Efficient Schemes for Estimating the Number of Affected Nodes In Very Large Networks

Alexander Landau
Efficient Schemes for Estimating the Number of Affected Nodes In Very Large Networks

Research Thesis

Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

Alexander Landau

Submitted to the Senate of the Technion — Israel Institute of Technology

Cheshvan 5770 Haifa October 2009
This Research Thesis was done under the supervision of Assoc. Prof. Reuven Cohen in the Department of Computer Science.

The generous financial support of the Technion is gratefully acknowledged.
# Contents

Abstract 1

Abbreviations and Notations 3

1 Introduction 5

2 “Not All At Once!” – A Generic Scheme for Estimating the Number of Affected Nodes 8
   2.1 Introduction ......................................................... 8
   2.2 Application Scenarios and Related Work .................... 9
      2.2.1 Application scenarios ........................................ 9
      2.2.2 Related work for the NATO! mathematical scheme .......... 11
   2.3 The Estimation Algorithm of NATO! ......................... 13
   2.4 Precision Analysis and Error Cancellation .................. 22
   2.5 The Effect of RPRT Loss ........................................ 33

3 H-NATO! – A Scalable Scheme for Preventing Feedback Implosion in a Large-Scale
Multi-Tier Sensor Network 37
   3.1 Introduction ......................................................... 37
   3.2 Related Work ....................................................... 38
   3.3 A Hierarchical Implementation of NATO! ...................... 40
   3.4 The H-NATO! Scheme and Its Analysis ...................... 42
      3.4.1 The New Scheme ............................................. 42
      3.4.2 A New Estimation Algorithm .............................. 45
      3.4.3 The Error of Eq. 3.2 ...................................... 48
   3.5 Simulation Study .................................................. 51
4 Using NATO! for Partially Reliable Large Scale Multicast Streaming in Broadcast Wireless Networks

4.1 Introduction .................................................. 54
4.2 Related Work .................................................. 55
4.3 Reliable Multicast with NATO! ............................. 57
   4.3.1 Optimization Criteria for Partially Reliable Multicast .. 57
   4.3.2 The Proposed Protocols ................................. 58
4.4 The Performance of the Proposed Protocols ............... 60
   4.4.1 Simulation Results for Protocol 1 and 2 ............... 60
   4.4.2 Comparison to a 2-round Reactive Protocol .......... 64
4.5 The Implementation of NATO! in a Contention Channel ... 65
   4.5.1 The Effect of RPR T Losses on NATO! ............... 65
   4.5.2 Combining NATO! with a Collision Resolution Protocol .. 66
   4.5.3 Simulation Results for the Performance of NATO! with Collision Resolution 68

5 Conclusions ...................................................... 71
# List of Figures

2.1 Estimation error vs. $N$ for $r = 100, 1,000$ and $10,000$ before the “error cancellation” algorithm is implemented ........................................ 19
2.2 Real values of the estimation error vs. $N$ for $r = 100, 1,000$ and $10,000$ before the “error cancellation” algorithm is implemented ........................................ 20
2.3 Estimation error and coefficient of variation vs. $N$ for $r = 1,000$ and $10,000$ before the “error cancellation” algorithm is implemented ................................. 21
2.4 The estimation error as a function of $N$ for $r = 10,000$ before error cancellation: analytic prediction vs. simulation results .................................................. 32
2.5 Estimation error vs. $N$ before and after “error cancellation” .................................................. 33
2.6 Estimation error and coefficient of variation vs. $N$ for $r = 1,000$ and $10,000$ after “error cancellation” .................................................. 34
2.7 Estimation error vs. $N$ for $S = 1$, and $5$ .................................................. 35
2.8 Estimation error vs. $N$ for $S = 10$, before and after “error cancellation” .................................................. 36
3.1 An example of a 3-tier sensor network hierarchy .................................................. 41
3.2 An example of the new scheme with two levels .................................................. 43
3.3 Estimation error vs. $N$ when Algorithm 1 is used in Scheme 2 .................................................. 44
3.4 The estimation error of Eq. 3.2 (actual error vs. our analysis) .................................................. 50
3.5 The estimation error of Eq. 3.2 and of Algorithm 4 .................................................. 51
3.6 Estimation error of Algorithm 4 vs. $N$ for $B = 100$ and 500 Level-1 gateways .................................................. 52
3.7 Estimation error vs. $N$ for $B = 200$, with a uniform and non-uniform distribution of affected nodes per Level-1 gateway .................................................. 52
3.8 Estimation error vs. number of gateway levels for 6,250,000 affected nodes and $N = 5$ and $10$ .................................................. 53
4.1 Protocol 1 for $\text{SNR}_{\text{min}} = 15.1$ dB, $s = 5$ and $K = 10, 20$ and $30$ .................................................. 60
4.2 Protocol 1 for $K = 10$, $s = 5$ and $SNR_{\text{min}} = 14.7, 15.1$ and 15.3 dB ............... 61
4.3 Protocol 1 for $SNR_{\text{min}} = 13.5$ dB, $K = 10$, $s=5, 10$ and 50 ............................... 61
4.4 Protocol 1 for $K = 10$, $s = 5$, and a different SNR distribution ................................. 62
4.5 Protocol 2 for $SNR_{\text{min}} = 14.7$ dB, $s = 5$ and $K = 10$ ............................... 63
4.6 Protocol 2 for $SNR_{\text{min}} = 15.1$ dB, $s = 5$ and $K =20$ and 30 ............................... 63
4.7 Estimation error vs. $N$ for $S = 1$ and 5 .................................................. 66
4.8 Number of POLLs required for getting the smallest $N$ RPRTs ............................... 69
4.9 Number of POLLs required for getting the smallest $N$ RPRTs using a uniform and a truncated exponential distribution ................................. 69
Abstract

In this thesis, we present a generic scheme that we call NATO! (Not All aT Once!) for estimating the size of a group of nodes affected by the same event in a large-scale network, such as a grid, a sensor network or a wireless broadband access network, while receiving only a small number of feedback messages from this group. Using the proposed scheme, a centralized gateway analyzes the transmission times of these feedback messages, defines a likelihood function for them, and then uses the Newton-Raphson method to find the number of affected nodes for which this function is maximized. We present a complete mathematical analysis for the precision of the proposed algorithm and provide tight upper and lower bounds for the estimation error. These bounds allow us to improve the precision of our estimation, bringing the error very close to 0.

NATO! has many interesting applications. For example, in an 802.16-like mobile wireless network, the base station can use it to discover a DoS attack. In grid networks, a management entity is typically responsible for thousands of nodes, many of which can fail at once due to a single event. NATO! can substantially reduce the number of failure reports received by a management station in such cases.

We then consider a huge hierarchical sensor network consisting of millions of sensors. We address the problem of how a centralized gateway can estimate the number of sensors affected by a certain event. We evaluate the use of NATO! in such a network and propose a scheme called H-NATO! (Hierarchical NATO!) for solving this problem in the most efficient way. We show that the error of the new scheme is very small even if the number of sensors experiencing an event is several millions.

Another application of NATO! we consider is partially reliable streaming multicast. Streaming multicast is one of the most important transport layer services in future broadcast wireless networks. This service enables the delivery of real-time voice and video applications to an almost unlimited number of mobile devices, by taking advantage of a base station’s ability to transmit a single packet to all the nodes in its cell. Using NATO!, we show how the base station can collect
information regarding the channel quality of thousands of receivers, and choose the best coding and transmission parameters based on this information.
### Abbreviations and Notations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPRT</td>
<td>Report message</td>
</tr>
<tr>
<td>MAC</td>
<td>Media Access Control</td>
</tr>
<tr>
<td>FEC</td>
<td>Forward Error Correction code</td>
</tr>
<tr>
<td>ARQ</td>
<td>Automatic Repeat Request</td>
</tr>
<tr>
<td>RTT</td>
<td>Round trip time</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

In this thesis we address a problem that arises in many modern networks, such as grid networks, satellite networks, sensor networks and broadband access wireless networks. Such networks consist of thousands of end devices (nodes) that are controlled or managed by a single centralized gateway. From time to time the end nodes send feedback messages to the gateway concerning local events about which the gateway has to be aware. While some events are only detected by a single node or a few nodes, some important events are likely to affect many nodes. The schemes proposed in this thesis are useful when it is crucial that the gateway not only be informed about such events but also have a rough estimation of the number of affected nodes, without having each node send a message to the gateway.

We call the first scheme \textit{NATO! (Not All aT Once!)} and define it in Chapter 2. The main idea is that after the gateway announces a possible event, every affected node waits a random amount of time before sending a feedback report message (RPRT). When the gateway receives enough RPRTs to estimate the number of affected nodes with good precision, it broadcasts a STOP message, telling the nodes that have not yet reported, not to send their RPRTs.

In sensor networks, sensors probe the surrounding environment and generate reports of the collected readings. Using wireless communication, these reports are sent to a control center, usually through a gateway deployed in the physical proximity of the sensors \cite{55}. Although one possible approach for designing a sensor application is to deploy homogeneous sensors and program each one to perform all possible application tasks, this yields a flat, non-scalable network of homogeneous nodes. An alternative, multi-tier approach is to use heterogeneous elements. In this approach, resource-constrained, low-power elements are in charge of performing the simple tasks, such as detecting scalar physical measurements, while stronger devices, called gateways, perform
complex tasks such as routing and resource management [5]. A large-scale cost-effective sensor network that consists of millions of nodes can be realized only by means of a multi-tier wireless network, where every tier employs a different wireless technology. Such an architecture allows the sensors to be controlled and alarms received quickly and reliably.

Chapter 3 addresses one of the most important problems in such a multi-tier, gigantic sensor network: collecting important real-time information from the sensors in a scalable way while preventing feedback implosion. In many sensor network applications, the root gateway needs to know not only about the occurrence of specific events, but also about their scale. For instance, the gateway is likely to be interested not only in the detection of a sudden temperature increase by the sensors, but also in the number of sensors experiencing this event. Having every sensor notify the gateway would result in feedback implosion due to the huge number of sensors. The scheme proposed in Chapter 3 informs the root gateway of the number of sensors experiencing a given event without requiring each of them to send its own notification message.

The main idea behind the proposed H-NATO! scheme (Hierarchical Not All at Once) is that only a small fraction of the sensors notify their area gateways, which then forward only some of these notification messages to the upper-level gateways. Thus, only a few messages (less than 10) arrive at the root gateway, which then processes them with our estimation algorithm and gets a precise estimate of the number of sensors experiencing the event. We show that the error of the new scheme is very small even if the number of nodes experiencing an event is several million. Moreover, we show that the communication cost of the proposed scheme is the minimum one could achieve, because each gateway needs to send only 2 messages.

Streaming multicast is one of the most important applications in future broadcast wireless networks such as WiMax/802.16 [24] and 3GPP/LTE [1]. This service enables the delivery of real-time voice and video services to practically unlimited number of mobile devices, by taking advantage of a base station’s ability to transmit a single packet to all the nodes in its cell.

In contrast to other multicast delivery service models in broadcast wireless networks – the on-demand service model and the push service model [39], for example – streaming multicast has two notable properties:

- full reliability is not possible;
- full reliability is not essential.

Full reliability is not possible due to (a) occasionally bad wireless channel conditions and intermittent disconnection introduced by mobility of the hosts, and (b) the streaming nature of the
broadcast data, which puts hard time limits – in the order of a few hundreds of milliseconds – on the time the delivery of every data block must be completed. Full reliability of streaming multicast is not essential because streaming applications such as audio and/or video broadcast can tolerate loss of data. If the loss is temporary, it might not even be noticed by the user due to the robustness of the audio/video codecs. If the loss is long in duration, e.g., due to a physical obstacle between the mobile node and the base station, the user is likely to want to continue receiving the audio/video broadcast despite the long blackout period. In fact, it is very common for users to join the streaming multicast group in the middle of the session, as they do when they listen to the radio or watch a TV station.

In Chapter 4 we propose to use NATO! to improve the performance of large-scale partially reliable multicast streaming in broadcast wireless networks. The use of NATO! is motivated by the following properties of such networks:

1. Continuous fading causes strong packet loss correlation over time. Since the number of repair rounds is usually limited, the sender may need to send in every round repair packets that will compensate not only for the loss in the previous round, but also for possible loss in the next round. Such packets are referred to as “proactive repairs” [3, 45]. The best way for the sender to estimate the number of proactive repairs is to have some information about the loss distribution in the previous rounds.

2. Continuous fading might also result in receivers whose PHY conditions are extremely bad. For reasons of cost effectiveness, a sender with a good estimate for the loss distribution may decide not to take such receivers into account when determining the number of repair packets to be sent.
Chapter 2

“Not All At Once!” – A Generic Scheme for Estimating the Number of Affected Nodes

2.1 Introduction

In this chapter, we describe the NATO! scheme. The most important contribution of this chapter is the development of a statistical analysis algorithm, to be employed by the gateway, for estimating the number of affected nodes. The estimation is based on the times at which the RPRTs are sent from the affected nodes to the gateway. This algorithm defines a likelihood function for the received RPRTs, and then uses the Newton-Raphson method to find the number of nodes for which this function is maximized. Using mathematical analysis, we provide tight upper and lower bounds for the estimation error. We show that the error is approximately $1/(N - 1)$, where $N$ is the number of sent RPRTs, and it is always positive. We use this property to bring the estimation error very close to 0.

The rest of this chapter is organized as follows. In Section 2.2 we present detailed application scenarios along with related work. In Section 2.3 we present the estimation algorithm, which is the core of the proposed NATO! scheme. In Section 2.4 we analyze the precision of this algorithm and find tight upper and lower bounds for the error. This analysis allows the error introduced by our estimation to be reduced and brought very close to 0 even if $N$ is very small compared to the total number of affected nodes. In Section 2.5 we analyze the effect of feedback losses on the precision of the estimation.
2.2 Application Scenarios and Related Work

2.2.1 Application scenarios

The proposed NATO! scheme is suitable for networks and systems that meet the following requirements:

(R1) The network consists of thousands of end nodes reporting to a single centralized gateway. Having each affected node send a separate report message (RPRT) to the gateway would result in one or more of the following implosion effects: (a) insufficient network resources for forwarding the messages to the gateway; (b) insufficient gateway CPU resources for processing all these messages; (c) delayed gateway response to the event.

(R2) It is not enough that the sender knows about the event. In order to correctly respond to it, the sender also needs a good estimate of the number \( r \) of nodes that have experienced this event.

(R3) There is a strict limit between the time the event takes place and the time the gateway needs to estimate the number of affected nodes.

(R4) The gateway is able to broadcast a START message and a STOP message to all of the nodes that might be affected by the event. The START message indicates that RPRT messages should be sent and the STOP message indicates that no more RPRT messages should be sent.

(R5) To ensure synchronous execution of the protocol, one of the following requirements must hold: (a) the gateway and the nodes share a global time, e.g., using a GPS; or (b) the delays from the gateway to all the nodes are almost equal, as in most of the single hop wireless/cellular networks when the base-station plays the role of the gateway; or (c) the gateway knows the delay \( D_i \) to/from every node \( i \).

We now present some application scenarios for which requirements R1-R5 hold, and present related work for each of them.

The detection of denial-of-service (DoS) attacks [52] in sensor networks is an important application scenario. Our NATO! scheme can discover many such possible attacks, especially when all sensors have direct wireless connectivity with the gateway. For instance, [37] describes an attack where the attacker prevents the sensors from receiving the gateway’s broadcast messages. Using the proposed NATO! scheme, the gateway can periodically estimate the number of sensors
that are able to receive its broadcast messages without requiring each of them to send an individual response. Attacks on the physical and MAC layers of sensor networks, such as jamming attacks [52], can be efficiently discovered by NATO! in a similar way.

The NATO! scheme can also be used to discover a DoS attack in a 802.16-like mobile wireless network [24]. Possible attacks that prevent connectivity from many hosts to the base-station are described in [10]. In this context, the 802.16 base-station plays the role of NATO!’s gateway. The base-station periodically invokes the scheme in order to ensure that the estimated number of responding nodes is roughly equal to the number of supposedly active nodes.

The third application we discuss is management of grid networks. As organizations deploy large, Internet-scale, computational and data grids, the necessity for systems that monitor and control vast amounts of available resources has become apparent. Such systems require collection of a substantial amount of monitoring data for a variety of tasks such as fault detection, performance analysis, performance tuning, performance prediction, and scheduling [50, 53].

For scalability issues, such monitoring and control systems have a hierarchical structure. Still, a management entity is typically responsible for thousands of nodes, and it is essential to minimize the amount of control information sent to/from each node. Because a grid consists of clusters of nodes, hundreds of nodes in the same cluster may be affected by an event. For instance, hundreds of farm PCs that are connected to the same storage will be affected when this storage fails. The NATO! scheme can substantially reduce the number of messages received by a grid management station or a grid local management station, in such cases. Even if the management station receives only 4-7 messages, it can still estimate the number of grid nodes affected by the event.

Reliable multicast in broadcast wireless/cellular/satellite networks is the fourth application we describe for the NATO! scheme. A prominent feature of these technologies is the base-station’s ability to transmit a single copy of a packet to a huge group of receivers. In a typical FEC-based reliable multicast, the sender creates from each data block $K + n$ packets. To decode the data block, a receiver must receive any $K$ of these packets. In a hybrid FEC/ARQ-based scheme [3, 4, 20, 30, 41, 45], receivers that have not received at least $K$ packets correctly notify the sender, by means of a NACK message, and the sender transmits additional repair packets. The number of such repair rounds is usually limited by real-time considerations.

One way to use NATO! for FEC-based reliable multicast without ARQ is as follows. Once every time-out period (e.g., once a second), the sender invokes NATO! in order to estimate the loss distribution for the considered multicast group. That is, the sender estimates the number of nodes in this group that have lost $p\%$ of multicast packets since the previous time-out, for several relevant values of $p$. Using this information, the sender can determine the number $n$ of proactive
repair packets that have to be transmitted in addition to the $K$ packets required for the decoding of every data block. This value of $n$ is used for the considered multicast group until the next time NATO! is invoked.

The last application scenario is the prevention of feedback implosion in sensor networks. Due to the requirement of NATO! that the gateway will initiate the transmission of RPRT messages by the nodes, we concentrate here on sensor networks where the gateway periodically asks the nodes to report about specific events, such as temperature exceeding some threshold. Most papers that address this problem adopt the concept of data aggregation, e.g., [13, 21, 25, 48]. The idea is that similar data messages sent by multiple sources are aggregated by the network nodes. The aggregation depends on many factors, such as message content, identification, urgency, or the processing and storage (caching) capability of the intermediate nodes. Although useful, the data aggregation concept significantly increases the cost and complexity of the sensor nodes. Moreover, it renders the sensor network vulnerable to eavesdropping and data tampering [16], or requires security association between neighboring sensors. Therefore, solutions that avoid aggregation might be useful in many applications. The proposed NATO! scheme can solve the problem of feedback implosion without using data aggregation.

We are not aware of any paper that addresses requirements R1-R5 like the NATO! scheme proposed in this chapter. Some works have addressed a slightly different problem: estimating the total number of receivers in a multicast group [7, 8, 33, 40]. These works take advantage of the strong correlation between successive measurements when the size of a multicast group is estimated. This correlation, used to reduce the cost of the estimation process, does not exist in the NATO! applications considered above.

### 2.2.2 Related work for the NATO! mathematical scheme

The authors of [8] discuss the $M/M/\infty$ model for receivers entering and exiting the multicast group. To avoid feedback implosion, not all the receivers send a message to the sender. Rather, each one sends a message with a predefined probability $p$. The sender uses $p$ and other parameters to estimate the number of receivers. The authors of [7] extend [8] by relaxing several assumptions and using different filters on the received feedback messages. As noted earlier, our NATO! scheme differs from the one proposed in [7, 8] in that we do not assume any correlation between two successive measurements. Furthermore, in our approach, the gateway sets a hard limit on the number of feedback messages it is willing to receive.

The authors of [40] also employ a timer-based approach. However, in their scheme receivers
should stop sending feedback messages after the first one is transmitted. In [17], the authors propose a probabilistic polling model for estimating the size \( n \) of a group. In this model, polling takes place over several rounds. In each round \( i \), the sender multicasts a polling request. A receiver sends a response message with probability \( p_i \). After \( k \) rounds, the sender estimates the value of \( n \) from the polling probabilities \( p_1, p_2, \ldots, p_k \) and from the number of responses \( r_1, r_2, \ldots, r_k \) in every round. This paper also shows how to map the models of [12] and [40] to its binomial estimation model. The main differences between the scheme of [17] and NATO! are as follows. In [17], the number of response messages depends on the round trip time (RTT). In NATO!, this number is mainly determined by the estimating station, and it may also be affected by the RTT. In [17], if more than one response message is received by the estimating station, only the reception time of the first message is taken into account, along with the total number of received messages. In contrast, in NATO! the estimating station takes into account the reception times of all the messages.

As in NATO!, the authors of [32] also employ the concept of maximum likelihood. However, they do so for the sake of estimating the size of a multicast group. They consider a sender broadcasting an RFB (Request for Feedback) message to a group of receivers. Each receiver sets a random timer using a known probability distribution function and sends a feedback message when it expires. When the first feedback message arrives at the sender, it broadcasts the next RFB, which also stops the receivers from sending additional responses to the previous RFB. The number of feedback messages received by the sender is therefore proportional to the length of the RTT. The fact that our work and [32] address different problems is translated into the following differences:

- We overcome feedback implosion by limiting the number of response messages sent by receivers, while [32] limits the time during which these messages are sent.
- Running the algorithm of [32] in a network with a small RTT will result in a single response message being received by the sender. In contrast, in NATO! the number of response messages is mainly affected by the required estimation precision.
- In our scheme, each measurement of the size of the group of affected nodes is independent of previous measurements, while in [32] the results of previous measurements are taken into account.

In [12] the authors propose a multi-round scheme. The gateway and every node choose a random key, and in each round the gateway asks all nodes matching the first \( n \) bits of the key to
reply. The value of \( n \) is initially set to the full length of the key and is decreased by one every round. The authors find the relationship between the number of nodes and the expected value of the first round in which at least one response is received. [43] reduces the number of rounds in [12]. Instead of halving the current estimation of the group size in each round, [43] raises it to the power of \( \alpha \), where \( 0 < \alpha < 1 \), thereby increasing the speed of convergence.

NATO! is an important building block in each of the application scenarios considered above. Still, problems specific to each of these applications must be closely examined and addressed. For example, in the context of reliable broadcast in wireless/cellular/satellite networks, one needs to show how to implement NATO! when the response messages are subject to collisions on the shared uplink. This and other issues are addressed in Chapter 4.

### 2.3 The Estimation Algorithm of NATO!

Let \( r \) be the number of affected nodes, namely, nodes that are supposed to send a response/report message (RPRT) to the gateway after receiving a START message. Our goal is to accurately estimate this number while limiting the number of RPRTs sent by affected nodes. When a node discovers that it is affected by an event that has to be reported to the gateway, it invokes the following algorithm:

**Algorithm 1** This algorithm is invoked by every affected node upon receiving a START\((t_0)\) message from the gateway, where \( t_0 \) is the time after which RPRT messages can be sent to the gateway.

- Choose a random timer in the range \([t_0, t_0 + T]\) using a known absolutely continuous probability distribution function \( F \).

- If the timer expires before a STOP message is received from the gateway, send a RPRT to the gateway. Otherwise, do not send.

Following this algorithm, the gateway receives during \([t_0, t_0 + T]\) a number of RPRT messages. Let this number be \( N \). Let \( f \) be the probability density function of \( F \). Let \( X_1, \ldots, X_N \) be random variables denoting the transmission times of the \( N \) RPRTs. Without loss of generality, these random variables are assumed to be ordered in non-decreasing order such that \( X_1 \leq X_2 \leq \ldots \leq X_N \). Finally, let \( x_1, \ldots, x_N \) denote the exact values of \( X_1, \ldots, X_N \) in a specific experiment.

Without loss of generality, in the following analysis we assume that \( t_0 = 0 \). We use the maximum likelihood method to estimate \( r \). Let \( f_{X_1, X_2, \ldots, X_N \mid r}(x_1, x_2, \ldots, x_N) \) be the joint density
function of $X_1, X_2, \ldots, X_N$ given that the number of affected nodes is $r$. This function is the probability density of the first $N$ order statistics of distribution $F$, for which it is known that [42]:

$$f_{X_1, X_2, \ldots, X_r}(x_1, x_2, \ldots, x_r)dx_1 \ldots dx_r =$$

$$= P(X_1 \in (x_1, x_1 + dx_1), \ldots, X_r \in (x_r, x_r + dx_r)) =$$

$$= r! P(Y_1 \in (x_1, x_1 + dx_1), \ldots, Y_r \in (x_r, x_r + dx_r)) =$$

$$= r!(F(x_1 + dx_1) - F(x_1)) \ldots (F(x_r + dx_r) - F(x_r)) =$$

$$= r! f(x_1) \ldots f(x_r) dx_1 \ldots dx_r,$$

where $Y_1, \ldots, Y_r$ are independent random variables from distribution $F$. Therefore,

$$f_{X_1, X_2, \ldots, X_r}(x_1, x_2, \ldots, x_r) = r! f(x_1) \ldots f(x_r).$$

In order to find the joint density of the first $N$ $X_i$'s, we integrate over $x_{N+1}, \ldots, x_r$:

$$f_{X_1, X_2, \ldots, X_N | r}(x_1, x_2, \ldots, x_N)$$

$$= \int \cdots \int_{x_N < x_{N+1} < \cdots < x_r} f_{X_1, X_2, \ldots, X_r}(x_1, x_2, \ldots, x_r) dx_{N+1} \ldots dx_r$$

$$= \int \cdots \int_{x_N < x_{N+1} < \cdots < x_r} r! f(x_1) \ldots f(x_r) dx_{N+1} \ldots dx_r$$

$$= r! \int_{x_N}^{T} dx_{N+1} \cdots \int_{x_{r-1}}^{T} dx_r f(x_1) \ldots f(x_r)$$

$$= r! \int_{x_N}^{T} dx_{N+1} \cdots \int_{x_{r-1}}^{T} dx_r \prod_{i=1}^{r-1} f(x_i) \cdot (1 - F(x_{r-1}))$$

$$= r! \int_{x_N}^{T} dx_{N+1} \cdots \int_{x_{r-3}}^{T} dx_{r-2} \prod_{i=1}^{r-3} f(x_i) \cdot \frac{(1 - F(x_{r-2}))^2}{2}$$

$$= \cdots = \frac{r!}{(r - N)!} \prod_{i=1}^{N} f(x_i) \cdot (1 - F(x_N))^{r-N}. \quad (2.1)$$

Define the likelihood function $L(r)$ to be

$$L(r) = f_{X_1, X_2, \ldots, X_N | r}(x_1, x_2, \ldots, x_N).$$
We now seek for the value of \( r \) that maximizes \( L(r) \). Such an \( r \) yields the maximum likelihood for getting the considered experiment’s outcome, \( x_1, \ldots, x_N \), and is therefore the most probable number of affected nodes. We find the maximum of \( L(r) \) by differentiation. Since \( L(r) \) is a product of other functions, it is hard to differentiate it directly. Since \( \ln \) is a monotonically increasing function, \( L(r) \) gets its maximum for the same value of \( r \) as \( l(r) \), where

\[
l(r) = \ln L(r) = \ln \frac{r!}{(r-N)!} + \ln f(x_1) + \ldots + \ln f(x_N) + (r-N) \ln(1-F(x_N)) = \ln(r-N+1) + \ldots + \ln r + r \ln(1-F(x_N)) + \text{const.}
\]

(2.2)

In this equation, \( \text{const} \) is a constant with respect to \( r \).

We now differentiate \( l(r) \) with respect to \( r \) and get

\[
l'(r) = \frac{1}{r} + \frac{1}{r-1} + \ldots + \frac{1}{r-N+1} + \ln(1-F(x_N)).
\]

Thus, in order to find the value of \( r \) which maximizes the likelihood function \( L(r) \), we need to find real values of \( r \) that satisfy the following equation:

\[
\frac{1}{r} + \frac{1}{r-1} + \ldots + \frac{1}{r-N+1} + \ln(1-F(x_N)) = 0.
\]

(2.3)

**Proposition 1** From the \( N \) possible real solutions of Eq. 2.3, the one that maximizes \( L(r) \) is the maximum one.

**Proof:** \( L(r) \) and \( l(r) \) get their maximum at the same \( r \). Thus it is enough to show that \( l(r) \) gets its maximum at the maximum solution of Eq. 2.3.

Since for every \( r \)

\[
l''(r) = -\frac{1}{r^2} - \frac{1}{(r-1)^2} - \ldots - \frac{1}{(r-N+1)^2} < 0,
\]

(2.4)

then any real root of Eq. 2.3 is a local maximum of \( l(r) \). The global maximum is one of the local maxima, so it remains to find which of the local maxima gives the highest value of \( l \).
Substituting

\[ \ln(1 - F(x_N)) = -\left( \frac{1}{r} + \frac{1}{r - 1} + \ldots + \frac{1}{r - N + 1} \right) \]

from Eq. 2.3 into Eq. 2.2 yields:

\[
l(r^*) = \ln r^* + \ldots + \ln(r^* - N + 1) \\
- r^* \left( \frac{1}{r^*} + \frac{1}{r^* - 1} + \ldots + \frac{1}{r^* - N + 1} \right) + \text{const} \\
= \ln r^* + \ldots + \ln(r^* - N + 1) - 1 \\
- \left( 1 + \frac{1}{r^* - 1} \right) - \ldots - \left( 1 + \frac{N - 1}{r^* - N + 1} \right) + \text{const}.
\]

This is a monotonically increasing function. Therefore, of all the roots \( r^* \) of Eq. 2.3, the one whose value is maximum will maximize both \( l(r) \) and \( L(r) \).

A practical method for solving Eq. 2.3 is as follows. Since the \( \ln \) term is constant, the equation has the form \( \frac{1}{r} + \frac{1}{r - 1} + \ldots + \frac{1}{r - N + 1} + c = 0 \). This function has vertical asymptotes at points \( r = 0, 1, \ldots, N - 1 \). From Eq. 2.4 it follows that the function decreases monotonically at every interval \( (i - 1, i) \), \( i = 1, \ldots, N - 1 \) and thus it has \( N - 1 \) roots at the interval \( (0, N - 1) \). The function also decreases monotonically at the interval \( (N - 1, \infty) \), and thus has its greatest root in this interval.

This is the root we are seeking. To find it, the sender can employ the Newton-Raphson method. Given an equation \( h(x) = 0 \) where \( h \) is a continuously differentiable function and given a starting point \( x_0 \), near which the equation root is located, the method iteratively finds an approximation for the root with any desirable precision. On the \( (n+1) \)-th iteration, \( x_{n+1} = x_n - \frac{h(x_n)}{h'(x_n)} \), where \( h'(x) \) is the derivative of \( h(x) \). The idea is to find the tangent of \( h \) at \( x_n \) and to set \( x_{n+1} \) to the point where the tangent crosses the \( x \)-axis, thereby getting closer to the root. In our case, \( h \) is given in Eq. 2.3 and \( x_0 = N - 1 + \varepsilon \), where \( \varepsilon \) is small positive number. This starting point was chosen for two reasons. First, there must not be asymptotes between the starting point and the root (therefore \( x_0 > N - 1 \)). Second, there must not be asymptotes between any \( x_n \) and the root, and since the function is monotonic at the interval \( (N - 1, \infty) \), it is implied that \( x_0 < \text{root} \). The whole process stops when \( |h(x_n)| \) gets sufficiently close to 0. In the simulation results shown later, we stopped when \( |h(x_n)| < 0.001 \), which usually holds after 9-10 iterations. The value \( x_n \) for the last (\( n \)th) iteration is taken to be the solution of Eq. 2.3, namely, the estimated value of \( r \).

To conclude, the algorithm executed by the gateway for estimating the number of affected nodes \( r \) is as follows.
Algorithm 2 The gateway algorithm:

- Broadcast/multicast a START($t_0$) message to all possible affected nodes.
- When $N$ RPRTs messages are received, broadcast/multicast a STOP message to all possible affected nodes.
- Use the Newton-Raphson method, as described above, to find the greatest real root of Eq. 2.3.

Theorem 1 The absolutely continuous distribution function $F$ does not affect the estimated value of $r$ as computed in Eq. 2.3.

Proof: According to Eq. 2.3, the only way the value of $r$ might depend on $F$ is through $-\ln(1 - F(x_N))$. However, we will show now that for every $i$, the value of $-\ln(1 - F(x_i))$ does not depend on $F$, namely, that the distribution of the random variable $Y = -\ln(1 - F(X_i))$ does not depend on $F$ given $r$ affected nodes.

Denote by $f_{X_i|r}(x)$, for $i = 1, \ldots, N$, the density function of $X_i$ given that the number of affected nodes is $r$. The function $f_{X_i|r}(x)$ is actually the probability density of the $i$th order statistic of distribution $F$, namely:

$$f_{X_i|r}(x) = \frac{d}{dx} F_{X_i|r}(x) = \frac{d}{dx} P(X_i \leq x | r \text{ affected nodes})$$

$$= \frac{d}{dx} P(\text{at least } i \text{ of the } r \text{ timers expire before } x)$$

$$= \frac{d}{dx} \sum_{j=i}^{r} \binom{r}{j} F(x)^j (1 - F(x))^{r-j} = \ldots$$

$$= r \left( \begin{array}{c} r - 1 \\ i - 1 \end{array} \right) F(x)^{i-1} (1 - F(x))^{r-i} f(x).$$

Then, we have $F_{X_i|r}(t) = \int_0^t r \left( \begin{array}{c} r - 1 \\ i - 1 \end{array} \right) F(x)^{i-1} (1 - F(x))^{r-i} f(x) \, dx$. 

17
Substituting \( y = F(x) \), so that \( dy = F'(x)dx = f(x)dx \), yields

\[
F_{X_i|\tau}(t) = r \left( \frac{r - 1}{i - 1} \right) \int_0^{F(t)} y^{i-1} (1 - y)^{r-i} dy
\]

(2.5)

\[
= r \left( \frac{r - 1}{i - 1} \right) \int_0^{F(t)} y^{i-1} \sum_{k=0}^{r-i} (-1)^k \binom{r - i}{k} y^k dy
\]

(2.6)

\[
= r \left( \frac{r - 1}{i - 1} \right) \sum_{k=0}^{r-i} (-1)^k \binom{r - i}{k} \int_0^{F(t)} y^{k+i} dy
\]

(2.7)

\[
= r \left( \frac{r - 1}{i - 1} \right) \sum_{k=0}^{r-i} (-1)^k \binom{r - i}{k} \frac{F(t)^{k+i}}{k+i}
\]

(2.8)

Let \( F_{Y|\tau}(z) \) be the distribution function of \( Y \) given there are \( r \) affected nodes. Hence,

\[
F_{Y|\tau}(z) = P(- \ln(1 - F(X_i)) \leq z| r \text{ affected nodes})
\]

\[
= P(1 - F(X_i) \geq e^{-z}| r \text{ affected nodes})
\]

\[
= P(F(X_i) \leq 1 - e^{-z}| r \text{ affected nodes})
\]

\[
= P(X_i \leq F^{-1}(1 - e^{-z})| r \text{ affected nodes})
\]

\[
= F_{X_i|\tau}(F^{-1}(1 - e^{-z}))
\]

\[
= r \left( \frac{r - 1}{i - 1} \right) \sum_{k=0}^{r-i} (-1)^k \binom{r - i}{k} \frac{(1 - e^{-z})^{k+i}}{k+i}
\]

Thus, given \( r \) affected nodes, \( Y \) does not depend on \( F \).

In [40] it was shown that a truncated exponential distribution outperforms a uniform distribution at reducing feedback implosion when the time interval during which RPRTs are to be sent is limited. Theorem 1 suggests that this is not the case when setting a hard limit on the number of RPRTs. The time interval \([t_0, t_0 + T]\) during which the RPRTs are transmitted is arbitrary too, since changing \( T \) results in changing the support of the distribution.

We implemented the algorithm proposed in Section 2.3 for two distribution functions: the uniform distribution \( f(x) = \frac{1}{T} \) and the truncated exponential distribution \( f(x) = \frac{1}{e^{\lambda} - 1} \cdot e^{\lambda x}, \) for \( x \in [0, T] \). In the simulation, the receivers draw timers from the above distributions and send a RPRT to the sender if fewer than \( N \) RPRTs have already been sent.

We have simulated 100, 1000 and 10,000 affected nodes. Figure 2.1 depicts the average error in estimating \( r \), namely \( |r_{\text{real}} - r_{\text{estimated}}|/r_{\text{real}} \), as a function of \( N \) for the uniform distribution.
and for the truncated exponential distribution. Each point in the graph is the average of 1000 different runs. This figure reveals that there is no noticeable difference between the uniform distribution and the truncated exponential distribution, as predicted by Theorem 1. It is evident that the greater \( N \) is, the better the estimation is.

Another important property of the proposed NATO! scheme is that it actually overestimates the number of affected nodes, i.e., with high probability \( r_{\text{estimated}} \geq r_{\text{real}} \). This is shown in Figure 2.2, which depicts the estimation error as \( (r_{\text{estimated}} - r_{\text{real}})/r_{\text{real}} \), rather than as \( |r_{\text{estimated}} - r_{\text{real}}|/r_{\text{real}} \). We can see that for all values of \( N \) and for any number of affected nodes, the error is always positive. Moreover, by comparing the graphs in Figure 2.2 to those in Figure 2.1, we can see that the curves are almost identical. This implies that \( (r_{\text{estimated}} - r_{\text{real}})/r_{\text{real}} \approx |r_{\text{estimated}} - r_{\text{real}}|/r_{\text{real}} \), or, in other words, that the difference between the actual and the estimated values results from the overestimation. We prove this property mathematically in Theorem 2 of Section 2.4, and show how it can be used in order to bring the estimation error very close to 0.

An important property of an estimator is its variation. Like the estimation error, whose value is normalized with respect to the number of affected nodes, the variation should be normalized as well. To this end, we use the coefficient of variation, defined as the ratio of the standard deviation to the mean. In Figure 2.3 we show the estimation error and the coefficient of variation for 1,000 and 10,000 affected nodes during 1,000 runs of the algorithm, for uniform timer distribution. Like the estimation error, the coefficient of variation decreases with \( N \). For small values of \( N \), it is approximately half of the estimation error, while for large values of \( N \) they are of the same order. This means that with high probability, the estimation error of a single estimation is very close to 0.
the mean estimation error, implying that NATO! performs very well not only when we average multiple runs, but also for a single run.

Implementation Notes:

1. RPRT messages are subject to loss due to collisions on a contention channel or to transmission errors. In Section 2.5 we show that NATO! can tolerate the loss of a small number of RPRTs, and that this number strongly depends on $N$. For example, 1 lost message when $N = 4$ or 5 lost messages when $N = 100$ have a small effect on the estimation error. To address the problem of message loss, we enhance NATO! with the following reliability mechanism. Upon receiving an RPRT, the gateway sends a confirmation to the sender. An RPRT sender that does not receive a confirmation within a time-out period resends its RPRT, and specifies the offset between its current local time and the time of the original RPRT.

2. In the protocol described so far, it was assumed that all the nodes are synchronized to a common clock or that all of them receive the START message almost at the same time. However, by requirement R5 from Section 2.2, NATO! can also be executed in a system where there is no common clock and there is a variable delay between the gateway and the various nodes, provided that the gateway knows the delay $D_i$ to/from every node $i$. In such a system the following adjustment should be used: Let $D_i$ be the delay from node $i$ to the gateway, and let $D = \max_i(D_i)$ be the maximum delay. The gateway adds the value of $D$ to the START message and each node $i$ should wait a time period of $D - D_i$ before running Algorithm 1.
3. To overcome a possible loss of the STOP message, the gateway must retransmit this message if it receives an RPRT after it sends a STOP, while taking into account the maximum round trip time from the gateway to the nodes.

4. When the delay from the gateway to the nodes is not negligible compared to the value of \( T \) from Algorithm 1, the gateway is likely to receive more RPRTs after sending a STOP message. This problem can be addressed by Algorithm 3 below.

**Algorithm 3** The gateway algorithm when the delay from the gateway to the nodes is not negligible compared to \( T \).

Let \( T_N(t) \) be the estimation made by the gateway at time \( t \) for the time elapsed until the \( N \)'th RPRT will be received. This estimation is based on the RPRTs received by the gateway until time \( t \). Let \( D \) be the propagation delay to the farthest node. When \( T_N(t) = D \), a STOP message has to be sent. Let \( N' \) be the number of RPRTs actually received by the gateway when it sends this STOP. If \( N' \neq N \), the gateway should use the value of \( N' \) rather than the value of \( N \) when solving Eq. 2.3.

The expected value of the time of the \( i \)'th RPRT can be computed using Eq. 2.5 as follows:

\[
E(X_i) = \int_0^T x \cdot r \left( \frac{r - 1}{i - 1} \right) F(x)^{i-1} (1 - F(x))^{r-i} f(x) \, dx.
\]
This integral can be computed numerically for any distribution $F$. For the uniform distribution it can also be computed analytically: substituting $f(x) = 1$ and $F(x) = x$ we get

$$E(X_i) = r \binom{r-1}{i-1} \frac{i!(r-i)!}{(r+1)!} = \frac{i}{r+1}.$$ 

Therefore, for a uniform distribution, if by time $t$ the gateway receives $n$ RPRTs, the $N$’th RPRT is expected to arrive at $\frac{t}{n} \cdot N$, and $T_N(t) = t \cdot (\frac{N}{n} - 1)$. It is evident from the graphs presented in Section 2.4 that even in the unlikely event that $N'$ is smaller than $N$ by 20-30%, the precision of the estimation is not affected at all.

### 2.4 Precision Analysis and Error Cancellation

In this section we analyze the precision of our algorithm. The importance of this section is twofold. First, we prove that the estimation error is approximately $\frac{1}{r(N-1)}$. Second, we use this analysis in order to further reduce the error, and to bring it very close to 0.

We have already shown that the absolutely continuous timer distribution function $F$ does not affect the result $r$ of the estimation. Therefore, in the following analysis we consider a uniform timer distribution on the interval $[0, 1]$. On that interval, $f(x) = 1$ and $F(x) = x$.

The gateway estimates the number of affected nodes by finding the maximal value of $r$ that solves the following equation:

$$\frac{1}{r} + \frac{1}{r-1} + \ldots + \frac{1}{r-(N-1)} = -\ln(1 - x_N).$$ (2.9)

Let this solution be $r = g(x_N)$. Let $R$ be a random variable denoting the estimated number of affected nodes, and let its expected value be $E(R)$. Our goal is to approximate the relative error $\frac{E(R) - r}{r}$.

We have seen that

$$f_{X_i | r}(x) = r \binom{r-1}{i-1} F(x)^{i-1}(1 - F(x))^{r-i} f(x).$$ (2.10)

Since $f(x) = 1$ and $F(x) = x$, then substituting $i = N$ into Eq. 2.10, we get

$$f_{X_N | r}(x) = r \binom{r-1}{N-1} x^{N-1}(1 - x)^{r-N}.$$
Therefore,

\[ E(R) = \int_0^1 g(x)f_{X_N|r}(x) \, dx. \] (2.11)

We used an iterative method in order to solve \( g(x) \). In what follows we seek for upper and lower bounds on this function. From Eq. 2.9 follows that

\[-\ln(1 - x_N) \geq \frac{N}{r} \quad \text{and} \quad -\ln(1 - x_N) \leq \frac{N}{r - (N-1)}.\]

Therefore,

\[-\frac{N}{\ln(1 - x_N)} \leq g(x_N) \leq -\frac{N}{\ln(1 - x_N)} + (N - 1). \] (2.12)

To find a lower bound we substitute the left-hand part of Eq. 2.12 into Eq. 2.11 and get

\[ E(R) \geq -rN \binom{r-1}{N-1} \int_0^1 \frac{x^{N-1}(1-x)^{r-N}}{\ln(1-x)} \, dx \]

\[ = -rN \binom{r-1}{N-1} \int_0^1 \frac{x^{r-N}(1-x)^{N-1}}{\ln x} \, dx. \] (2.13)

Next, we note that

\[ \int_0^1 \frac{x^n(1-x)^k}{\ln x} \, dx = \sum_{i=0}^{k} (-1)^i \binom{k}{i} \ln(n + i + 1), \] (2.14)

for \( n \geq 0, k \geq 1, \) and \( \int_0^1 \frac{x^n}{\ln x} \, dx = -\infty \) for \( n \geq 0, k = 0. \) This means that for \( N = 1, E(R) = \infty \) and the relative estimation error is also infinite. Thus, from now on we assume that \( N \geq 2. \) In the following equations, a sum or a product whose lower limit is greater than its upper limit is considered to be equal to 0 or to 1 respectively. Substituting Eq. 2.14 into Eq. 2.13, yields:

\[ E(R) \geq -rN \binom{r-1}{N-1} \]

\[ \cdot \sum_{i=0}^{N-1} (-1)^i \binom{N-1}{i} \ln(r - (N-1) + i) \]

\[ = -rN \frac{(N-1)!}{(N-1)!} \prod_{i=0}^{N-2} (r - (N-1) + i) \]

\[ \cdot \sum_{i=0}^{N-1} (-1)^i \binom{N-1}{i} \ln(r - (N-1) + i). \] (2.16)
We can expand the right item of this product as follows:

\[
\sum_{i=0}^{N-1} (-1)^i \binom{N - 1}{i} \ln(r - (N - 1) + i)
\]

\[
= \ln(r - (N - 1))
\]

\[
+ \sum_{i=1}^{N-2} (-1)^i \left[ \binom{N - 2}{i - 1} + \binom{N - 2}{i} \right]
\]

\[
\cdot \ln(r - (N - 1) + i) + (-1)^{N-1} \ln r
\]

\[
= \ln(r - (N - 1))
\]

\[
+ \sum_{i=0}^{N-3} (-1)^{i+1} \binom{N - 2}{i} \ln(r - (N - 1) + i+1)
\]

\[
+ \sum_{i=1}^{N-2} (-1)^i \binom{N - 2}{i} \ln(r - (N - 1) + i)
\]

\[
+ (-1)^{N-1} \ln r.
\] (2.17)
We now group in Eq. 2.17 those \( \ln \) terms that are adjacent but differ in sign, and we get

\[
\ln(r - (N - 1)) \\
- \sum_{i=0}^{N-3} (-1)^i \binom{N-2}{i} \ln(r - (N - 1) + i + 1) \\
+ \sum_{i=1}^{N-2} (-1)^i \binom{N-2}{i} \ln(r - (N - 1) + i) \\
+ (-1)^{N-1} \ln r = \\
\ln \frac{r - (N - 1)}{r - (N - 1) + 1} + \\
\sum_{i=1}^{N-3} (-1)^i \binom{N-2}{i} \ln \frac{r - (N - 1) + i}{r - (N - 1) + i + 1} + \\
+ (-1)^{N-2} \ln \frac{r - 1}{r} = \\
= - \ln \left(1 + \frac{1}{r - (N - 1)}\right) - \\
\sum_{i=1}^{N-3} (-1)^i \binom{N-2}{i} \ln \left(1 + \frac{1}{r - (N - 1) + i}\right) - \\
- (-1)^{N-2} \ln \left(1 + \frac{1}{r - 1}\right) = \\
= - \sum_{i=0}^{N-2} (-1)^i \binom{N-2}{i} \ln \left(1 + \frac{1}{r - (N - 1) + i}\right).
\]
Substituting this into Eq. 2.16 yields

\[ E(R) \geq \frac{rN}{(N-1)!} \prod_{i=0}^{N-2} (r - (N - 1) + i) \] (2.18)
\[ \cdot \sum_{i=0}^{N-2} (-1)^i \binom{N-2}{i} \ln \left(1 + \frac{1}{r - (N - 1) + i}\right) \]
\[ = \frac{rN}{(N-1)!} \sum_{i=0}^{N-2} (-1)^i \binom{N-2}{i} \]
\[ \cdot \prod_{j=0}^{N-2} \left(\prod_{j \neq i} (r - (N - 1) + j)\right) \]
\[ \cdot \ln \left(1 + \frac{1}{r - (N - 1) + i}\right)^{r-(N-1)+i} . \]

The sequence

\[ \left\{ \left(1 + \frac{1}{r - (N - 1) + i}\right)^{r-(N-1)+i} \right\}_{i=0}^{N-2} \]

is monotonically increasing and upper bounded by e. Denote

\[ p = \ln \left(1 + \frac{1}{r - (N - 1)}\right)^{r-(N-1)} . \]

Then, \(0 < p < 1\), and for large enough values of \(r - (N - 1)\), \(p\) is close to 1. We now get:

\[ E(R) \geq \frac{p \cdot r \cdot N}{(N-1)!} \]
\[ \cdot \sum_{i=0}^{N-2} (-1)^i \binom{N-2}{i} \left[ \prod_{j=0}^{N-2} \left(\prod_{j \neq i} (r - (N - 1) + j)\right) \right] . \]

We continue with a series of propositions that will help us to simplify the above expression.

**Proposition 2** Let \(c_k\) be the coefficient of \(r^k\) in the polynomial \(\prod_{j=0}^{N-2} (r - (N - 1) + j)\), for \(1 \leq k \leq N - 2\). Then, there exists a polynomial \(p(x)\) of degree \(N - 2 - k\) such that \(c_k = p(i)\).
Proof: For $1 \leq k \leq N - 2$,

\[
c_k = \sum_{A_1} \text{product of the factors } -[(N - 1) - j] = \\
= \sum_{A_2} \text{product of the factors } -[(N - 1) - j] + \\
\cdot \sum_{A_3} \text{product of the factors } -[(N - 1) - j] = \\
= a_k + [(N - 1) - i]c_{k+1},
\]

where:

- $A_1$ is the set of all ways to choose $N - 2 - k$'s from $\{0, \ldots, N - 2\}\{i\}$.
- $A_2$ is the set of all ways to choose $N - 2 - k$'s from $\{0, \ldots, N - 2\}$.
- $A_3$ is the set of all ways to choose $N - 3 - k$'s from $\{0, \ldots, N - 2\}\{i\}$.
- $a_k$ is a constant with respect to $i$.

For $k = N - 2$, $c_{N-2} = 1$. Therefore, the proposition holds for $c_{N-2}$ with the constant polynomial $p(x) = 1$.

By reverse induction, suppose that the proposition holds for $k + 1$. Then, there is a polynomial $q(x)$ of degree $N - 3 - k$ such that $c_{k+1} = q(i)$. Define $p(x) = a_k + (N - 1)q(x) - iq(x)$. Then, $p(x)$ is a polynomial of degree $N - 2 - k$, and $p(i) = a_k + (N - 1)q(i) - iq(i) = a_k + [(N - 1) - i]c_{k+1} = c_k$.

Proposition 3 For all $n \geq 0$, $\sum_{i=0}^{n} \binom{n}{i}(-1)^i = 0$. For all $k \geq 1$ and $n \geq k+1$, $\sum_{i=k}^{n} \binom{n}{i}(-1)^i(i-1) \ldots (i-(k-1)) = 0$.

Proof: The first claim follows immediately from the binomial formula:

\[
\sum_{i=0}^{n} \binom{n}{i}(-1)^i = \sum_{i=0}^{n} \binom{n}{i}(-1)^i1^{n-i} = (1 - 1)^n = 0.
\]

The second claim is proved by differentiating $(x - 1)^n$ $k$ times, first as a composite function and
then after expansion using the binomial formula.

\[
[(x-1)^{n}]^{(k)} = n(n-1)\ldots(n-(k-1))(x-1)^{n-k}
\]

\[
[(x-1)^{n}]^{(k)} = \left[\sum_{i=0}^{n} \binom{n}{i}(-1)^{n-i}x^{i}\right]^{(k)} = (-1)^{n}\sum_{i=k}^{n} \binom{n}{i}(-1)^{i} \cdot i(i-1)\ldots(i-(k-1))x^{i-k}.
\]

By substituting \(x = 1\), the proof is completed.

\[\square\]

**Proposition 4** Every polynomial \(p(x) = \sum_{i=0}^{k} a_{i} x^{i}\) of degree \(k\) can be written as a linear combination of the polynomials in the set \(B_{k} = \{1, x, x(x-1), x(x-1)(x-2), \ldots, x(x-1)\ldots(x-(k-1))\}\).

**Proof:** By induction on \(k\). For \(k = 0\), \(p(x) = a_{0}\) is certainly a linear combination of the polynomials in \(B_{0} = \{1\}\). For a general \(k \geq 1\),

\[
p(x) = a_{k} \cdot x(x-1)\ldots(x-(k-1)) + [p(x) - a_{k} \cdot x(x-1)\ldots(x-(k-1))],
\]

where the polynomial in brackets is of degree \(k - 1\). Thus, by the induction hypothesis, it can be written as a linear combination of the polynomials in \(B_{k-1}\).

\[\square\]

**Proposition 5** For every \(k \geq 0\) and \(n \geq k + 1\), \(\sum_{i=0}^{n} \binom{n}{i}(-1)^{i}p_{k}(i) = 0\), where \(p_{k}(x)\) is a polynomial of degree \(k\).

**Proof:** By Proposition 4 we can write

\[
p_{k}(i) = b + a_{0}i + a_{1}i(i-1) + \ldots + a_{k-1}i(i-1)\ldots(i-(k-1)).
\]

28
Then, by Proposition 3,

\[
\sum_{i=0}^{n} \binom{n}{i} (-1)^i p_k(i)
\]

\[
= b \sum_{i=0}^{n} \binom{n}{i} (-1)^i + a_0 \sum_{i=1}^{n} \binom{n}{i} (-1)^i + \ldots
\]

\[
+ a_{k-1} \sum_{i=k}^{n} \binom{n}{i} (-1)^i (i-1) \ldots (i-(k-1))
\]

\[
= 0.
\]

We will now use the above propositions to simplify the expression for \(E(R)\) from Eq. 2.19:

\[
\sum_{i=0}^{N-2} (-1)^i \binom{N-2}{i} \left[ \prod_{j=0, j \neq i}^{N-2} (r - (N-1) + j) \right]
\]

\[
= \sum_{i=0}^{N-2} (-1)^i \binom{N-2}{i} \left[ N-2 \sum_{k=0}^{N-2} c_k r^k \right]
\]

\[
= \sum_{k=1}^{N-2} r^k \left[ \sum_{i=0}^{N-2} \binom{N-2}{i} (-1)^i c_k \right]
\]

\[
+ r^0 \sum_{i=0}^{N-2} \binom{N-2}{i} (-1)^i c_0.
\]

By Proposition 2, \(c_k\) can be written as a polynomial of degree \(N-2-k\) for the variable \(i\). Thus, by Proposition 5, the first term vanishes. Note that \(c_0 = (-1)^{N-2} \frac{(N-1)!}{(N-1)-1} = (-1)^N \frac{(N-1)!}{(N-1)-1}\), and
so the second term can be expanded as follows:

\[ r^0 \sum_{i=0}^{N-2} \binom{N-2}{i} (-1)^i c_0 \]

\[ = (-1)^N \sum_{i=0}^{N-2} \binom{N-2}{i} (-1)^i \frac{(N-1)!}{N-1-i} \]

\[ = (-1)^N (N-1)! \sum_{i=0}^{N-2} \binom{N-2}{i} (-1)^i \frac{1}{i+1} \]

\[ = (N-1)! \sum_{i=0}^{N-2} (-1)^i \frac{(N-2)!}{(i+1)!(N-2-i)!} \]

\[ = (N-1)! \frac{1}{N-1} \sum_{i=0}^{N-2} (-1)^i \frac{(N-1)!}{(i+1)!(N-2-i)!} \]

\[ = (N-2)! \sum_{i=0}^{N-2} (-1)^i \frac{(N-1)!}{(i+1)!(N-2-i)!} \]

\[ = (N-2)! \sum_{i=1}^{N-1} (-1)^{i-1} \frac{(N-1)!}{i!} \]

\[ = -(N-2)! \sum_{i=1}^{N-1} (-1)^i \frac{(N-1)!}{i!} \]

\[ = -(N-2)! \left( \sum_{i=0}^{N-1} (-1)^i \binom{N-1}{i} - 1 \right) \]

\[ = (N-2)! , \]

where the last equality follows from Proposition 3.

Therefore, from Eq. 2.19, it follows that

\[ E(R) \geq p \cdot r \cdot \frac{N}{(N-1)!} (N-2)! = p \cdot r \cdot \frac{N}{N-1} \]

\[ = p \cdot r \left( 1 + \frac{1}{N-1} \right) . \] (2.20)

This completes the lower bound analysis for \( E(R) \).

To find an upper bound for \( E(R) \), we employ similar techniques. In the following equations,
a number above a relation symbol denotes the number of an equivalent equation for the lower bound:

\[
E(R) = \int_0^1 g(x)f_{X_N|r}(x) \, dx
\]

\[
\text{Eq. 2.12} \quad \leq -rN \left( \frac{r - 1}{N - 1} \right) \int_0^1 \frac{x^{N-1}(1-x)^{r-N}}{\ln(1-x)} \, dx \\
+ (N - 1) \int_0^1 f_{X_N|r}(x) \, dx
\]

\[
= -rN \left( \frac{r - 1}{N - 1} \right) \int_0^1 \frac{x^{-N}(1-x)^{N-1}}{\ln x} \, dx \\
+ (N - 1)
\]

\[
\text{Eq. 2.15} \quad = -rN \left( \frac{r - 1}{N - 1} \right) \sum_{i=0}^{N-1} (-1)^i \binom{N - 1}{i} \\
\cdot \ln(r - (N - 1) + i) + (N - 1)
\]

\[
\text{Eq. 2.18} \quad = \left( \frac{rN}{(N - 1)!} \sum_{i=0}^{N-2} (-1)^i \binom{N - 2}{i} \right) \\
\cdot \left[ \prod_{j=0}^{N-2} (r - (N - 1) + j) \right] \\
\cdot \ln \left( 1 + \frac{1}{r - (N - 1) + i} \right)^{r-(N-1)+i} \\
+ (N - 1).
\]

Denote \( q = \ln \left( 1 + \frac{1}{r-1} \right)^{-r-1} \). Then, \( 0 < q < 1 \), and for large enough values of \( r - 1 \), \( q \) is close to 1. Then,

\[
E(R) \leq \frac{q \cdot r \cdot N}{(N - 1)!} \left[ \sum_{i=0}^{N-2} (-1)^i \binom{N - 2}{i} \right] \\
\cdot \left[ \prod_{j=0}^{N-2} (r - (N - 1) + j) \right] + (N - 1)
\]

\[
\text{Eq. 2.20} \quad = \frac{q \cdot r \cdot N}{N - 1} + (N - 1)
\]

\[
= q \cdot r \left( 1 + \frac{1}{N - 1} \right) + (N - 1). \quad (2.21)
\]
This completes the upper bound analysis. From Eq. 2.20 and Eq. 2.21 we conclude that:

\[
\begin{align*}
    p \cdot r \left(1 + \frac{1}{N-1}\right) &\leq E(R) \leq q \cdot r \left(1 + \frac{1}{N-1}\right) + (N-1) \\
    p \left(1 + \frac{1}{N-1}\right) &\leq \frac{E(R)}{r} \leq q \left(1 + \frac{1}{N-1}\right) + \frac{N-1}{r} \\
    \frac{p}{N-1} - (1-p) &\leq \frac{E(R)-r}{r} \leq \frac{q}{N-1} - (1-q) + \frac{N-1}{r}.
\end{align*}
\]

(2.22)

Eq. 2.22 shows the upper and lower bounds on the estimation error as a function of the number of RPRTs \(N\) and the real number of affected nodes \(r\).

**Theorem 2** When \(1 \ll N \ll r\), the estimation error is approximately \(\frac{1}{N-1}\). Moreover, this error is positive, which means that our algorithm overestimates the number of affected nodes.

**Proof:** The proof follows from Eq. 2.22. When \(1 \ll N \ll r\), \(p\) and \(q\) are close to 1, and we have \(E(R) \approx r\left(1 + \frac{1}{N-1}\right)\).

In Figure 2.4 we show again the estimation error as obtained from simulations for 10,000 affected nodes. However, this time we compare this error to our analytic prediction \(f(N) = \frac{1}{N-1}\) and see excellent agreement between the two curves.

Using the above analysis, we now show how to eliminate the estimation error with no further
cost. Let \( r_{\text{real}} \) be the real number of affected nodes and \( r_{\text{estimate}} \) be the estimated number using our estimation algorithm. From the above analysis, we know that

\[
r_{\text{real}} \approx r_{\text{estimate}} \cdot \frac{N - 1}{N}.
\]

(2.23)

Figure 2.5 shows simulation results of this process for 1,000 and 10,000 affected nodes before and after applying “error cancellation”. It is evident that after applying error cancellation our algorithm reduces the error to approximately 0 even when the number of RPRTs is very small.

For a constant \( a \) and a random variable \( X \), \( E(aX) = aE(X) \) and \( \text{var}(aX) = a^2 \text{var}(X) \). Thus, \( \text{std}(aX) = a \text{std}(X) \) and the coefficient of variation of \( aX \) is \( \frac{\text{std}(aX)}{E(aX)} = \frac{\text{std}(X)}{E(X)} \), which is equal to the coefficient of variation of \( X \). Since the effect of error cancellation is multiplication of \( r_{\text{estimate}} \) by a constant \( \frac{N - 1}{N} \) (Eq. 2.23), the coefficient of variation is not affected. In Figure 2.6 we depict the estimation error and coefficient of variation after error cancellation for 1,000 and 10,000 affected nodes. We can see that, as before error cancellation, the coefficient of variation is negligible for large values of \( N \) and is small enough for practical small values of \( N \).

2.5 The Effect of RPRT Loss

So far we have assumed that RPRTs are not lost. We have also presented, in the Implementation Notes of Section 2.3, a simple mechanism for the retransmission of lost RPRTs by their senders. In this section we study the effect of RPRT losses on NATO! and show why such a retransmission
mechanism is indeed necessary.

Suppose that by the time the gateway receives the \( N \)’th RPRT, \( S \) RPRTs have been lost. Therefore, the RPRT considered by the gateway to be the \( N \)’th is actually the \( N + S \)’th. This influences Eq. 2.9, which the gateway solves to find \( r \). This equation now should read:

\[
\frac{1}{r} + \frac{1}{r-1} + \ldots + \frac{1}{r-(N-1)} = -\ln(1 - x_{N+S}).
\]  

(2.24)

Going through the analysis in Section 2.4 and replacing occurrences of \( N \) with \( N + S \) when \( N \) indicates the index of the \( N \)’th RPRT (as opposed to places where it indicates the number of RPRTs), we get:

\[
-p \frac{S-1}{N+S-1} - (1-p) \leq \frac{E(R) - r}{r} \leq -q \frac{S-1}{N+S-1} - (1-q) + \frac{N-1}{r}.
\]  

(2.25)

In this equation, \( p = \ln \left(1 + \frac{1}{r-(N+S-1)} \right)^{r-(N+S-1)} \) and \( q = \ln \left(1 + \frac{1}{r-1} \right)^{r-1} \). This time, when \( r \gg N \) and \( N + S \gg 1 \), the estimation error is approximately

\[
\frac{E(R) - r}{r} \approx -\frac{S-1}{N+S-1}.
\]  

(2.26)

Figure 2.7 shows the estimation error in the presence of RPRT loss. We consider 1 or 5 RPRT
losses, and show the error using both simulation and Eq. 2.26. A few conclusions can be drawn from this figure. First, as in Figure 2.4, we can see very good agreement between the simulation and the analysis. Second, comparing Figure 2.7(a) to Figure 2.7(b) reveals no major difference. This implies that, as before, the error is independent of \( r \). Third, as expected (see Eq. 2.26), the error now is negative. Thus, in the presence of RPRT loss, our method underestimates the number of affected nodes instead of overestimating it. However, the most important conclusion we draw from the figure is that when RPRTs are subject to loss, NATO! should be enhanced with a mechanism for RPRT retransmissions, as proposed in the Implementation Notes.

An interesting observation is that when \( S = 1 \), the error almost vanishes. This is also implied by Eq. 2.26, which predicts a 0 error in this situation. This fact has a rather simple explanation. As noted above, RPRT losses cause our method to underestimate the error, while with no RPRT losses the error is overestimated. When \( S = 1 \), the two counter-forces cancel each other, resulting in an error very close to 0.

We can apply the same process used in the end of Section 2.4 to reduce the effect of RPRT loss on the precision of our algorithm. In this case, Eq. 2.23 is replaced by the following equation:

\[
    r_{\text{real}} \approx r_{\text{estimate}} \cdot \frac{N + S - 1}{N}. \tag{2.27}
\]

Figure 2.8 presents the error for \( S = 10 \) lost RPRTs before and after applying the error cancellation process. Note, however, that in order to implement this process when \( S > 0 \), the gateway needs to know the value of \( S \), which is not always possible.
Figure 2.8: Estimation error vs. $N$ for $S = 10$, before and after “error cancellation”
Chapter 3

H-NATO! – A Scalable Scheme for Preventing Feedback Implosion in a Large-Scale Multi-Tier Sensor Network

3.1 Introduction

Consider a 4-tier, large-scale sensor network. The bottom tier of such a network consists of millions of sensors deployed throughout a country or a continent and divided into areas. WiFi technology is used to give each sensor direct wireless connectivity with an area gateway. There are thousands of such gateways, divided into regions that are controlled by regional gateways. The area gateways can communicate with their regional gateways using WiMax. Finally, the regional gateways are connected to a centralized root gateway by means of a satellite channel.

This architecture has several advantages compared to a flat network with homogeneous devices [28], the most important of which are as follows:

- The sensors do not need to participate in a routing protocol, which would require them to expend a lot of energy because of the network size [49].

- The routing of a message from a sensor to the root gateway is fast, reliable, and simple. In the 4-tier model considered above, the messages go through three broadcast domains: from a sensor to an area gateway, then to a regional gateway, and finally to the root gateway. If the various gateways are connected to a permanent power supply, they do not need to switch their communication module on and off. Hence, it takes less than a second for a message to
reach the root gateway [58].

- This is probably the only architecture that can be used for a huge sensor network consisting of millions of devices and covering a gigantic area [6], without being restricted by topography, weather, and other constraints.

In this chapter, we evaluate the use of the NATO! scheme of Chapter 2 in large-scale hierarchical sensor networks in order to inform the root gateway of the number of sensors experiencing a given event without requiring each of them to send its own notification message. We conclude that NATO! has a high communication cost when implemented in a hierarchical network, and then we describe the H-NATO! scheme that has a much lower communication cost.

The rest of this chapter is organized as follows. In Section 3.2 we present related work. In Section 3.3 we describe the existing NATO! scheme and extend it for a hierarchical sensor network. In Section 3.4 we present a new scheme, analyze it, and show that it can reduce the communication cost by a factor of 10. In Section 3.5 we present simulation results.

3.2 Related Work

In this section we present works related to the considered sensor network architecture and works related to real-time estimation schemes.

Many papers develop and study routing problems for hierarchical sensor networks. One of the two most common approaches is to designate some of the sensors as cluster heads, through which messages from the rest of the sensors will be routed to a central gateway. LEACH [22] selects these cluster heads randomly and periodically rotates this role. PEGASIS [31] builds paths from the sensors to the central gateway, as routing is done in flat sensor networks [23].

The multi-tier approach considered in this chapter is different, and is more suitable for very large scale sensor networks. Here we assume that strong gateways, which are connected to a continuous power source, are located in advance in the same way that cellular base-stations are located in a cellular network. In [56], the authors provide a time synchronization algorithm for such networks. In [27], the authors employ a hierarchical sensor network for a video over IP application. In this application, home sensors detect user presence and send report messages via intermediate access points to an area head end. The latter is in charge of starting, halting, and prioritizing streaming information originating at every camera.

Habitat monitoring can also take advantage of hierarchical sensor networks. In [34], the authors spread sensors on Great Duck Island that measure environmental characteristics such as tem-
perature, pressure and humidity. The authors of [46] extend this idea and provide fault tolerance and distributed storage in their sensor network.

In [54] the authors address the problem of gateway positioning in a hierarchical sensor network in order to increase sensor lifetime. A 3-tier hierarchical sensor network built using off-the-shelf hardware and software is presented in [11]. A model for hierarchical in-network aggregation in sensor networks is presented in [15].

A common criterion for a sensor to send its measurement is the fulfillment of a certain condition or event regarding the measured attribute. For example, sensors measuring temperature in some geographical region may send messages to a central gateway once the temperature exceeds a given threshold. These messages may be synchronous or asynchronous. In the asynchronous model [35, 36], whenever a certain condition or event takes place, a sensor sends a message. In the synchronous model [57], the central gateway or cluster head periodically polls the sensors, and those experiencing the event send a reply.

While receiving messages from all the sensors experiencing a given event may be useful in certain circumstances, it may result in feedback implosion if a large group of sensors send their messages simultaneously. One solution to this problem is data aggregation and directed diffusion [26], in which sensors or intermediate gateways merge received messages and send the combined message to the central gateway. Our scheme addresses this problem in a different way, by asking the reporting sensors to wait a random time before sending their messages, and by allowing only the first sensors to send their reports. Using an estimation scheme like the one proposed in Chapter 2 for a flat wireless network, a gateway can estimate the actual number of sensors experiencing the considered event in its region. We elaborate on this scheme in Section 3.3.

Other estimation schemes were optimized for different application scenarios. In [7, 8] the authors estimate the number of receivers in a multicast group. To avoid feedback implosion, not all the receivers send a message to the sender. Rather, each one sends a message with a predefined probability $p$. The sender uses $p$ and other parameters to estimate the number of receivers.

A timer based scheme for estimating the number of hosts contending for access to a shared ALOHA channel is given in [29]. The scheme is derived from the probability $p_f$, of a successful access during the first transmission given a successful delivery of the packet. To this end, each contending host indicates in every transmitted packet whether this packet is being transmitted for the first time, or whether it has already experienced a collision.
3.3 A Hierarchical Implementation of NATO!

In Chapter 2, we propose the NATO! scheme for estimating the number of affected nodes in a 1-tier (flat) network. In this section we summarize this scheme and then show how it can be extended for a multi-tier network.

The scheme is optimized for a flat network where a gateway is able to broadcast a message to all the nodes, and the delays between the nodes and the gateway are equal. According to this scheme, the gateway broadcasts a START($t_0$) message indicating that after time $t_0$ it wishes to receive RPRTs from a subset of the affected nodes. The affected nodes are those that experience the considered event. Every affected node chooses a random timer in the range $[t_0, t_0 + T]$, using a known probability distribution function $F$, and if when the timer expires, the other affected nodes have sent fewer than $N$ RPRTs to the gateway, the considered node should send a RPRT.

As a result of this scheme, the gateway receives $N$ RPRTs during $[t_0, t_0 + T]$. It then executes Algorithm 1 of Chapter 2 followed by an error cancellation step as defined in Eq. 2.23. It is shown in Chapter 2 that in a flat network (one broadcast domain) with thousands of affected nodes this algorithm yields an estimation error of 1%.

We now modify this scheme to fit a hierarchical sensor network. We consider a logical tree structure with $L$ levels of gateways. The root is referred to as a Level-$L$ gateway and it has direct wireless connectivity with multiple Level-($L - 1$) gateways. The latter have direct wireless connectivity with Level-($L - 2$) gateways, and so on. At the bottom level, each Level-1 gateway has direct wireless connectivity with the sensors. Figure 3.1 shows an example for $L = 2$. Some of the sensors experience an event. Recall that these sensors are also referred to as “affected nodes.”

Scheme 1, presented below, implements NATO! hierarchically. The main idea behind this scheme is that a Level-$i$ gateway relays towards the root the first $N$ RPRTs it receives from its Level-($i - 1$) gateways.

**Scheme 1 (using NATO! hierarchically)**

A) *The root gateway broadcasts a START($t_0$) message, where $t_0$ is the time after which RPRTs messages can be sent from the affected nodes to the Level-1 gateways.*

B) *At time $t_0$, each affected node $v$ performs the following steps:*

   B1) *It chooses a random time $t$ in the range $[t_0, t_0 + T]$, using a known probability distribution function $F$.***
B2) If the number of RPRTs sent before $t$ by the other affected nodes to the gateway of $v$ is smaller than $N$, $v$ sends a RPRT to this gateway. Otherwise, it does not send a RPRT.

C) For every level $i$: By time $t_0 + i \cdot T$, every Level-$i$ gateway has the $N$ times $x_1, \ldots, x_N$ associated with the first $N$ RPRTs it has received from its downstream gateways (if $i > 1$) or affected nodes (if $i = 1$). For every $j$ from 1 to $N$, if the number of RPRTs sent to its parent Level-($i+1$) gateway before $t_0 + i \cdot T + x_j$ is smaller than $N$, the considered Level-$i$ gateway should send to its parent gateway a RPRT at time $t_0 + i \cdot T + x_j$.

In Scheme 1, by time $t_0 + nT$, the root gateway knows the transmission time of the first $N$ RPRTs sent by the affected nodes and it can run Algorithm 1 to estimate the number of affected nodes.

Scheme 1 requires each Level-$i$ gateway to receive $N$ RPRTs. This is in addition to 2 control messages, START and STOP, sent by each gateway. Thus, the communication cost of the scheme is equal to $|G|(N + 2)$, where $|G|$ is the total number of gateways. Because we do not make any assumption regarding the distribution of affected nodes in the network, any estimation scheme must involve all the gateways. Moreover, in every scheme every gateway must receive a message when the scheme is initiated by the root and send a message after it acquires data. Hence, we believe that the minimum number of messages requires by any scheme is $3|G| + C$, where $C$ is a constant that does not depend on the network structure. In the next section we present a new scheme that achieves this lower bound.
3.4 The H-NATO! Scheme and Its Analysis

3.4.1 The New Scheme

In this section we propose an alternative scheme that significantly reduces the communication cost without affecting the precision. The main idea behind this scheme is as follows. The root gateway needs to get the $N$ smallest reporting times of the network nodes. However, this requirement should not necessarily be translated into the requirement that every Level-\textit{i} gateway will get the $N$ smallest reporting times of affected nodes in its subtree. If the number of Level-\textit{i} gateways is much larger than $N$, there is a very small probability that several RPRTs of affected nodes from the subtree of the same Level-\textit{i} gateway will fall into the list of the $N$ smallest RPRT transmission times to be received by the Level-\textit{(i + 1)} gateway. Therefore, it might be sufficient for every Level-\textit{i} gateway, where $1 \leq i \leq L - 1$, to get only the $N_1 < N$ smallest reporting times in its subtree. In such a case, the communication cost of the scheme is $N + \sum_{i=1}^{L-1} N_i \cdot B_i$, where $B_i$ is the number of Level-\textit{i} gateways.

This idea is translated into the following scheme:

\textbf{Scheme 2 (H-NATO!)}

A) The root gateway broadcasts a \textit{START}(\textit{t}_0) message, where \textit{t}_0 is the time after which RPRT messages can be sent from the affected nodes to their Level-1 gateways.

B) At time \textit{t}_0, each affected node \textit{v} performs the following steps:

B1) It chooses a random time \textit{t} in the range $[\textit{t}_0, \textit{t}_0 + \textit{T}]$, using a known probability distribution function \textit{F}.

B2) If the number of RPRTs sent before \textit{t} to the Level-1 gateway of \textit{v} is smaller than \textit{N}_1, then \textit{v} should send a RPRT to this gateway. Otherwise, it should not send a RPRT.

C) For every level $i \leq L - 1$: By \textit{t}_0 + \textit{i} \cdot \textit{T}, every Level-$i$ gateway \textit{g} has received the \textit{N}_i times $x_1, \ldots, x_{N_1}$ associated with the \textit{N}_i first RPRTs it has received from its downstream gateways (if $i > 1$) or affected nodes (if $i = 1$). For every $1 \leq j \leq \textit{N}_i$, if the number of RPRTs sent to the parent Level-$(i + 1)$ gateway of \textit{g} before \textit{t}_0 + \textit{i} \cdot \textit{T} + x_j$ is smaller than \textit{N}_{i+1}, \textit{g} sends to its parent Level-$(i + 1)$ gateway a RPRT at \textit{t}_0 + \textit{i} \cdot \textit{T} + x_j.

D) The root gateway uses the \textit{N} RPRTs it receives as input to Algorithm 4, to be described later, in order to estimate the number of affected nodes.
Implementation Notes:

1. In both Scheme 1 and Scheme 2, it was assumed that all gateways and sensors are synchronized to a common clock or that all gateways and sensors of the same level receive the START message at the same time. Such an assumption can be easily realized in the considered wireless networks (WiFi, WiMax, satellite, etc.). However, H-NATO! can also be executed in a system where there is no common clock and there is a variable delay between a gateway and its downstream gateways or sensors, provided that each gateway knows the delay $D_j$ to/from every downstream gateway or sensor $j$, and that $j$ knows the delay to its parent gateway. In such a system the following adjustment should be used. Let $D_j$ be the delay from a Level-$i$ gateway to node $j$, and let $D = \max_j(D_j)$ be the maximum delay. The Level-$i$ gateway adds the value of $D$ to the START message it broadcasts, and each node $j$ should wait a time period of $D - D_j$ before running Scheme 2.

2. RPRT messages are subject to loss due to collisions on a contention channel or to transmission errors. To address this problem, we enhance Scheme 2 with the following reliability mechanism. Upon receiving a RPRT, a Level-$i$ gateway sends a confirmation to the sender. A RPRT sender that does not receive a confirmation within a time-out period resends its RPRT, and specifies the offset between its current local time and the time of the original RPRT.

Figure 3.2 depicts the new scheme for 2 levels, when $N_1 = 1$. In what follows we concentrate...
on the case where \( N_i = 1 \) for \( 1 \leq i \leq L-1 \). In such a case this scheme reduces the communication cost of Scheme 1 by a factor of \( N \). Moreover, the communication cost for the new scheme with \( N_i = 1 \) for \( 1 \leq i \leq L-1 \) is probably the best one may achieve, because in such a case every gateway is required to send approximately two messages, i.e., a START and a STOP.

We are now looking for an estimation algorithm to be used by the root gateway in step (D) of Scheme 2. With high probability, even if \( N_i = 1 \) for every \( i \leq L-1 \), the root receives the \( N \) almost smallest RPRT times sent by the affected nodes. This may suggest that Algorithm 1 of can be used in this case as well. In Figure 3.3 we see the estimation error as a function of \( N \) when running Scheme 2 with Algorithm 1 while \( N_i = 1 \) for every \( i \leq L-1 \). The estimation error is defined by \( \frac{\text{estimated} - \text{real}}{\text{real}} \), where \( \text{estimated} \) is the estimated number of affected nodes, and \( \text{real} \) is the real number. For this graph we simulated a network with two levels of gateways: 1 root and 100 Level-1 gateways. Each Level-1 gateway has a random number of affected nodes, uniformly distributed in the range \([1,100]\). We can see that the error is significant, and that it does not converge to 0 when \( N \) increases. There are two reasons for this error. First, each Level-1 gateway may have a different number of affected nodes. Thus, the transmission times of the RPRTs sent by these gateways to the root are not from the same distributions. Algorithm 1, however, requires that all affected nodes use the same (known) distribution when deciding the transmission time of their RPRT message. Second, the \( N \) RPRTs received by the root are not guaranteed to be those with the earliest transmission time. For example, if an affected node of the first Level-1 gateway sends the first RPRT and another affected node of the same gateway sends the next RPRT, this second RPRT is not forwarded by the Level-1 gateway to the root. Consequently, the input to the algorithm is not accurate.
3.4.2 A New Estimation Algorithm

We now develop a new estimation algorithm to be executed by the root in Scheme 2. We assume that the root knows the number $B$ of Level-$(L - 1)$ gateways, the number $N$ of RPRTs it receives, and the times these RPRTs were sent by the affected nodes. The algorithm requires that each Level-$(L - 1)$ gateway send at most one RPRT to the root, containing the smallest RPRT time sent by all affected nodes in its subtree. This means that for every $1 \leq i \leq L - 1$, $N_i = 1$. Recall that with these values, the communication cost of Scheme 2 is minimized.

Let $F$ be a uniform distribution function over the interval $[0, 1]$. Thus, for every $x \in [0, 1]$, $F(x) = x$ and $f(x) = 1$. Let $B$ be the number of Level-$(L - 1)$ gateways. Let random variable $X_i$ be the time a Level-$(L - 1)$ gateway $i$ receives a RPRT from its descendants, where $i = 1, \ldots, B$. Let $Y_1, \ldots, Y_B$ be the sequence $X_1, \ldots, X_B$ sorted in ascending order. Thus, the times during which the root gateway receives the $N$ RPRTs are $Y_1, \ldots, Y_N$. Let $m_i$ be the total number of affected nodes in the subtree of gateway $i$. Since $X_i$ is the minimum of $m_i$ independent uniform random variables, we get:

$$P(X_i > x) = P(U > x)^{m_i} = (1 - x)^{m_i},$$

where $U$ is a uniform random variable on $[0, 1]$.

Therefore,

$$F_{X_i}(x) = P(X_i \leq x) = 1 - (1 - x)^{m_i}$$

and

$$f_{X_i}(x) = \frac{d}{dx} F_{X_i}(x) = m_i \cdot (1 - x)^{m_i-1}.$$
The joint density of $Y_1, \ldots, Y_N$ is:

$$f_{Y_1,\ldots,Y_N}(y_1, \ldots, y_N)dy_1 \cdots dy_N$$

$$= P(\text{one } X_i \text{ is in } (y_1, y_1 + dy_1), \ldots, \text{one } X_i \text{ is in } (y_N, y_N + dy_N),$$

all other $X_i$'s are in $(y_N, 1))$

$$= \sum_{(i_1, \ldots, i_N) \in I} P(X_{i_1} \in (y_1, y_1 + dy_1)) \cdot \ldots \cdot$$

$\cdot P(X_{i_N} \in (y_N, y_N + dy_N))$

$\cdot P((B - N) \text{ other } X_i \text{'s are in } (y_N, 1))$

$$= \sum_{(i_1, \ldots, i_N) \in I} m_{i_1} (1 - y_1)^{m_{i_1} - 1}dy_1 \cdot \ldots \cdot$$

$\cdot m_{i_N} (1 - y_N)^{m_{i_N} - 1}dy_N$

$\cdot (1 - y_N)^{\sum_{j \in \{1, \ldots, B\} \setminus \{i_1, \ldots, i_N\}} m_j},$

where

$$I = \{(i_1, \ldots, i_N) | \forall j: 1 \leq i_j \leq B \land$$

$\forall j, k: i_j \neq i_k\}$$

is the set of $N$-tuples of pairwise different indexes in the range $1, \ldots, B$.

Therefore,

$$f_{Y_1,\ldots,Y_N}(y_1, \ldots, y_N)$$

$$= \sum_{(i_1, \ldots, i_N) \in I} m_{i_1} (1 - y_1)^{m_{i_1} - 1} \cdot \ldots \cdot$$

$\cdot m_{i_N} (1 - y_N)^{m_{i_N} - 1}$

$\cdot (1 - y_N)^{\sum_{j \in \{1, \ldots, B\} \setminus \{i_1, \ldots, i_N\}} m_j},$

We now use the maximum likelihood method to find the values of $m_1, \ldots, m_N$ that maximize $f_{Y_1,\ldots,Y_N}(y_1, \ldots, y_N)$, where $y_1, \ldots, y_N$ are the times when the $N$ RPRTs are received by the root. First, note that $f_{Y_1,\ldots,Y_N}(y_1, \ldots, y_N)$ is a symmetric function of $m_1, \ldots, m_N$. In other words, any permutation of $m_1, \ldots, m_N$ yields the same value of $f$. Therefore, the maximum of $f$ is achieved for $m_1 = \ldots = m_N$. The intersection of $f$ with the plane defined by $m_1 = \ldots = m_N$ in the


\[(N + 1)\text{-dimensional space is a function}
\]

\[g(m) = \sum_{(i_1, \ldots, i_N) \in I} m(1 - y_1)^{m-1} \cdot \ldots \cdot m(1 - y_N)^{m-1} \cdot (1 - y_N)^{\sum_{j \in \{1, \ldots, B\} - \{i_1, \ldots, i_N\} \cdot m},}
\]

whose maximum coincides with the maximum of \(f\). Thus, we should find the value of \(m\) that maximizes \(g(m)\). To this end, we now rewrite \(g(m)\) in the following way:

\[
g(m) = \sum_{(i_1, \ldots, i_N) \in I} m(1 - y_1)^{m-1} \cdot \ldots \cdot m(1 - y_N)^{m-1} \cdot (1 - y_N)^{(B-N)m}
= |I| \cdot m(1 - y_1)^{m-1} \cdot \ldots \cdot m(1 - y_N)^{m-1} \cdot (1 - y_N)^{(B-N)m}
= \frac{B!}{(B-N)!} \cdot m^N \cdot [(1 - y_1) \ldots (1 - y_N)]^{m-1} \cdot (1 - y_N)^{(B-N)m}.
\]

Since \(\ln(\cdot)\) is a monotonically increasing function, \(g(m)\) gets its maximum at the same \(m\) as \(\ln g(m)\).

\[
\ln g(m) = \ln \frac{B!}{(B-N)!} + N \ln m + (m - 1) \ln[(1 - y_1) \ldots (1 - y_N)]
+(B - N)m \ln(1 - y_N).
\]

We now compute the derivative:

\[
(ln g(m))' = \frac{N}{m} + \ln[(1 - y_1) \ldots (1 - y_N)]
+(B - N) \ln(1 - y_N).
\]
Equating \((\ln g(m))'\) to 0 and solving for \(m\) yields \(m\) equal to
\[
N \div \ln[(1 - y_1) \ldots (1 - y_N)] + (B - N) \ln(1 - y_N). \tag{3.1}
\]

Since \((\ln g(m))'' = -\frac{N}{m^2} < 0\) holds for every \(m\), the above value is indeed a maximum, and the total number \(r\) of affected nodes estimated by the root is
\[
N \cdot B \div \ln[(1 - y_1) \ldots (1 - y_N)] + (B - N) \ln(1 - y_N). \tag{3.2}
\]

### 3.4.3 The Error of Eq. 3.2

We will now find the relative error of Eq. 3.2. Let random variable \(R\) be the result of substituting \(Y_i\) for \(y_i\) in Eq. 3.2, and let random variable \(M\) be the result of the same substitution in Eq. 3.1. Thus, the estimation error is
\[
\frac{E(R) - r}{r} = \frac{B \cdot E(M) - Bm}{Bm} = \frac{E(M) - m}{m},
\]
where
\[
E(M) = \int_{0<y_1<\ldots<y_N<1} -dy_1 \ldots dy_N \div N \cdot \ln[(1 - y_1) \ldots (1 - y_N)] + (B - N) \ln(1 - y_N)
\]
\[
\cdot \frac{B!}{(B - N)!} \cdot m^N \cdot [(1 - y_1) \ldots (1 - y_N)]^{m-1}
\]
\[
\cdot (1 - y_N)^{(B-N)m}.
\]

Substituting \(x_i = 1 - y_i\) yields:
\[
E(M) = -m^N N \div \frac{B!}{(B - N)!} \int_{0<x_N<\ldots<x_1<1} \left(\prod_{i=1}^{N} x_i\right)^{m-1} \cdot x_N^{(B-N)m} \div \ln[\prod_{i=1}^{N-1} x_i \cdot x_N^{B-N+1}] \cdot dx_1 \ldots dx_N
\]
\[
= -m^N N \div \frac{B!}{(B - N)!} \int_{0<x_N<\ldots<x_1<1} \left(\prod_{i=1}^{N} x_i \cdot x_N^{B-N}\right)^{m-1} \cdot x_N^{B-N} \div \ln[\prod_{i=1}^{N-1} x_i \cdot x_N^{B-N+1}] \cdot dx_1 \ldots dx_N.
\]
Define $t_1 = \prod_{i=1}^{N} x_i \cdot x_N^{B-N} = \prod_{i=1}^{N-1} x_i \cdot x_N^{B-N+1}$, $t_2 = x_2, \ldots, t_N = x_N$. Then, $x_1 = \frac{t_1}{\prod_{i=2}^{N} t_i \cdot t_N^{B-N}}$. The Jacobian matrix $\frac{\partial (x_1, \ldots, x_N)}{\partial (t_1, \ldots, t_N)}$ is an upper triangular matrix with the values $\frac{\partial x_1}{\partial t_1}, \ldots, \frac{\partial x_N}{\partial t_N}$ on its diagonal. For $i \geq 2$, $\frac{\partial x_i}{\partial t_i} = 1$ and $\frac{\partial x_1}{\partial t_1} = \frac{1}{\prod_{i=2}^{N} t_i \cdot t_N^{B-N}}$. Therefore, the determinant of this matrix (the Jacobian) is $J = \frac{1}{\prod_{i=2}^{N} t_i \cdot t_N^{B-N}}$.

Thus,

$$E(M) = -m^N N \frac{B!}{(B - N)!} \cdot \int_D \frac{t_1^{m-1}}{\ln t_1 \cdot \prod_{i=2}^{N} t_i} dt_1 \ldots dt_N,$$

where $D$ is a domain in $\mathbb{R}^N$ defined by $D = \{ (t_1, \ldots, t_N) | 0 < t_N < \ldots < t_2 < \prod_{i=2}^{N-1} t_i \cdot t_N^{B-N+1} < 1 \}$.

We solve this integral by iterated integration with the following bounds.

If $N = 2$,

$$0 < t_1 < 1$$

$$B^{-\sqrt{t_1}} < t_2 < \sqrt{t_1}.$$ 

If $N \geq 3$,

$$0 < t_1 < 1$$

$$B^{-\sqrt{t_1}} < t_2 < 1$$

$$B^{-\sqrt{\frac{t_1}{t_2}}} < t_3 < t_2$$

$$\cdots$$

$$B^{-\sqrt{\frac{t_1}{t_2 \cdots t_{N-2}}}} < t_{N-1} < t_{N-2}$$

$$B^{-\sqrt{\frac{t_1}{t_2 t_3 \cdots t_{N-1}}}} < t_N < \sqrt{\frac{t_1}{t_2 t_3 \cdots t_{N-1}}}.$$
We can show analytically that for every practical value of $N$ (e.g., $N \leq 15$):

$$\int_D \frac{t_1^{m-1}}{\ln t_1 \cdot \prod_{i=2}^N t_i} \, dt_1 \ldots dt_N =$$

$$= \begin{cases} 
- \frac{1}{mB(B-1)} & N = 2 \\
- \frac{1}{(N-1)m^{N-1} \cdot (B-1)^{(N-2)} \cdot (B-N+1)} & N \geq 3 
\end{cases}$$

Therefore,

$$E(M) = \begin{cases} 
2m & N = 2 \\
m \frac{N}{N-1} \frac{B}{B-1} & N \geq 3
\end{cases}$$

Thus, the estimation error is

$$\frac{E(M)}{m} - 1 = \begin{cases} 
\frac{1}{N-1} \frac{B}{B-1} - 1 & N = 2 \\
\frac{N}{N-1} \frac{B}{B-1} - 1 & N \geq 3
\end{cases}$$

(3.3)

When $B$ is large, $\frac{B}{B-1} = 1 + \frac{1}{B-1} \approx 1$. In this case the error is $\frac{N}{N-1} - 1 = \frac{1}{N-1}$.

In Figure 3.4 we present the results for implementing Scheme 2 with Eq. 3.2 in a 2-level network with $B = 200$ Level-1 gateways. Each Level-1 gateway has a random number $m$ of affected nodes, where $m$ is uniformly distributed in the range $[1, 1000]$. The graph shows two curves: the upper one shows the actual estimation error while the lower one shows the error found by our analysis (Eq. 3.3). The $x$-axis is the number $N$ of RPRTs received by the root and the $y$-axis is the estimation error. We see an excellent agreement of the curves for $N \geq 3$.

Since the error is always positive, we can significantly improve the precision of our estimation algorithm by multiplying the value found by Eq. 3.2 by $\frac{N-1}{N}$. To summarize, the algorithm to be
employed by the root in Scheme 2 is as follows:

**Algorithm 4**

1. Receive \( N \) RPRTs from the Level-\((L - 1)\) gateways at times \( y_1 < \ldots < y_N \).

2. Calculate the number of affected nodes by Eq. 3.2 and multiply the result by \( \frac{N-1}{N} \).

### 3.5 Simulation Study

We simulated a 2-gateway-level network with \( B = 200 \) Level-1 gateways, each with a random number \( m \) of affected nodes, uniformly distributed in the range \([1, 1000]\). We implemented Scheme 2 with Algorithm 4 (i.e., Eq. 3.2 with error cancellation) and calculated the estimation error given by \( \frac{r_{\text{real}} - r_{\text{estimated}}}{r_{\text{real}}} \).

Figure 3.5 depicts two curves. The top one is the estimation error before applying error cancellation (i.e., using Eq. 3.2), and the bottom one is the actual error of Algorithm 4 after multiplying Eq. 3.2 by \( \frac{N-1}{N} \). As we can see, the error of Algorithm 4 is very close to 0 for \( N \geq 5 \), whereas the curve of Eq. 3.2 tends to 0 only for much greater values of \( N \).

Figure 3.6 compares the estimation error for \( B = 100 \) and for \( B = 500 \) Level-1 gateways as a function of \( N \). We see that for both cases, the error is within the 2-3% range, and that \( B \) has no effect on the error, as predicted by Eq. 3.3.

One of the most important properties of the proposed scheme is that it works very well even if the reporting nodes are not evenly distributed in the network. It is not difficult to see that with a non-uniform distribution it is impossible to estimate the number of affected nodes without having
Figure 3.6: Estimation error of Algorithm 4 vs. $N$ for $B = 100$ and 500 Level-1 gateways

Figure 3.7: Estimation error vs. $N$ for $B = 200$, with a uniform and non-uniform distribution of affected nodes per Level-1 gateway

every gateway involved. Thus, our estimation scheme, which requires every gateway to send only two messages, it probably the most efficient scheme that can be designed. In Figure 3.7 we compare the estimation error for $B = 200$ Level-1 gateways and two different node distributions. The top curve (“Uniform”) depicts the case when each gateway has a uniformly distributed random number of affected nodes in the range $[1, 1000]$, as in the previous graphs. The bottom curve (“Non uniform”) depicts the case when each Level-1 gateway $i$ has $i \cdot 20$ affected nodes. The figure shows that in both cases the error is very small, i.e., less than 2%.

Finally, Figure 3.8 depicts the estimation error as a function of the number of gateway levels in the hierarchy, with 6,250,000 affected nodes. Each Level-1 gateway has 50 affected nodes. For the 2-level case we use $6,250,000/50=125,000$ Level-1 gateways. For the 3-level case we use 125,000 Level-1 gateways and $6,250,000/(50)^2 = 2,500$ Level-2 gateways. For the 4-level case we use
125,000 Level-1 gateways, 2,500 Level-2 gateways and 50 Level-3 gateways. For all cases, each non-root Level-\(i\) gateway has 50 Level-\((i + 1)\) gateways or affected nodes. The top curve shows the estimation error for \(N = 5\) and the bottom for \(N = 10\).

Several conclusions can be drawn from the figure. While \(N = 10\) leads to a lower estimation error, the improvement is not dramatic, whereas the cost is twice the cost of \(N = 5\). Second, while the error increases with the number of levels, it is reasonable (\(< 7\%\)) for any practical number of levels. The error and the number of levels correlate because it is more likely that two or more RPRTs need to be forwarded by the same Level-\((L - 1)\) gateway when the number of levels increases, which is not possible with Scheme 2.
Chapter 4

Using NATO! for Partially Reliable Large Scale Multicast Streaming in Broadcast Wireless Networks

4.1 Introduction

The IETF RMT (Reliable Multicast Transport) Working Group has been working for the last 10 years on many issues related to large-scale reliable multicast. This WG has defined several important solutions (see [1] for a good overview), while adopting the most important research results published on this topic. Therefore, we believe that the best way to present our proposed scheme is in the context of the framework developed by the RMT WG.

The protocol developed by the RMT WG for large-scale reliable multicast streaming is called NORM (NACK oriented reliable multicast) [3]. To achieve some level of reliability, NORM uses Transport layer FEC (Forward Error Correction). In a typical FEC-based reliable multicast, the sender creates from each data block $K+n$ packets, and every receiver must receive any $K$ of these packets in order to decode the data block. NORM allows FEC to be combined with the Transport layer ARQ (Automatic Repeat reQuest), thereby defining a Transport layer hybrid FEC/ARQ scheme [3, 20, 30, 41, 45]. With such a scheme, receivers that have not received enough ($K$) packets correctly notify the sender by sending NACK messages. The sender then decides whether to send additional repair packets, and how many such packets to send. The number of such repair rounds might be limited because of real-time, buffer space, or similar considerations.

In this chapter we propose to use NATO! in the framework of NORM in order to improve the
performance of large-scale partially reliable multicast streaming in broadcast wireless networks. With NORM, the sender can only infer the maximum number of lost packets experienced by the receivers. This information is not sufficient for the selection of an optimal transmission strategy because it does not include the number of receivers experiencing this loss. Our scheme executes several instances of NATO! in parallel. Each such instance queries receivers that lost a certain number of packets. Given this loss information, the sender builds a histogram containing the number of receivers versus the number of lost packets, and uses this histogram to choose the optimal number of packets to transmit in order to satisfy each of the defined optimization criteria.

The rest of this chapter is organized as follows. In Section 4.2 we discuss related work. In Section 4.3 we summarize the main concepts of the NATO! scheme and then present two protocols for reliable multicast based on this scheme. In Section 4.4 we discuss the performance of the proposed protocols, and compare it to the performance of reactive 2-round protocols that do not use the NATO! scheme. In Section 4.5 we discuss the implementation of NATO! in a wireless broadband access network, where report messages sent by the receivers to the base station are subject to arbitrary loss and delay.

### 4.2 Related Work

The main idea in this chapter is to allow the base station to understand the loss distribution of the multicast group in order to improve reliable multicast performance. We are not aware of papers that have addressed a similar problem. However, several papers have addressed a closely related problem: estimating the total number of receivers in a multicast group [7, 8, 33, 40]. Our problem differs not only in that we need to find a full distribution rather than a single estimate, but also in that we do not rely on the correlation between successive measurements to estimate the size of the multicast group. In the aforementioned papers, this correlation is used to reduce the cost of the estimation process. Node mobility makes this useless in our case. Moreover, some of these papers solve the opposite problem, i.e., they assume a known number of receivers in the group and find the best timer distribution to ensure good feedback suppression and avoid implosion.

In Section 2.2.2, a survey of several estimation approaches for the number of receivers in a multicast group is presented. All these approaches limit the number of receivers that send a NACK to the sender, mainly by asking a receiver to send its NACK with a certain probability and not to send it with the complement probability. When these NACKs are received, the sender uses their total number and the times at which they were received to estimate the total number of receivers. Some papers (e.g. [32]) use only the first arrived NACK; others (e.g. [17]) assume a
binomial loss distribution, therefore solving only part of the problem.

In [29], the authors propose a timer based scheme for estimating the number $n$ of hosts contending for an access to a shared ALOHA channel. The purpose of the estimation is to set the access probability to the channel to $1/n$, thereby maximizing the throughput. The scheme is derived from the probability $p_{fs}$ of a successful access during the first transmission given a successful delivery of the packet. To this end, with each transmission of a packet, each contending host indicates whether this packet is being transmitted for the first time, or whether it has already experienced a collision. Although the problem solved in [29] is different from the one we study, it seems at first glance that the same mechanism can also be used to estimate the number of receivers missing some packets sent by the base station, e.g., by having these receivers contend for transmitting their RPRTs. However, this is not the case, mainly because the scheme in [29] must allow most of the receivers to transmit their packets, whereas in our model only a very small fraction of bad receivers are allowed to transmit their RPRTs. Moreover, the scheme in [29] does not impose an upper bound on the time period during which response messages should be sent by the receivers.

As indicated in Section 4.1, we also address the implementation of NATO! in a broadband access wireless network, where the transmissions of hosts to the base station are subject to loss due to contention. The problem is that any loss of a NATO! report message is translated into a significant estimation error. MAC layer retransmissions do not solve this problem because they do not guarantee that the message will be received by the base station before it must complete the estimation. To overcome this problem, we propose to integrate NATO! into a contention resolution protocol. For this part of the chapter, there are many related works, starting with the publication of the Aloha algorithm by Abramson [2] in 1970. In 1977, Capetanakis [14] first proposed the concept of collision resolution. He showed that collision resolution algorithms are stable when the packet arrival rate is “not too high.” Independently of Capetanakis, Tsybakov and Mikhailov [51] proposed the use of collision resolution algorithms and advanced their analysis.

Unlike ALOHA-based access protocols, the tree collision resolution algorithm [14, 51] guarantees that any group of colliding hosts will be able to resolve the collisions and send their data packets on the shared medium. The main idea is that the nodes not involved in a collision wait for it to resolve, while the colliding nodes are randomly split into two sets, $L$ and $R$. The nodes in $L$ send their packets before those in $R$. Each additional collision results in another split of the contending nodes. Finally, after every node in $L$ transmits its packet successfully, the nodes in $R$ can use the channel. In [47], the authors describe a variant of the tree algorithm that runs collision resolution rounds while leaving slots for new transmissions.
Over the years, many modifications and improvements have been proposed to the original tree protocol. For example, consider the case where every contending station is mapped to the $R$ set. In this case, the next slot must be empty and the one afterwards must contain a collision. In [38], a modified protocol is proposed where the nodes in the $R$ set avoid transmission after an idle slot. Rather, each of them reselects a random L or R set. Another improvement to the throughput, called FCFS, is proposed in [18]. The tree structure is eliminated, and only two subsets of nodes have to be considered: the current and the next. This and other enhancements are described in [19], which presents an excellent survey of multiple access protocols.

4.3 Reliable Multicast with NATO!

NATO!, as described in Chapter 2, is a generic scheme that allows a centralized gateway (the base station, in our case) to estimate the number of nodes affected by some event. In the following sections, we use NATO! to enhance partially reliable multicast transmissions.

4.3.1 Optimization Criteria for Partially Reliable Multicast

We consider large multicast groups of up to thousands of members. The sender is configured to use up to $R$ transmission rounds for every data block, where $R \geq 1$. In round $i$, the sender sends $n_i$ FEC (repair) packets, where $\sum_{i=1}^{R} n_i \geq K$. In order to decode the data block, a receiver needs to receive at least $K$ of these packets. If $R > 1$, the sender needs to receive a feedback message from the receivers after every round except the last one, in order to decide how many packets should be transmitted in the next round.

To use NATO! and to discuss its performance, we need to define an optimization criterion for the performance of a “partially reliable” multicast protocol. We believe that no single optimization criterion can fit every system. In this chapter we consider the following two criteria:

**OC-1** Maximize $G/B$, where $G$ is the expected number of receivers that are able to decode the data block by receiving at least $K$ packets, and $B$ is the total bandwidth used by the sender for encoding all the $\sum_{i=1}^{R} n_i$ packets of this data block.

**OC-2** Minimize the consumption of bandwidth $B$ used by the sender for encoding all the $\sum_{i=1}^{R} n_i$ packets of a given data block, while guaranteeing that with probability $\geq p$, every receiver whose SNR is above some threshold will be able to decode the data block, by receiving at least $K$ packets.
4.3.2 The Proposed Protocols

For the protocols proposed in this section, each node periodically computes the moving average of its SNR (signal-to-noise ratio), using the following procedure:

**Procedure Find-local-SNR()**

1. \( SNR = (1 - \alpha) \cdot SNR + \alpha \cdot NewlyMeasuredSNR \)

2. Return \( SNR \).

In this procedure, \( 0 < \alpha < 1 \) is the weight of the last local SNR measurement while \( 1 - \alpha \) is the weight of the previously computed moving average. The value of \( \alpha \) depends on the mobility of the node. When the node is highly mobile, \( \alpha \) should be closer to 1, whereas for a static node, \( \alpha \) should be closer to 0.

The protocols proposed in this section are 1-round protocols that invoke NATO! periodically in order to discover the distribution of the expected loss encountered by the nodes in the multicast group. In Section 4.4.2 we compare them to a 2-round reactive protocol and show that for many sets of parameters they perform better with regard to OC-1 and OC-2.

The following procedure finds the expected loss distribution by mapping an SNR value into the corresponding probability to receive a packet for the considered modulation and coding scheme (see [9] for further details):

**Procedure Find-loss-distribution()**

1. Choose a range \([p_{min}, 1]\) of probabilities. Only receivers whose SNRs are translated to probabilities in this range are targeted.

2. Using a mapping between an SNR value and the corresponding probability to correctly receive a packet transmitted using the considered modulation and coding scheme, find the SNR range \([SNR_{min}, SNR_{max}]\) for the selected range of probabilities.

3. Divide the SNR range into \( s \) subranges. For each subrange, execute NATO! to find the number of nodes whose SNR falls within this subrange.

4. Let \( p_i \) be the corresponding probability to the left bound of the \( i \)'th subrange, where \( p_1 \leq p_2 \leq \ldots \leq p_s \). Let \( r_i \) be the number of nodes falling into the \( i \)'th subrange as determined by NATO!. Return the vector \( \{(p_i, r_i)|i = 1, \ldots, s\} \).

Note that the subranges in step 2 are not necessarily equal.
The first protocol we propose is for OC-1. Let \( \{(p_i, r_i) \mid i = 1, \ldots, s\} \) be the vector returned by \text{Find-loss-distribution}(), and let \( V_i \) be the set of nodes corresponding to element \((p_i, r_i)\) in this vector. For a given \( i \), the expected number of packets a node in \( V_i \) receives if \( n \) packets are transmitted is \( \geq np_i \). Thus, the nodes in \( V_i \) are expected to decode the data block correctly if \( np_i \geq K \) holds. Denote \( l_n = \min\{i \mid p_i \geq \frac{K}{n}\} \). Then, \( G_n/B_n = (\sum_{i=l_n}^s r_i)/n \). To summarize:

**Protocol 1 (for OC-1)**

1. Run \text{Find-loss-distribution}() to calculate \((p_i, r_i)\) for \( i = 1, \ldots, s \).
2. For every \( n \) from \( K \) to the maximum number of packets the base station can use for the considered data block, calculate \( g_n = G_n/B_n = (\sum_{i=l_n}^s r_i)/n \).
3. Return the value of \( n \) that maximizes \( g_n \).

The second protocol is for OC-2. Procedure \text{Find-loss-distribution} is invoked to find the pairs \((p_i, r_i)\) for the subranges \( i = 1, \ldots, s \). Let \( r \) be the total number of nodes in the multicast group. The value of \( r \) is found using NA TO!. Let \( p \) be the percentage of nodes that should receive at least \( K \) packets. As in Protocol 1, the expected number of such nodes is \( \sum_{i=l_n}^s r_i \). For OC-2 we want this number to be \( \geq p \cdot r \). To summarize:

**Protocol 2 (for OC-2)**

1. Run \text{Find-loss-distribution}() to calculate \((p_i, r_i)\) for \( i = 1, \ldots, s \).
2. Starting with \( n = K \), calculate \( h_n = \sum_{i=l_n}^s r_i \) for every \( n \) until \( h_n \geq p \cdot r \).
3. Return \( n \).

If NA TO! is invoked every 3 seconds, say, and it is used for about \( K \) queries, where \( K \) is the number of packets a host needs to receive in order to decode a multicast data block (a typical value is 10), the total cost of NA TO! is equal to 10-30 uplink messages per second per multicast group. This is about 5-10% of the number of packets sent on the downlink for a typical streaming application with the same value of \( K \).
4.4 The Performance of the Proposed Protocols

4.4.1 Simulation Results for Protocol 1 and 2

We now present simulation results for Protocol 1 and 2 from Section 4.3.2. We simulated 1,000 nodes in the multicast group. The SNR value of each node is randomly chosen using several probability distribution functions. The modulation scheme we consider is rate 1/2 coded 64 Quadrature Amplitude Modulation (64-QAM 1/2). For this modulation, the considered $[SNR_{min}, SNR_{max}]$ range is $[11.5, 15.5]$ dB [9]. A node with a lower SNR value has 100% loss rate, while a node with a higher SNR value has 0% loss rate. The SNR range is divided into $s$ equal subranges and the number of nodes in each subgroup is found with N/A TO! The number $K$ of packets required to decode a data block is also variable, chosen in accordance with [41] and [44].

Figures 4.1–4.4 depict the results for Protocol 1. For these figures, the SNR of the participating nodes is chosen using a uniform distribution from the range $[SNR_{min}, 15.5]$ dB. Other distributions show similar results. The $x$-axis is $n - K$, i.e., the number of proactive packets, transmitted by the sender in excess to the minimum $K$ packets. The $y$-axis is the corresponding $G/B$ value.

Figure 4.1 shows different values of $K$. In all cases the number of required proactive packets (beyond $K$) for getting the maximum $G/B$ is small (1–3). Moreover, the smaller $K$ is, the smaller this number is, and the larger the corresponding value of $G/B$ is. This is expected since one proactive packet is likely to satisfy more receivers when $K = 10$ but not when $K = 20$ or 30. It might seem that choosing $K = 1$ would yield the best results, since in this case one proactive packet is very likely to satisfy all the receivers, which maximizes the value of $G/B$. However, $K = 1$ counteracts the benefit of FEC: if a data block consists of a single packet, the probability to correctly receive this packet is much lower than if the data block is split into 12 packets for
Figure 4.2: Protocol 1 for $K = 10$, $s = 5$ and $SNR_{min} = 14.7, 15.1$ and $15.3$ dB

Figure 4.3: Protocol 1 for $SNR_{min} = 13.5$ dB, $K = 10$, $s=5, 10$ and $50$

$K = 10$. Thus, $K$ should not be chosen in order to maximize $G/B$. In fact, the inverse is true: once $K$ is chosen, it is possible to maximize $G/B$ and achieve OC-1 by using Protocol 1.

Figure 4.2 shows the results for the same case as Figure 4.1, except that the changed parameter is not $K$ but the SNR range. The left bound of the SNR range is 14.7, 15.1 or 15.3 dB, which correspond to 80%, 90% or 95% of the range. We see that the curves for 15.1 or 15.3 dB coincide, while the 14.7 dB curve has a lower maximum that is reached at a larger value of $n - K$. This is because when the SNR is high, one proactive packet satisfies many receivers. Additional packets increase $B$ with almost no effect on $G$. However, for a lower SNR, additional proactive packets significantly contribute to $G$, thus increasing $G/B$ up to a certain number of packets.

In Figure 4.3 we show different curves for three different values of $s$, the number of subranges into which the SNR range is divided. Recall that NATO! is executed for every such subrange. Thus, more subranges give a more accurate result at higher estimation cost. As opposed to the previous graphs, here $SNR_{min} = 13.5$ dB because, when setting $SNR_{min} = 15.1$ dB, the range
15.1–15.5 dB is small enough and different values of $s$ do not affect the results. We see in this figure that for smaller values of $s$ spikes appear in the graph, while for high values of $s$, e.g., 50, the curve becomes smoother. These spikes are the result of the coarse division of the receivers into SNR subranges. With a small value of $s$, the width of the SNR subrange is large, and receivers near the top of the subrange are still considered to have the lowest SNR in the range. However, in spite of the spikes, the maximum has approximately the same $x$-coordinate regardless of $s$. In other words, even for relatively low values of SNR (e.g., 13.5 dB), small values of $s$ are enough to get the precise number of required proactive packets. This is all the more true for high values of SNR (e.g., 15.1 dB).

The graphs in Figure 4.4 differ from the previous graphs in that the SNR distribution among nodes is not uniform. In the upper curve, 800 of the 1000 nodes have an SNR of 15.1 dB and 200 nodes have an SNR of 13.9 dB. For the lower curve the setting is different: 600 nodes have 15.1 dB and 400 nodes have 13.5 dB. Both cases simulate a situation where some nodes are close to the base station and thus have a good SNR, while the other nodes are behind some obstacle (e.g., a tall building) and have a bad SNR. The graph shows 2 peaks for each curve with a gap between them. The first peak corresponds to the satisfaction of the nodes with the higher SNR, whereas the second peak corresponds to the satisfaction of the nodes with the lower SNR. As soon as enough proactive packets are sent to satisfy the high-SNR receivers, sending more packets does not increase $G$ but does increase $B$, thus lowering the value of $G/B$. This happens until enough packets are sent to also satisfy the receivers with a low SNR, at which point $G/B$ increases in the form of a spike.

Next, we show simulation results for Protocol 2. In Figures 4.5 and 4.6 the $x$-axis is still the number of proactive packets, but the $y$-axis shows the number of satisfied receivers. Obviously,
the curves are monotonically increasing: the more packets sent, the more satisfied receivers there are. The figures have a “target” curve that is the probability $p$, as defined in OC-2, multiplied by the number of receivers. As soon as the number of satisfied receivers is above this threshold, the base station knows that each receiver is satisfied with probability $\geq p$, which is the objective of OC-2. In both figures $p$ was chosen to be 97%.

Figure 4.5 shows the number of satisfied receivers for $K = 10$ when $SNR_{min} = 14.7$ dB. As we can see, with 3 proactive packets, at least 97% of the receivers are satisfied. When setting $SNR_{min} = 15.1$ dB, the portion of satisfied receivers exceeds 97% even for 1 proactive packet.

In Figure 4.6, the curves show the difference in the number of satisfied receivers when $K = 20$ or $K = 30$ packets have to be received in order to decode the data block. As before, the threshold was set to 97%. Naturally, when $K = 30$, more proactive packets are required.
4.4.2 Comparison to a 2-round Reactive Protocol

We now compare Protocols 1 and 2 to the following simple 2-round reactive protocol with regard to OC-1 and OC-2. While the following protocol is not explicitly defined in [3], it works in the spirit of the framework of [3].

**Protocol 3** *(a generic 2-round reactive protocol)*

1. Send $K$ packets.
2. Find $l$, the maximum number of lost packets.
3. Send $l$ more packets.

As explained in [3], in order to implement step 2 in the protocol, each non-satisfied receiver draws a random timer from a truncated exponential distribution. The base station continuously announces the maximum number of repair packets requested by other receivers. When the timer expires, a receiver checks whether its repair needs are superseded by the already sent repair requests. Only if this is not the case does it send its own request.

While our protocols have only 1 round, Protocol 3 is a 2-round protocol. Using NATO! we are able to perform precise loss measurements once in several rounds, and thus combine all transmissions into one round.

If there are many receivers, then for every SNR range, there is a very high probability that at least one receiver will receive the minimum possible number of packets. For example, for SNR $= 15.1$ dB, the maximum loss rate is 10%. Thus, the maximum number of lost packets will be $0.1 \cdot K$. Looking at Figure 4.1, we see that the values $0.1 \cdot K$ for $K = 10$, 20 and 30 correspond to the x-coordinate of the maximum. This means that for this setting, Protocol 3 and Protocol 1 perform equally well despite the fact that Protocol 1 uses only one round.

If SNR$_{min} = 14.7$ dB and $K = 10$, a setting corresponding to the lower curve of Figure 4.2, Protocol 3 sends 2 more packets in step 3. This is because an SNR of 14.7 dB corresponds to 20% loss, and thus $0.2 \cdot K = 2$ more packets are transmitted. But the maximum of this curve, and thus of $G/B$, is achieved for 3 packets. A bigger difference arises when some receivers are close to the base station and some are behind an obstacle, as in the lower curve of Figure 4.4. Here, the worst SNR is 13.5 dB. Thus, Protocol 3 transmits 5 more packets, whereas Protocol 1 sends only 2 packets and achieves a much higher $G/B$.

The comparison of Protocol 3 with Protocol 2 shows similar results. With the setting of Figure 4.6, the minimum SNR corresponds to 10% loss. Thus, Protocol 3 sends 2 or 3 more packets,
for $K = 20$ or $30$ respectively, and achieves the same target threshold achieved by Protocol 2. However, in Figure 4.5, the setting of $\text{SNR}_{\text{min}} = 14.7$ dB and $K = 10$ result in Protocol 3 sending 2 packets. These 2 packets achieve a threshold of only $80\%$ instead of the desired $97\%$.

To summarize, the main conclusions we draw from the simulation study are:

- Protocols 1 and 2 achieve the desired goals of OC-1 and OC-2 respectively.
- Protocols 1 and 2 perform as well as a 2-round reactive protocol for some set of parameters.
- For many sets of parameters, Protocols 1 and 2 outperform a 2-round reactive protocol.
- Coarse division of the SNR range, which reduces the overhead of NATO!, is good even for bad SNR conditions.

### 4.5 The Implementation of NATO! in a Contention Channel

#### 4.5.1 The Effect of RPRT Losses on NATO!

The excellent precision of NATO! is achieved under the condition that all the RPRTs messages are received by the gateway (the base station in our case). When this is not the case, the precision is reduced. In Section 2.5 we studied the effect of RPRT losses on NATO! In what follows we state the main results of Section 2.5 and show why we need to integrate a mechanism for collision resolution into NATO!.

Suppose that by the time the gateway receives the $N$’th RPRT, $S$ RPRTs have been lost. Therefore, the RPRT considered by the gateway to be the $N$’th is actually the $N + S$’th. This influences the equation the gateway solves to find $r$, which now should read:

$$\frac{1}{r} + \frac{1}{r-1} + \ldots + \frac{1}{r-(N-1)} + \ln(1 - F(x_{N+S})) = 0. \quad (4.1)$$

The estimation error is then:

$$-pS \frac{1}{N+S-1} - (1-p) \leq \frac{E(R) - r}{r} \leq -qS \frac{1}{N+S-1} - (1-q) + \frac{N-1}{r}. \quad (4.2)$$

In this equation, $p = \ln \left(1 + \frac{1}{r(N+S-1)}\right)^{r-(N+S-1)}$ and $q = \ln \left(1 + \frac{1}{r-1}\right)^{r-1}$. This time, when
Figure 4.7: Estimation error vs. $N$ for $S = 1$ and 5

$r \gg N$ and $N + S \gg 1$, we get

$$r_{\text{real}} \approx \frac{r_{\text{estimated}} (N - 1 + S)}{N}. \quad (4.3)$$

Unlike the case $S = 0$, where the gateway multiplies $r_{\text{estimated}}$ by $\frac{N-1}{N}$, here it cannot get rid of the error, because the value of $S$ is unknown. Therefore, the gateway’s algorithm when losses are possible is Algorithm 2 of Chapter 2 without an error correction step.

Figure 4.7 shows the estimation error in the presence of RPRT loss. We consider 1 or 5 RPRT losses, and show the error using both simulation and Eq. 4.3. A few conclusions can be drawn from this figure. First, we can see very good agreement between the simulation and the analysis. Second, comparing Figure 4.7(a) to Figure 4.7(b) reveals no major difference. This implies that, as before, the error is independent of $r$. However, the most important conclusion we draw from the figure is that when RPRTs are subject to loss, NATO! should be enhanced with a mechanism for RPRT retransmissions.

4.5.2 Combining NATO! with a Collision Resolution Protocol

When the loss rate is small, e.g., due to transmission errors, a simple ACK-based scheme allows every receiver to guarantee the receipt of its RPRT by the gateway. However, when NATO! is executed over a shared uplink channel, as in the application considered in this chapter, a more efficient reliability scheme is required. We now show how to efficiently implement NATO! over such a channel.

The main idea is to run NATO! in conjunction with a distributed protocol for collision resolu-
tion [14, 51]. When a receiver needs to send a RPRT message, it draws a random number from the interval \((0, 1]\) using a uniform distribution. The base station needs to get only the RPRTs with the \(N\) smallest numbers (times). This is done by means of POLL messages broadcast by the base station on the downlink. Each such a message specifies an interval \((t_1, t_2]\). Receivers whose drawn number falls into this interval send their RPRTs in an uplink time slot specifically allocated by the base station for this purpose. Let the \(i\)'th POLL message be POLL\((i)\). Let the interval specified by this message be \((t^{i}_1, t^{i}_2]\), where \(t^{i}_2 - t^{i}_1 = \Delta^i\). The initial values are \(t^{1}_1 = 0\) and \(t^{1}_2 = \Delta^1 = \Delta\), where \(\Delta\) is a constant of the algorithm. Hence, the interval announced by POLL\((1)\) is \((0, \Delta]\).

After the base station sends POLL\((i)\), there are three possible cases:

[R1] No RPRT is received for \((t^{i}_1, t^{i}_2]\). In this case the base station sends the next POLL message, POLL\((i + 1)\), with \(t^{i+1}_1 = t^{i}_2\) and \(\Delta^{i+1} = 2\Delta^i\). In other words, the interval is shifted by \(\Delta^i\) units and its width is doubled.

[R2] Exactly one RPRT is received, whose value is \(t\). If this is the \(N\)'th RPRT to be received, the protocol stops. If this is not the \(N\)'th RPRT, the base station sends a new POLL message, POLL\((i + 1)\), in order to obtain the next RPRT. For this POLL, \(t^{i+1}_1 = t\) and \(\Delta^{i+1} = \Delta^i\). In other words the interval for POLL\((i + 1)\) starts at \(t\) and has the same length as POLL\((i)\). This is because this interval length is likely to contain exactly one RPRT in the next POLL too.

[R3] A collision occurs due to the transmission of two or more RPRTs. In such a case the base station makes a binary search for the first colliding RPRT on the interval \((t^{i}_1, t^{i}_2]\). If this RPRT is the \(N\)'th one, the algorithm stops. Otherwise, after this RPRT is found, say at time \(t\) by POLL\((j)\), the algorithm sets \(t^{j+1}_1 = t\) and \(\Delta^{j+1} = \frac{1}{2}\Delta^i\). In other words the interval for POLL\((i + 1)\) starts at \(t\) and has half of the length of POLL\((i)\). This is because this interval length is likely to contain less RPRTs than POLL\((i)\), and with luck, exactly one RPRT.

The binary search on the interval \((a, b]\) works as follows. The sender sends a POLL, for which \(t_1 = a\) and \(t_2 = \text{mid}\), where \(\text{mid} = \frac{1}{2}(a + b)\). Now there are several possible cases:

(a) If no RPRT is received, the binary search is recursively executed on the interval \((\text{mid}, b]\).

(b) If exactly one RPRT is received for time \(t\), the binary search stops and the algorithm searches for the next RPRT in \((t, t + \frac{1}{2}\Delta^i]\) as indicated above.

(c) If a collision of two or more RPRTs occurs, the binary search is executed recursively on the interval \((a, \text{mid}]\).  

67
4.5.3 Simulation Results for the Performance of NATO! with Collision Resolution

We now present simulation results for the performance of the above protocol. The bandwidth and the time consumed by this protocol are both functions of the number of POLL messages, because each POLL requires one round trip and one uplink slot. Hence, we measure the performance of this scheme in terms of this number.

We simulate the scheme by having $r$ receivers, from which the base station needs to receive $N$ RPRTs. To receive these RPRTs, the base station sends POLL messages as dictated by the scheme. This procedure is repeated 50 times for every value of $N$ in the range of $[1, 10]$. Our graphs depict the average number of POLLs during these 50 runs as a function of $N$. In all the graphs the x-axis is $N$ and the y-axis is the number of POLLs.

Figure 4.8 depicts several curves with different initial values of $\Delta$. These values are relevant only when the algorithm starts running. Some curves in Figure 4.8 are not linear. For example, the curve of $\Delta = 0.1$ when $r = 1,000$. These curves start with a steeper slope that decreases as $N$ gets larger, until they reach the slopes of the corresponding linear curves. The initial slope is highly dependent on the initial value of $\Delta$. For small values of $N$, the interval size is doubled or halved at each POLL until it reaches a value that leads to the least number of replies for a POLL. For larger values of $N$ ($N \geq 6$), the curves have a much less steeply inclined slope, which is the result of using the better interval size as “calculated” for small $N$’s.

We can see that when $N$ becomes larger, the interval size is likely to adapt to the number of contending users. Hence, all the curves have the same slope. From Figure 4.8(a) it is also evident that selecting a too small $\Delta$ (e.g. 0.0001) is better than selecting one (e.g. 0.01 or 0.1) that is too large. Since a small value of $N$ is enough for an accurate estimation using the error cancellation technique described in Section 2.4, using $\Delta = 0.001$ could be acceptable for both small and large values of $r$.

During the above discussion we assumed that the receivers select the times for their RPRTs from a uniform distribution. As mathematically proven in Section 2.4, any other distribution will result in the same quality of estimation for a similar number of RPRTs. However, we now show that by choosing a different distribution we can reduce the number of POLL messages required to receive the first $N$ RPRTs.

While $\Delta = 1/r$ will result in fewer POLLs than other values of $\Delta$, we cannot select such a value since $r$ is unknown. As discussed above, it is better to choose a value smaller than $1/r$. For a value of $\Delta$ bigger than $1/r$, collisions rather than empty slots, and the required binary search to resolve them, are the major reason for requiring several POLLs per RPRT. Since we are interested
only in the first $N$ RPRTs, where $N$ is a small number, a good distribution should ensure that RPRTs are less frequent at the beginning of the interval, perhaps at the expense of being more frequent near its end. An example of such a distribution is the truncated exponential distribution on the interval $(0, 1)$: $f(x) = \frac{1}{e^{\lambda}-1} \cdot \lambda e^{\lambda x}$, where $\lambda > 0$.

Figure 4.9 shows the number of POLL messages needed for $N$ RPRTs with truncated exponential distribution and $\lambda = 1$, compared to the uniform distribution, for $r = 1,000$ and $r = 10,000$ affected nodes. Four curves are shown in each figure. Consider first the bottom two curves, which represent the case where $\Delta = 1/r$ (in Figure 4.9(a) they almost fully overlap). The upper one is for the uniform distribution while the lower is for the exponential distribution. For this setting, the exponential distribution performs similarly or only marginally better than the uniform one, be-
cause $\Delta = 1/r$ is a good choice, as explained above. On the other hand, selecting a bad value for $\Delta$, such as $\Delta = 0.1$ (the top two curves in Figure 4.9(a) and 4.9(b)), shows a clear improvement of the exponential distribution over the uniform one.
Chapter 5

Conclusions

This thesis is the first to explicitly show the correlation between the number of RPRTs sent by a group of affected nodes and the ability of the gateway to precisely estimate the size of this group. In Chapter 2 we developed a statistical analysis algorithm for estimating the number of affected nodes. The algorithm, which is based on the times the RPRTs are received, defines the likelihood function for the received RPRTs and then uses the Newton-Raphson method to find the number of receivers for which this function is maximized. We analyzed the error of our algorithm and showed that when $1 \ll N \ll r$, where $r$ is the number of affected nodes and $N$ is the number of RPRTs, this error is positive and approximately equals to $1/(N - 1)$. We used this important result to correct the estimation of our algorithm, and to bring its error very close to 0.

Chapter 3 addresses the problems of feedback suppression in hierarchical sensor networks. We considered a hierarchical sensor network and adjusted the NATO! scheme to be used by a root gateway in order to estimate the number of sensors experiencing a given event. To reduce the communication cost, we developed a new scheme and a new algorithm to be employed by the root gateway. We believe that no other scheme can perform the same task with a lower communication cost. We analyzed the new scheme mathematically, calculated its estimation error, and showed that the actual error is very close to the analytical result. Using this observation, we improved the estimation algorithm and substantially reduced its error.

Finally, Chapter 4 employed the generic NATO! protocol for the sake of partially reliable multicast streaming in broadband access wireless networks. We defined two optimization criteria for partially reliable multicast streaming and developed a protocol based on NATO! to address each of them. The main idea behind the proposed protocols is to find the loss distribution at the receivers and to deduce from it how many packets have to be sent in order to satisfy each
optimization criterion. Using simulations, we studied the performance of both protocols with respect to the relevant optimization criteria. We also defined a 2-round reactive protocol for both optimization criteria and showed that each of our NATO!-based protocols not only abolishes the need for multiple rounds but also performs better than the corresponding 2-round reactive protocol.
Bibliography


Shimon Nof - ב. נאט''ו -Normalized Streaming
This work considers the use of streaming, particularly in the context of NATO, for broadcast and multicast streaming. The primary concern is the maintenance of reliability and quality of service, especially during multicast situations.

The solution presented aims to ensure reliability in broadcast and multicast streaming, particularly in the context of NATO. The proposed solution utilizes NAK-Oriented Reliable Multicast (NORM) for efficient communication.

Technion - Computer Science Department - Ph.D. Thesis  PHD-2009-13 - 2009
NATO messages are encoded and decoded in a hierarchical manner, with each level of the hierarchy handling a specific set of tasks.

The messages are transmitted through the NATO network, which was designed to handle a large number of messages efficiently. The network is structured in a hierarchical manner, with each level performing a specific set of tasks.

The infrastructure of the network includes a central hub that receives and processes the messages from the various sensors. The central hub then distributes the messages to the appropriate destinations, such as the command centers.

The network is designed to handle a large number of messages, and it is optimized to ensure that the messages are delivered accurately and efficiently. The network is constantly monitored and maintained to ensure that it remains operational at all times.

In conclusion, the NATO network is a highly efficient and reliable system that is designed to handle a large number of messages and ensure that they are delivered accurately and efficiently.
BK is also known as the best platform for modern networks. In this paper, we explore the interconnectedness of networks, a concept that is central to today’s world. We focus on understanding the fundamental requirements of network design and the challenges in their implementation.

Grids, like communication networks, require efficient and reliable data transmission. Sensing networks, wireless networks, or physical networks can all be categorized as thousands of interconnected nodes, each responsible for managing traffic at the edge. These networks are managed centrally, with control signals transmitted to the network backbone. This process is repeated at the network backbone, which is responsible for managing traffic at the central level.

Technion - Computer Science Department - Ph.D. Thesis PHD-2009-13 - 2009

The NKI
باحثת נועשה בחנניל פורפ"ה רדאובן כهى בפקולטה למבדי המחשב

אני מודה לטעני על התמיכה המתמדת והונית בהשתיים

Technion - Computer Science Department - Ph.D. Thesis PHD-2009-13 - 2009
שיטות יעילות לHttpServletRequest מספים האזרחים
המורשת מיוצרת ברשויות הגדולות נאדה

חיבת על מחקר

לשם مليי חלקי של הדירישות לקבלת התואר
דוקטור לפילוסופיה

ולוגיה לדג腳

הוגש לסנט הטכניון – מכון טכנולוגי לישראל
תיכון 2009
אוקטובר, חיפה
שיטות יעילות להערכה מספר החמאיים
המושגים מאמרים ברשאות גדולים מאז

אלכסנדר לאפר

Technion - Computer Science Department - Ph.D. Thesis PHD-2009-13 - 2009