 5 illustrates an example of the scheduler admitting a new Type 2 persistent resource allocation. In the optimal packing scenario, the scheduler packs the new persistent resource allocation (2d) so as to share the same logical time slot as a pre-existing persistent resource allocation (2b). This leaves the 5th logical time slot completely unoccupied, therefore the system can admit any type of future persistent resource allocation request, including a Type 1 persistent resource allocation request. In the non-optimal packing scenario, the scheduler accommodates the new persistent resource allocation (2d) on a previously unoccupied logical time slot. The consequence of this is that there are no longer any completely free logical time slots; therefore the system cannot admit a future Type 1 persistent resource allocation request.

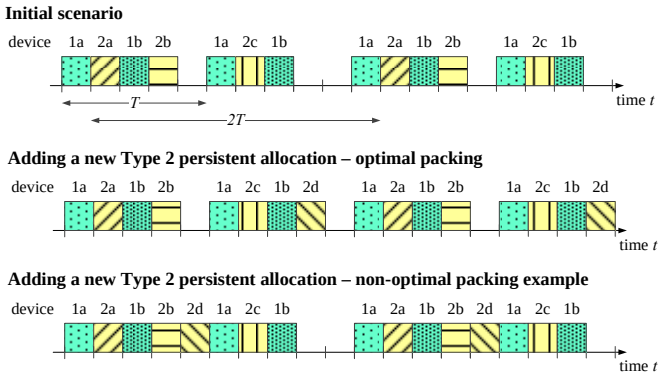


Figure 5. Packing of a New Persistent Resource Allocation

When a persistent resource allocation terminates, the resultant packing of active persistent resource allocations may be non-optimal and then it is necessary to perform re-packing in order to maximize the probability that a future persistent resource allocation request can be admitted. Since re-packing involves changing the logical resources of an ongoing persistent allocation, it involves signalling and synchronisation between the scheduler and associated device.

Figure 6 illustrates an example of the scheduler performing a re-packing exercise.

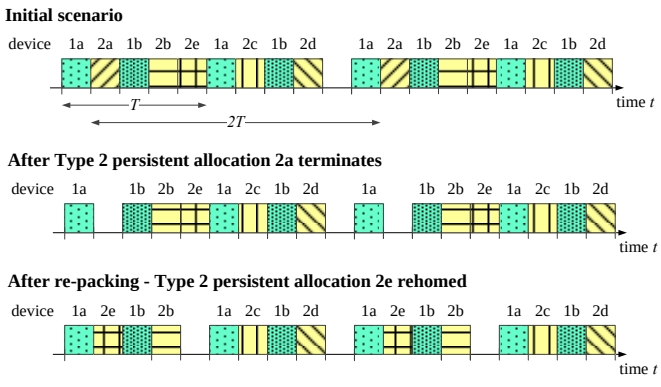


Figure 6. Re-packing After a Persistent Allocation Terminates

After Type 2 persistent resource allocation 2a terminates, persistent resource allocations 2b and 2d share a logical timeslot, but persistent resource allocations 2c and 2e remain on different logical time slots. Without re-packing, a future Type 1 persistent allocation request cannot be accommodated. In Figure 6, persistent resource allocation 2e is rehomed to share the same logical timeslot as persistent resource allocation 2c (alternatively persistent resource allocation 2c could be rehomed to share the same logical timeslot as persistent resource allocation 2e). This leaves the 5th logical time slot completely unoccupied, therefore the system can admit any type of future persistent allocation request, including a Type 1 persistent allocation request.

In this paper, we assume that the scheduler performs optimal packing and re-packing because then the criterion to admit new persistent resource allocations is dependent only upon the total remaining spare capacity i.e. it does not depend upon the detailed allocation map of ongoing persistent resource allocations to logical timeslots. As per the previous discussion in this section, in the absence of optimal packing and re-packing, the likelihood of maintaining a completely free logical time slot which can accommodate any persistent resource allocation type is reduced and therefore the blocking probability for at least Type 1 persistent resource allocations which require a whole logical time slot will be higher.

The packing and re-packing algorithms are not within the scope of this paper because we can calculate the blocking probability knowing only the end state of the system after packing/repacking (which is straightforward to do with simply periodic and complex periodic traffic types).

2.7 Actions of the Scheduler

On receiving a persistent resource allocation request of a certain type, the scheduler determines whether it can accommodate the request based only on whether there is sufficient available persistent resource allocation capacity remaining. If the request can be accommodated, the new allocation is packed as tightly as possible with any other ongoing persistent resource allocations as discussed in Section 2.6.

When a persistent resource allocation terminates, the scheduler determines whether the remaining ongoing persistent resource allocations are still optimally packed as tightly as possible. If not, it initiates a re-packing as discussed in Section 2.6.

3 ANALYSIS

3.1 Introduction

The system described in Section 2 is clearly a shared multirate loss system in which a fixed and finite amount of resources for persistent allocations are shared between devices with different data rate requirements. This type of system is characterised by the EMLM [19][20] which specifies the blocking probability for each distinct type of traffic with different data rate requirements assuming a Poisson arrival process. The EMLM model is generic and allows for different resource sharing policies which specify constraints (or the lack thereof) in the resources which can be allocated to different device types. In this section, we map our persistent resource allocation system to the EMLM and analyse different resource sharing policies as determined by the simply periodic and complex periodic systems. The primary focus of this analysis is the blocking probability for persistent resource allocations of different periods. We first begin with a recap of the EMLM using the same terminology as in [19]. A more detailed overview can be found in Appendix A.

3.2 Erlang Multirate Loss Model (EMLM)

In the EMLM [19][20], there are k distinct request types where Type i has a Poisson arrival process with rate λ_i, an expected residency time of $1/\mu_i$ and a resource requirement of b_i resource

units, where b_i is an integer. The total number of resource units to be shared between all traffic types is denoted by C, where C is also an integer.

Given a state $n = (n_1, \dots, n_k)$ where n_i is the current number of allocated Type i resources, and a set of allowed states Ω, the probability of being in any one valid state is given by [19]:

$$P(n) = \begin{cases} \prod_{j=1}^{k} \frac{a_j^{n_j}}{n_j!} G^{-1}(\Omega), & n \in \Omega \\ 0, & n \notin \Omega \end{cases} \quad (7)$$

where:

$$G(\Omega) = \sum_{n \in \Omega} \left(\prod_{j=1}^{k} \frac{a_j^{n_j}}{n_j!} \right) \quad (8)$$

and $a_i = \lambda_i/\mu_i$ is the offered load associated with Type i traffic.

The blocking probability P_i for a Type i request is the sum of the state probabilities for all valid states in which a Type i request cannot be accommodated because of insufficient available resource. This can be represented as:

$$P_i = \frac{G(B_i^+)}{G(\Omega)} \quad (9)$$

where B_i^+ is the set of blocking states for a Type i request defined as follows:

$$B_i^+ = \{n \in \Omega : n_i^+ \notin \Omega\} \quad (10)$$

and $n_i^+ = (n_1, \dots, n_i + 1, \dots, n_k)$.

The set of allowed states Ω is determined by the resource sharing policy. The simplest and most unrestricted resource sharing policy is complete sharing in which any available resource units can be allocated to any type of incoming request provided there is sufficient remaining available capacity. The set of valid states $\Omega_{complete}$ for complete sharing is thus defined as follows:

$$\Omega_{complete} = \left\{ n : 0 \le n \cdot b = \sum_{j=1}^{k} n_j b_j \le C \right\} \quad (11)$$

For complete sharing, a Type i request requiring b_i resource units is blocked if and only if the remaining available resource units $C - n \cdot b$ is less than b_i. So the set of blocking states ${B_i^+}_{complete}$ is given by:

$${B_i^+}_{complete} = \left\{ n : 0 \le C - n \cdot b = C - \sum_{j=1}^{k} n_j b_j < b_i \right\} \quad (12)$$

A more restricted resource sharing policy is dynamic bin sharing in which the total number of resource units C is divided into bins of size b_1 resource units. b_1 is assumed to be a divisor of C, therefore the number of bins is C/b_1. b_1 is also assumed to be the maximum size of allocated resource to any request type and an integer multiple of all other allocated resource sizes. Bins are dynamically assigned to a subset of request types such that only request types from that subset can be accommodated in a bin during the period in which at least one request type from the subset is active in that bin. Any request type can be allocated to an empty bin so the association between bins and resource type subsets is dynamic.

We consider s subsets of resource types where the p^{th} subset has k_p members with resource unit requirements $\Phi^{<p>} = \{b_1^{<p>}, \dots, b_{k_p}^{<p>}\}$ and $b_1^{<p>}$ is an integer multiple of $b_i^{<p>}$ for $1 < i \le k_p$. For convenience in the analysis which follows for the mapping to complex periodic systems, the first and only the first member of each subset of resource types is common to all subsets i.e. the only common member of subsets $\Phi^{<p>}$ and $\Phi^{<q>}$ (where $p \ne q$) is $b_1^{<p>} = b_1^{<q>} = b_1$ or alternatively $\Phi^{<q>} \cap \Phi^{<q>} = b_1 \; \forall \, p, q: 1 \le p < q \le s$. The state vector of the p^{th} subset can be written $n^{<p>} = (n_1^{<p>}, n_2^{<p>}, \dots, n_{k_p}^{<p>})$. Due to the fact that the first member of each subset is common, $n_1^{<p>} = n_1^{<q>} = n_1 \forall \, p, q: 1 \le p < q \le s$. Therefore the state vector of the system is given by $n = (n_1, n_2^{<1>}, \dots, n_{k_1}^{<1>}, n_2^{<2>}, \dots, n_{k_2}^{<2>}, \dots, n_2^{<s>}, \dots, n_{k_s}^{<s>})$.

With dynamic bin sharing, the number of completely and partially occupied bins $N_{occupied\ bins}$ required to accommodate a state given the restrictions on bin sharing between request types must be less than or equal to the number of bins C/b_1. The set of valid states $\Omega_{dyn\ bin}$ with s subsets of request types is therefore defined as follows:

$$\Omega_{dyn\ bin} = \{n : 0 \le N_{occupied\ bins} b_1 \le C\} \quad (13)$$

where:

$$N_{occupied\ bins} = n_1 + \sum_{p=1}^{s} \left\lceil \frac{\sum_{j=2}^{k_p} n_j^{<p>} b_j^{<p>}}{b_1} \right\rceil \quad (14)$$

Now we consider the set of blocking states ${B_i^+}_{dyn\ bin}$. For a Type 1 request for b_1 resource units, which requires a completely free bin to serve, blocking occurs when all bins are completely or partially occupied. Therefore:

$${B_1^+}_{dyn\ bin} = \{n : N_{occupied\ bins} b_1 = C\} \quad (15)$$

For a Type i request from subset p for $b_i^{<p>} \in \Phi^{<p>}$ resource units where $i > 1$, blocking occurs when all bins are completely or partially occupied and the remaining capacity in the bins currently assigned to request types with periods from subset $\Phi^{<p>}$ is less than $b_i^{<p>}$. Therefore the set of blocking states ${B_i^{<p>+}}_{dyn\ bin}$ is given by:

$$\begin{aligned} &{B_i^{<p>+}}_{dyn\ bin} \\ &= \left\{ n : N_{occupied\ bins} b_1 = C, \left\lceil \frac{\sum_{j=2}^{k_p} n_j^{<p>} b_j^{<p>}}{b_1} \right\rceil b_1 - \sum_{j=2}^{k_p} n_j^{<p>} b_j^{<p>} < b_i^{<p>} \right\} \\ &= \left\{ n : N_{occupied\ bins} = \frac{C}{b_1}, \left\lceil \frac{\sum_{j=2}^{k_p} n_j^{<p>} b_j^{<p>}}{b_1} \right\rceil - \sum_{j=2}^{k_p} \frac{n_j^{<p>} b_j^{<p>}}{b_1} < \frac{b_i^{<p>}}{b_1} \right\} \end{aligned} \quad (16)$$

3.3 Mapping of Persistent Allocation Model to EMLM

Given the set of persistent resource allocation periods $\Psi = \{T_1, T_2, \dots, T_k\}$ supported by the system, we consider the minimum super period SP measured in physical time slots for which all supported periods have an integer number of cycles. This allows the amount of resource units required by each persistent allocation period to be compared across the same time scale. Since at most one persistent allocation can be accommodated in each physical timeslot, SP also represents the total number of resource units C to be shared between all devices in the corresponding EMLM model and is given by:

$$C = SP = \mathrm{lcm}(T_1, T_2, \dots, T_k) = \left[\mathrm{lcm}\left(\frac{T_2}{T}, \dots, \frac{T_k}{T}\right)\right] T \tag{17}$$

assuming $T_1 = T$ is the lowest/base persistent resource allocation period supported by the system and all other persistent resource allocation periods $\{T_i\}$ are integer multiples of T.

The number of resource units b_i occupied by a Type i persistent resource allocation request is then given by:

$$b_i = \frac{C}{T_i} = \left[\mathrm{lcm}\left(\frac{T_2}{T}, \dots, \frac{T_k}{T}\right)\right] \frac{T}{T_i} \tag{18}$$

Therefore the maximum number m_i of Type i persistent allocations that can be granted in the system is given by:

$$m_i = \frac{C}{b_i} = T_i \tag{19}$$

This occurs when the persistent resource allocation capacity C is fully occupied by Type i persistent allocations to the exclusion of any other type of persistent allocations i.e. when the state vector $n = (0, \dots, m_i, \dots, 0)$.

3.4 Simply Periodic System

In a simply periodic system in which the set of supported periods for persistent resource allocations is defined by Eq. (3), the largest supported period T_k is an integer multiple of all other supported periods and so $\mathrm{lcm}(T_2/T, \dots, T_k/T) = T_k/T$. Therefore Eq. (17) and Eq. (18) reduce to:

$$C = T_k \tag{20}$$

$$b_i = \frac{C}{T_i} = \frac{T_k}{T_i} = c_{i,k} \tag{21}$$

A simply periodic system for persistent resource allocations maps to an EMLM system with a complete sharing resource scheme assuming optimum packing and re-packing. This is because, with reference to Eq. (3), if there are sufficient resource units to accommodate a Type i persistent allocation request with period T_i physical time slots, then it is alternatively possible to accommodate $c_{i,j}$ Type j persistent allocation requests (where $j > i$) with period $T_j = c_{i,j} T_i$ physical time slots where $c_{i,j}$ is a positive integer. In other words, the ability to admit a Type i persistent allocation request depends only upon there being at least $b_i = c_{i,k}$ resource units available from the complete pool of C resource units; there are no other restrictions.

Therefore the set of valid states $\Omega_{\text{simply periodic}}$ for the simply periodic system is given by substituting Eq. (20) and Eq. (21) into Eq. (11) as follows:

$$\Omega_{\text{simply periodic}} = \left\{ n: 0 \le n \cdot b = \sum_{j=1}^{k} n_j \frac{T_k}{T_j} \le T_k \right\} \tag{22}$$

The set of blocking states for a Type i persistent allocation request is given by substituting Eq. (20) and Eq. (21) into Eq. (12) as follows:

$$B_{i\ \text{simply periodic}}^{+} = \left\{ n: 0 \le C - n \cdot b = T_k - \sum_{j=1}^{k} n_j \frac{T_k}{T_j} < \frac{T_k}{T_i} \right\} \tag{23}$$

Therefore the blocking probability $P_{i_{\text{simply periodic}}}$ for a Type i persistent allocation request is given by reference to Eq. (9) as:

$$P_{i_{\text{simply periodic}}} = \frac{G\left(B_{i\ \text{simply periodic}}^{+}\right)}{G\left(\Omega_{\text{simply periodic}}\right)} = \frac{\sum_{n \in B_{i\ \text{simply periodic}}^{+}} \left(\prod_{j=1}^{k} \frac{a_j^{n_j}}{n_j!}\right)}{\sum_{n \in \Omega_{\text{simply periodic}}} \left(\prod_{j=1}^{k} \frac{a_j^{n_j}}{n_j!}\right)} \tag{24}$$

3.5 Complex Periodic System

In a complex periodic system in which the set of supported periods for persistent allocations is the union of subsets $\Psi = \bigcup_{p=1}^{s} \Psi^{<p>}$ each of which are individually simply periodic with cross subset period factors which are coprime (see Eq. (4) and Eq. (5)), the value of $\mathrm{lcm}(T_2/T, \dots, T_k/T) = \left(\prod_{p=1}^{s} T_{k_p}^{<p>}\right)/T^s$ where $T_{k_p}^{<p>}$ is the largest supported period in the subset $\Psi^{<p>}$. Therefore Eq. (17) and Eq. (18) reduce to:

$$C = \frac{\prod_{p=1}^{s} T_{k_p}^{<p>}}{T^{s-1}} \tag{25}$$

$$b_i^{<r>} = \frac{C}{T_i^{<r>}} = \frac{\prod_{p=1}^{s} T_{k_p}^{<p>}}{T^{s-1} T_i^{<r>}} \tag{26}$$

A complex periodic system for persistent allocations maps to an EMLM system with a dynamic bin sharing resource scheme assuming optimum packing and re-packing. This is because persistent resource allocations with periods from $\Psi^{<p>}$ cannot be time multiplexed in the same logical time slot as those with periods from $\Psi^{<q>}$ (where $p \neq q$) and vice versa without time domain collisions. The number of bins is equal to the number of logical timeslots as defined by the base period T for persistent resource allocations.

Therefore the set of valid states $\Omega_{\text{complex periodic}}$ for the complex periodic system is given by substituting Eq. (25) and Eq. (26) into Eq. (13) and Eq. (14) as follows:

$$\Omega_{\text{complex periodic}} = \left\{ n: 0 \le N_{\text{occupied timeslots}} \le T \right\} \tag{27}$$

where:

$$N_{\text{occupied timeslots}} = n_1 + \sum_{p=1}^{s} \left\lceil \sum_{j=2}^{k_p} \frac{n_j^{<p>} T}{T_j^{<p>}} \right\rceil \tag{28}$$

The set of blocking states for a Type i persistent allocation request is given by substituting Eq. (25) and Eq. (26) into Eq. (15) and Eq. (16) as follows.

For a Type 1 request:

$$B_{1\ \text{complex periodic}}^{+} = \{n: N_{\text{occupied timeslots}} = T\} \quad (29)$$

For a Type i request (excluding a Type 1 request) with a period that is a member of subset $\Psi^{<p>}$, the set of blocking states $B_{i\ \text{complex periodic}}^{<p>+}$ is given by:

$$B_{i\ \text{complex periodic}}^{<p>+} = \left\{ n: N_{\text{occupied timeslots}} = T, \left\lceil \sum_{j=2}^{k_p} \frac{n_j^{<p>} T}{T_j^{<p>}} \right\rceil - \sum_{j=2}^{k_p} \frac{n_j^{<p>} T}{T_j^{<p>}} < \frac{T}{T_i^{<p>}} \right\} \quad (30)$$

Therefore the blocking probability $P_{i\ \text{complex periodic}}^{<p>}$ for a Type i persistent allocation request with a period that is a member of subset $\Psi^{<p>}$ is given by reference to Eq. (9) as:

$$P_{i\ \text{complex periodic}}^{<p>} = \frac{G\left(B_{i\ \text{complex periodic}}^{<p>+}\right)}{G\left(\Omega_{\text{complex periodic}}\right)} = \frac{\sum_{n \in B_{i\ \text{complex periodic}}^{<p>+}} \left(\prod_{j=1}^{k} \frac{a_j^{n_j}}{n_j!} \right)}{\sum_{n \in \Omega_{\text{complex periodic}}} \left(\prod_{j=1}^{k} \frac{a_j^{n_j}}{n_j!} \right)} \quad (31)$$

4 Results and Validation

4.1 Introduction

In this section, we validate the theoretical models for the blocking probability P_i for a Type i request developed in Section 3 for simple and complex periodic systems of persistent uplink resource allocations assuming optimal packing and re-packing of such allocations. This involves comparing the predicted results from the theoretical models for various offered load scenarios against those arising from a discrete event simulation of a custom OPNET model. We also demonstrate via simulation the effect on blocking probabilities when optimal packing and re-packing of persistent resource allocations is not employed.

In the simulation, there is a single traffic source node for each of 4 types of persistent resource allocation. For a Type i request, the associated node generates requests with exponentially distributed inter-arrival times with rate λ_i and exponentially distributed service times with an expected residency time of $1/\mu_i$ such that the offered load is $a_i = \lambda_i/\mu_i$. It should be stressed that the offered load refers to the persistent resource allocations of a given type themselves; individual persistent resource allocations of a given type may carry raw data with different profiles e.g. one persistent resource allocation of Type i may carry periodic data while another may carry randomly arriving data according to a certain arrival distribution. The properties of the raw data are not considered in this paper, we assume that a request for a persistent resource allocation of a given type is appropriate to the raw data it is intended to carry.

At the time each request is generated in the simulation, a scheduler node determines whether the request can be accommodated as a persistent resource allocation based upon the current state n of persistent allocations currently being serviced and in particular whether there is sufficient available persistent resource allocation capacity remaining. If the request can be accommodated, the scheduler packs the new allocation as tightly as possible with any other ongoing persistent resource allocations as discussed in Section 2.7 and then the current state of the scheduler is updated; if it cannot, the request is effectively discarded to be served by dynamic scheduling instead and a blocked request count for that specific type of request is incremented. Therefore a sample blocking probability can be determined for each type of request by evaluating the ratio of the blocked request count versus the total request count for that specific request type. This sample blocking probability from the simulation can be compared to the theoretical blocking probability for different offered traffic load sets $\{a_i\}$.

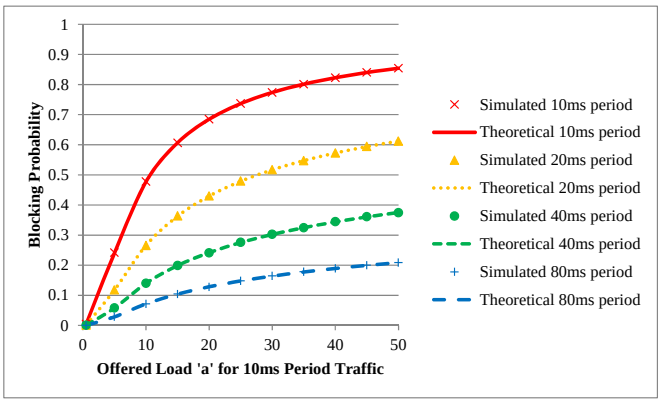
When a persistent resource allocation terminates in the simulation, the scheduler determines whether the remaining ongoing persistent resource allocations are still optimally packed as tightly as possible. If not, it initiates a re-packing as discussed in Section 2.7.

4.2 Simply Periodic System

In this section, we validate a simply periodic system with four persistent uplink resource allocation types with request periods $T_1 = T = 10\text{ms}$, $T_2 = 20\text{ms}$, $T_3 = 40\text{ms}$ and $T_4 = 80\text{ms}$. Clearly each request period T_j is an integer multiple of each lower request period T_i and the period ratio $c_{i,j}$ is given by $c_{i,j} = T_j/T_i = 2^{j-i}$.

We assume a physical time slot duration of 1ms, therefore there are 10 logical time slots to be shared between the four resource allocation types given that $T = 10\text{ms}$ is the lowest/base persistent resource allocation period supported by the system. We employ a base offered traffic load for persistent resource allocations of $a_i = \lambda_i/\mu_i = 5$ $(i \in \{1, 2, 3, 4\})$ and vary the offered traffic load for each resource allocation type in turn.

Figure 7(a)-(d) illustrate the blocking probability of each of the four resource allocation types as the offered traffic load of one of the types is varied while the others are held constant. Clearly there is close agreement between the simulated and theoretical results in all cases. The type with the smallest persistent uplink resource allocation period of 10ms is the hardest to accommodate since it requires a whole logical time slot to be available at the time of the request, and therefore the blocking probability is the highest for this type. Conversely, the type with the largest persistent uplink resource allocation period of 80ms is the easiest to accommodate since it requires only 1/8th of the resources associated with a logical timeslot to be available at the time of the request, and therefore the blocking probability is the lowest for this type.



(a)

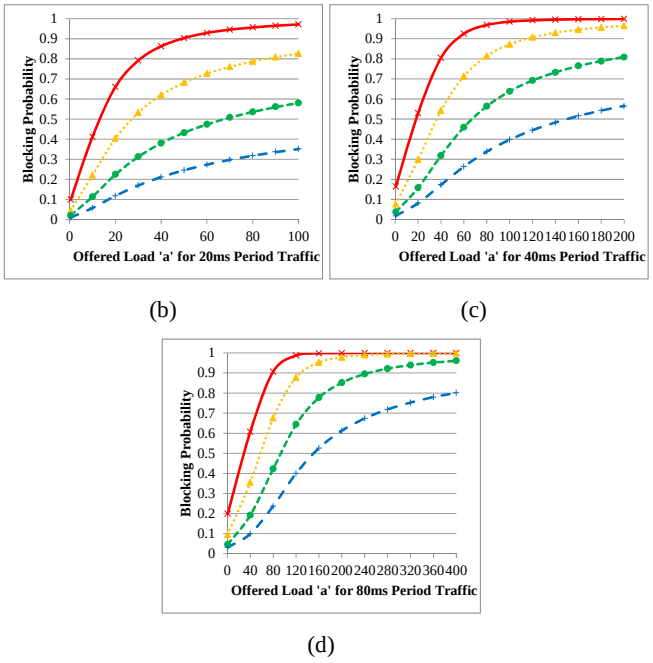


Figure 7. Blocking Probability for all Traffic Types vs Offered Traffic Load in a Simply Periodic System, Base Load of a=5 for all Traffic Types, Variation in Offered Traffic Load for: (a) 10ms Period Traffic (b) 20ms Period Traffic (c) 40ms Period Traffic (d) 80ms Period Traffic

4.3 Complex Periodic System

In this section, we validate a complex periodic system with persistent uplink resource allocation type periods grouped into two subsets $\Psi^{<1>}$ and $\Psi^{<2>}$ which are each individually simply periodic. The first subset $\Psi^{<1>} = \{10\text{ms}, 20\text{ms}, 40\text{ms}\}$ such that each request period $T_j^{<1>}$ is an integer multiple of each lower request period $T_i^{<1>}$ and the period ratio $c_{i,j}^{<1>}$ is given by $c_{i,j}^{<1>} = T_j^{<1>}/T_i^{<1>} = 2^{j-i}$. The second subset $\Psi^{<2>} = \{10\text{ms}, 30\text{ms}, 90\text{ms}\}$ such that each request period $T_j^{<2>}$ is an integer multiple of each lower request period $T_i^{<2>}$ and the period ratio $c_{i,j}^{<2>}$ is given by $c_{i,j}^{<2>} = T_j^{<2>}/T_i^{<2>} = 3^{j-i}$. The base period $T = 10\text{ms}$ is common to both subsets for convenience to show their simply periodic nature. An arbitrary logical time slot can be shared between persistent uplink resource allocation types with periods drawn solely from $\Psi^{<1>}$, and separately between types with periods drawn solely from $\Psi^{<2>}$, but not simultaneously between types with periods drawn from both $\Psi^{<1>}$ and $\Psi^{<2>}$ due to time domain collisions.

We again assume a physical time slot duration of 1ms, therefore there are 10 logical time slots to be shared between the five resource allocation types given that $T = 10\text{ms}$ is the lowest persistent resource allocation period supported by the system. We employ a base offered traffic load of $a = 5$ for each persistent allocation type request and vary the offered traffic load for each resource allocation type in turn.

Figure 8(a)-(e) illustrate the blocking probability of each of the five resource allocations types as the offered traffic load of one of the types is varied while the others are held constant. Clearly there is close agreement between the simulated and theoretical results in all cases. In general, as with the simply periodic system, resource types with larger periods are easier to accommodate because they require less resources and therefore their blocking probability is lower for any given scenario. However, the fact that certain types cannot share logical timeslots creates some interesting exceptions. For example, we see in Figure 8(e) that the blocking probability for the resource type with a 40ms period becomes larger than that for the resource type with a 30ms period as the offered load for the resource type with a 90ms period increases. This is due to the fact that the resource type with a 30ms period can share a logical timeslots with the resource type with a 90ms period, but the resource type with a 40ms period cannot.

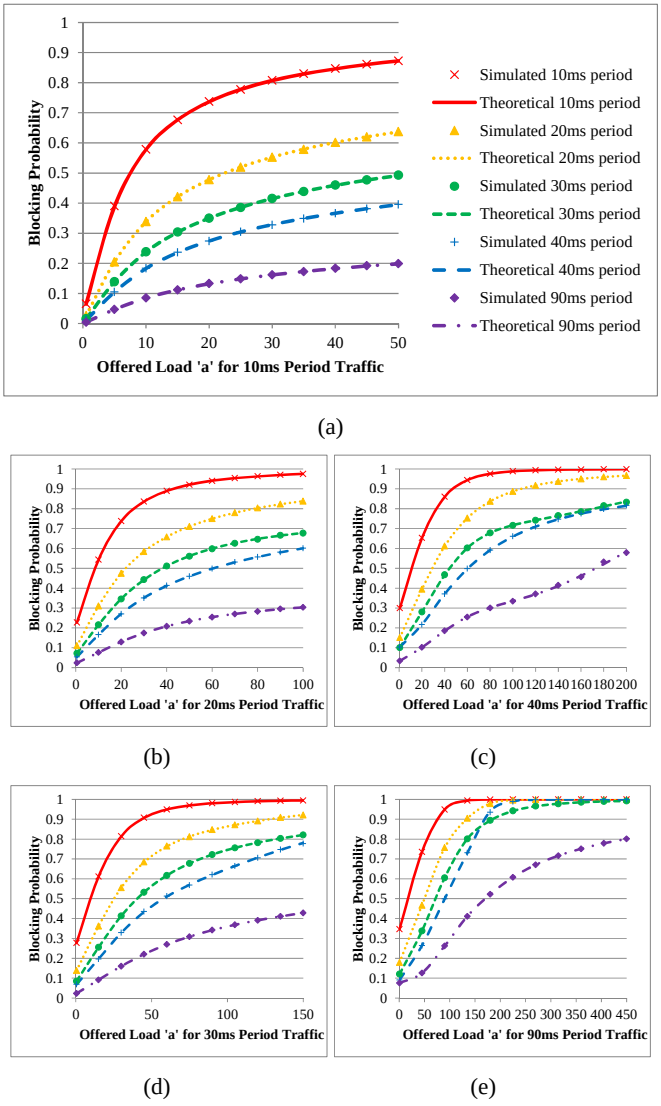


Figure 8. Blocking Probability for all Traffic Types vs Offered Traffic Load in a Complex Periodic System, Base Load of a=5 for all Traffic Types,Variation in Offered Traffic Load for: (a) 10ms Period Traffic (b) 20ms Period Traffic (c) 40ms Period Traffic (d) 30ms Period Traffic (e) 90ms Period Traffic

4.4 Effect of Not Employing Optimal Packing and Re-Packing

As discussed in Section 2, optimal packing and re-packing maximises the probability of maintaining a completely free logical time slot which can accommodate any persistent resource allocation type and in particular a Type 1 persistent resource allocation which requires a whole logical time slot. In this section, we investigate the

effect of not employing such optimal packing and re-packing via simulation. In particular:

- When a persistent resource allocation terminates, the remaining ongoing persistent resource allocations are not re-packed.

- When a new persistent resource allocation request occurs, the scheduler searches for a logical time slot that can accommodate the request. The search order of logical time slots is always the same and the search terminates immediately if and when an available logical time slot that can accommodate the request is discovered. Consequently there is still some implicit packing of persistent resource allocation requests together, but it is not necessarily an optimized packing. Different allocation algorithms could be employed with different results; we illustrate this particular algorithm as an example.

Figure 9 illustrates the simulated blocking probability of each of the four resource allocation types for the simply periodic system discussed in Section 4.2 when optimal packing and re-packing is not employed. A comparison is made to the simulated/theoretical blocking probability when optimal packing and re-packing is used.

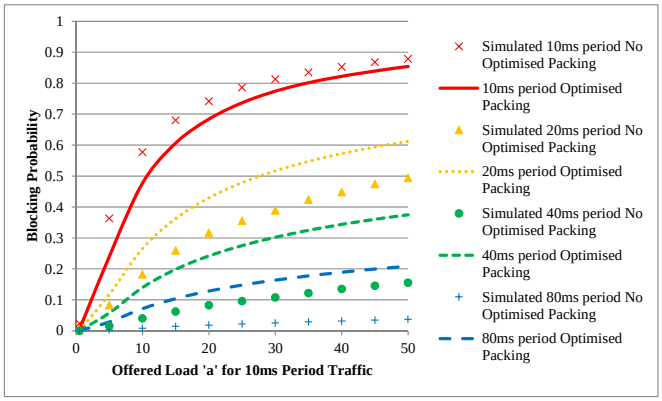


Figure 9. Blocking Probability for all Traffic Types vs Offered Traffic Load in a Simply Periodic System, Base Load of a=5 for all Traffic Types, Variation in Offered Traffic Load for 10ms Period Traffic

It is clear from Figure 9 that optimal packing and re-packing reduces the blocking probability of Type 1 persistent resource allocations by maximising the probability of maintaining a completely free logical timeslot. However, this is achieved at the expense of increasing the blocking probability of other types of persistent resource allocations. This is to be expected because when the blocking probability of Type 1 requests is reduced, more Type 1 traffic can be carried as a persistent resource allocation and therefore the available persistent resources for other types of traffic are reduced. It should be noted that a Type 1 persistent resource allocation request is equivalent in terms of resource volume to two Type 2 persistent resource allocation requests, four Type 3 persistent resource allocation requests or eight Type 4 persistent resource allocation requests in this simulation. Therefore reducing the blocking probability of Type 1 persistent resource allocations by only a relatively small amount increases the blocking probability of other types of traffic by a more significant margin.

Operationally, it is a subjective decision whether to use optimal packing and re-packing. Type 1 persistent resource allocations are the most difficult type of traffic to accommodate with the highest blocking probability, and optimal packing and re-packing provides a means to reduce this blocking probability at the expense of signalling and synchronisation to maintain an optimally packed resource set and increasing the blocking probability of other types of persistent resource allocation requests.

5 Conclusions

In this paper, we have derived and validated equations for the blocking probability of persistent resource allocations of different periods in simply periodic and complex periodic systems. This is applicable to wide area wireless systems supporting multiple M2M applications in which the use of persistent resource allocations can increase system efficiency and improve power management of devices. The utility of the blocking probability models is that they allow accurate assessment of the split between the M2M offered traffic of each type that can be carried over persistent resource allocations and that which is blocked from receiving a persistent resource allocation and must therefore be carried using conventional dynamic packet-by-packet scheduling. This facilitates system configuration (particularly in terms of the set of periods supported for persistent allocations), resource dimensioning and system performance characterisation.

Regarding system design and configuration, an operator is faced with many choices when offering persistent resource allocations for M2M applications. The most fundamental aspect is the set of periods offered. It is doubtful that a simply periodic system can provide a sufficiently granular set of periods to support a wide range of applications and therefore it is likely that a complex periodic system will be employed in practice. The smaller the period of a persistent resource allocation, the higher the blocking probability will be for the same incident load. However, we have seen that the operator does have some flexibility here in that employing optimal packing and re-packing of persistent allocation resources reduces the blocking probability of persistent resource allocations corresponding to the smallest period offered in the system at the expense of signalling and synchronisation and increasing the blocking probability of other types of persistent resource allocation requests. In addition, it is possible that an operator could attempt to equalise the blocking probabilities for the different persistent resource allocation types further in the admission control function by placing quotas on the number of persistent resource allocations of a given period which can be ongoing at any arbitrary time. This is not a technique we have investigated in this paper but it may be of use to operators facing different business requirements.

Future work will focus on the characterisation of the blocking probability for more general disciplines that allow a more granular set of persistent resource allocation periods to be supported than simply periodic and complex periodic systems. In addition, if a persistent resource allocation request from a device is blocked, the current paradigm is that the packets from this device will all be served via dynamic scheduling. A refinement of this scheme is to allow such a device to transition to a persistent resource allocation of the appropriate period once sufficient persistent resources become available. We also plan to characterise such a system.

Acknowledgment

This work has been supported by Ausgrid and the Australian Research Council (ARC).

REFERENCES

1. Potsch T, Khan Marwat SN, Zaki Y, Gorg C, Influence of future M2M communication on the LTE system, *Wireless and Mobile Networking Conference (WMNC), 2013 6th Joint IFIP*, pp.1-4, 23-25 April 2013, doi: 10.1109/WMNC.2013.6549000
2. Ghavimi F, Chen Hsiao-Hwa, M2M Communications in 3GPP LTE/LTE-A Networks: Architectures, Service Requirements, Challenges and Applications, *Communications Surveys & Tutorials, IEEE* , October 2014, doi: 10.1109/COMST.2014.2361626
3. oneM2M, oneM2M-TR-0001-UseCase, *oneM2M Use cases collection*, v0.0.5, Sep. 23, 2013.
4. 3GPP TS 22.368 V12.4.0 (2014-06), Service requirements for Machine-Type Communications (MTC), Stage 1, Release 12
5. IEEE 802.16p-2012, IEEE Standard for Air Interface for Broadband Wireless Access Systems-Amendment 1: Enhancements to Support Machine-to-Machine Applications
6. Laya A, Alonso L, Alonso-Zarate J, Is the Random Access Channel of LTE and LTE-A Suitable for M2M Communications? *A Survey of Alternatives, Communications Surveys & Tutorials, IEEE*, **vol.16, no.1**, pp.4-16, 1Q 2014, doi: 10.1109/SURV.2013.111313.00244
7. Hasan M, Hossain E, Niyato D., Random access for machine-to-machine communication in LTE-advanced networks: issues and approaches, *Communications Magazine, IEEE*, **vol.51, no.6**, pp.86-93, June 2013, doi: 10.1109/MCOM.2013.6525600
8. 3GPP TS 36.321 V8.10.0 (2011-09), Evolved Universal Terrestrial Radio Access (E-UTRA); Medium Access Control (MAC) protocol specification, Release 10
9. Shao-Yu Lien, Kwang-Cheng Chen, Massive Access Management for QoS Guarantees in 3GPP Machine-to-Machine Communications, *Communications Letters, IEEE*, **vol. 15, no. 3**, pp. 311-313, March 2011, doi: 10.1109/LCOMM.2011.011811.101798
10. You C, Zhang Y, A radio resource scheduling scheme for periodic M2M communications in cellular networks, *Wireless Communications and Signal Processing (WCSP), 2014 Sixth International Conference on*, pp.1-5, 23-25 Oct. 2014, doi: 10.1109/WCSP.2014.6992085
11. Gotsis Antonis G, Athanasios S Lioumpas, Angeliki Alexiou, Analytical modelling and performance evaluation of realistic time-controlled M2M scheduling over LTE cellular networks. *Transactions on Emerging Telecommunications Technologies* **24, no. 4 (2013)**: pp. 378-388, doi: 10.1002/ett.2629
12. Hu Jin, Ju Young Lee, Dan Keun Sung, On the efficiency of persistent scheduling for non-periodic real-time services in IEEE 802.16e system, *Personal Indoor and Mobile Radio Communications (PIMRC), 2010 IEEE 21st International Symposium on*, pp.1481,1486, 26-30 Sept. 2010, doi: 10.1109/PIMRC.2010.5671974
13. Afrin Nusrat, Brown Jason, Khan Jamil Y, An Adaptive Buffer Based Semi-persistent Scheduling Scheme for Machine-to-Machine Communications over LTE, *Next Generation Mobile Apps, Services and Technologies (NGMAST), 2014 Eighth International Conference on*, pp.260-265, 10-12 Sept. 2014, doi: 10.1109/NGMAST.2014.48
14. Afrin N, Brown J, Khan JY, Performance evaluation of an adaptive semi-persistent LTE packet scheduler for M2M communications, *Signal Processing and Communication Systems (ICSPCS)*, 2014 8th International Conference on, pp.1-7, 15-17 Dec. 2014, doi: 10.1109/ICSPCS.2014.7021125
15. Brown J, Afrin N, Khan, JY, Delay Models for Static and Adaptive Persistent Resource Allocations in Wireless Systems, *Mobile Computing, IEEE Transactions on*, **vol.PP, no.99**, 2015, doi: 10.1109/TMC.2015.2492546
16. Daniel E Flath, *Introduction to Number Theory*. New York: Wiley, 1989.
17. Myoung-Jo Jung, Yeong Rak Seong, Cheol-Hoon Lee, Optimal RM scheduling for simply periodic tasks on uniform multiprocessors, *Proceedings of the 2009 International Conference on Hybrid Information Technology (ICHIT '09). ACM*, New York, NY, USA, Pages 383-389, doi: 10.1145/1644993.1645064
18. D Muller, Accelerated Simply Periodic Task Sets for RM Scheduling, *Embedded Real Time Software and Systems*, May 2010.
19. Kaufman J, Blocking in a Shared Resource Environment, *Communications, IEEE Transactions on*, **vol.29, no.10**, pp.1474,1481, Oct 1981, doi: 10.1109/TCOM.1981.1094894
20. Logothetis MD, Moscholios ID, Teletraffic models beyond Erlang, *ELEKTRO*, 2014, pp.10-15, 19-20 May 2014, doi: 10.1109/ELEKTRO.2014.6847860
21. Yuan Zhang, Tree-Based Resource Allocation for Periodic Cellular M2M Communications, *Wireless Communications Letters, IEEE*, **vol.3, no.6**, pp.621,624, Dec. 2014, doi: 10.1109/LWC.2014.2366769

APPENDIX A: DERIVATION OF THE EMLM

In this appendix, we provide a condensed derivation of the EMLM as first presented in [19] to aid the reader who is not familiar with this model.

THE MODEL

The resource has a finite capacity of C units, customers arrive at a mean rate λ and have two requirements:

1) a spatial requirement - b units of the resource

2) a temporal requirement - the resource units are required for τ units of time.

No assumption is made about which b units of the resource are needed - any b units will do. Thus, in the message storage context, we assume that a message of size b can be stored in any b (not necessarily contiguous) units.

A. General Assumptions

Al) k customer types, each with distinct spatial and/or temporal requirements. k is finite, but arbitrary.

A2) A customer whose spatial requirement cannot be satisfied is blocked and departs without further affecting the system.

B. Stochastic Assumptions

Bl) The customer arrival process is a stationary Poisson process (mean rate λ).

B2) The spatial requirement b is an arbitrary discrete random variable ($P\{b = b_i\} = q_i, i = 1, \dots, k$).

B3) A customer with spatial requirement b_i has residency time τ_i whose distribution has a rational Laplace transform and whose mean is denoted by $1/\mu_i$.

Assumptions B1) and B2) imply that customers with resource requirements b_i arrive according to a Poisson process with mean rate $\lambda_i = \lambda q_i$.

Conspicuously lacking thus far is any description of how customers share the resource. Thus, for example, a resource may be completely shared, or certain customers may have exclusive use of portions of the resource with the excess commonly shared. Obviously, the resource sharing policy has a strong effect on the blocking experienced by different customers. The state distribution exhibited later allows for completely arbitrary resource sharing policies.

THE STATE DESCRIPTION

In order to set notation and discuss the state distribution, we assume (in this section only) that all residency times are exponentially distributed. Given this assumption, it is easy to see that the appropriate state description is:

$$n = (n_1, \dots, n_k)$$

where n_i = number of type i customers using the resource. If each of the C units of the resource is viewed as a server, a type i customer simultaneously requires b_i servers. However, unlike blocking models with batched Poisson arrivals, all b_i servers must be simultaneously relinquished when the customer departs. Note that the above state description captures this effect. The remaining notation needed is as follows.

Ω = set of allowable states (determined by the resource sharing policy in effect)

$$n_i^+ = (n_1, \dots, n_i + 1, \dots, n_k).$$

$$n_i^- = (n_1, \dots, n_i - 1, \dots, n_k).$$

$$\delta_i^+(n) = \begin{cases} 1 & if\ n_i^+ \in \Omega \\ 0 & otherwise \end{cases}$$

$$\delta_i^-(n) = \begin{cases} 1 & if\ n_i^- \in \Omega \\ 0 & otherwise \end{cases}$$

$$n \cdot b = \sum_{i=1}^{k} n_i b_i$$

Because each resource sharing policy gives rise to a set Ω of allowable states, it is useful to view any set Ω as a resource policy provided that:

$$n \in \Omega \Rightarrow n_i \geq 0 \qquad i = 1, \dots, k \qquad n \cdot b \leq C$$

Realistic policies, of course, impose much more structure on such sets.

The performance measure of primary interest is the probability P_{b_i} that a type i arrival (requiring b_i units of the resource) is blocked. If $P(\cdot)$ denotes the state distribution, given that policy Ω is in effect, then:

$$P_{b_i} = \sum_{n \in B_i^+} P(n)$$

where $B_i^+ = \{n \in \Omega : n_i^+ \notin \Omega\}$.

The state distribution is given by the following.

Theorem 1: The state distribution corresponding to an arbitrary resource sharing policy Ω is given by:

$$P(n) = \prod_{i=1}^{k} \frac{a_i^{n_i}}{n_i!} G^{-1}(\Omega), \qquad all\ n \in \Omega \tag{A.1}$$

where:

$$G(\Omega) = \sum_{n \in \Omega} \left(\prod_{i=1}^{k} \frac{a_i^{n_i}}{n_i!} \right) \tag{A.2}$$

The type i offered load $a_i = \lambda_i / \mu_i$ where $\lambda_i = \lambda q_i$ and $1/\mu_i$ is the mean residency of type i customers.

Proof: The Markovian equilibrium balance equation, valid for an arbitrary resource sharing policy, can be written down by inspection:

$$\left[\sum_{i=1}^{k} \lambda_i \delta_i^+(n) + \sum_{i=1}^{k} n_i \mu_i \delta_i^-(n) \right] P(n) = \sum_{i=1}^{k} \lambda_i \delta_i^-(n) P(n_i^-) + \sum_{i=1}^{k} (n_i + 1) \mu_i \delta_i^+(n)\, P(n_i^+) \quad for\ all\ n \in \Omega \tag{A.3}$$

Moreover, the local balance equation is given by:

$$\lambda_i \delta_i^-(n) P(n_i^-) = n_i \mu_i \delta_i^-(n) P(n) \quad i = 1, \dots, k \quad all\ n \in \Omega \tag{A.4}$$

The proof follows from the observation that the distribution (A.1) is a solution to the local balance equation (A.4), and that any solution to (A.4) must also be a solution to (A.3).

Corollary: The blocking experienced by a type i customer $i = 1, \dots, k$ when policy Ω is in effect is given by

$$P_{b_i} = \frac{G(B_i^+)}{G(\Omega)}$$