Adaptive Scheduling of High Bandwidth Periodic Packet Switch Sources for Congestion Handling

Suranga Prasad

(169231K)

Degree of Master of Science

Department of Electronic and Telecommunication Engineering

University of Moratuwa
Moratuwa, Sri Lanka

April 2022

Abstract

Industrial Internet of Things (IIoT) networks are quite prevalent in many types of sectors today. Also, in the near future, many sensors are estimated to be connected and used. With the development of sensor technologies, they transmit extra data to the core network. Optimization methods are crucial to cater to this high bandwidth traffic without expanding the aggregation servers and links.

This work has proposed a two-tier IIoT network architecture with several scheduling algorithms at tier one (Data Loggers). The network has been simulated using Omnet++. The traffic distribution at the aggregation layer has been compared for several scheduling algorithms to get the best possible method to optimize the network without using a centralized scheduler or controller. The results show that out of the proposed algorithms, the best methods in terms of distributing traffic are Initial Uniform Slot Selection (IUSS) and Initial Bernoulli Slot Selection (IBSS) methods.

Declaration

I declare that this is my own work and this thesis does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my thesis, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature

Date

The above candidate has carried out research for the Masters thesis under my supervision.

Signature of the supervisor

Date

Acknowledgement

This work was carried out during the years 2017-2022 at the Department of Electronic and Telecommunication Engineering, University of Moratuwa.

I owe my deepest gratitude to my supervisors Dr. Upeka Premaratne for his guidance to successfully complete this research. I would like to thank Dr. Tharaka Samarasinghe, Dr. Kasun Hemachandra, Mr. Damith Kandage and all the authors and owners of the materials that have been referred in this thesis and finally, I would like to thank all the people who supported to successfully complete the research.

Contents

List of Figures

List of Tables

List of Abbreviations

CAPEX Capital Expenditure

DL Data-Logger

FIFO First-in/first-out

GA Global-Aggregator

IBSS Initial Bernoulli Slot Selection

IIoT Industrial Internet of Things

IoT Internet of Things

IUSS Initial Uniform Slot Selection

LAN Local Area Network

QoS Quality of Service

RBSS Random Bernoulli Slot Selection

RCSP Rate Controlled Static Priority

RR Round Robin

RUSS Random Uniform Slot Selection

RUSSPI Random Uniform Slot Selection with Path Initialization

SF Switching Fabric

SP Static Priority

Nomenclature

τ	Message generation time
a	Lower bound on data
b	Shape parameter
i	A message generating node
k	Sampling period number
m	Number of Bernoulli trials
M_s	Number of sensors connected with one data logger
M_{dl}	Number of data loggers connected with aggregator
N	Number of time slots in a sampling period
n	Selected time slot by a data logger
p	Probability of success
T	Sampling time period
t	First message sending time
x	Random variable
Y_c	Packet size of a SYN and ACK messages
Y_d	Packet size of a data chunk

Chapter 1

INTRODUCTION

The available bandwidth of communication networks has seen a significant increase in the recent past with the further promise of additional bandwidth due to significant developments in the terahertz region [1] [2]. However, demand for bandwidth from Industry 4.0 applications continues to fill up this newly available bandwidth. This is mainly due to the proliferation of IIoT [3] [4] to allow increased connection of sensors and the integration of big data algorithms to extract meaningful results [5] [6] [7] [8] [9] with applications such as predictive maintenance [10] [11], forensic analysis (traceability, security etc.) [8] [12] [13], model predictive control [14] and self-optimization. The downside of this is the ensuing congestion which negatively affects performance by causing higher latency [15] and packet drops (i.e., data loss). In the case of the machine-to-machine traffic, methods for facilitating congestion include methods such as data reduction through compression [16] [17] and event-based sampling [18] [19] [20] ,of which the latter can compromise accuracy.

This thesis focuses on distributing network traffic through buffering and scheduled transmission. This requires the data whether compressed or not, to be delay-tolerant to some extent. Furthermore, periodic data is also desired, or the time-division scheduling will become resource inefficient due to under-utilization. Examples of such sources include smart meters and data loggers such as those of a chemical plant. If all such sources were to transmit at once, a large traffic burst may result in a higher risk of packet drops [21] [22] which could be potentially detrimental in the event of a critical packet being dropped. Therefore, by evening out the traffic, such a situation can be avoided. This thesis looks into strategies for distributing such traffic in a collision-free manner.

Chapter 2

LITERATURE SURVEY

2.1 Network Traffic Modeling

To illustrate different types of traffic, different models have to be used. For instance, packet-switched network traffic of a backbone link can be described using a sum of Bernoulli sources [23]. The main attributes of traffic sources are their average data rate, burstiness, and correlation [24]. Average data rate refers to the amount of data transferred per unit of time. Burstiness is a sense that involve data that is transmitted intermittently, and it affects network congestion and data losses. The correlation is the relation between packet arrivals at different times.

Since simple traffic models are essentially counting processes that count the number of packets that arrive in a specific time interval, they are sometimes referred to as point processes. These point processes can produce a random sequence representing the time intervals between packets. Several random processes are combined to create more complex traffic patterns [25] [24].

The use of fluid flow models is the other extreme in traffic modelling [25]. The traffic is divided into flows based on average and burst data rates in fluid flow modelling. The goal of these models is to look into traffic at the aggregate level, such as Ethernet traffic or traffic arriving at Internet service provider ingress and egress points. Individual packet arrivals and departures are not considered in fluid flow models [26].

2.1.1 Poisson Distribution Model

The Poisson Model is one of the most extensively used and oldest traffic models. In traditional telephone networks, the memory-less Poisson distribution is the most commonly used model for traffic analysis [26]. If the arrivals come from many independent sources (known as Poisson sources), the Poisson distribution is appropriate. The mean and variance of the distribution are equal to the parameter λ [26].

Poisson processes have several interesting analytic characteristics. First, the superposition of separate Poisson processes yields a new Poisson process with a rate equal to the total component rates. Second, the independent increment property renders Poisson a memoryless process. As a result, queuing issues involving Poisson arrivals are considerably simplified [27]. Third, Poisson processes are commonly used in traffic applications that include many distinct traffic streams, each of which can be highly general. The theoretical basis for this phenomenon is known as Palm's Theorem. This theory has been widely applied in many fields. It was mathematically improved by Khintchine and is widely relevant in many fields [28].

Therefore the theorem is called as Palm-Khintchine theorem. It states that aggregate traffic can be approximated with a Poisson process under specific conditions and for a more significant number of streams [29]. So, traffic streams on main communications arteries are mainly considered Poisson processes. Nevertheless, aggregated traffic does not need to be a Poisson process always.

2.1.2 Self-Similar Traffic

Self-Similar signals replicate themselves in time scale with multiplicative changes [30]. Thus a self-similar waveform will have the same shape if we scale the time axis up or down [24]. To explain traffic on Ethernet LANs and variable-bit-rate video services self-similar statistical model can be used [31] [32]. The main attribute of this model is the presence of "equally-looking" bursts at any time scale [26].

The result of self-similarity is for introducing long-range auto-correlation into the traffic stream, which is perceived in practice. This is a reason for higher traffic volumes, although the average traffic density is low. So, if a switch or router is selected due to Poisson traffic-based simulations, it will be experienced unexpected buffer overflow and packet losses [24].

2.1.3 Pareto Distribution Process

This distribution is different from a normal distribution, and it tells, that most of the data exist at the extremes. The probability and cumulative distribution functions are mentioned below.

$$f(x) = \frac{ba^b}{x^{b+1}} \tag{2.1}$$

$$F(x) = 1 - \left(\frac{a}{x}\right)^b \tag{2.2}$$

The parameter a is the lower bound, b is the shape, and the random variable x has values limited in the range $a \leq x \leq \infty$. This distribution is applied to self-similar model arrivals in network traffic [33]. The essential attributes of the Pareto distribution are that it has an infinite variance when $b \geq 2$ and achieves an unlimited mean when $b \leq 2$.

2.1.4 Flow Traffic Models

Flow traffic models obscure the intricacies of the many traffics moving through the network and substitute them with flows with several defining factors. The models that result are simple to create, measure, and monitor [24].

A flow contains consistent addressing and service requirements for an end-to-end application [34]. These specifications establish a flow specification, sometimes known as a flow spec, used in throughput and service modelling. Single flows from different sessions or applications are joined to form composite flows with the same path, connection, or service needs [26]. When the network reaches a specific hierarchy level, composite flows are integrated into backbone flows. It's easier to incorporate flow features and deal with a smaller data collection when you describe flows in this way. Depending on the Quality of Service (QoS) required by the users, a core router could divide incoming data into individual flows, composite flows, and backbone flows [24]. As a result, the number of service queues is reduced, and the scheduling method in the router is simplified.

In most networks, most flows are low-performance backbone flows; some composite flows will also present, but high-performance individual flows will be limited. High-performance flows will influence the design of the switch's scheduling algorithm and the size and number of queues required because they often have demanding latency and/or bandwidth requirements [24].

2.1.5 Leaky Bucket Algorithm

Traffic bursts which lead to data losses and network congestions could be observed in any network. So, several mechanisms have been proposed to regulate and smooth out these bursts, such as traffic shaping and policing [24].

The leaky bucket is a traffic shaping algorithm that controls the maximum traffic rate arriving from a source. This algorithm defines a maximum rate that the aggregation node can handle. The leaky bucket forwards the traffic if the input rate is lower than the maximum. If the input rate is higher than the maximum rate, leaky bucket forward traffic at the maximum rate and excess data is buffered. If the buffer is fully utilized, the excess data will be lost [35].

2.2 Scheduling Algorithms

To use limited resources of the network elements effectively, a scheduling algorithm must be applied to each switch or router. The most critical resources are link bandwidth and buffer spaces. The key features of a scheduler in a network are, providing required QoS, selecting the least damaging packet to drop during congestions and providing impartial sharing of network resources among different users [24].

- Packet selection policy

  Several packets will demand access to a specific output port in an ordinary switch or router. The next packet to be transmitted should be selected and rest of the packets must be kept in an intermediate buffer. According to the service types of the queues, multiple categorization policies could be established according to the types of queues. Some programs, for example, have strict real-time latency and jitter limits, whilst other applications accept to change their behaviour depending on the network condition. Many Internet services are handled utilizing a best-effort packet transport policy without having bandwidth or delay assurances at the time of writing [36] [37].

- Packet dropping policy

  Because packets must be kept in each network switching node, buffer storage in a switch is a resource that must be shared among multiple users or sessions. The switch buffers fill up during periods of congestion, and the scheduler must determine which packets to drop [38] [39] [40].

- Fair Sharing Policy

  Users must share switch resources such as available output link bandwidth and local buffer space. Due to the many types of services provided by the switch, equitable resource sharing is not the ideal solution. Instead, the scheduler must distribute these finite resources fairly so that each user receives their fair portion based on their service class. Isolation or protection is another issue related to equitable sharing. This is required because not all users abide by their agreed-upon data rate. When this happens, the misbehaving user begins to consume resources at the expense of other users who are behaving properly [41].

2.2.1 Scheduling as a Problem of Optimization

From the foregoing, it is intelligible that the scheduling problem is an optimization problem, as the scheduler distributes limited resources among user traffic, affecting the quality of

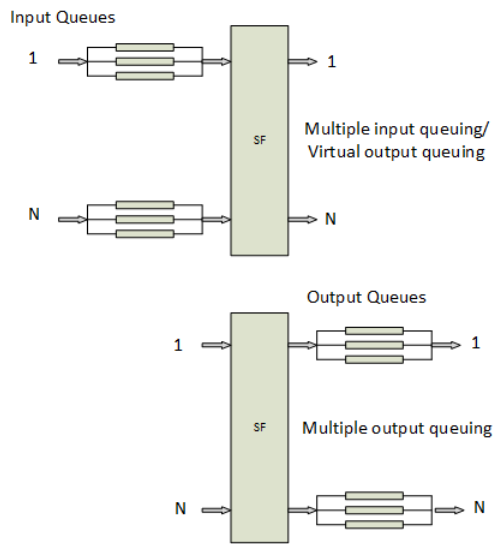

Figure 2.1: Virtual output queuing (VOQ) and output queuing switches with different queues for each service class

QoS. The scheduling techniques mentioned in the following sections are entirely heuristic, and there is no mathematical proof that the proposed scheduling policy is optimal [24].

Based on the transportation problem optimization technique, the author created a hierarchical scheduler [42]. Over the use of the transportation problem technique [42], [43], different QoS kinds such as delay-sensitive traffic and bandwidth-sensitive traffic can be optimized through the same switch.

Finding the best answer to a transportation challenge is complex and may take some time. However, using computational geometric notions, the author devised a simple greedy method for addressing the transportation problem. Because the approach uses simple add/subtract operations, it is quick to compute [42].

2.2.2 Scheduler Location

A scheduler should be placed in the switch or router where packets are buffered and where packets must share a resource like a switch fabric or the output links.

Each service class should have its dedicated buffer if the switch can support multiple quality services. Figure 2.1 shows two buffer locations in switches. The SF rectangles represented the switching fabric responsible for providing connectivity between each input and output port. The buffers are situated at each input port in the top figure of an input queuing switch. The figure on the bottom shows an output queuing switch where the buffers are located at each output port. Multiple buffers are displayed at each input or output port when numerous service classes are supported.

Regardless of the buffering mechanism used, packets will compete for shared resources, necessitating the use of a scheduling algorithm. There are three types of shared resources depicted in the diagram: (1) the buffer storage space; (2) the switch fabric (SF); and (3) the output links (i.e., available bandwidth) [24].

2.2.3 Scheduler Design Challenges

Scheduler designers have to face several issues and challenges, which have been reviewed in the following subsections.

- Priority

  Equality is not acceptable for network traffic handling in many cases. Because some packets are more important than others. So, a parameter to differentiate packets from each other is called priority value [44]. There are two types of methods to assign priority for packets. They are static and dynamic. Static priority does not change over time, whereas dynamic priority allocation changes according to the network's time and traffic volume.

- Degree of Aggregation

  The scheduler must store state information of each user or connection to deliver guaranteed QoS. When using that kind of scheduler, designers have to face several difficulties such as constraints on scaling, implementation challenges, and mapping between application and network service parameters [45]. Many states reduce the scheduler's performance and limit the number of users that can handle. Other schedulers group numerous connections into classes to decrease state information and workload. This method is more productive because it deals with enormous network traffic and has a configurable per-hop behaviour [46]. Even when the quality level varies, the differentiated services scheduler maintains stable ratios of quality of service ratios between service classes. The cost of aggregating traffic is the loss of deterministic QoS guarantees, as each connection's state is lost. High-level aggregation servers assure QoS on a probabilistic basis. In other words, because the scheduler only maintains track of groups of users, it loses precise information about each user's condition. The number of states that must be reviewed and updated is reduced. For groups of users, guarantees on QoS can be made, but each user cannot be guaranteed a precise quality of service [24].

- Work-Conserving Versus Nonwork-conserving

  A work-conserving method is idle when all of the priority queues are empty. A nonwork-conserving method may not transfer a packet even when the priority queues are full.

This may occur, for example, to reduce delay jitter [47]. In theory, this is an excellent idea, but it is never implemented. Typically, work-conserving algorithms produce less average delay than nonwork-conserving algorithms.

Work-conserving methods include generalized processor sharing [48], weighted fair queuing [49], virtual clock [50], weighted round-robin [51], delay earliest-due date [52], and deficit round-robin [53]. Non-work-conserving techniques include stop-and-go, jitter earliest-due date [54], and Rate Controlled Static Priority (RCSP).

- Packet Drop Policy

  Schedulers must choose which packet to serve next and which packets to drop if the system's resources become overburdened. Dropping packets can be done in various ways, including dropping packets that come after the buffer reaches a specific level of occupancy (this is known as tail dropping) [55]. Another alternative is to drop packets based on their priority from any point in the buffer. A third option is to drop packets at random whenever the system's resources become overloaded [24] [38] [39] [40].

2.2.4 Rate-Based Versus Credit-Based Scheduling

Rate-based and time-based scheduling methods are two types of scheduling methods. A rate-based scheduler chooses packets depending on the data rate that the service class has been assigned. Virtual clock [56], weighted fair queuing [57] [58], hierarchical round-robin [59] [60], deficit round-robin [61] [62], rate-controlled static priority [63] are few examples for rate-based scheduling methods. Rate-based procedures provide for a wide range of approaches to determining a service policy [64]. Some of the strategies are difficult to execute because they necessitate complicated mathematical calculations and knowledge of the statuses of various flows. However, because they translate these requirements into equal bandwidth, they can only make guarantees based on the maximum delay or delay jitter limitations. This is the algorithm's weak point because offering bandwidth guarantees to achieve latency overlooks the traffic stream's real bandwidth requirements. This would result in an unfair allocation of bandwidth to other services with legitimate QoS bandwidth requirements [24].

Time-based schedulers select packets based on the arrival time. The earliest-due-date for delay [65], earliest-due-date for jitter [66] [65], and lowest response time [67] are few examples for time-based scheduling methods. Scheduler-based approaches necessitate keeping track of particular packet arrival timings and calculating packet priority based on the packet arrival time and the deadline. To give end-to-end delay and delay jitter assurances, the scheduler must be deployed at all switching nodes, and incoming traffic must closely match the predicted model [24].

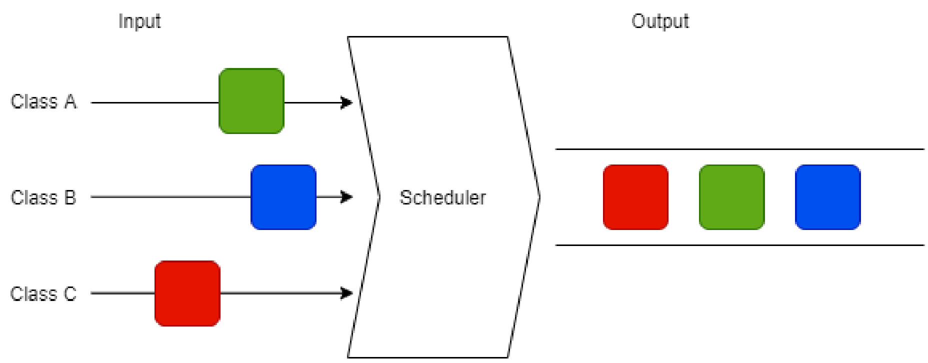

Figure 2.2: FIFO Scheduling

2.2.5 Analysis of Common Scheduling Algorithms

The rest of this chapter goes over various scheduling algorithms, ranging from those that provide best-effort traffic with no performance guarantees to those that support guaranteed services with bandwidth and delay limitations.

1. First-in/first-out (FIFO)

   FIFO is a prevalent method which serves based on the arrival time of the user [68] [69]. All users have equal priority, and the arbitration rule is based on packet arrival time. This approach is commonly used to store incoming packets in queues that can be linked to input or output ports. Figure 2.2 shows the FIFO algorithm.

   FIFO is straightforward to set up, and packet insertion and deletion are especially simple, requiring no session information to be maintained. However, because the server cannot discriminate between packets belonging to different users, this becomes a disadvantage. As a result, certain users may misbehave and fill up a considerable percentage of the queue, increasing the likelihood of other users' packets being dropped [24].

   When all incoming flows are queued in a single queue, greedy users that engage in long bursts of activity will take up the most of the queue capacity, putting other flows at risk. Packets at the back of the line are dropped when the queue is full. This is why this strategy is referred to as FIFO with tail drop [70] [71]. Because of its simplicity, this method is used by the majority of routers.

   FIFO does not provide per-connection latency or rate guarantees since priority is determined by arrival time. Limiting the queue size so that the maximum delay equals the time it takes to serve a full queue is one technique to create delay constraints. Naturally, once the queue size is restricted, there is a chance that an arriving packet will be deleted if it arrives when the queue is already full [24]. The number of sessions should be limited to minimize the likelihood of packet discard.

2. Static Priority (SP) Scheduler

   Different priorities are allocated to different queues in a static priority scheduler. Incoming data is forwarded to the appropriate queues based on its priority. Only if all of the higher priority queues are empty do the scheduler service packets in a lower priority queue [69] [72] [73].

3. Round Robin (RR) Scheduler

   Backlogged sessions are served in a defined order by the round-robin scheduler [74] [75]. When the scheduler finishes transmitting a packet from session 1, for example, it advances to session 2 and checks if any packets are ready to be transmitted. The scheduler returns to the initial session after running through all of the sessions, hence the name round-robin [24].

   This scheduler's key features include ease of hardware implementation and protection of all sessions, including best-effort sessions [76]. A greedy or uncontrollable user will be unable to send more than one packet every round. As a result, other users will not be penalized. Such an user will only succeed in overflowing its buffer, resulting in data loss. However, the long packets will be transferred because the scheduler serves total packets. Other sessions with short packets to send may experience a delay due to this.

2.3 Aggregated Traffic Patterns Analysis

Traffic generated from distributed sensor nodes is aggregated into the Internet of Things (IoT) architecture at different levels. Those points can be a gateway or load balancers of an IoT cloud system. To evaluate the resource requirement of such systems, analytical approaches are essential, since test-beds and simulations are constrained by several factors. From this section, we will specially discuss the superposition of periodic traffic processes. For a large number of asynchronous nodes, the *Palm-Khintchine* theorem can be used [77] [29] to approximate the aggregated traffic to a Poisson process.

2.3.1 Periodic behavior of sensor networks

We can roughly classify IoT sensor networks into two types according to their communication modes. They are periodic and event-based networks [78] [79]. An example of an event-driven application is a motion detection sensor in a smart home. They have triggered an event only if detected a moving object. But, even in this example, someone can assume this has a periodic behavior as well. For instance, leaving and returning to the home almost equals in every day. So it is predictable and periodic [80].

Naturally, most of the sensor networks types are periodic. A well-known example of these types of sensor networks is smart grids. This includes management, supervision, and maintenance of power generation and distribution network other than collecting of current power usage from industrial and residential smart meters [81].

In this thesis, we have considered the sensor nodes are generating messages and transmitting towards the aggregating points periodically.

Definition 1. *Periodic Traffic: If the traffic generating process is periodic, a single node i generates messages at time $\tau_{i,k}$ in sampling periodic k (which is a natural number) such that*

$$\tau_{i,k} = t_i + kT_i \qquad (k \in \mathbb{N}) \tag{2.3}$$

The time between two messages are equals to a constant T_i and the first message sending time $t_i = \tau_{i,0}$.

2.3.2 Superposition of a large number of binomial random processes

According to the de Moivre–Laplace theorem, which is a special case of the central limit theorem, states that the normal distribution may be used as an approximation to the binomial distribution under certain conditions. In particular, the theorem shows that the probability mass function of the random number of "successes" observed in a series of m independent Bernoulli trials, each having probability p of success (a binomial distribution with m trials), converges to the probability density function of the normal distribution with mean mp and standard deviation $mp\sqrt{1-p}$, as n grows large, assuming p is not 0 or 1.

Chapter 3

THE SYSTEM MODEL AND PROBLEM FORMULATION

3.1 Problem Formulation

Due to the power constraints, it is infeasible to design a large-scale wireless sensor network using centralized algorithms [82]. As a solution, distributed IoT network models were introduced. Due to the decentralized architecture, it is really important to have a method to manage congestion using a proper scheduling method [83]. It can reduce the link and central nodes' resource utilization, traffic drop, and delay.

FIFO and *RR* scheduling methods are simple to implement. But, if the input rate is higher than the scheduler's output rate, packets are dropped without considering the message's priority. Static priority and adaptive priority methods are important if the network considers the QoS. But, these scheduling methods are based on a centralized server. To implement the network, proper central nodes are required. Most probably, they increase the CAPEX cost.

Time slot synchronization algorithms [84] [85] [86] are specially designed for decentralized networks. For time synchronization, sensors have to exchange their information with each other. This adds extra signaling traffic to the network. Similarly, protocol based congestion control methods [87] [88] [89] [90] also add extra signalling traffic to the network. On other hand, sensors and other network elements should be customized to handle those special protocols.

3.2 System Model

The thesis suggested a two-tier network model for a decentralized sensor network with simple random slot selection methods to minimize the congestion at the aggregation layer by evening

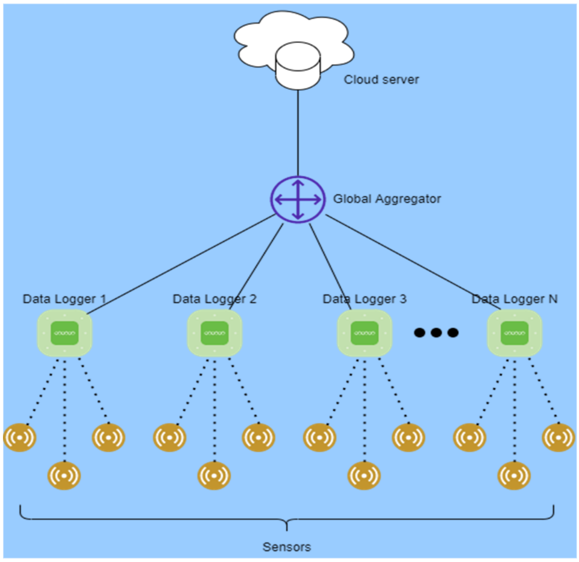

Figure 3.1: System Model

out the traffic without using centralized management servers or adding extra traffic to the network. Data-Logger (DL) are the access layer (Tire 1) nodes and Global-Aggregator (GA) is the aggregation layer (Tier 2) node.

A single sensor is a traffic source that transmits messages randomly to DLs. Each DL has connected with M_s number of sensors. DL i can store data and transmit it to the GA in a randomly selected time slot in each sampling period T_i. The GA has connected with M_{dl} number of DLs (Figure 3.1).

DLs can transmit the traffic in three different methods:

1. Source is synchronized, $t_i = t$, and $T_i = T$, $\forall i$;

2. Source is asynchronous and sampling period is same, $t_i \neq t_j$ and $T_i = T$;

3. Source is asynchronous and sampling period is different, $t_i \neq t_j$ and $T_i \neq T$.

Here, T_i is the sampling period of DL i and t_i is the first message transmission time of DL i. Periodic batch arrivals can be used to model synchronous sources. Then the superimposed process takes on the same characteristics as the individual processes. However, it is logical to presume that the IoT traffic sources are asynchronous in a practical scenario. From this thesis, homogeneous asynchronous periodic traffic with the same periodicity has been mainly discussed.

Since DLs are connected to many sensors, they transmit many data packets in a selected time slot. From this thesis, we have analyzed the traffic distribution at the GA by proposing several slot selection methods to transmit data from DLs to the IoT cloud.

Chapter 4

MAIN RESULTS

As mentioned in earlier sections our main target is to re-distribute the network traffic to minimize the buffering and congestion at the aggregation layers. This could be achieved by using the proper evening out method at DLs without using a centralized controlling mechanism (Figure 3.1). From this section, we have discussed several slot selection methods (Figure4.1) to even out the traffic over time.

4.1 Traffic Distribution at Data-Logger

Messages arriving to the DLs from sensor nodes is an example of an arrival process which can be mathematically described as stochastic point processes [77]. From the *Palm-Khintchine theorem*, it can be assumed the traffic at the DL is a Poisson process. Once messages arrive at a DL, it should buffer them until the next transmission time slot.

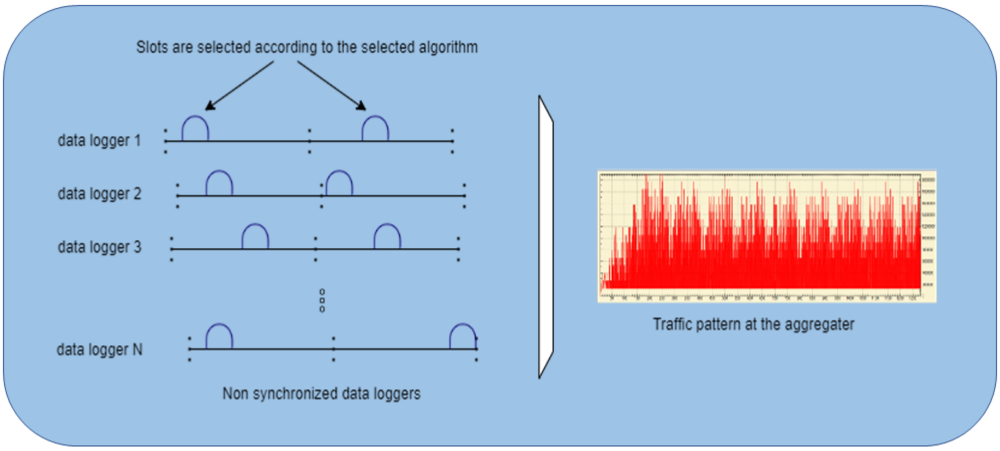

Figure 4.1: Random Slot Selection

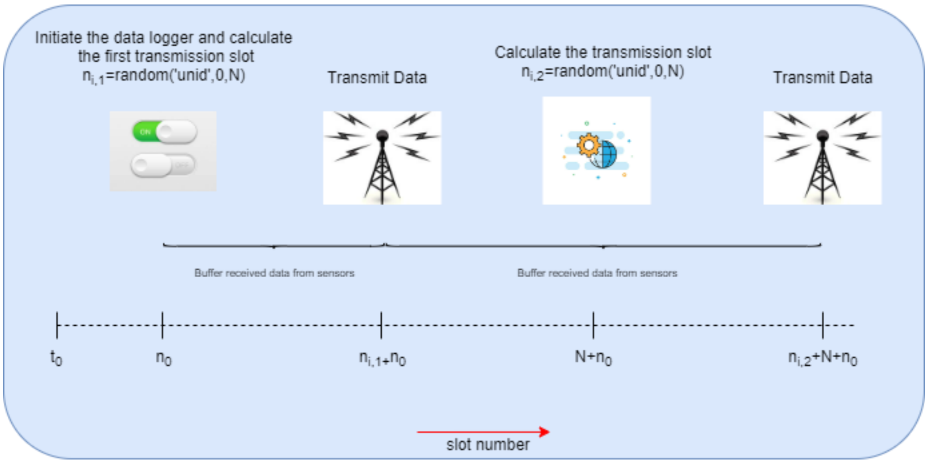

Figure 4.2: RUSS slot selection

4.2 Traffic Distribution at Global-Aggregator

The individual traffic flows from each data logger occur during a large time slot. Therefore, the summation of the identical randomly selected slots will form a multinomial distribution. For a single slot, the traffic distribution will be the marginal distribution of the multinomial distribution, which is binomial. When the number of DLs is large, it will converge to a Gaussian distribution by the Laplace De-Moivre theorem.

However, from the Palm-Khintchine Theorem, the individual packets of the logged data will follow a Poisson arrival distribution and result in a Poisson sum. This Poisson sum will converge to a Gaussian distribution with equal mean and variance. Therefore, the test of equal variance and mean of the Gaussian distribution is done to check the Poisson model or multinomial model best describes the result.

4.3 Random Uniform Slot Selection (*RUSS*)

In this method, DLs were initialized randomly within the first sampling period to simulate asynchronous data sources and assumed the sampling period is T for every DL. One sampling period has been divided into N number of time slots. The transmission time slot n is a discrete uniform random variable and has to be calculated in every sampling period. After selecting the time slot, buffered data can be transmitted to the GA. Figure 4.2 shows the slot selection process of this method. In the figure, t_0 and n_0 represent the initial time and initial time slot respectively. $n_{i,k}$ is the selected time slot of the i^{th} DL at k^{th} sampling period $(i, k \in \mathbb{N})$.

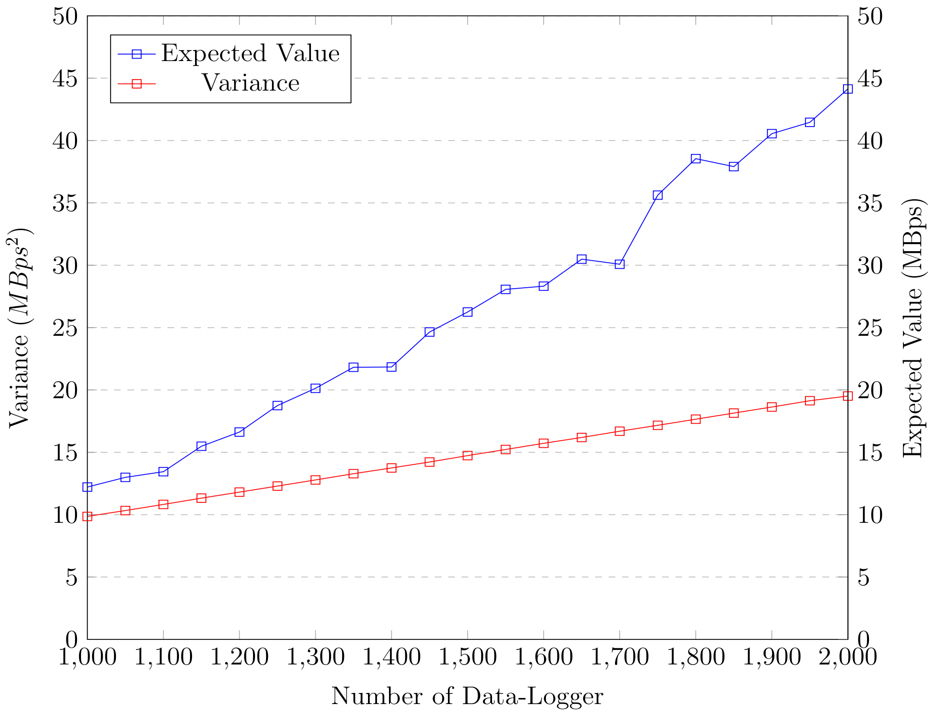

Figure 4.3: Expected value and Variance of the distribution at the aggregator ($RUSS$)

Algorithm 1 Random Uniform slot selection ($RUSS$)

```
for each Local Aggregator do
    if initialized = False then
        Select the initial time randomly
        initialized = True
    else if initialized = True then
        for each sampling time period do
            n=select the transmission time slot according to a Uniform distribution
            transmit data at n^th time slot
```

Figure 4.3 shows the deviation of the expected value and the variance of the traffic distribution at the GA. The number of DLs were increased from 1000 to 2000, while each DL was connected with 100 sensor nodes. We assume the size of messages transmitted by a sensor node equals 10kB.

4.4 Initial Uniform Slot Selection (*IUSS*)

The initialization part is the same as the earlier method. The sampling period is T for every DL, and it has divided into N number of time slots. The transmission time slot n is a discrete uniform random variable and has to be calculated only in the first sampling period after the initialization. After selecting the time slot, it can be used to transmit data in every

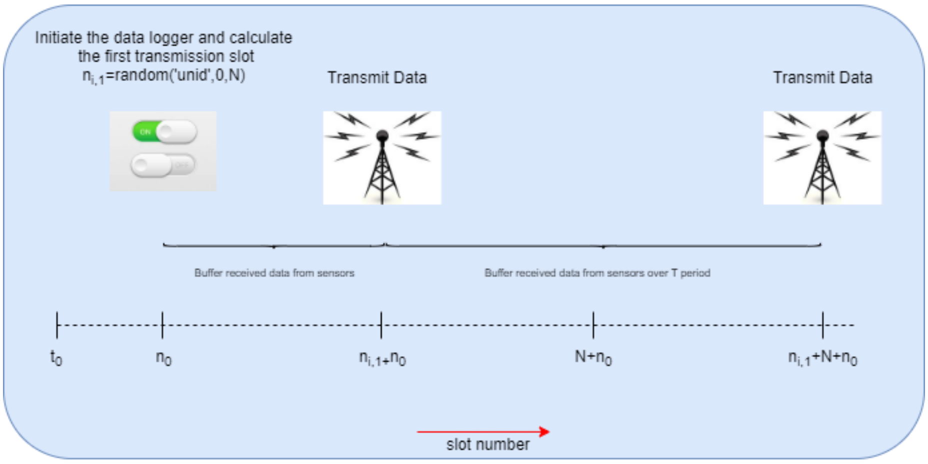

Figure 4.4: IUSS slot selection

sampling period (Figure 4.4).

Algorithm 2 Initial Uniform Slot Selection (IUSS)

```
for each data logger do
    if initialized = False then
        Select the initial time randomly
        initialized = True
    else if initialized = True then
        for each sampling time period do
            if time-slot-selected = False then
                n=select the transmission time slot according to a Uniform distribution
                time-slot-selected = True
                transmit data at n^th time slot
            else if time-slot-selected = True then
                transmit-data-at n^th time slot
```

Figure 4.5 shows the deviation of the expected value and the variance of the traffic distribution at the GA. The number of DL ware increased from 1000 to 2000, while each DL connected with 100 sensor nodes. We assumed the size of messages transmitted by a sensor node was equal to 10kB (Figure 4.5). Quantile-Quantile plot (Figure 4.6) proved that the traffic distribution for a higher number of DLs has been converged to a Gaussian distribution.

4.5 Initial Bernoulli Slot Selection (*IBSS*)

The initialization part is the same as the earlier methods. The sampling period is T for every DL, divided into N number of time slots. The transmission time slot n is selected according to a Bernoulli trial. The probability of success p should be equal to $1/N$. If any DL fails to

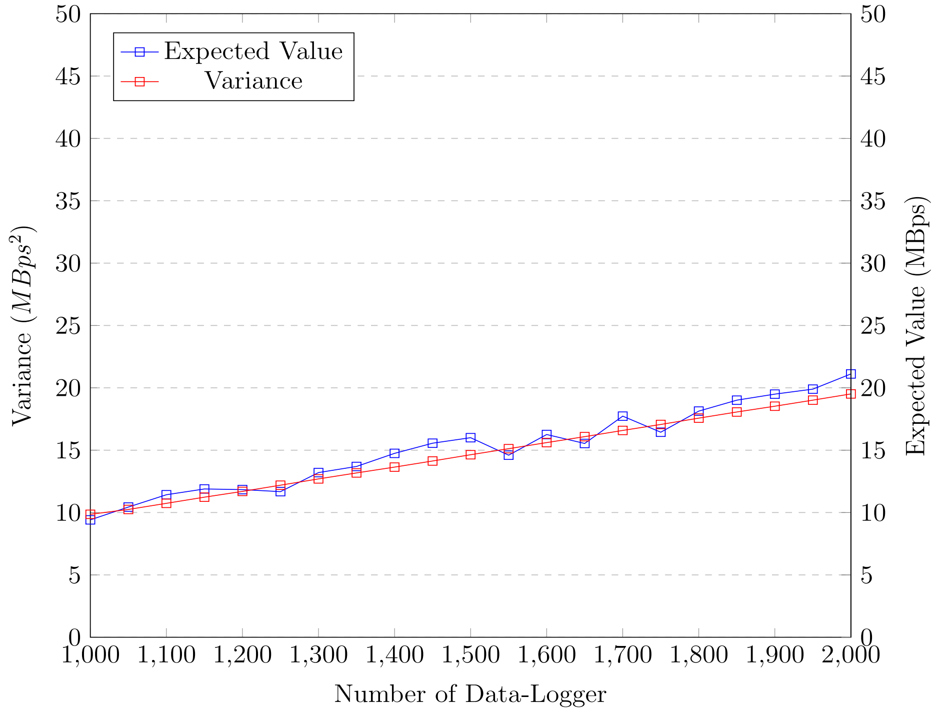

Figure 4.5: Expected value and Variance of the distribution at the aggregator (*IUSS*)

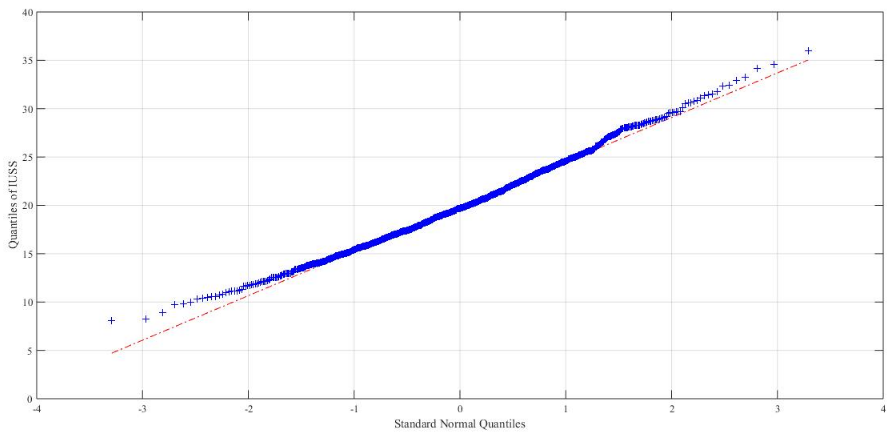

Figure 4.6: Quantiles comparison of the distribution of *IUSS* mode (for larger expected traffic volume)

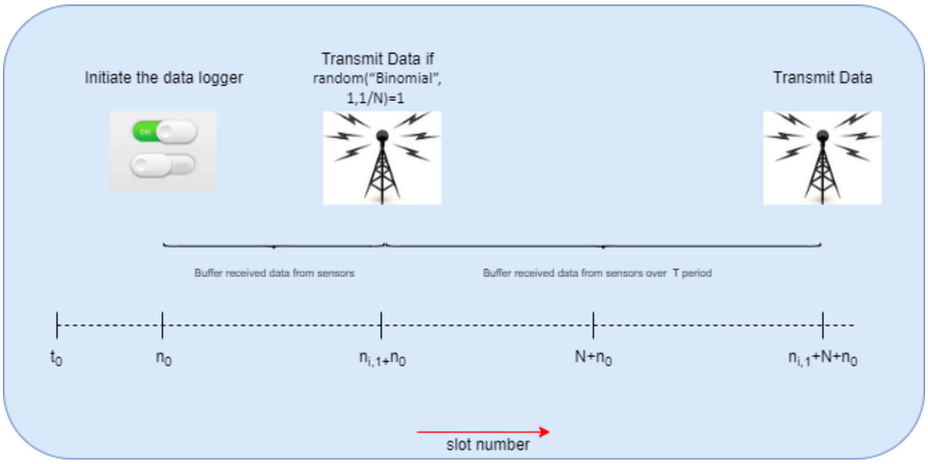

Figure 4.7: IBSS slot selection

select the time slot within the first sampling period T, data will be transmitted at the final time slot (N^{th} time slot). Then the time slot of those DLs should be selected again in the next sampling period. This process should be performed until every DL selects a time slot according to the Bernoulli process. After selecting the time slot, it can be used to transmit data in every sampling period (Figure 4.7).

Algorithm 3 Initial Bernoulli Slot Selection (*IBSS*)

for each Data Logger
    if initialized = False **then**
        Select the initial time randomly
        initialized=True
    else if initialized = True **then**
        for each sampling time period **do**
            if time-slot-selected = False **then**
                for n in 0:N **do**
                    n=select and transmit data if $random('Binomial', 1, 1/N) = 1$
                time-slot-selected = True
            else if time-slot-selected = True **then**
                transmit-data-at n^{th} time slot
do

Un-synchronized DL are essential in this method, since each DL cannot select the transmission slot at the first N time slots. If the DLs are synchronized, many of them transmit data at the last time slot of the interval. So, it would generate a large traffic spike at the GA. Figure 4.8 shows the simulation results for such model.

If DLs are initialized asynchronously, we can minimized the impact of above issue. Figure 4.9 shows the simulation results for asynchronous DLs.

Figure 4.10 shows the deviation of the expected value and the variance of the traffic

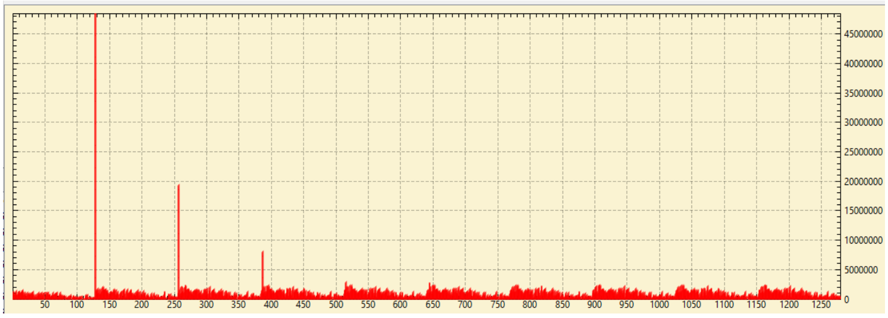

Figure 4.8: Traffic distribution at the GA from set of synchronized DLs

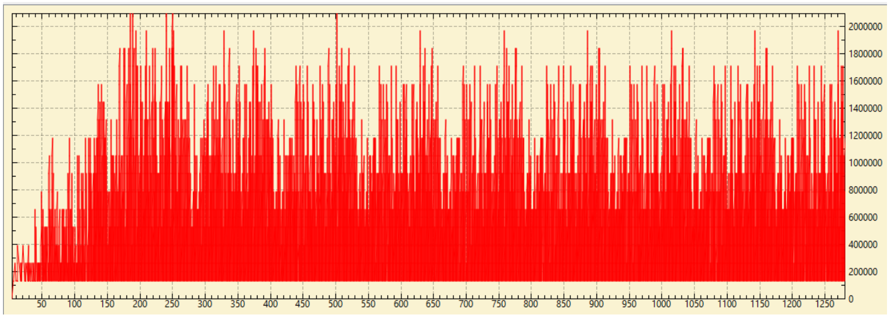

Figure 4.9: Traffic distribution at the GA from set of asynchronized DLs

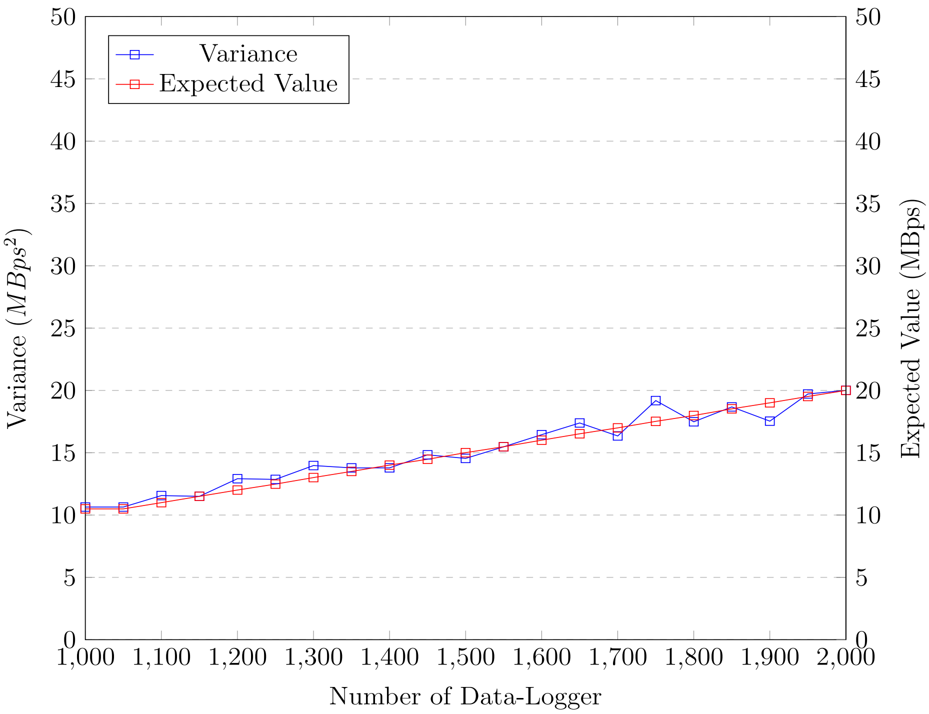

Figure 4.10: Expected value and Variance of the distribution at the aggregator (*IBSS*)

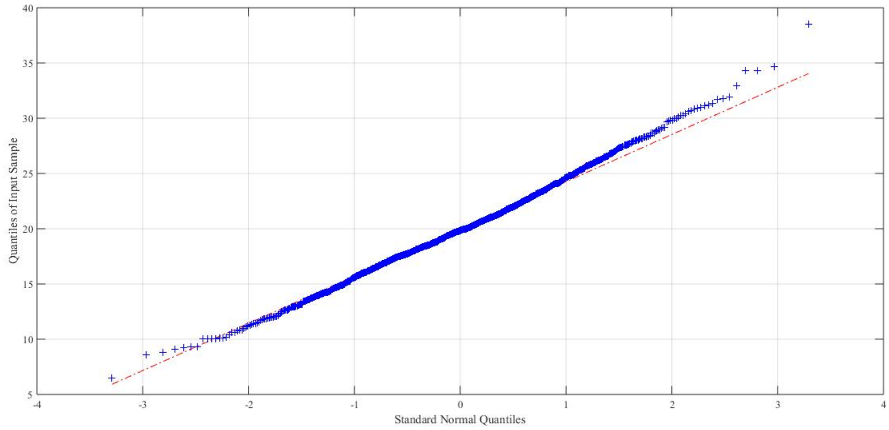

Figure 4.11: Quantiles comparison of the distribution of *IBSS* mode (for larger expected traffic volume)

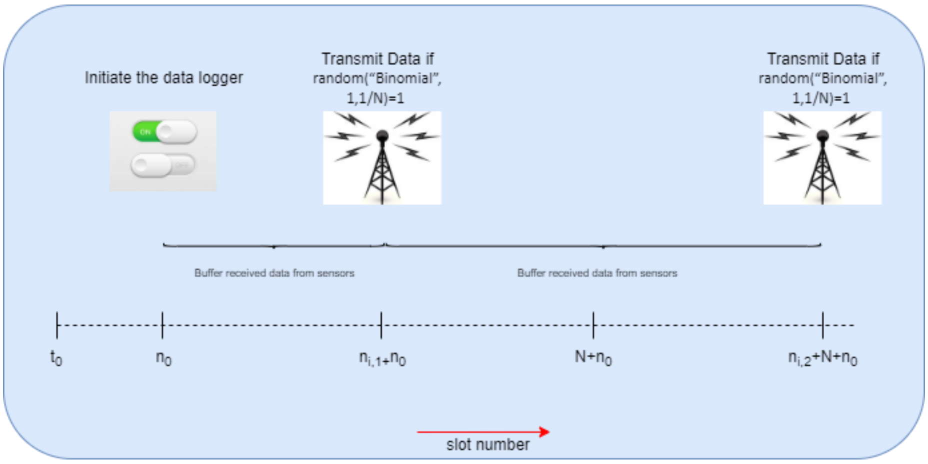

Figure 4.12: RBSS slot selection

distribution at the GA. The number of DLs were increased from 1000 to 2000, while each DL was connected with 100 sensor nodes. We assume the size of messages transmitted by a sensor node equals 10kB (Figure 4.10). Quantile-Quantile plot (Figure 4.11) proved that the traffic distribution for a higher number of DLs is a Gaussian distribution.

4.6 Random Bernoulli Slot Selection (*RBSS*)

This method is almost equals with IBSS method. The difference is, the time slot should be selected in every sampling period (Figure 4.12).

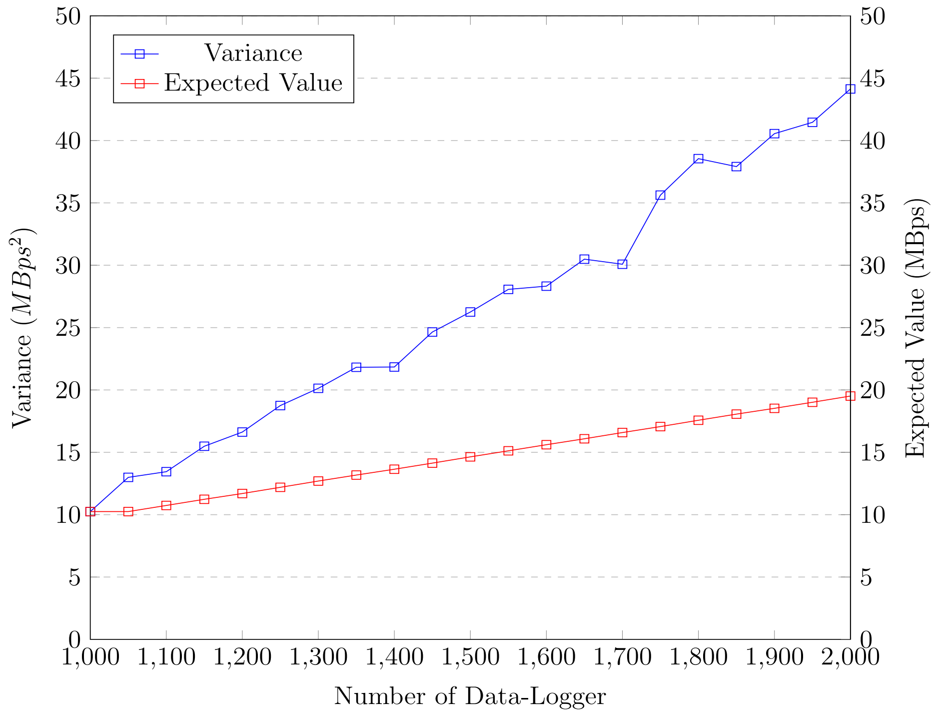

Figure 4.13: Expected value and Standard deviation of the distribution at the aggregator (*RBSS*)

Algorithm 4 Random Bernoulli Slot Selection (*RBSS*)

1: **for** each Data Logger
2:     **if** initialized = False **then**
3:         Select the initial time randomly
4:         initialized=True
5:     **else if** initialized = True **then**
6:         **for** each sampling time period **do**
7:             **for** n in 0:N **do**
8:                 n=select and transmit data if $random('Binomial', 1, 1/N) = 1$
do

4.7 Random Uniform Slot Selection with Path Initialization (*RUSSPI*)

Initialization of the DLs is performed similarly to the earlier methods. Slot selection also equals with the *IUSS*. In this method, DLs partition data into smaller packets (size of the packet Y_d) and transmit them one by one to the IoT Cloud. Before sending each packet, DL should initialize the path with GA using “SYN” and “ACK” messages.

This method is proposed for IoT networks requiring a lower packet loss rate and the correct data sequence. To set up this algorithm, DLs should have more computational capabilities. Also, the packet transmission rate between DL and GA is lower than in earlier methods. These reasons lead to a higher cost.

Algorithm 5 Random Uniform Slot Selection with Path Initialization *RUSSPI* logic

```
for each Data-Logger do
    if initialized = False then
        Select the initial time randomly
        initialized=True
    else if initialized = True then
        for each sampling time period do
            select the transmission time according to a Uniform RV
            Transmit SYN message
            if ACK received then
                transmit the first data chunk
                if ACK for the above data chunk received then
                    transmit the next data chunk
                    Continue until all stored data transmitted.
            else if ACK not received then
                Wait predefined amount of time
                send the SYN message again and wait for ACK
```

Controlling messages (SYN and ACK) add extra traffic to the system. If the packet size of SYN and ACK messages (Y_c) are larger enough with respect to a data chunk, the added extra traffic is considerable. But if $Y_d >> Y_c$, the added traffic would be negligible.

This is simulated using *Omnet++* to know the best value for the data chunk to transmit from DLs. For this simulation, eight sensors were connected with one DL, and 1024 such DL were connected with one GA. Assuming the number of slots per one sampling period (T) is $N = 128$, measured the re-transmissions (number of re-transmitted bits) for several link capacities. Outputs are shown in Figure 4.14 and Table 4.1.

4.8 Comparison

According to the figures 4.3 and 4.13, the mean (expected value) and variance are different in RUSS and RBSS methods. But, according to the figures 4.5 and 4.10, the mean and variance are almost equals in IUSS and IBSS methods. Which means, Poisson model describe the IUSS and IBSS methods whereas RUSS and RBSS methods are described by multinormial model.

Figures 4.15, 4.16, 4.21 and 4.18 show the distribution of the packet drops when up-link size of the aggregator is 20Mbps. We can clearly observe higher packet drop ratio in *RUSS* and *RBSS* methods. Figures 4.19, 4.20, 4.17 and 4.22 show the traffic distribution variation

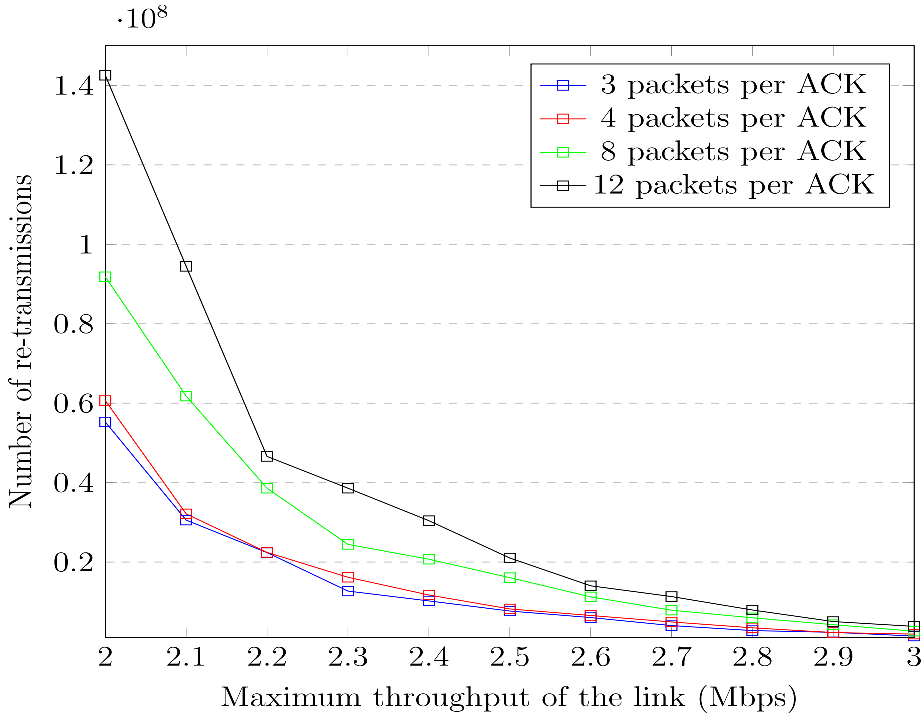

Figure 4.14: Number of re-transmissions for different capacities

Number of re-transmissions for different capacities				
Maximum throughput of the link (Mbps)	3 packets per acknowledgement	4 packets per acknowledgement	8 packets per acknowledgement	12 packets per acknowledgement
2.0	55274900	60664100	91836400	142566000
2.1	30555800	32137900	61787900	94450000
2.2	22377800	22433800	38590700	46596000
2.3	12698400	16177400	24443200	38596000
2.4	10216400	11708700	20717400	30426100
2.5	7674540	8201120	16082800	21028700
2.6	6029140	6550020	11241800	14024100
2.7	4006400	4922810	7885610	11318000
2.8	2747930	3443650	5965380	7931060
2.9	2311550	2243420	4233870	5019500
3.0	1365570	1857590	2593370	3805790

Table 4.1: Number of re-transmissions

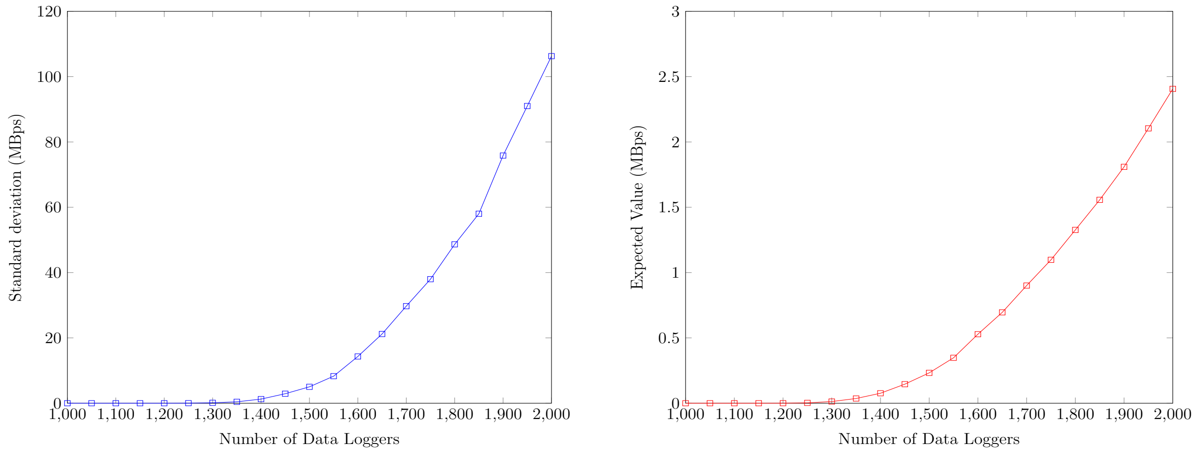

Figure 4.15: Expected value and Standard deviation of the packet drops at the aggregator *RUSS*

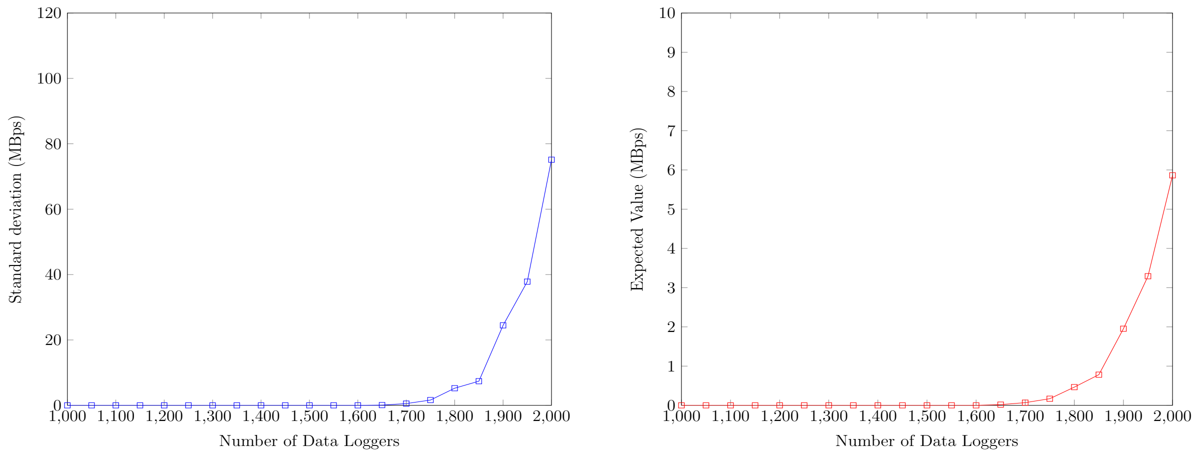

Figure 4.16: Expected value and Standard deviation of the packet drops at the aggregator *IUSS*

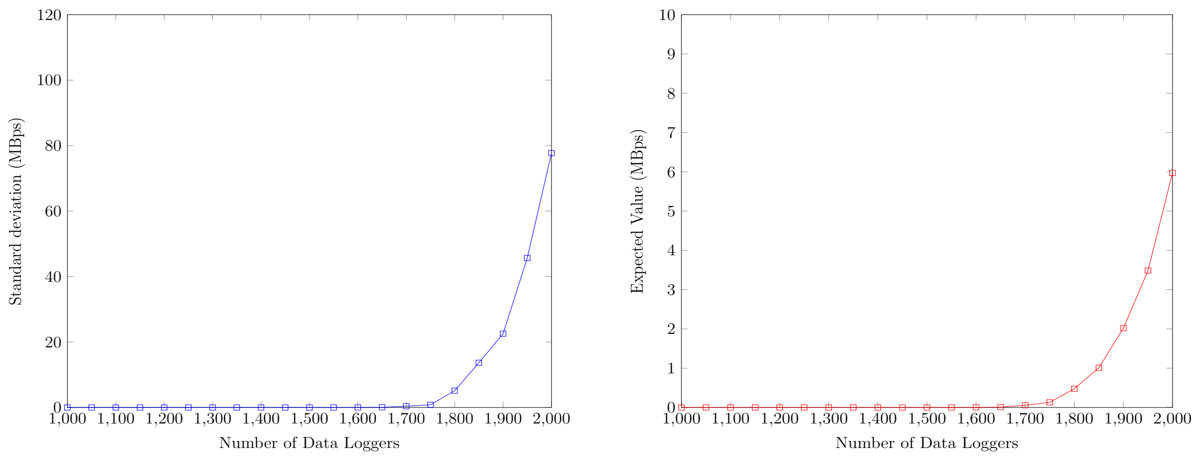

Figure 4.17: Expected value and Standard deviation of the packet drops at the aggregator *IBSS*

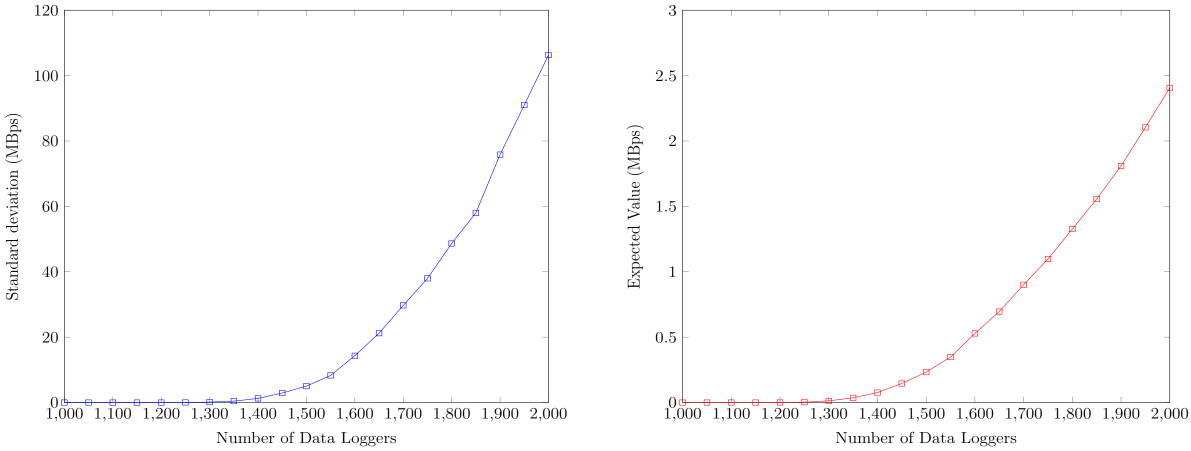

Figure 4.18: Expected value and Standard deviation of the packet drops at the aggregator *RBSS*

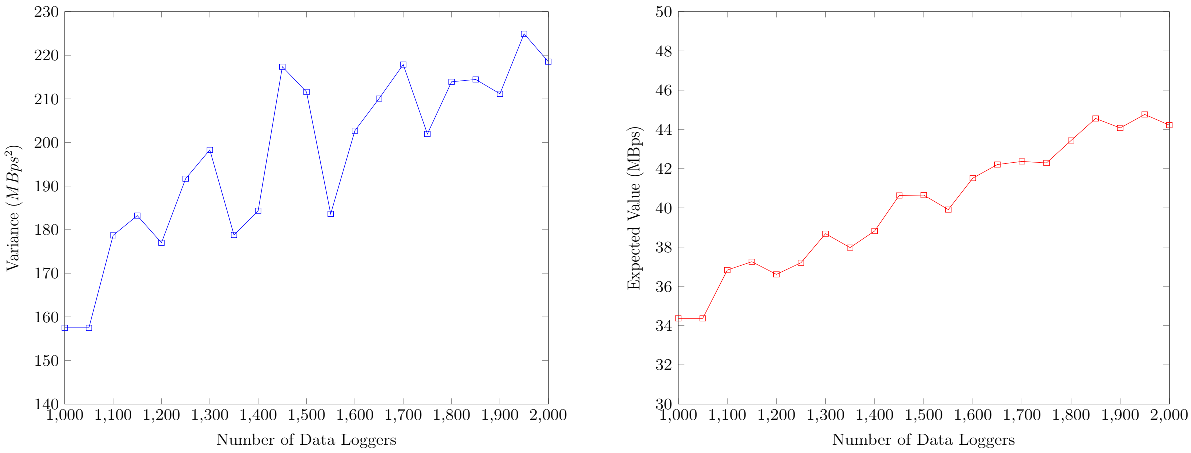

Figure 4.19: RUSS-IoT traffic combined with packet switched network backbone traffic

of each methods, after combining with a packet switched network traffic which was modeled by referring [23].

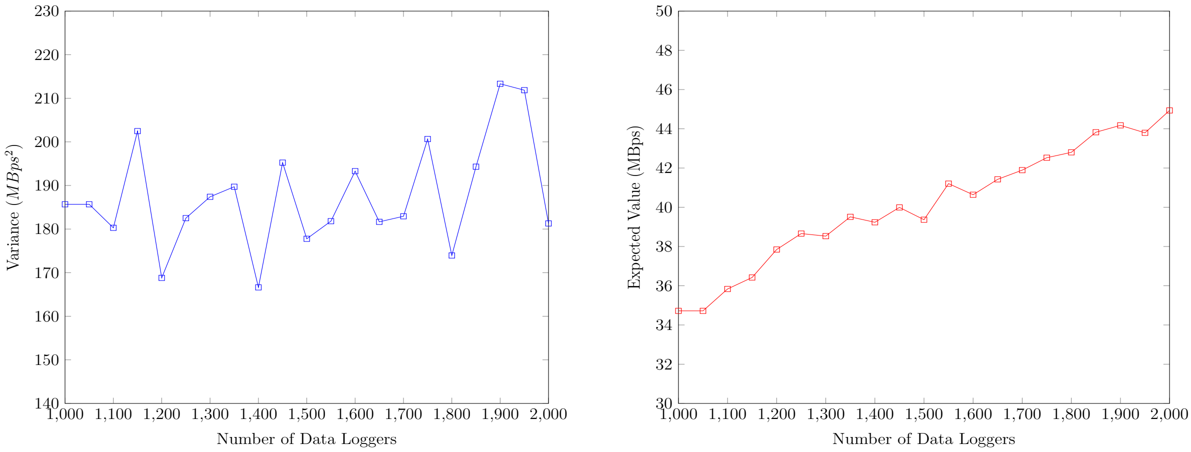

Figure 4.20: IUSS-IoT traffic combined with packet switched network backbone traffic

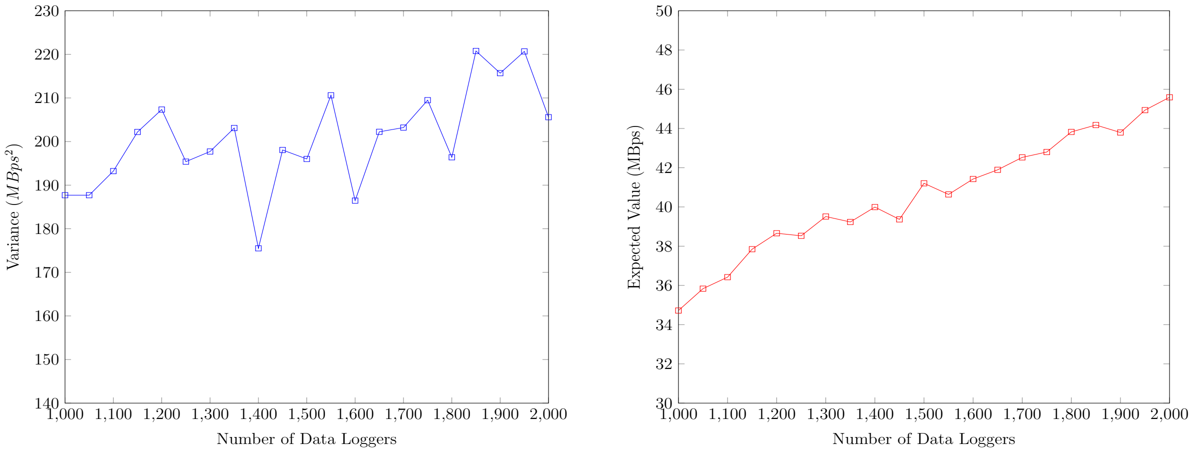

Figure 4.21: IBSS-IoT traffic combined with packet switched network backbone traffic

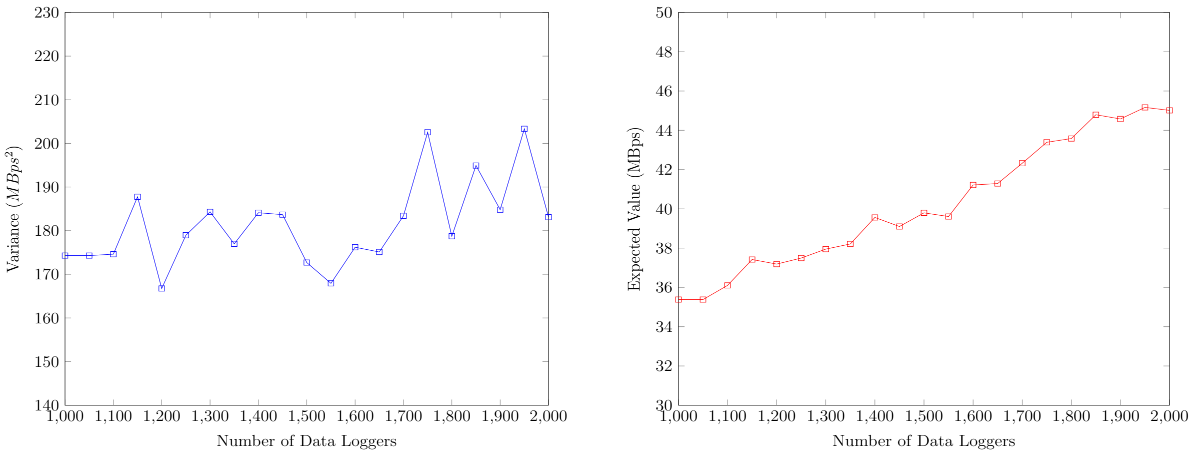

Figure 4.22: RBSS-IoT traffic combined with packet switched network backbone traffic

Chapter 5

CONCLUSION

In this thesis, we have discussed several traffic transmission algorithms to even out the distribution at the aggregation nodes over time. To compare the outcomes, coefficient of variation of each model for different number of DLs has been plotted in Figure 5.1 and Figure 5.2.

Figure 5.1 shows the coefficient of variation among different distribution methods when the aggregation layer is receiving traffic only from the IoT network. In this graph, there are clear differences among different distribution methods. *IUSS* and *IBSS* methods have lower coefficient of variation than other two methods (*RUSS* and *RBSS*). This means *IUSS* and *IBSS* methods are better than the other two methods to even out the traffic at the aggregation layer. Also, *RUSS* and *RBSS* methods required more CPU power to calculate the slots at each transmission period. So, both *RUSS* and *RBSS* are effective and simple methods to implement at the access layer in IoT sensor networks.

Figure 5.2 shows the coefficient of variation among different distribution methods while the IoT network is combined with a packet-switched network backbone at the aggregation layer. Since the normal user traffic is dominant, the variation of “coefficient of variation” is almost equal in each distribution method. However, the coefficient of variation has decreased for a larger number of DLs, which indicates the traffic has been more evenly distributed at the aggregation layer when the number of DLs are increasing.

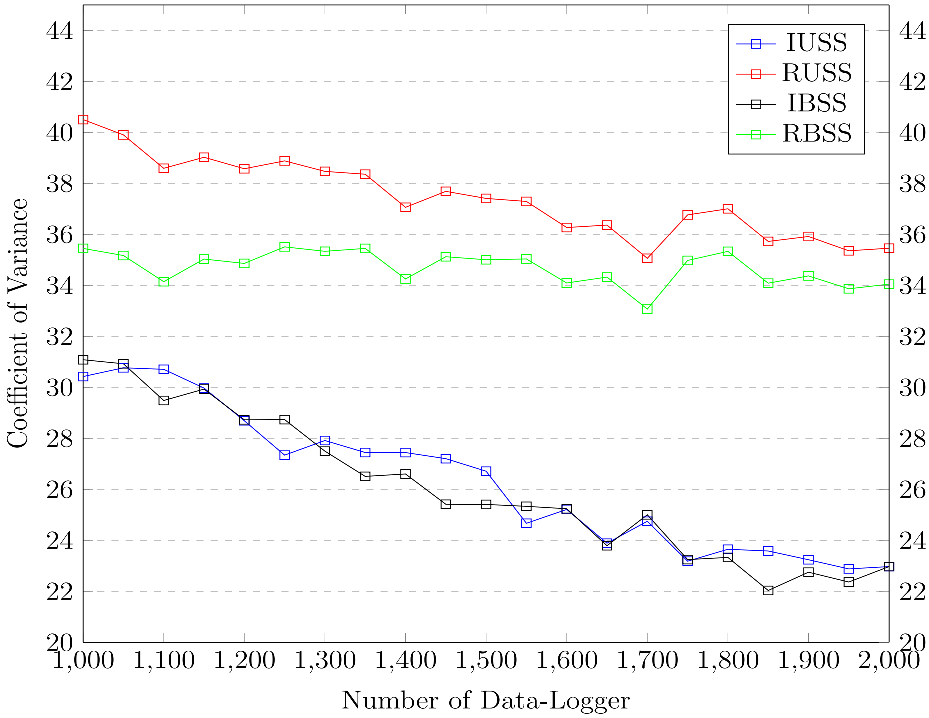

Figure 5.1: Coefficient of variation ($\sigma/\mu * 100$) for standalone IoT traffic

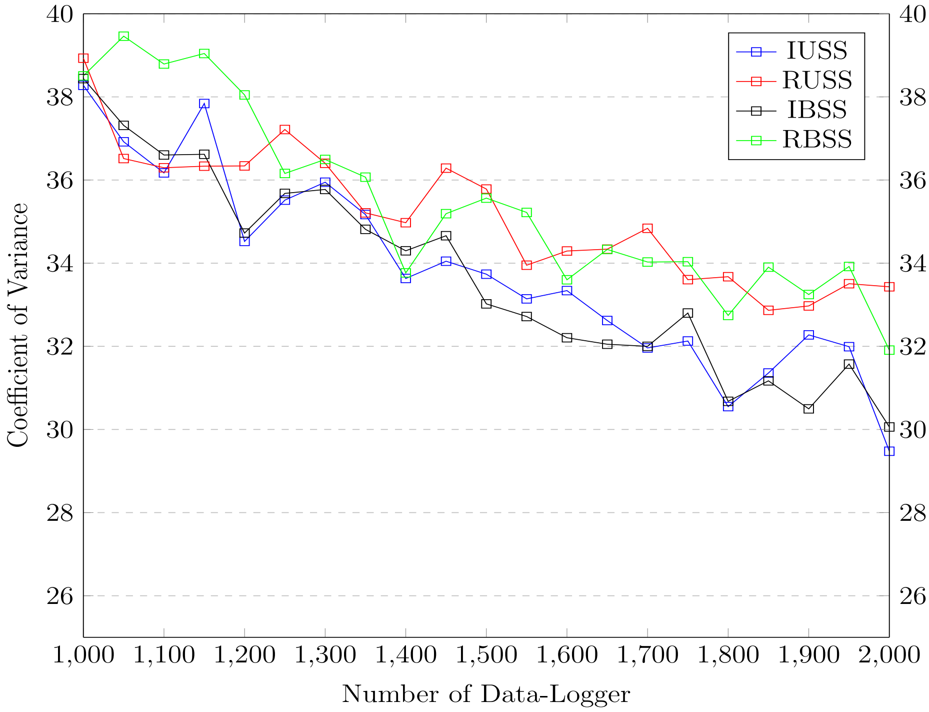

Figure 5.2: Coefficient of variation ($\sigma/\mu * 100$) for IoT and packet switched network backbone traffic

Bibliography

[1] J. Wilamowski, B.M. Irwin, "Industrial communication systems," *CRC Press*, 2011.

[2] S. A. Busari, K. M. S. Huq, S. Mumtaz, and J. Rodriguez, "Terahertz massive mimo for beyond-5g wireless communication," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–6.

[3] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial internet of things: Challenges, opportunities, and directions," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 11, pp. 4724–4734, 2018.

[4] F. Nizzi, T. Pecorella, M. Bertini, R. Fantacci, M. Bastianini, C. Cerboni, A. Buzzigoli, M. Gattoni, and A. Fratini, "Evaluation of iot and videosurveillance applications in a 5g smart city: the italian 5g experimentation in prato," in *2018 AEIT International Annual Conference*, 2018, pp. 1–6.

[5] Z. Qin, D. Wu, Z. Xiao, B. Fu, and Z. Qin, "Modeling and analysis of data aggregation from convergecast in mobile sensor networks for industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4457–4467, 2018.

[6] C. Kan, H. Yang, and S. Kumara, "Parallel computing and network analytics for fast industrial internet-of-things (iiot) machine information processing and condition monitoring," *Journal of Manufacturing Systems*, vol. 46, pp. 282–293, 2018.

[7] Y. Yu, R. Chen, H. Li, Y. Li, and A. Tian, "Toward data security in edge intelligent iiot," *IEEE Network*, vol. 33, no. 5, pp. 20–26, 2019.

[8] T. Hussain, K. Muhammad, J. D. Ser, S. W. Baik, and V. H. C. de Albuquerque, "Intelligent embedded vision for summarization of multiview videos in iiot," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2592–2602, 2020.

[9] X. Wang, L. T. Yang, Y. Wang, L. Ren, and M. J. Deen, "Adtt: A highly efficient distributed tensor-train decomposition method for iiot big data," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 3, pp. 1573–1582, 2021.

[10] H.-L. Truong, "Integrated analytics for iiot predictive maintenance using iot big data cloud systems," in *2018 IEEE International Conference on Industrial Internet (ICII)*, 2018, pp. 109–118.

[11] M. Cakir, M. A. Guvenc, and S. Mistikoglu, "The experimental application of popular machine learning algorithms on predictive maintenance and the design of iiot based condition monitoring system," *Computers Industrial Engineering*, vol. 151, p. 106948, 2021.

[12] X. Yan, Y. Xu, X. Xing, B. Cui, Z. Guo, and T. Guo, "Trustworthy network anomaly detection based on an adaptive learning rate and momentum in iiot," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 6182–6192, 2020.

[13] P. Eden, A. Blyth, K. Jones, H. Soulsby, P. Burnap, Y. Cherdantseva, and K. Stoddart, "Scada system forensic analysis within iiot," in *Cybersecurity for Industry 4.0.* Springer, 2017, pp. 73–101.

[14] F. Núñez, S. Langarica, P. Díaz, M. Torres, and J. C. Salas, "Neural network-based model predictive control of a paste thickener over an industrial internet platform," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2859–2867, 2020.

[15] Y. Li, M. Alizadeh, M. Yu, R. Miao, H. Liu, Y. Zhuang, F. Feng, L. Tang, Z. Cao, M. Zhang, and F. Kelly, "Hpcc: high precision congestion control," 08 2019, pp. 44–58.

[16] B. Valeske, A. Osman, F. Römer, and R. Tschuncky, "Next generation nde sensor systems as iiot elements of industry 4.0," *Research in Nondestructive Evaluation*, vol. 31, no. 5-6, pp. 340–369, 2020.

[17] S. Chen, Z. Wang, H. Zhang, G. Yang, and K. Wang, "Fog-based optimized kronecker-supported compression design for industrial iot," *IEEE Transactions on Sustainable Computing*, vol. 5, no. 1, pp. 95–106, 2020.

[18] K.-E. Åarzén, "A simple event-based pid controller," *IFAC Proceedings Volumes*, vol. 32, no. 2, pp. 8687–8692, 1999, 14th IFAC World Congress 1999, Beijing, Chia, 5-9 July.

[19] U. Premaratne, S. Warnakulasooriya, and R. Nandana, "Characterization of event based sampling encoders for iiot using input-output mutual information," *IEEE Transactions on Industrial Informatics*, vol. 17, 09 2020.

[20] M. Emara, H. Elsawy, and G. Bauch, "A spatiotemporal model for peak aoi in uplink iot networks: Time versus event-triggered traffic," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 6762–6777, 2020.

[21] H. Rezaei, M. U. Chaudhry, H. Almasi, and B. Vamanan, "Icon: Incast congestion control using packet pacing in datacenter networks," in *2019 11th International Conference on Communication Systems & Networks (COMSNETS).* IEEE, 2019, pp. 125–132.

[22] M. Gowtham and K. Subramaniam, "Congestion control and packet recovery for cross layer approach in manet," *Cluster Computing*, vol. 22, no. 5, pp. 12 029–12 036, 2019.

[23] U. Premaratne and U. S. Premarathne, "A sum of bernoulli sources approximation for packet switched network traffic in backbone links," *IEEE Communications Letters*, vol. 24, no. 1, pp. 141–145, 2019.

[24] F. Gebali, *Analysis of computer and communication networks.* Springer Science & Business Media, 2008.

[25] D. R. Cox and V. Isham, *Point processes.* CRC Press, 1980, vol. 12.

[26] V. S. Frost and B. Melamed, "Traffic modeling for telecommunications networks," *IEEE Communications Magazine*, vol. 32, no. 3, pp. 70–81, 1994.

[27] V. Frost and B. Melamed, "Traffic modeling for telecommunications networks," *IEEE Communications Magazine*, vol. 32, no. 3, pp. 70–81, 1994.

[28] A. Khintchine, "Mathematical methods in the theory of queueing," *London 1960.*, p. 124, 1955.

[29] F. Metzger, T. Hoßfeld, A. Bauer, S. Kounev, and P. E. Heegaard, "Modeling of aggregated iot traffic and its application to an iot cloud," *Proceedings of the IEEE*, vol. 107, no. 4, pp. 679–694, 2019.

[30] A. Erramilli, M. Roughan, D. Veitch, and W. Willinger, "Self-similar traffic and network dynamics," *Proceedings of the IEEE*, vol. 90, no. 5, pp. 800–819, 2002.

[31] M. E. Crovella and A. Bestavros, "Self-similarity in world wide web traffic: Evidence and possible causes," *IEEE/ACM Transactions on networking*, vol. 5, no. 6, pp. 835–846, 1997.

[32] Z. Sahinoglu and S. Tekinay, "On multimedia networks: self-similar traffic and network performance," *IEEE Communications Magazine*, vol. 37, no. 1, pp. 48–52, 1999.

[33] A. Adas, "Traffic models in broadband networks," *IEEE communications Magazine*, vol. 35, no. 7, pp. 82–89, 1997.

[34] J. D. McCabe, *Practical computer network analysis and design.* Morgan Kaufmann Publishers Inc., 1997.

[35] N. Yamanaka, Y. Sato, and K. Sato, "Performance limitation of the leaky bucket algorithm for atm networks," *IEEE Transactions on Communications*, vol. 43, no. 8, pp. 2298–2300, 1995.

[36] D. Jurca and P. Frossard, "Video packet selection and scheduling for multipath streaming," *IEEE Transactions on Multimedia*, vol. 9, no. 3, pp. 629–641, 2007.

[37] A. Bilami, M. Lalam, M. Daoui, and B. Djamah, "A cell selection policy for an input-buffered packet switch," *International Journal of Computers and Applications*, vol. 28, no. 3, pp. 234–242, 2006.

[38] M. Ke, Y. Nenghai, and L. Bin, "A new packet dropping policy in delay tolerant network," in *2010 IEEE 12th International Conference on Communication Technology*. IEEE, 2010, pp. 377–380.

[39] M. A. Labrador and S. Banerjee, "Packet dropping policies for atm and ip networks," *IEEE Communications Surveys*, vol. 2, no. 3, pp. 2–14, 1999.

[40] M. J. Neely, "Intelligent packet dropping for optimal energy-delay tradeoffs in wireless downlinks," *IEEE Transactions on Automatic Control*, vol. 54, no. 3, pp. 565–579, 2009.

[41] N. McKeown and T. Anderson, "A quantitative comparison of iterative scheduling algorithms for input-queued switches," *Computer Networks and ISDN Systems*, vol. 30, pp. 2133–2168, 1998.

[42] G. B. Dantzig and M. N. Thapa, *Linear programming 1: introduction*. Springer Science & Business Media, 2006.

[43] S. S. Rao, *Engineering optimization: theory and practice*. John Wiley & Sons, 2019.

[44] A. Burns, *Preemptive priority based scheduling: An appropriate engineering approach*. University of York, Department of Computer Science, 1993.

[45] J. A. Cobb, M. G. Gouda, and A. El-Nahas, "Time-shift scheduling-fair scheduling of flows in high-speed networks," *IEEE/ACM transactions on networking*, vol. 6, no. 3, pp. 274–285, 1998.

[46] K. Nichols, S. Blake, F. Baker, and D. Black, "Definition of the differentiated services field (ds field) in the ipv4 and ipv6 headers," RFC 2474, 1998.

[47] D. Stiliadis and A. Varma, "Latency-rate servers: a general model for analysis of traffic scheduling algorithms," *IEEE/ACM Transactions on networking*, vol. 6, no. 5, pp. 611–624, 1998.

[48] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the single-node case," *IEEE/ACM transactions on networking*, vol. 1, no. 3, pp. 344–357, 1993.

[49] A. Parekh and R. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the multiple node case," *IEEE/ACM Transactions on Networking*, vol. 2, no. 2, pp. 137–150, 1994.

[50] B. Grot, S. W. Keckler, and O. Mutlu, "Preemptive virtual clock: A flexible, efficient, and cost-effective qos scheme for networks-on-chip," in *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, 2009, pp. 268–279.

[51] M. Katevenis, S. Sidiropoulos, and C. Courcoubetis, "Weighted round-robin cell multiplexing in a general-purpose atm switch chip," *IEEE Journal on selected Areas in Communications*, vol. 9, no. 8, pp. 1265–1279, 1991.

[52] D. Ferrari and D. C. Verma, "A scheme for real-time channel establishment in wide-area networks," *IEEE journal on Selected Areas in communications*, vol. 8, no. 3, pp. 368–379, 1990.

[53] M. Shreedhar and G. Varghese, "Efficient fair queuing using deficit round-robin," *IEEE/ACM Transactions on networking*, vol. 4, no. 3, pp. 375–385, 1996.

[54] D. C. Verma, H. Zhang, and D. Ferrari, *Delay jitter control for real-time communication in a packet switching network.* International Computer Science Institute, 1991.

[55] V. M. Vokkarane, J. P. Jue, and S. Sitaraman, "Burst segmentation: an approach for reducing packet loss in optical burst switched networks," in *2002 IEEE International Conference on Communications. Conference Proceedings. ICC 2002 (Cat. No. 02CH37333)*, vol. 5. IEEE, 2002, pp. 2673–2677.

[56] L. Zhang, "Virtualclock: A new traffic control algorithm for packet-switched networks," *ACM Transactions on Computer Systems (TOCS)*, vol. 9, no. 2, pp. 101–124, 1991.

[57] M. E. G. Mustafa, S. A. Talab *et al.*, "The effect of queuing mechanisms first in first out (fifo), priority queuing (pq) and weighted fair queuing (wfq) on network's routers and applications," *Wireless Sensor Network*, vol. 8, no. 05, p. 77, 2016.

[58] F. Lu, G. M. Voelker, and A. C. Snoeren, "Weighted fair queuing with differential dropping," in *2012 Proceedings IEEE INFOCOM.* IEEE, 2012, pp. 2981–2985.

[59] C. Kim, T.-W. Yoo, and B.-T. Kim, "A hierarchical weighted round robin epon dba scheme and its comparison with cyclic water-filling algorithm," in *2007 IEEE International Conference on Communications.* IEEE, 2007, pp. 2156–2161.

[60] D.-s. Back, K. Pyun, S.-M. Lee, J. Cho, and N. Kim, "A hierarchical deficit round-robin scheduling algorithm for a high level of fair service," in *2007 International Symposium on Information Technology Convergence (ISITC 2007)*, 2007, pp. 115–119.

[61] A. Kortebi, S. Oueslati, and J. Roberts, "Implicit service differentiation using deficit round robin," *ITC19*, 2005.

[62] M. Shreedhar and G. Varghese, "Efficient fair queueing using deficit round robin," in *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, 1995, pp. 231–242.

[63] H. Zhang and D. Ferrari, "Rate-controlled static-priority queueing," in *IEEE INFOCOM'93 The Conference on Computer Communications, Proceedings.* IEEE, 1993, pp. 227–236.

[64] C. Aras, J. Kurose, D. Reeves, and H. Schulzrinne, "Real-time communication in packet-switched networks," *Proceedings of the IEEE*, vol. 82, no. 1, pp. 122–139, 1994.

[65] S.-L. Tsao, "Extending earliest-due-date scheduling algorithms for wireless networks with location-dependent errors," in *Vehicular Technology Conference Fall 2000. IEEE VTS Fall VTC2000. 52nd Vehicular Technology Conference (Cat. No. 00CH37152)*, vol. 1. IEEE, 2000, pp. 223–228.

[66] K.-H. Teh, P.-Y. Kong, and S. Jiang, "Proactive earliest due-date scheduling in wireless packet networks," in *International Conference on Communication Technology Proceedings, 2003. ICCT 2003.*, vol. 2. IEEE, 2003, pp. 816–820.

[67] S. Han and K. G. Shin, "Fast restoration of real-time communication service from component failures in multi-hop networks," in *Proceedings of the ACM SIGCOMM'97 conference on Applications, technologies, architectures, and protocols for computer communication*, 1997, pp. 77–88.

[68] A. C. Sembiring, J. Tampubolon, D. Sitanggang, M. Turnip *et al.*, "Improvement of inventory system using first in first out (fifo) method," in *Journal of Physics: Conference Series*, vol. 1361, no. 1. IOP Publishing, 2019, p. 012070.

[69] R. Guérin and V. Peris, "Quality-of-service in packet networks: basic mechanisms and directions," *Computer networks*, vol. 31, no. 3, pp. 169–189, 1999.

[70] Z. Lotker and B. Patt-Shamir, “Nearly optimal fifo buffer management for diffserv,” in *Proceedings of the twenty-first annual symposium on Principles of distributed computing*, 2002, pp. 134–143.

[71] M. Christiansen, K. Jeffay, D. Ott, and F. D. Smith, “Tuning red for web traffic,” *ACM SIGCOMM Computer Communication Review*, vol. 30, no. 4, pp. 139–150, 2000.

[72] L. Bic and A. C. Shaw, *The logical design of operating systems.* Prentice-Hall, Inc., 1988.

[73] J. S. Gero and T. Mc Neill, “An approach to the analysis of design protocols,” *Design studies*, vol. 19, no. 1, pp. 21–61, 1998.

[74] R. V. Rasmussen and M. A. Trick, “Round robin scheduling–a survey,” *European Journal of Operational Research*, vol. 188, no. 3, pp. 617–636, 2008.

[75] E. L. Hahne, “Round-robin scheduling for max-min fairness in data networks,” *IEEE Journal on Selected Areas in communications*, vol. 9, no. 7, pp. 1024–1039, 1991.

[76] S. Ramesh and S. K. Kasera, “Best effort session-level congestion control,” in *2007 IEEE International Conference on Network Protocols.* IEEE, 2007, pp. 236–245.

[77] V. B. Iversen, *Teletraffic engineering and network planning.* DTU Fotonik, 2015.

[78] N. Nikaein, M. Laner, K. Zhou, P. Svoboda, D. Drajic, M. Popovic, and S. Krco, “Simple traffic modeling framework for machine type communication,” in *ISWCS 2013; The Tenth International Symposium on Wireless Communication Systems*, 2013, pp. 1–5.

[79] M. Laner, N. Nikaein, P. Svoboda, M. Popovic, D. Drajic, and S. Krco, “Traffic models for machine-to-machine (m2m) communications: types and applications,” in *Machine-to-machine (M2M) Communications.* Elsevier, 2015, pp. 133–154.

[80] P. S. I. Petizn and A. Nogueira, “Characterization and modeling of m2m video surveillance traffic.” IARIA Int., 2012, pp. 65–70.

[81] R. H. Khan and J. Y. Khan, “A comprehensive review of the application characteristics and traffic requirements of a smart grid communications network,” *Computer Networks*, vol. 57, no. 3, pp. 825–845, 2013.

[82] W. Dargie and C. Poellabauer, *Fundamentals of wireless sensor networks: theory and practice.* John Wiley & Sons, 2010.

[83] J. Mocnej, A. Pekar, W. K. Seah, P. Papcun, E. Kajati, D. Cupkova, J. Koziorek, and I. Zolotova, "Quality-enabled decentralized iot architecture with efficient resources utilization," *Robotics and Computer-Integrated Manufacturing*, vol. 67, p. 102001, 2021.

[84] M. Bocca, O. Kaltiokallio, and N. Patwari, "Radio tomographic imaging for ambient assisted living," in *International Competition on Evaluating AAL Systems through Competitive Benchmarking*. Springer, 2012, pp. 108–130.

[85] G. Pabbisetty and H. Mori, "A simple decentralized timeslot synchronization algorithm for large-scale wireless iot networks," in *2021 16th Annual Conference on Wireless On-demand Network Systems and Services Conference (WONS)*. IEEE, 2021, pp. 1–4.

[86] K. D. Colling, P. R. Carpenter, L. Koleszar, and P. E. Schmidt, "Distributed ad hoc network protocol using synchronous shared beacon signaling," Feb. 26 2013, uS Patent 8,385,322.

[87] H.-J. Byun and J.-T. Lim, "On a fair congestion control scheme for *TCP* vegas," *IEEE Communications Letters*, vol. 9, no. 2, pp. 190–192, 2005.

[88] S. Ha, I. Rhee, and L. Xu, "Cubic: a new *TCP*-friendly high-speed *TCP* variant," *ACM SIGOPS operating systems review*, vol. 42, no. 5, pp. 64–74, 2008.

[89] J. Wang, J. Wen, Y. Han, J. Zhang, C. Li, and Z. Xiong, "Cubic-fit: A high performance and *TCP* cubic friendly congestion control algorithm," *IEEE Communications Letters*, vol. 17, no. 8, pp. 1664–1667, 2013.

[90] J. Wang, J. Wen, J. Zhang, and Y. Han, "*TCP*-fit: An improved *TCP* congestion control algorithm and its performance," in *2011 Proceedings IEEE INFOCOM*. IEEE, 2011, pp. 2894–2902.