REIHE INFORMATIK
12/2001

Extremum Feedback for Very Large Multicast Groups

Jörg Widmer and Thomas Fuhrmann
Universität Mannheim
Praktische Informatik IV
L15, 16
D-68131 Mannheim

Extremum Feedback for Very Large Multicast Groups

Jörg Widmer* Thomas Fuhrmann†

Mai 25, 2001

Abstract

In multicast communication, it is often required that feedback is received from a potentially very large group of responders while at the same time a feedback implosion needs to be prevented. To this end, a number of feedback control mechanisms have been proposed, which rely either on tree-based feedback aggregation or timer-based feedback suppression. Usually, these mechanisms assume that it is not necessary to discriminate between feedback from different receivers. However, for many applications this is not the case and feedback from receivers with certain response values is preferred (e.g., highest loss or largest delay).

In this paper, we present modifications to timer-based feedback suppression mechanisms that introduce such a preference scheme to differentiate between receivers. The modifications preserve the desirable characteristic of reliably preventing a feedback implosion.

Keywords: multicast feedback; extremum detection; feedback suppression; biased feedback

1 Introduction

Many multicast protocols require receiver feedback. Feedback can be used for negative acknowledgements in reliable multicast [5], for control and identification functionality for multicast transport protocols [12], for status reporting from receivers for congestion control [11], and for the reporting of allocation clashes in multicast address allocation mechanisms [8]. In such scenarios, the size of the receiver set is potentially very large. Sessions with several million participants may be common in the future and without an appropriate feedback control mechanism a severe feedback implosion is possible.

Some multicast protocols arrange receivers in a tree hierarchy. This hierarchy can be used to aggregate receiver feedback at the inner nodes of the tree to effectively solve the feedback implosion problem. However, in many cases such a tree will not be available (e.g., for satellite links) or cannot be used for feedback aggregation (e.g., in networks without router support). For this reason, we will focus on feedback control using timer-based feedback suppression throughout the remainder of the paper.

Pure end-to-end feedback suppression mechanisms do not need any additional support except from the receivers themselves and can thus be used for arbitrary settings. The basic mechanism of feedback suppression is to use random feedback timers at the receivers. Feedback is sent when the timer expires unless it is suppressed by a notification that another receiver (with a smaller timeout value for its feedback timer) already sent feedback.

Most of the mechanisms presented so far assume that there is no preference as to which receivers send feedback. As we will see, for many applications this is not sufficient. Those applications require the feedback to reflect an extreme value for some parameter within the group. Multicast congestion control, for example, needs to get feedback from the receiver(s) experiencing the worst network conditions. Other examples are the polling of a large number of sensors for extreme values, online auctions where one is interested in the highest bids, and the detection of resource availability in very large distributed systems.

In this paper we propose several algorithms that favour feedback from receivers with certain characteristics while preserving the feedback implosion avoidance of the original feedback mechanism. Our algorithms can therefore be used to report extrema from very large multicast groups.

Past work related to this paper is presented in section 2. In section 3 we summarize basic properties of timer-based feedback algorithms and give some definitions to be used in our analysis. Depending on the amount of knowledge about the distribution of the values to be reported we distinguish extremum detection and feedback bias. With the former we just detect extreme values without forcing early responses from receivers with extreme values. This variant which requires no additional information about the distribution of the values is studied in section 4. With the latter we exploit knowledge about the value distribution by biasing the timers of responders. Biased feedback is studied in section 5. In both sections, we give a theoretical analysis of the properties of our feedback mechanisms and present

*Praktische Informatik IV, University of Mannheim, Germany
†The Boston Consulting Group, Munich, Germany

simulations that corroborate our findings. We conclude the paper and give an outlook on future work in section 6.

2 Related work

Feedback suppression algorithms have already been widely studied and employed. Good scalability to very large receiver sets can be achieved by exponentially distributing the receivers' feedback times. A method of round-based polling of the receiver set with exponentially increasing response probabilities was first proposed in [2] to be used as a feedback control mechanism for multicast video distribution. It was later refined by Nonnenmacher and Biersack [10], using a single feedback round with exponentially distributed random timers at the receivers. In [7], the authors compare the properties of different methods of setting the timer parameters with exponential feedback and give analytical terms and simulation results for feedback latency and response duplicates. However, none of these papers consider preferential feedback.

A simple scheme to gradually improve the values reported by the receivers is presented in [1]. Receivers continuously give feedback to control the sending rate of a multicast transmission. Since the lowest rate of the previous round is known, feedback can be limited to receivers with as low a rate. The rate must be adjusted by the largest possible increase during one round to be able to react to improved network conditions. After several rounds, the sending rate will reflect the smallest feedback value of the receiver set. While not specifically addressed in the paper, this scheme could be used in combination with exponential feedback timers for suppression within the feedback rounds to reliably prevent a feedback implosion. However, with this scheme it may still take a number of rounds to obtain the optimum feedback value.

To our knowledge, the only work that is directly concerned with altering a non-topology based feedback suppression mechanism to solicit responses from receivers with specific metric values is presented in [3]. The authors discuss two different mechanisms, Targeted Slotting and Damping (TDS) and Targeted Probabilistic Iterative Polling (TIPP). For TDS, response values are divided into classes and the feedback mechanism is adjusted such that response times for the classes do not overlap. Responders within a better class always get to respond earlier than lower-class responders. Thus, the delay before feedback is received scales linearly with the number of empty "high" classes. Furthermore, it is not possible to obtain real values as feedback without the assignment of classes. To prevent implosion when many receivers fall into the same class, the response interval of a single class is divided into subintervals and the receivers are randomly spread over these intervals. It was shown in [10, 7] that a uniform distribution of response times scales very poorly to large receiver sets. TIPP provides better scalability by using a polling mechanism based on the scheme presented in [2], thus having more favourable characteristics than uniform feedback timers. However, separate feedback rounds are still used for each possible feedback class. This results in very long feedback delays when the number of receivers is overestimated and the number of feedback classes is large. Underestimation will lead to a feedback implosion. As a solution, the authors propose estimating the size of the receiver set before starting the actual feedback mechanism. Determining the size of the receiver set requires one or more feedback rounds. In contrast, the mechanisms discussed in this paper only require a very rough upper bound on the number of receivers and will result in (close to) optimal feedback values within a single round. A further assumption for TDS and TIPP is that the distribution of the response values is known by the receivers. In most real scenarios this distribution is at best partially known or even completely unknown. If however the distribution is known, a feedback mechanism that guarantees optimum response values and at the same time prevents a feedback implosion can be built. Such a mechanism is presented in section 5.

3 General considerations

Let us first summarize some general considerations about feedback on which we will later base our analysis. For feedback suppression with exponentially distributed timers, each receiver gives feedback according to the following mechanism[1]:

Algorithm 1 *Let N be an estimated upper bound on the number of potential responders and T an upper bound on the amount of time by which the sending of the feedback can be delayed in order to avoid feedback implosion.*

Upon receipt of a feedback request each receiver draws a random variable x uniformly distributed in $(0, 1]$ and sets its feedback timer to

$$t = T \max(0; 1 + \log_N x) \tag{1}$$

When a receiver is notified that another receiver already gave feedback, it cancels its timer. If the feedback timer expires without the receiver having received such a notification, the receiver sends the feedback message.

[1] See [7]. Extending the suggestions in [10] this algorithms sets the parameter of the exponential distribution to its optimal value $\lambda = \ln N$ and additionally introduces an offset of N^{-1} at $t = 0$ into the distribution that further improves the feedback latency.

The set of potential responders is formed by the participants that simultaneously want to give feedback. If no direct estimate is possible, N can be set to an upper bound on the size of the entire receiver set.

Time is divided into feedback rounds, which are either implicitly or explicitly indicated to the receivers. In case continuous feedback is required, a new feedback round is started at the end of the previous one (i.e., after the first receiver gave feedback).

The choice of parameters is critical for the functioning of the mechanism. While the mechanism is relatively insensitive to overestimation of the size of the receiver set, underestimation will result in a feedback implosion. Thus, a sufficiently large value for N should be chosen. Similarly, the maximum feedback delay T should be significantly larger than the network latency[2] τ among the receivers since for $T \approx \tau$ a feedback implosion is inevitable.

The expected delay until the first feedback message is sent is

$$\begin{aligned} E[D] &= \frac{T}{\ln N} \int_{1/N}^{1} \frac{(1-x)^n}{x} dx \\ &\simeq T(1 - \log_N n) \end{aligned} \quad (2)$$

and the expected number of feedback messages is

$$E[M] = N^{\tau/T} \left(\frac{n}{N} + \left(1 - \frac{1}{N}\right)^n - \left(1 - \frac{1}{N^{\tau/T}}\right)^n \right) \quad (3)$$

where n is the actual number of receivers. A proof of Equation (2) is given in Appendix A and a proof of Equation (3) can be found in [7]. From the latter equation we learn that $E[M]$ remains fairly constant over a large range of n (as long as $n \lesssim N$).

3.1 Unicast vs. multicast feedback channels

When receivers are able to multicast packets to all other receivers, feedback cancellation is immediate in that the feedback that ends the feedback round is received by other receivers at roughly the same time as by the sender.

However, the mechanism described in the previous section also works in environments where only the sender has multicast capabilities, such as in many satellite networks or networks where source-specific multicast [4] is deployed. In that case, feedback is first unicast back to the sender which then multicasts a feedback cancellation message to all receivers. This incurs an additional delay of half a round-trip time, thus roughly doubling the feedback latency of the system (in the case of symmetric transmission delays between the sender and the receivers.)

In order to safeguard against loss of feedback cancellation messages with unicast feedback channels, we note that it may be necessary to let the sender send multiple cancellation messages in case multiple responses arrive at the sender and/or to repeat the previous cancellation message after a certain time interval. Loss of cancellation messages is critical since a delayed feedback cancellation is very likely to provoke a feedback implosion.

3.2 Message Piggybacking

The feedback requests and the cancellation messages from the sender can both be piggybacked on data packets to minimize network overhead. In case a unicast feedback channel is used, piggybacking has to be done with great care since at low sending rates the delayed cancellation messages may provoke a feedback implosion. This undesired behavior is likely to occur when the inter-packet spacing between data packets gets close to the maximum feedback delay.

The problem can be prevented by not piggybacking but sending a separate cancellation message at low data rates (i.e., introducing an upper bound on the amount of time by which a cancellation message can be delayed). If separate cancellation messages are undesirable, it is necessary to increasing the maximum feedback delay T in proportion to the time interval between data packets.

3.3 Removing latency bias

Plain exponential feedback favours low-latency receivers since they get the feedback request earlier and are thus more likely to suppress other feedback with an early response. In case all (or some) of the receivers know their own latency τ as well as an upper bound on the latency for all receivers τ_{max}, it is possible to (partially) remove this bias. Receivers simply schedule the sending of the feedback message for time $t + (\tau_{max} - \tau)$ instead of t.

In fact, this unbiasing itself introduces a slight bias *against* low-latency receivers in case unicast feedback channels are used. While the first feedback message is unaffected, subsequent duplicates are more likely to come from high-latency receivers, since they will receive the feedback suppression notification from the sender later in time.

If it is not necessary to remove the latency bias, the additional receiver heterogeneity generally improves the suppression characteristics of the feedback mechanism, as dem-

[2] With network latency we denote the average time between the sending of a feedback response by any one of the receivers and the receipt (of a notification) of this response by other receivers.

onstrated in [10]. Similar considerations hold for the suppression mechanisms discussed in the following sections.

4 Extremum detection

Let us now consider the case where not only an arbitrary response from the group is required but an extreme value for some parameter from within a group. Depending on the purpose the required extremum can be either a maximum or a minimum. Without loss of generality we will formulate all algorithms as maximum detection algorithms.

4.1 Basic Extremum Detection

An obvious approach to introduce a feedback preference scheme is to extend the normal exponential feedback mechanism with the following algorithm:

Algorithm 2 *Let $v_1 > v_2 > \ldots > v_k > 0$ be the set of response values of the receivers.*

Upon receipt of a feedback request each receiver sets a feedback timer according to Algorithm 1. When a receiver with value v is notified that another receiver already gave feedback with $v' \geq v$, it cancels its timer. If the feedback timer expires without having received such notification (i.e., for all notifications $v' < v$ or no notifications were received at all), the receiver sends a feedback message with value v.

With this mechanism the sender will always obtain feedback from the receiver with the largest response value within one feedback round.

Let us now analyse the algorithm in detail: Following Equation (3) we use n for the actual number of potential responders and denote the expected number of feedback messages in Algorithm 1 with $R(n) := E[M]$. Let p_i be the fraction of responders with value v_i. For $k = 1$ the problem reduces to Algorithm 1 and we expect $R(n)$ feedback messages. For $k = 2$ we can reduce the problem to the previous case by assuming that every v_1 responder responds with both a v_1 and a v_2 message. Hereby, we can treat both groups independently from each other while preserving the fact that v_1 responders also stop further (unnecessary) responses from v_2 responders. Summing up both expected values we have $R(p_1 n) + R(n)$ messages. However p_1 of the v_2 messages were sent by v_1 responders and are thus duplicates. Subtracting these duplicates we obtain $R(p_1 n) + p_2 R(n)$ for the expected number of responses.

This argument can be extended to the general case

$$\begin{aligned} E[M] &= R(p_1 n) + \frac{p_2}{p_1 + p_2} R(p_1 n + p_2 n) \\ &\quad + \frac{p_3}{p_1 + p_2 + p_3} R(p_1 n + p_2 n + p_3 n) \\ &\quad + \cdots + p_k R(n) \\ &= \sum_{i=1}^{k} \frac{p_i}{P_i} R(P_i n) \end{aligned} \tag{4}$$

where $P_i := p_1 + p_2 + \ldots + p_i$ and thus $P_k = 1$. According to [7], $R(n)$ remains approximately constant over wide ranges of n. Assuming $R(n) \simeq R$, $p_i \simeq \frac{1}{k}$, and $k \gg 1$ we have

$$\begin{aligned} E[M] &\simeq \left(1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k}\right) R \\ &\simeq (\ln k + C) R \end{aligned} \tag{5}$$

where $C = 0.577\ldots$ denotes the Euler constant.

From this analysis we see that the number of possible feedback values has an impact on the expected number of feedback messages. For a responder set with a real-valued feedback parameter this results in $E[M] \simeq \ln(n) R$.

4.2 Class-based Extremum Detection

Although this logarithmic increase is well acceptable for a number of applications, the algorithm's properties can easily be further improved by the introduction of feedback classes. Within those classes no differentiation is made between different feedback values. Note that it is not necessary to choose a fixed size for all classes. The class size can be adapted to the required granularity for certain value ranges. In case a fixed number of classes is used, the expected number of feedback messages increases only by a constant factor over normal exponential feedback. This increase is expectedly observed in the simulation results shown in Figure 1. As the number of classes approaches the number of receivers, the increase in feedback messages follows

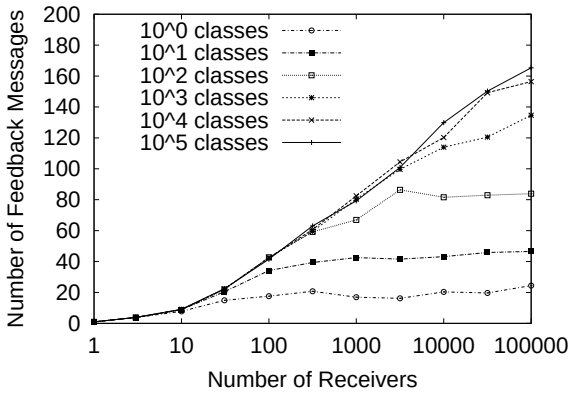


Figure 1: Uniformly sized classes

more and more the logarithmic increase for real-valued feedback as stated in Equation (5). For all simulations in this paper we use the parameters $N = 100,000$ and $T = 4\tau$ and average the results over 200 simulation runs, unless stated otherwise.

By adjusting the classes' positions depending on the actual value distribution, the number of classes required to cover the range of possible feedback values can be reduced without increasing the intervals' actual size. Thereby, the granularity of the feedback suppression (i.e., to what extent less optimal values can suppress better values) remains unchanged while the number of feedback messages is reduced.

Figure 2 gives a schematic overview of this mechanism. The first diagram shows the classless version of the feedback algorithm. Here, each time a feedback message v_i is sent the suppressed value range (shaded area) increases to $[0; v_i]$. A total of four feedback messages is sent in this example. The second diagram shows the same distribution of feedback for the case of static classes. Without loss of generality we assume equally sized classes of size δ and $v_1 \in [0; \delta]$ for this example. After receipt of the first feedback message v_1 the entire range $[0; \delta]$ of the lowest feedback class is suppressed. Only when a value outside this class is to be reported another message is sent, resulting in three feedback messages in total. The third diagram shows the case of dynamically adjusted classes. Upon receipt of the first feedback message v_1 the suppression limit is immediately raised to $v_1 + \delta$ and thus the value range $[0; v_1 + \delta]$ is now being suppressed. Through this mechanism feedback is reduced to only two messages.

With the above considerations, an elegant way to introduce feedback classes is the modification of Algorithm 2 to suppress feedback not only upon receipt of values strictly larger than the own value v but also upon receiving values $v' \geq (1 - q)v$, resulting in an adaptive feedback granularity dependent on the absolute value of the optimum.

Algorithm 3 *Let q be a tolerance factor with $q \in [0; 1]$. Modify Algorithm 2 such that a responder with value v cancels its timer if another responder has already sent feedback for value v' with $v' \geq (1 - q)v$.*

For $q = 0$ the algorithm is equivalent to Algorithm 2, whereas for $q = 1$ we obtain Algorithm 1.

Assuming the values v_i to be evenly distributed between rv_{max} and v_{max} $(0 < r < 1)$ we have approximately k feedback classes[3], where $k < \frac{\ln r}{\ln 1-q}$. For a value range $0 < v_i < 1$ we can assume $k < \frac{-\ln n}{\ln 1-q}$, thus setting r inversely proportional to the number of receivers since the receiver set is too small to cover the whole range of possible values.

Approximating further with

$$p_i = \frac{(1-q)^{i-1} - (1-q)^i}{1-r} = q\frac{(1-q)^{i-1}}{1-r}$$

and

$$P_i = \frac{1-(1-q)^i}{1-r}$$

we have

$$E_{max}[M] \quad < \quad qR \sum_{i=1}^{k} \frac{(1-q)^{i-1}}{1-(1-q)^i} \tag{6}$$

The mechanism strongly benefits from the feedback classes being more densely populated near the maximum than the classes near the minimum, resulting in a much lower expected number of feedback messages than the previous algorithm. Note that for small r the number of members with $v < (1-q)^{-1}r$ can be very small. Eventually, these feedback classes will contain only a single member and we therefore loose the desired suppression effect that leads to a sub-logarithmic increase of feedback messages. In maximum search this effect cannot be observed since already a single response in the larger feedback classes near the maximum will suppress all feedback from the potentially large number of small classes.

To demonstrate the effect we will calculate the expected number of feedback messages for a minimum search scenario: The feedback values v_i are again evenly distributed between rv_{max} and v_{max}, but in contrast to Algorithm 3 a responder cancels its timer if a response with $v' \leq (1-q)v$ was received. The algorithm produces the minimal value of the group within a factor of q. Note that as far as the expected number of feedback messages is concerned, this mechanism is equivalent to a maximum search with small class sizes for classes close to the optimum.

The feedback classes are in the opposite order as compared to our previous calculation.

$$p_i = \frac{(1-q)^{k-i} - (1-q)^{k-i+1}}{1-r} = q\frac{(1-q)^{k-i}}{1-r}$$

and

$$P_i = \frac{(1-q)^{k-i} - (1-q)^k}{1-r}$$

Thus

$$\begin{aligned} E_{min}[M] \quad &< \quad qR \sum_{i=1}^{k} \frac{1}{1-(1-q)^i} \\ &\approx \quad (1-q)\, E_{max}[M] + kqR \end{aligned} \tag{7}$$

[3] We assume the parameter range $(r, 1)$ to be fully covered by the feedback classes which is not strictly the case for this algorithm. This approximation thus overestimates the expected number of feedback messages.

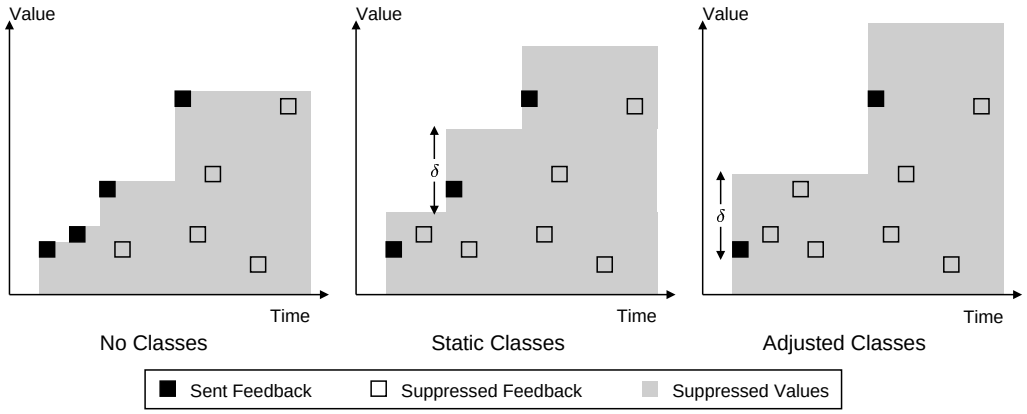


Figure 2: Class-based suppression with variable class position

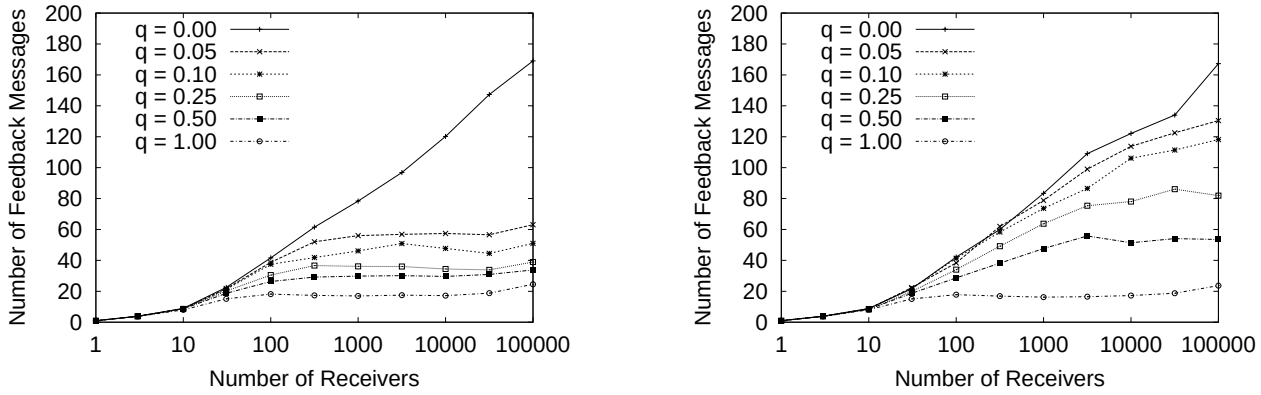


Figure 3: Number of feedback messages for maximum search (left) and minimum search (right)

Hence, for small r (large k) the sum is significantly larger than in the previous case.

Both scenarios have been simulated with various values for q. The sub-logarithmic increase of feedback messages is seen in both plots shown in Figure 3. But only in the maximum search case where the feedback-classes near the search goal are densely populated the strong class-induced suppression dominates the $\ln(n)$ scale-effect.

Numeric values for the upper limits on the expected number of feedback messages in both scenarios can be obtained from Equations (6) and (7). Some example values are shown in Table 1. These limits match well with the results of our simulations.

q	0.05	0.10	0.25	0.50	1.00
Maximum search	3.64	3.00	2.19	1.59	1.00
Minimum search	7.91	7.00	5.64	3.80	1.00

Table 1: Upper limits (as factor of R) for the expected number of feedback messages ($r = 10^{-2}$)

As mentioned before, Algorithm 3 guarantees a maximum deviation from the true optimum of a factor of q. It is worthwhile to note that this factor really is an upper bound on the deviation. Almost always the reported values will be much closer to optimal since the sender can choose the best one of all the responses given. The deviation of the best reported value from the optimum for different tolerance factors q is depicted in Figure 4. On average, with normal exponential suppression (i.e., $q = 100\%$) the best reported value lies within 10% of the optimum, for $q = 50\%$ the deviation drops to less than 0.15%, for $q = 10\%$ we obtain less than 0.02% deviation, etc. Thus, even for relatively high q with consequently only a moderate increase in the number of feedback messages, the best feedback values have only a marginal deviation from the optimum.

5 Biased Feedback

The previously described algorithms yield considerable results for various cases of extremum detection. However,

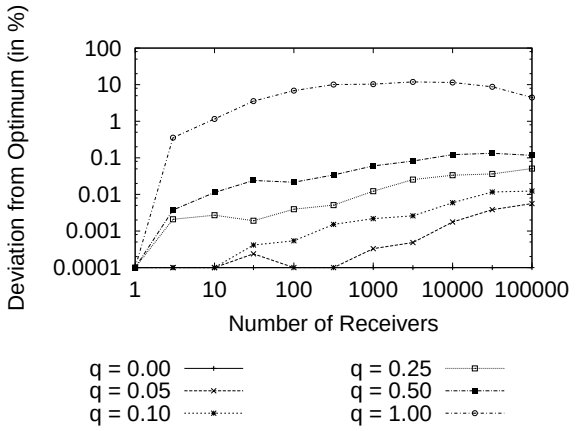


Figure 4: Feedback quality with different tolerance values

they will not affect the expected value of the first feedback message but only result in improved expected values for subsequent messages. In certain cases, the algorithms can be further improved by biasing the feedback timers. Increasing the probability that $t_1 < t_2$ if $v_1 > v_2$ results in better feedback behaviour but we must carefully avoid a feedback implosion for cases where many large values are present in the responder group.

If we know the probability distribution of the values v we can achieve our goal of minimizing the number of responses by the following algorithm:

Algorithm 4 *Let $P(v) = P(v' < v)$ be the probability distribution function of the values v within the group of responders. We follow Algorithm 1, but instead of drawing a random number we set the feedback time directly to*

$$t = T \max(0; 1 + \log_N(1 - P(v)))$$

Clearly, duplicate feedback responses are now only due to network latency effects since the maximal responder is guaranteed to have the earliest response time. However, the feedback latency is strongly coupled to the actual set of feedback values. Moreover, if the probability distribution of this specific set does not match the distribution used in the algorithm's calculation, feedback implosion is inevitable. For this reason, Algorithm 4 should only be used if the distribution of feedback values is well known for each individual set of values.

The latter condition is crucial. In general, it does *not* hold for values from causally connected responders. Consider for example the loss rate for multicast receivers: if congestion occurs near the sending site, all receivers will experience a high rate of packet loss simultaneously. Since the time-average distribution does not show this coherence effect the algorithm presented above will produce feedback implosion, if used to solicit responses from high-loss receivers. Due to this effect, the application of this simple mechanism is quite limited. It can be used, for example, with application level values where no coherence is generated within the network.

A simple way to adopt the key idea of value-based feedback bias is to mix value-based response times with a random component. This mechanism can be applied in various cases where coherence effects prohibit the application of algorithm 4. Let us study an example:

Algorithm 5 *Apply Algorithm 1 but modify the feedback time to*

$$\begin{aligned} t &= T \max(0; (1 - v) + v(1 + \log_N x)) \\ &= T \max(0; (1 + \log_N x^v)) \end{aligned} \quad (8)$$

Here, the feedback time consists of a component linearly dependent on the feedback value and a component for the exponential feedback suppression. The feedback time t is increased in proportion to decreasing feedback values v and a smaller fraction of T is used for the actual suppression. As long as at least one responder has a sufficiently early feedback time to suppress the majority of other feedback this distribution of timer values greatly decreases the number of duplicate responses while at the same time increasing the quality of the feedback (i.e., the best reported value with respect to the actual optimum value of the receiver set). Furthermore, in contrast to pure extremum detection algorithms this mechanism improves the expected feedback value of the first response as well as subsequent responses.

However, the feedback suppression characteristics of the above mechanism still depend at least to some extent on the value distribution at the receivers. Some extreme cases such as $v = 0$ for all receivers will always result in a feedback implosion. A more conservative approach is to not combine bias and suppression but use a purely additive bias.

Algorithm 6 *Apply Algorithm 1 but modify the feedback time to*

$$t = T \max(0; \gamma(1 - v) + (1 - \gamma)(1 + \log_N x)) \quad (9)$$

with $\gamma \in [0; 1]$.

To retain the same upper bound on the maximum feedback delay, it is necessary to split up T and use a fraction of T to spread out the feedback with respect to the response values and the other fraction for the exponential timer component. As long as $(1 - \gamma)T$ is sufficiently large compared to the network latency τ, an implosion as in the above example is no longer possible.

To better demonstrate the characteristics of these modifications, Figures 5 to 7 show how the feedback time changes with respect to response values compared to normal unbiased feedback according to Algorithm 1. A single set of

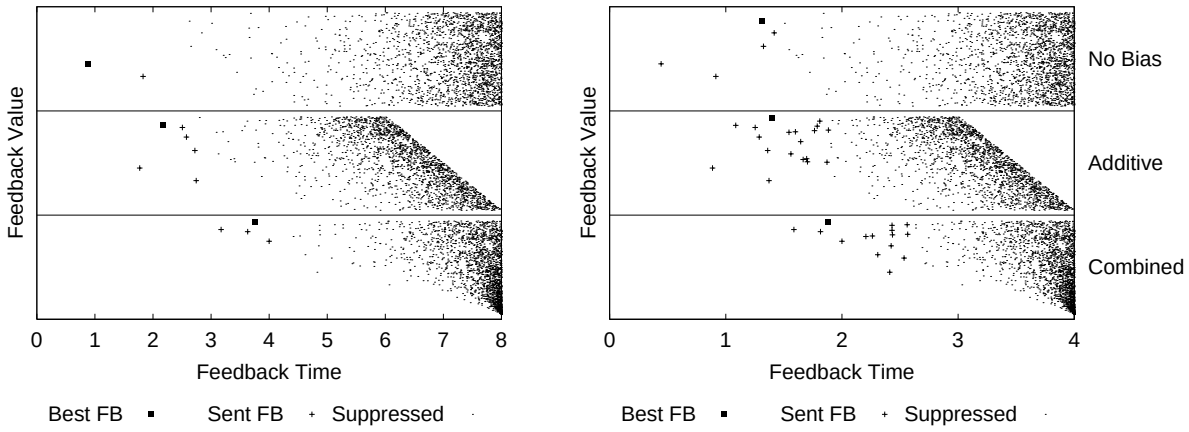


Figure 5: Feedback time and value (uniform distribution of values)

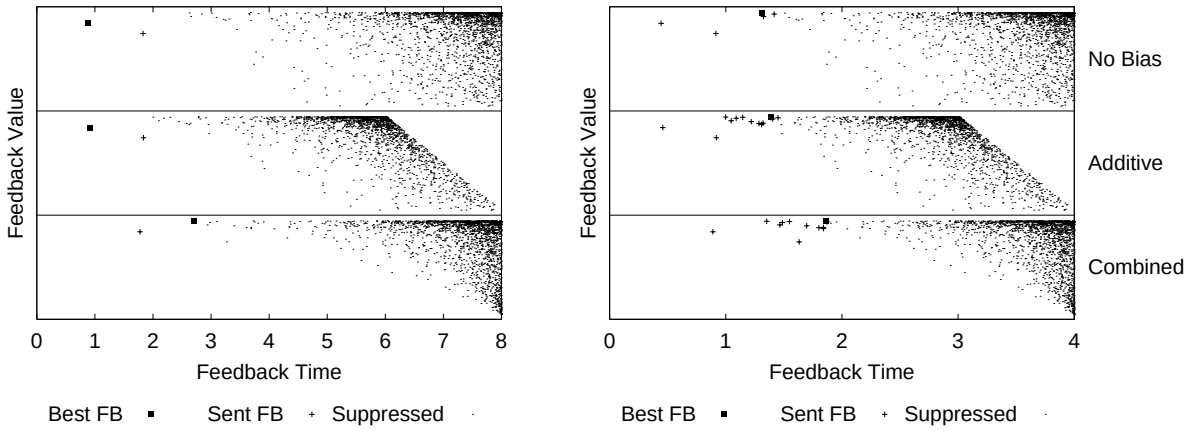


Figure 6: Feedback time and value (exponential distribution of values)

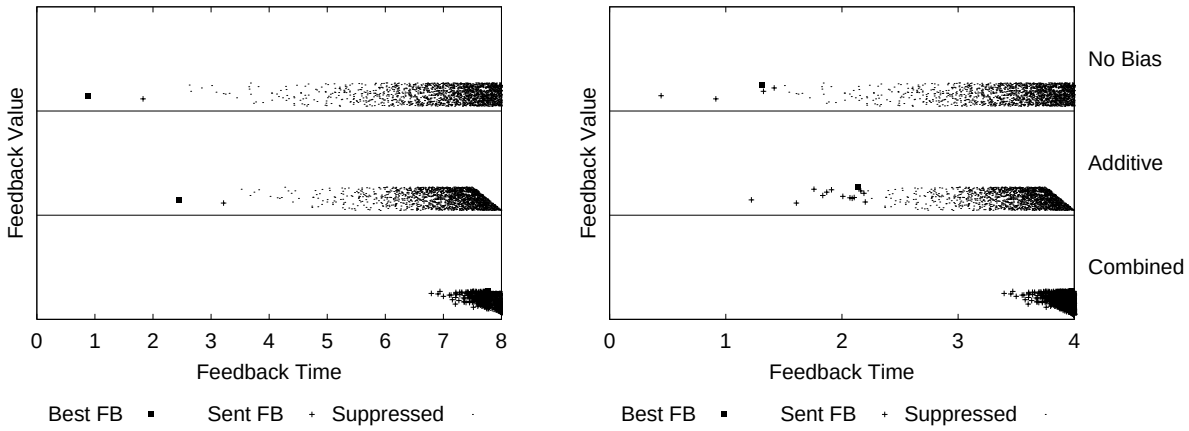


Figure 7: Feedback time and value (truncated uniform distribution of values)

random variables was used for all the simulations to allow a direct comparison of the results. For the simulations, the parameters N and n were set to $10,000$ and $2,000$ respectively.[4] In these simulations we do *not* consider maximum search but only how feedback biasing affects the distribution of feedback timers. Thus, to isolate the effect of feedback biasing, only a single feedback class was used such that the first cancellation notification suppresses *all* subsequent feedback. All simulations were carried out with $T = 4\tau$ as well as $T = 8\tau$ to demonstrate the impact of the feedback delay on the number of feedback responses. Each graph shows the feedback times in τ of the receiver set along the x-axis and the corresponding response values on the y-axis for each of the three feedback mechanisms *no bias* (Algorithm 1), *combined bias* (Algorithm 5), and *additive bias* (Algorithm 6) with $\gamma = 1/4$. Suppressed feedback messages are marked with a dot, feedback that is sent is marked with a cross, and the black square indicates which of these feedback messages had a value closest to the actual optimum of the receiver set.

In the graphs in Figure 5, the response values of the receivers are uniformly distributed. When no feedback bias is used, the first response that suppresses the other responses is random in value. In contrast, both feedback biasing methods result in the best reported feedback value being very close to the actual optimum. The number of sent feedback messages is higher with the two biasing methods since a smaller fraction of T is used for feedback suppression. Naturally, the number of feedback messages also increases when T is smaller (as depicted in the right graph).

In Figure 6, the same simulations were carried out for an exponential distribution of response values with a high probability of being close to the optimum. (When a reversed exponential distribution with most values far from the optimum is used, the few good values suppress all other feedback and again a feedback implosion is always prevented.) As can be seen from the graph, feedback suppression works well even when the actual distribution of response values is no longer uniform. For a uniform as well as an exponential distribution of response values, the *combined bias* suppression method results in fewer feedback messages while maintaining the same feedback quality.

However, as mentioned before, combining bias and suppression permits a feedback implosion when the range of feedback values is smaller than anticipated. In this case, the bias results in an unnecessary delaying of feedback messages, thus reducing the time that can be used for feedback suppression. In Figure 7, the response values are distributed uniformly in $[0; 0.25]$ instead of $[0; 1]$. For $T = 4\tau$, the time left for feedback suppression is τ, resulting in a scenario where no suppression is possible and each receiver will send feedback. Even when $T = 8\tau$ and thus a time of 2τ can be used for the feedback suppression, the number of feedback messages is considerably larger than in simulations with an additive bias. The exact numbers for the feedback responses of the three methods are given in Table 2.

[4]Note that using $n = N = 10,000$ instead of $n = 2,000$ would *reduce* the probability of an implosion since the probability that one early responder suppresses all others increases.

Feedback Time	No Bias	Additive	Combined
T=4, Uniform	5	19	15
T=4, Exponential	5	14	12
T=4, Truncated	5	14	2000
T=8, Uniform	2	6	4
T=8, Exponential	2	2	2
T=8, Truncated	2	2	334

Table 2: Number of responses with the different biasing methods

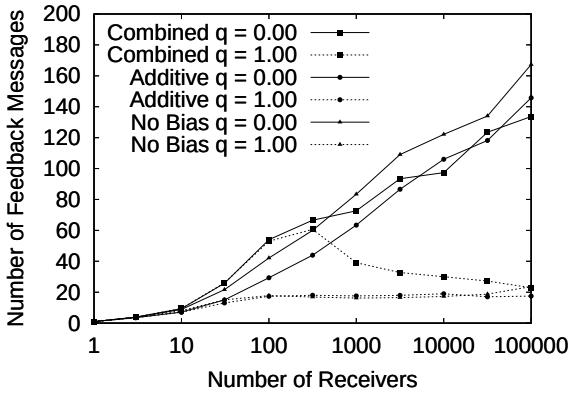


Figure 8: Number of responses with feedback biasing

For suppression to be effective, the amount of time reserved for the exponential distribution of the feedback timers should not be smaller than 2τ. Thus, the feedback implosion with Algorithm 5 can be prevented by bounding v such that $vT > 2\tau$ (i.e., using $v' = \max(v; 2\tau/T)$ instead of v in Equation (8)).

The outcome of a single experiment is not very representative since the number of feedback messages is extremely dependent on the feedback values of the early responders. As for the previously discussed feedback mechanisms, we depict the number of feedback messages for combined and additive bias averaged over 200 simulations in Figure 8.

The main advantage of the feedback bias is that the expected response value for early responses is improved. This not only reduces the time until close to optimal feedback is received (with unbiased feedback and class-based suppression, close to optimal feedback is likely to arrive at the

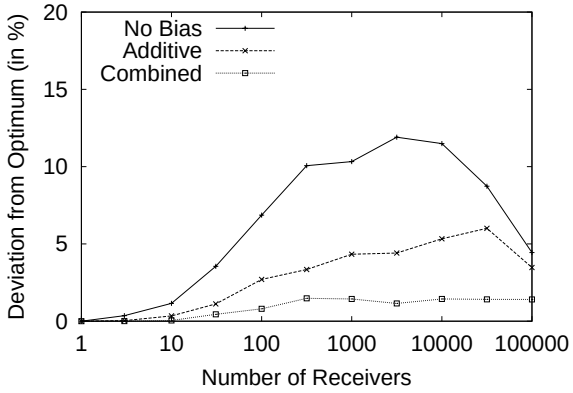


Figure 9: Deviation of best responses value from optimum

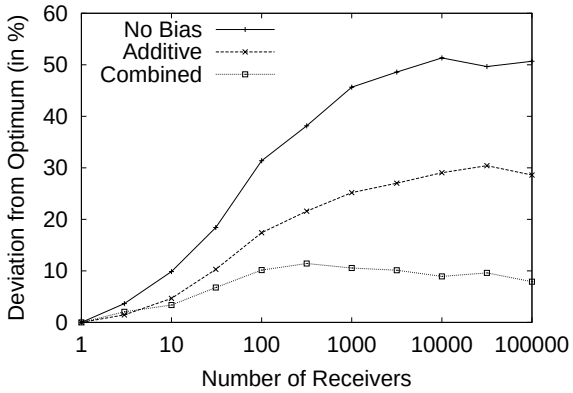


Figure 10: Deviation of first response value from optimum

end of a feedback round) but also reduces the number of responses with less optimal feedback.

Figure 9 shows how the feedback quality improves compared to the normal exponential feedback suppression when biasing the feedback timer. The maximum deviation is reduced from about 15% to 6% for additive bias and to less than 2% for combined bias.

While a similar increase in feedback quality can be achieved by using feedback classes (at the expense of an increased number of feedback messages) only with a feedback bias is it possible to improve the quality of the *first* feedback message. In case a close to optimal value is needed very quickly, using either Algorithm 5 or Algorithm 6 can be beneficial. Figure 10 depicts the average deviation of the value of the first feedback message from the optimum. Here, the increase in quality is much more obvious than in the previous case. With all unbiased feedback mechanisms, the first reported value is random and thus the average deviation is 50% (for large enough n) whereas the combined and the additive biased feedback mechanisms achieve average deviation values around 10% and 30% respectively.

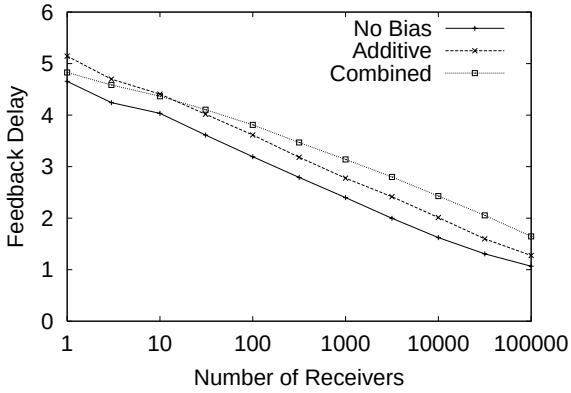


Figure 11: Average feedback delay

Lastly, the expected delay until the first feedback message is received is of concern. While all mechanisms adhere to the upper bound of T, feedback can be expected earlier in most cases. In Figure 11 we show the average feedback delay for biased and unbiased feedback mechanisms. For all algorithms the feedback delay decreases logarithmically for an increasing number of receivers. The exact run of the feedback curve depends on the amount of time used for suppression. For this reason, unbiased feedback delay drops faster than biased feedback, since a bias can only delay feedback messages compared to unbias feedback. In case the number of receivers is estimated correctly (i.e., $n = N$), the feedback delay for unbiased feedback drops to τ, the minimum delay possible for such a feedback system. Biased feedback delay is slightly higher with approximately 1.5τ.

6 Conclusions

In this paper we presented mechanisms that improve upon the well-known concept of exponential feedback suppression in case feedback of some extreme value of the group is needed. We discuss two orthogonal methods to improve the quality of the feedback given. If no information is available about the distribution of the values at the receivers, a safe method to obtain better feedback is to modify the suppression mechanism to allow the sending of high valued feedback even after a receiver is notified that a different receiver already gave feedback. We give exact bounds for the expected increase in feedback messages for a given improvement in feedback quality. If more information about the distribution of feedback values is available or certain worst-case distributions are very unlikely, it is furthermore possible to bias the feedback timer. The better the feedback value the earlier the feedback is sent, thus suppressing later feedback with less optimal values. The modified suppression mechanism and the feedback biasing can be used in

combination to further improve the feedback process.

The mechanisms discussed in this paper have been included in the TCP-friendly Multicast Congestion Control Protocol (TFMCC) [13]. It uses class-based feedback cancellation as well as feedback biasing to determine the current limiting receiver (i.e., the receiver with the lowest expected throughput of the multicast group). The protocol depends on short feedback delays in order to quickly respond to congestion. Selecting the correct receiver as current limiting receiver is critical for the functioning of the protocol since a wrong choice may compromise the TCP-friendly characteristics of TFMCC. In that sense, the feedback mechanism is an important part of the TFMCC protocol.

Extremum feedback is not yet included in any other application, but we believe a number of applications can benefit from it.

6.1 Future Work

In the future we would like to continue this work in several directions.

Most applications need to consider only one type of feedback value. Nevertheless, it may sometimes be useful to get multivalued feedback, for example to monitor some critical parameters of a large network, where changes in each of the parameters are equally important. It may not always be possible to aggregate different types of values to one single "ranking" value. In this case, a multivalued feedback mechanism clearly has better suppression characteristics than separate feedback mechanisms for each of the relevant values.

Another important step will be the combination of knowledge about the value distribution within the responder group with implosion avoidance features. Several mechanisms to estimate the size of the receiver set from the feedback time and the number of feedback messages with exponential feedback timers have been proposed [9, 6]. Combining such estimation methods with extremum feedback, it should be possible to estimate the distribution of response values at the receivers in case this distribution is not known. For continuous feedback, this knowledge can then be used to generate feedback mechanisms based on Algorithm 4.

Taking these considerations one step further, in some cases the maximum change of the relevant state during one feedback round is bounded. For example, in the case of TFMCC, the measurements to determine round-trip time and loss event rate are subject to smoothing, thus limiting the maximum rate increase and decrease per round-trip time. In case (partial) information about the current distribution of feedback values is known (e.g., from the previous feedback round), it is possible to infer the worst case distribution of the next feedback round. This allows to further improve the feedback algorithm by tailoring it to the specific distribution.

7 Acknowledgements

Thomas Fuhrmann would like to thank Ernst Biersack for supporting this work by hosting him during a sabbatical stay at Eurecom.

References

[1] BASU, A., AND GOLESTANI, J. Architectural issues for multicast congestion control. In *International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)* (June 1999).

[2] BOLOT, J.-C., TURLETTI, T., AND WAKEMAN, I. Scalable feedback control for multicast video distribution in the Internet. *Proceedings of the ACM SIGCOMM* (Sept. 1994), 58 – 67.

[3] DONAHOO, M. J., AND AINAPURE, S. R. Scalable multicast representative member selection. In *IEEE INFOCOM* (March 2001).

[4] FENNER, B., HANDLEY, M., HOLBROOK, H., AND KOUVELAS, I. Protocol independent multicast - sparse mode (pim-sm): Protocol specification, Nov. 2000. Internet draft draft-ietf-pim-sm-v2-new-01.txt, work in progress.

[5] FLOYD, S., JACOBSON, V., LIU, C., MCCANNE, S., AND ZHANG, L. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM Transactions on Networking 5*, 6 (Dec. 1997), 784 – 803.

[6] FRIEDMAN, T., AND TOWSLEY, D. Multicast session membership size estimation. In *IEEE Infocom* (New York, NY, Mar. 1999).

[7] FUHRMANN, T., AND WIDMER, J. On the scaling of feedback algorithms for very large multicast groups. *Computer Communications 24*, 5-6 (Mar. 2001), 539 – 547.

[8] HANDLEY, M. Session directories and scalable Internet multicast address allocation. In *Proc. ACM Sigcomm* (Vancouver, B.C., Canada, Sept. 1998), pp. 105 – 116.

[9] LIU, C., AND NONNENMACHER, J. Broadcast audience estimation. In *IEEE Infocom* (Tel Aviv, Israel, Mar. 2000), pp. 952–960.

[10] NONNENMACHER, J., AND BIERSACK, E. W. Scalable feedback for large groups. *IEEE/ACM Transactions on Networking 7*, 3 (June 1999), 375 – 386.

[11] RIZZO, L. pgmcc: A TCP-friendly single-rate multicast congestion control scheme. In *Proc. ACM SIGCOMM* (Stockholm, Sweden, August 2000), pp. 17 – 28.

[12] SCHULZRINNE, H., CASNER, S., FREDERICK, R., AND JACOBSON, V. Rtp: A transport protocol for real-time applications. *RFC 1889* (January 1996).

[13] WIDMER, J., AND HANDLEY, M. Extending equation-based congestion control to multicast applications. In *Proc. ACM SIGCOMM* (San Diego, CA, Aug. 2001). to appear.

A Proof of Equation (2)

We will show that

$$\int_{1/N}^{1} \frac{(1-x)^n}{x}\,dx < \ln N - \ln n - C + \frac{n}{N}$$

Therefore, we first prove

$$-\sum_{k=1}^{n} \binom{n}{k} \frac{(-1)^k}{k} = \sum_{k=1}^{n} \frac{1}{k} \tag{10}$$

using $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$, $\binom{n-1}{k-1}\frac{1}{k} = \frac{1}{n}\binom{n}{k}$, and complete induction. Clearly Equation (10) holds for $n = 1$. Then

$$\begin{aligned}
&-\sum_{k=1}^{n} \binom{n}{k} \frac{(-1)^k}{k} \\
&= -\sum_{k=1}^{n-1} \binom{n-1}{k} \frac{(-1)^k}{k} - \sum_{k=1}^{n-1} \binom{n-1}{k-1} \frac{(-1)^k}{k} - \frac{(-1)^n}{n} \\
&= \sum_{k=1}^{n-1} \frac{1}{k} - \frac{1}{n} \sum_{k=1}^{n-1} \binom{n}{k} (-1)^k - \frac{(-1)^n}{n} \\
&= \sum_{k=1}^{n-1} \frac{1}{k} + \frac{1}{n} - \frac{1}{n} \sum_{k=0}^{n} \binom{n}{k} (-1)^k \\
&= \sum_{k=1}^{n} \frac{1}{k}
\end{aligned}$$

This result is approximated from below by $\ln n + C$ where $C = 0.577\ldots$ is the Euler constant.

Thus, we have

$$\begin{aligned}
&\int_{1/N}^{1} \frac{(1-x)^n}{x}\,dx \\
&= -\int_{1/N}^{1} \sum_{k=0}^{n} \binom{n}{k} (-x)^{k-1}\,dx \\
&= \left[\ln x + \sum_{k=1}^{n} \binom{n}{k} \frac{(-x)^k}{k}\right]_{1/N}^{1} \\
&= \ln N + \sum_{k=1}^{n} \binom{n}{k} \frac{(-1)^k}{k} - \sum_{k=1}^{n} \binom{n}{k} \frac{(-1/N)^k}{k} \\
&< \ln N - \ln n - C + \frac{n}{N} - \sum_{k=2}^{n} \binom{n}{k} \frac{(-1/N)^k}{k} \\
&< \ln N - \ln n - C + \frac{n}{N}
\end{aligned}$$

For a proof of Equation (3) we refer the reader to [7].

B Additional Simulation Graphs

On the following pages we present a more comprehensive overview of the graphs created from the simulations. However, we will not discuss each graph in detail.

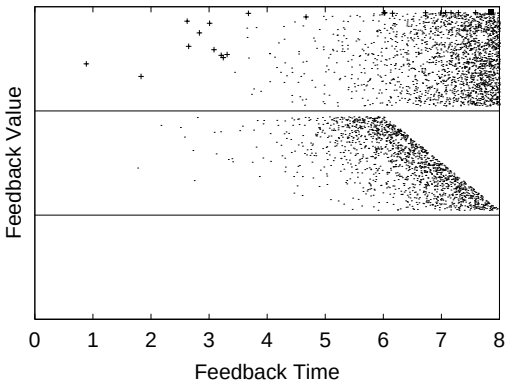
In Figures 12-14, we show the results of simulations similar to the ones depicted in Figures 5-7. with tolerance value of $q = 0.0$ instead of $q = 1.0$. Again, we also include Table 3 for the exact numbers of feedback messages in the experiments.

Feedback Time	No Bias	Additive	Combined
T=4, Uniform	56	65	41
T=4, Exponential	56	85	52
T=4, Truncated	56	78	2000
T=8, Uniform	22	21	14
T=8, Exponential	22	28	21
T=8, Truncated	22	28	416

Table 3: Number of responses ($q = 0.0$)

All other figures consist of four graphs each, depicting the average number of feedback messages, the feedback time, the feedback quality of the best feedback obtained, and the feedback quality of the first feedback message. Only for the first minimum search and maximum search graphs do we plot curves for intermediate values of q. All other graphs only show curves for $q = 0$ and $q = 1$. The following algorithms were simulated:

- uniformly sized feedback classes (Figure 15)
- unbiased feedback suppression for minimum search (Figure 16)
- unbiased feedback suppression for maximum search (Figure 17)
- feedback suppression for minimum search with combined bias (Figure 18)
- feedback suppression for minimum search with additive bias and $\gamma = 0.25$ (Figure 19)
- feedback suppression for minimum search with additive bias and $\gamma = 0.5$ (Figure 20)
- unbiased feedback suppression for minimum search with a twice as large T (Figure 21)
- feedback suppression for minimum search with additive bias, $\gamma = 0.5$ and a twice as large T (Figure 22)


Feedback Value
0
1
2
3
4
5
6
7
8
Feedback Time

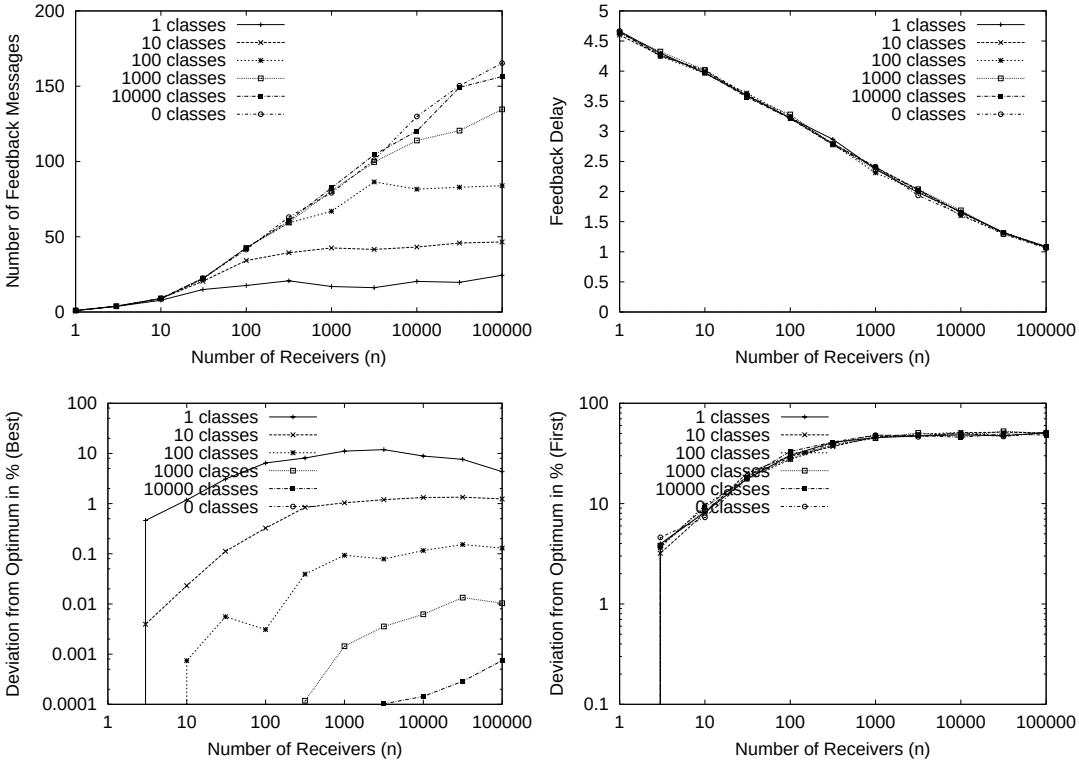


Figure 15: Uniformly sized feedback classes

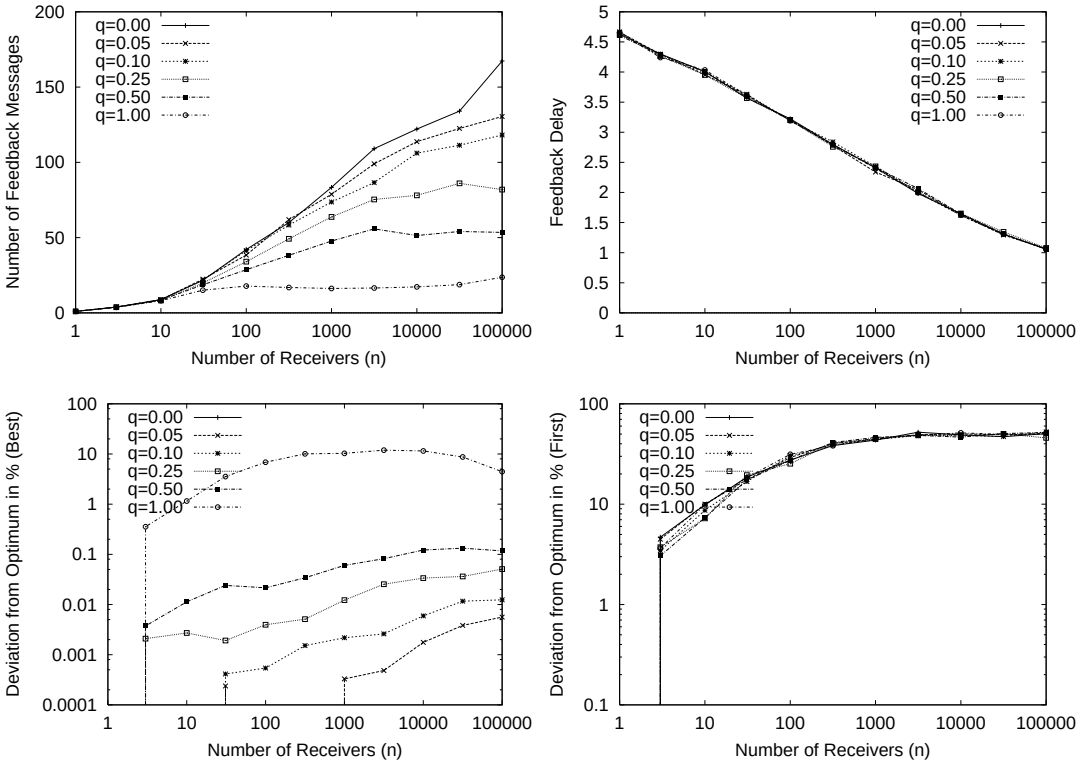


Figure 16: No bias, minimum search

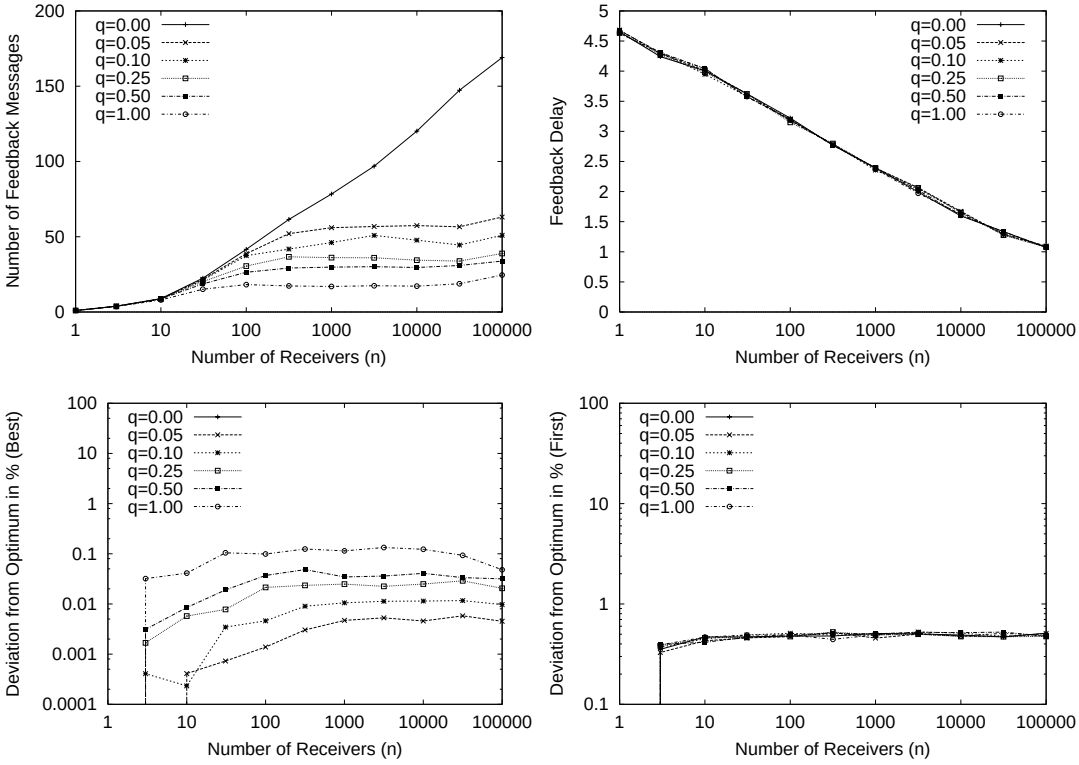


Figure 17: No bias, maximum search

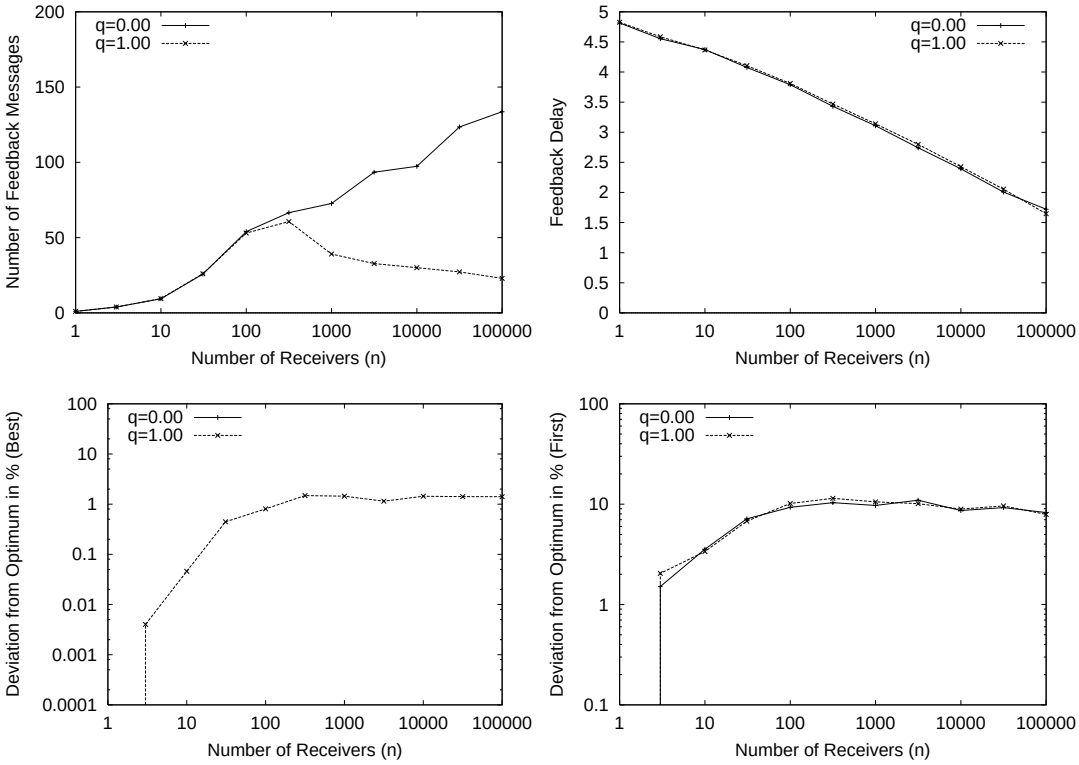


Figure 18: Combined bias, minimum search

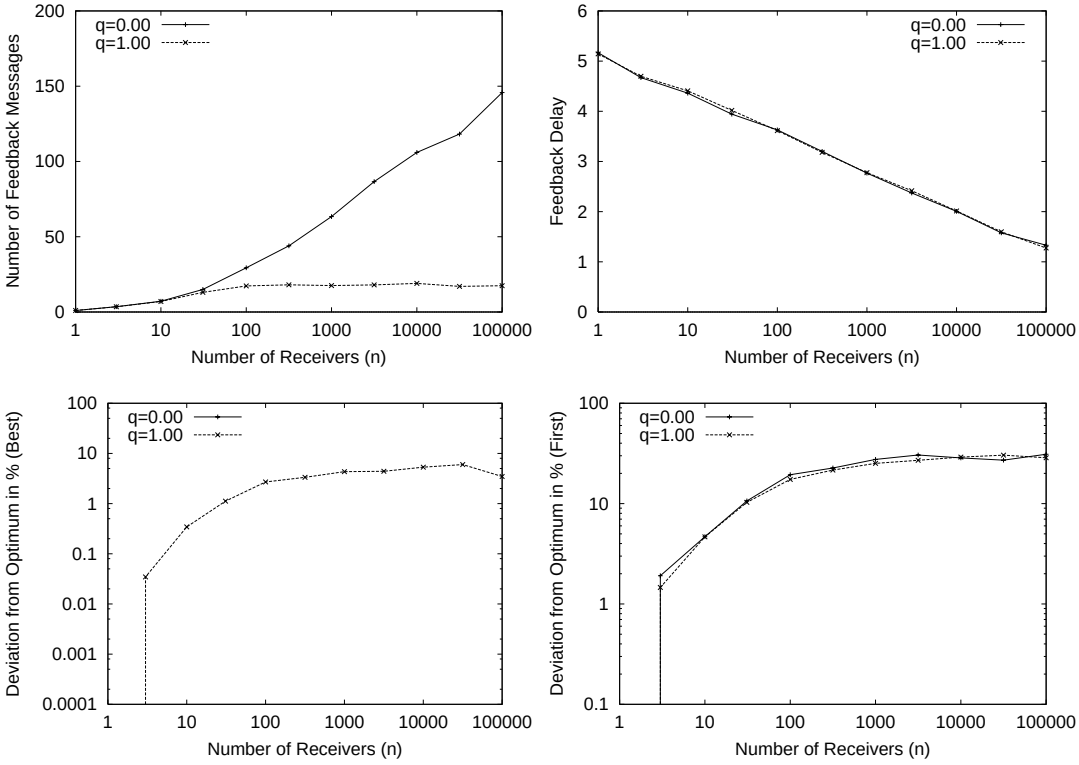


Figure 19: Additive bias, minimum search (γ = 0.25)

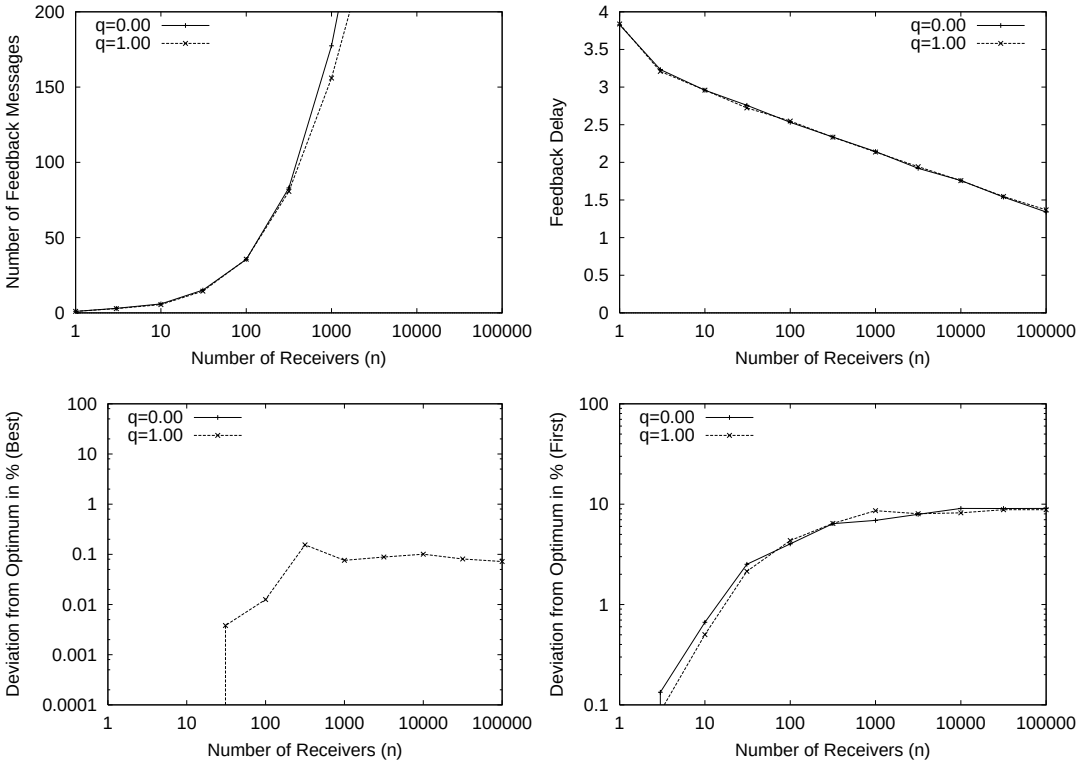


Figure 20: Additive bias, minimum search (γ = 0.5)

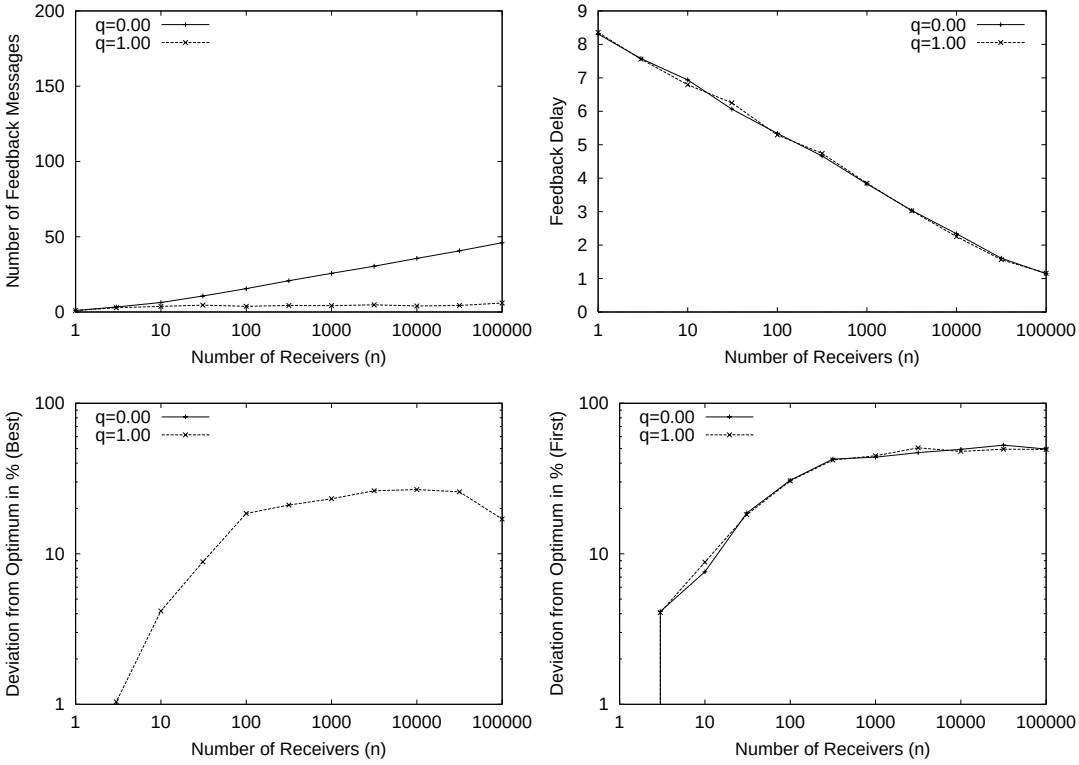


Figure 21: No bias, minimum search ($T = 8\tau$)

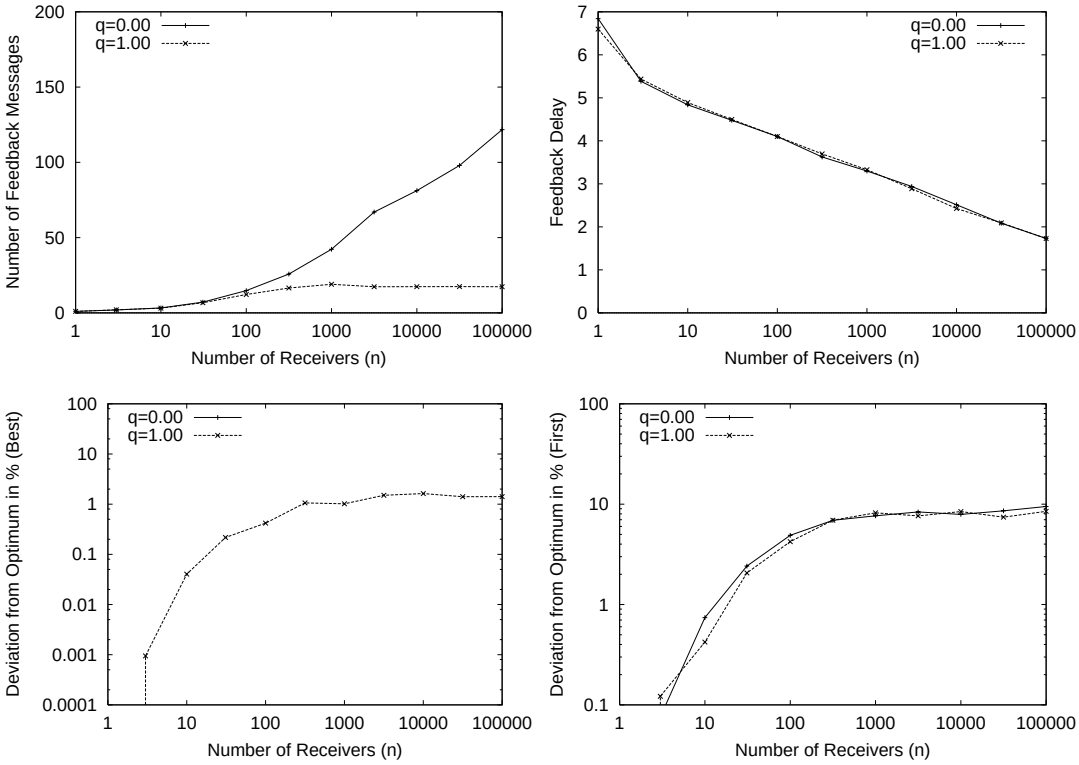


Figure 22: Additive bias, minimum search ($\gamma = 0.5, T = 8\tau$)