Improving the Scalability of an Application-Level Group Communication Protocol *

Ayman EL-SAYED Vincent ROCA
INRIA Rhône-Alpes, Planète project, France
http://www.inrialpes.fr/planete/people/elsayed|roca
ayman.elsayed|vincent.roca@inrialpes.fr

Abstract

In this paper we analyze the scalability of a control protocol providing a user-level group communication service. This control protocol, called HBM (Host Based Multicast), can be used when native multicast routing is not available. HBM is by nature centralized, everything being controlled by a single host, called Rendez-Vous Point, or RP. This feature naturally leads to scalability issues. Yet we show in this paper that scalability can very easily be improved. In particular the control information being exchanged between the group members and the RP can be rate limited not to exceed a given threshold. This is made possible by a careful modeling of the control traffic exchanges and is validated by many experiments carried out with our prototype.

1 Introduction

Group communication traditionally requires that each node at each site has access to a native multicast routing service. If intra-domain multicast (within a LAN or a site) is widely available, this is different for inter-domain multicast. Today many ISPs are still reluctant to provide a wide-area multicast routing service because of technical or marketing reasons [2].

Application-level multicast proposals [3][4][5][7] offer a practical solution to this problem. They enable every host to participate in multicast sessions efficiently, no matter whether is has access to native multicast or not. The HBM protocol [6] is one such protocol. HBM is by nature centralized, everything being controlled by a single host, called Rendez-Vous Point, or RP. This feature naturally leads to scalability issues. The goal of this paper is to show that scalability can be very easily improved, using one of several different approaches.

The remaining of the paper is organized as follows: section 2 introduces our HBM proposal; section 3 explains how the HBM scalability can be improved; section 4 gives an account of experiments that were carried out with our HBM implementation; results are discussed in section 5 and section 6 concludes this paper.

2 Introduction to the Host Based Multicast Protocol

This section quickly introduces the protocol and the scalability problems arisen. More details can be found in [6].

2.1 Protocol Description

Basic Idea: The HBM protocol automatically creates a virtual overlay topology between the various group members (sources and receivers), using point-to-point UDP tunnels between them. Everything is under the control of a single host, the *rendez-vous point* (or RP). This RP knows the members, their features, and the communication costs between them. He is responsible of the distribution topology calculation and its dissemination among group members.

Periodic topology update: A dynamic adaptation of the topology is required for several reasons:

- to reflect the changing networking conditions,
- because of new members joining the group, who are initially grafted on the existing topology in a sub-optimal way,
- after the departure of members, deliberately, after a crash, or because of a network failure,
- because recovery actions taken by the RP after a partitioning lead to a sub-optimal overlay topology.

Therefore (1) all the members periodically evaluate the new communication costs between them (or a subset of them, as we will see later) and inform the RP, and (2)

*0-7803-7661-7/03/$17.00 ©2003 IEEE.

the RP periodically calculates a new topology and informs each member. These two mechanisms are fully asynchronous.

Control messages: Several control messages are defined. In this paper we only consider *the metric update (or MU) messages*, sent by a member to the RP in order to update the RP's communication cost database, and the *topology update (or TU) messages*, sent by the RP to the members in order to inform them of the new topology. Since a TU message only contains the direct neighborhood, a different message is sent to each member.

All the control messages are textual and use an XML syntax (figures 1). Using XML offers major advantages in terms of simplicity, extensibility, debugging and logging capabilities. This is fully compliant with the more general trend where control information is textual (e.g. RTSP, SDP, HTTP, SMIL...), and data communications in binary format (e.g. RTP, TCP/IP protocols...). Note that compressing these messages is an immediate way of improving scalability in terms of bit rate at the expense of additional processing load, especially on the RP. *This optimization is NOT considered in this paper but could easily be applied.*

These control messages use a (Type/Length/Value) format and are carried out in dedicated member-to-RP TCP connections.

2.2 Discussion of the Scalability Aspects

The centralized nature of HBM naturally limits the scalability in terms of the number of simultaneous members in a session. The number and size of the metric update and topology update messages can quickly waste a significant amount of bandwidth on the network and processing power at the RP. Besides, the necessity for each member to know other members and to periodically evaluate the new communication costs with them also limits the scalability of the solution.

More generally we believe that any overlay multicast solution based on point-to-point communications and that tries to create "not too bad" topologies (note that gossiping solutions like [1] do not belong to this category) has scalability limitations, even if this is more acute in a centralized solution.

Yet we show in this paper that HBM scalability can be easily improved, using one of three different approaches, and that the control information exchanged between the group members and the RP can be rate limited. HBM and many other overlay multicast solutions target applications, like collaborative working environments, that only need a reasonable scalability, in the order of a few hundreds of members at most. Therefore our tests are limited to 200 participants. Anyway, a single HBM member per site is sufficient when intra-domain multicast is possible in this site, since this HBM member will hide all the internal members and behave as a reflector for them. Therefore, the true number of members in a session, as seen by the application, can be largely higher than the number of members known by HBM.

3 Improving the Scalability

After a detailed modeling of the control traffic overhead, this section introduces four strategies to increase the scalability of the HBM protocol.

3.1 Mathematical Model

3.1.1 Metric Update (MU) Message Incoming Rate

Let N be the number of members in the session, $T_{mu}(N)$ the metric update period at a member, $s_{mu}(N)$ the size of a single metric update message, s_{mu_header} the fixed size of a message header, $n_{rmu}(N)$ the number of records in each metric update message, each record being s_{rmu} bits long (assumed to be a constant). We assume that these parameters are the same for all members. The incoming rate, from the RP point of view, for all metric update messages, $R_{mu}(N)$, is given by:

$$\begin{aligned} R_{mu}(N) &= \frac{N * s_{mu}(N)}{T_{mu}(N)} \qquad (1) \\ with: \; s_{mu}(N) &= s_{mu\_header} + n_{rmu}(N) * s_{rmu} \end{aligned}$$

3.1.2 Topology Update (TU) Message Outgoing Rate

Let $T_{tu}(N)$ be the topology update period at the RP. Let n_l be the total number of links in the virtual topology. Since each member needs a link to get connected, it follows that $n_l = N - 1$. Having more links would create loops which is avoided in the present work (i.e. we do not take into account the possibility of having additional links for improved robustness as in [6]). Since each link is common to two members, a record for a link is sent twice, in two different topology update messages.

Let $s_{all\_tu}(N)$ be the size of all topology update messages sent after a topology update, s_{tu_header} the fixed size of a message header, s_{rtu} the size (assumed to be a constant) in bits of each record in each message. $s_{all\_tu}(N)$ is given by:

$$s_{all\_tu}(N) = N * s_{tu\_header} + 2 * n_l * s_{rtu}$$

```
<metricupdate>2                        # Message start for node id=2
 records=5                             # Number of metrics
<record>2, 1                           # metric between 2 and 1
<metric>10.10, 0.12</metric>           # RTT=10.10ms, loss=0.12%
</record>
<record>2, 3
<metric>1.60, 0.010</metric>
 </record>
<record>2, 4
<metric>30.10, 0.195</metric>
</record>
<record>2, 5
 <metric>2.50, 0.001</metric>
</record>
<record>2, 6
<metric>3.10, 0.012</metric>
</record>
</metricupdate>                        # Message end
```

(a) Metric Update (MU) message

```
<topologyupdate>2                      # Message start for node id =2
 records=3                             # Number of links =3
<record>2, 5                           # link between 2 and 5
 type=1                                # type of link = 1: tree link
groups=1                               # Number of groups in this link =1
<group>16843232, 1111</group>          # group ip=224.1.1.1, port = 1111
</record>
<record>2, 3
 type=1
groups=2
<group>16843232, 1111</group>          # group ip=224.1.1.1, port = 1111
<group>33620448, 2222</group>          # group ip=224.1.1.2, port = 2222
</record>
<record>2, 6
type=1
groups=1
<group>33620448, 2222</group>          # group ip=224.1.1.2, port = 2222
</record>
</topologyupdate>                      # Message end
```

(b) Topology Update (TU) message

Figure 1: An example of MU and TU control messages.

The outgoing rate, from the RP point of view, for all topology update messages, $R_{tu}(N)$, is given by:

$$R_{tu}(N) = \frac{s_{all\_tu}(N)}{T_{tu}(N)} \qquad (2)$$

3.1.3 Total Rate of Control Messages

It follows that the total rate, $R_{ctrl}(N)$, for all HBM control messages is the sum of $R_{mu}(N)$ and $R_{tu}(N)$:

$$\begin{aligned} R_{ctrl}(N) &= R_{mu}(N) + R_{tu}(N) \\ R_{ctrl}(N) &= \frac{N * s_{mu\_header} + N * n_{rmu}(N) * s_{rmu}}{T_{mu}(N)} + \\ &\quad \frac{N * s_{tu\_header} + 2 * (N-1) * s_{rtu}}{T_{tu}(N)} \end{aligned} \qquad (3)$$

3.2 Strategies to Reduce the Control Overhead

3.2.1 General ideas

In order to limit the control traffic overhead, $R_{ctrl}(N)$ must not be greater than a given threshold. This threshold can be either a fixed predefined value, or be expressed as a given percentage, α, of the sum of both the average data traffic rate, R_{data} and the control traffic rate, $R_{ctrl}(N)$. Let's consider this second case (RTP/RTCP do the same). Therefore:

$$\begin{aligned} R_{ctrl}(N) &\leq \alpha * (R_{ctrl}(N) + R_{data}) \\ R_{ctrl}(N) &\leq \frac{\alpha}{1-\alpha} * R_{data} = R_{ctrl\_max} \end{aligned} \qquad (4)$$

For a given $T_{mu}(N)$ and $T_{tu}(N)$, let $N_{n\_rmu\_max}$ be the maximum N such that having $n_{rmu} = N - 1$ records, i.e. the maximum, in each metric update message satisfy this constraint. $N_{n\_rmu\_max}$ is the solution of a second order equation: $R_{ctrl}(N) = R_{ctrl\_max}$ (see equation 3).

This value of N is a major threshold, since for groups having less than $N_{n\_rmu\_max}$ members, an optimal solution is feasible, each member generating messages containing the new communication cost to all other members, while the total control message rate remains below a maximum threshold.

On the contrary, *for groups having more than $N_{n\_rmu\_max}$ members, specific measures must be taken in order to satisfy equation 4.* Several strategies are possible, depending on the protocol parameters we play upon:

Strategy 1 (none) is the reference, and *does not* include any optimization. Therefore, as N grows, the constraint of equation 4 is not necessarily fulfilled.

strategy 2 limits the number of records, $n_{rmu}(N)$, in a metric update message. When $N > N_{n\_rmu\_max}$, this number of records is progressively reduced until a lower predefined bound is reached, n_{rmu_min}, at $N = N_{n\_rmu\_min}$. Then, for $N > N_{n\_rmu\_min}$ the number of records remains constant and equal to n_{rmu_min}.

strategy 3 limits the number of records too in a metric update message. The difference with strategy 2 is that when $N > N_{n\_rmu\_max}$, the number of records remains constant, n_{rmu_max}, rather than being reduced. In order to keep the total control message rate below the upper bound, the $T_{mu}(N)$ period is progressively increased when $N > N_{n\_rmu\_max}$.

strategy 4 also limits the number of records in each metric update message, but less aggressively than the previous two strategies. To compensate, the $T_{mu}(N)$ period is aggressively increased when $N > N_{n\_rmu\_max}$.

Therefore these strategies try to find a balance between the various protocol parameters (figure 2) by playing with the frequency and the size of the control messages. In some cases the messages are more frequent but contain less information, and vice-versa. Each of these strategies is now introduced with more details.

3.2.2 Strategy 1: Non optimized

The first strategy, not optimized, is controlled by the following equation set:

$$\begin{aligned} n_{rmu}(N) &= N-1 \\ T_{mu}(N) &= max(T_{mu}, T_{m\_eval} * (N-1)) \\ T_{tu}(N) &= T_{tu} \end{aligned}$$

where T_{m_eval} is the time (assumed constant) required to evaluate the communication cost between two members (section 3.3). Since the evaluation of $(N-1)$ metrics grows linearly with N, the $T_{mu}(N)$ period is only constant (T_{mu}) when $T_{m\_eval} * (N-1) \leq T_{mu}$, and then equal to $T_{m\_eval} * (N-1)$. Note that the same constraint applies to the three other strategies.

3.2.3 Strategy 2

The second strategy is controlled by the following equation set:

$$n_{rmu}(N) = \begin{cases} N-1 & \text{if } N \leq N_{n\_rmu\_max} \\ n_{rmu\_min} & \text{if } N > N_{n\_rmu\_min} \\ n_{rmu\_max} * \frac{N_{n\_rmu\_max}}{N} & \text{otherwise} \end{cases}$$

$$T_{mu}(N) = \begin{cases} max(T_{mu}, T_{m\_eval} * (N-1)) & \text{if } N \leq N_{n\_rmu\_max} \\ T_{m\_eval} * n_{rmu\_max} * \frac{N}{N_{n\_rrm\_min}} & \text{if } N > N_{n\_rmu\_min} \\ T_{m\_eval} * n_{rmu\_max} & \text{otherwise} \end{cases}$$

$$T_{tu}(N) = \begin{cases} T_{tu} & \text{if } N \leq N_{n\_rmu\_max} \\ T_{tu} * \frac{s_{tu}(N)}{s_{tu}(N_{n\_rmu\_max})}) & \text{if } N > N_{n\_rmu\_max} \end{cases}$$

This strategy essentially consists in a $n_{rmu}(N)$ adaptation. The $T_{mu}(N)$ period is adapted for the reasons explained in section 3.2.2. The $T_{tu}(N)$ period also needs an adaptation. Indeed, as N grows, more topology update messages are required (one per member), and this is not caught with a $n_{rmu}(N)$ adaptation. Therefore the topology update period is slightly increased for groups larger than $N_{n\_rmu\_max}$ members.

An important question is: "what is an appropriate minimum number of records in a metric update message, n_{rmu_min}?". This parameter obviously depends on the maximum topology fanout (i.e. the maximum number of neighbors, which in turns depends on the topology creation algorithm). n_{rmu_min} must be at least equal to this maximum fanout to discover potential congestion problems on the topology. Some extra records with a random subset of the remaining members are then added to enable the RP to improve the topology.

3.2.4 Strategy 3

The third strategy is controlled by the following equation set:

$$n_{rmu}(N) = \begin{cases} N-1 & \text{if } N \leq N_{n\_rmu\_max} \\ n_{rmu\_max} & \text{if } N > N_{n\_rmu\_max} \end{cases}$$

$$T_{mu}(N) = \begin{cases} max(T_{mu}, T_{m\_eval} * (N-1)) & \text{if } N \leq N_{n\_rmu\_max} \\ T_{m\_eval} * n_{rmu\_max} * \frac{N}{N_{n\_rmu\_max}} & \text{if } N > N_{n\_rmu\_max} \end{cases}$$

$$T_{tu}(N) = \begin{cases} T_{tu} & \text{if } N \leq N_{n\_rmu\_max} \\ T_{tu} * \frac{s_{tu}(N)}{s_{tu}(N_{n\_rmu\_max})}) & \text{if } N > N_{n\_rmu\_max} \end{cases}$$

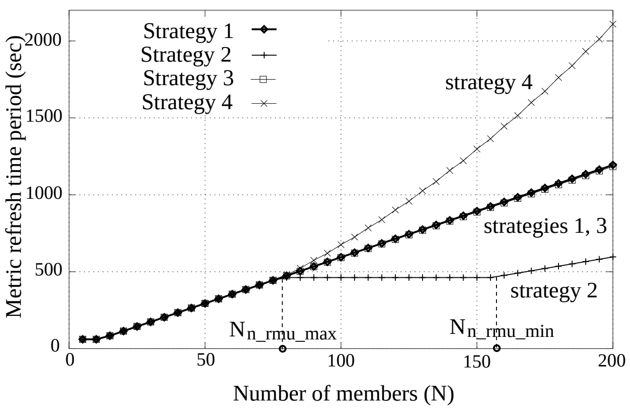
The main difference with strategy 2 concerns the $n_{rmu}(N)$ adaptation. This parameter remains constant above $N_{n\_rmu\_max}$ members, while the $T_{mu}(N)$ is immediately increased. It means that this strategy progressively reduces the number of metric update messages but each of them refreshes a higher number of communication costs with other members.

3.2.5 Strategy 4

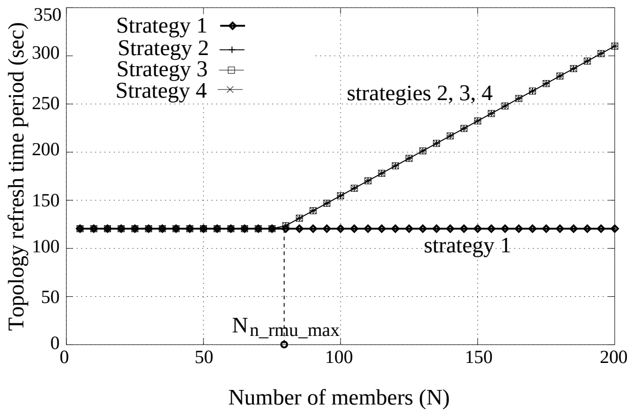
Finally the fourth strategy is controlled by the following equation set:

$$n_{rmu}(N) = \begin{cases} N-1 & \text{if } N \leq N_{n\_rmu\_max} \\ n_{rmu\_max} + \frac{N-(n_{rmu\_max}+1)}{\beta} & \text{if } N > N_{n\_rmu\_max} \end{cases}$$

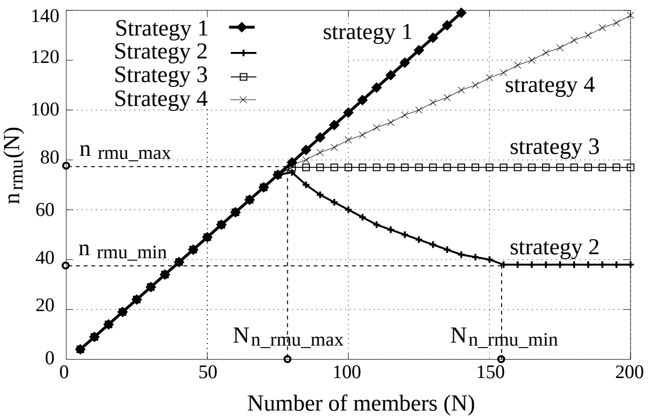
$T_{mu}(N) =$



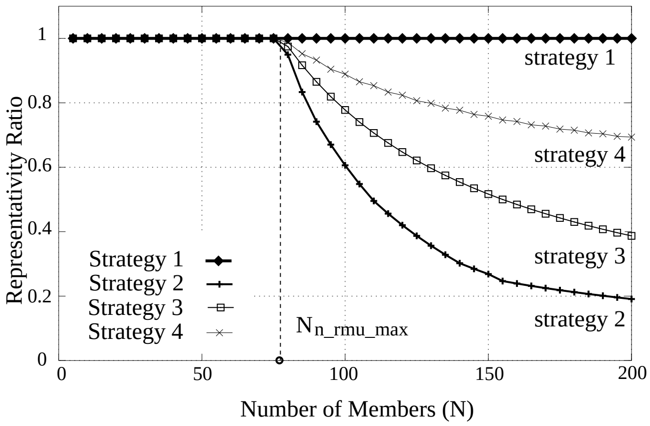
(a) Metric update period, $T_{mu}(N)$, at a member.



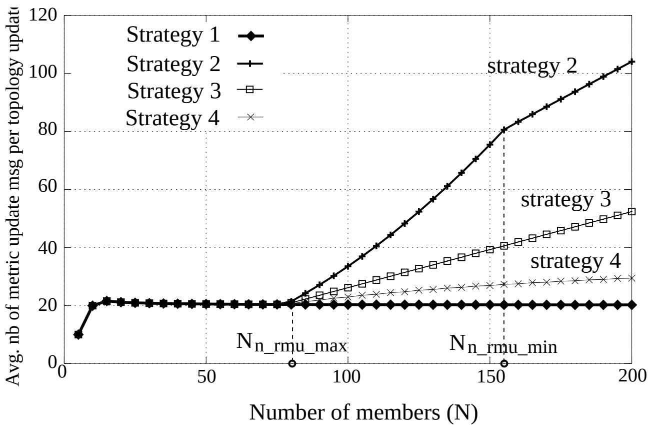
(b) Topology update period, $T_{tu}(N)$, at the RP.



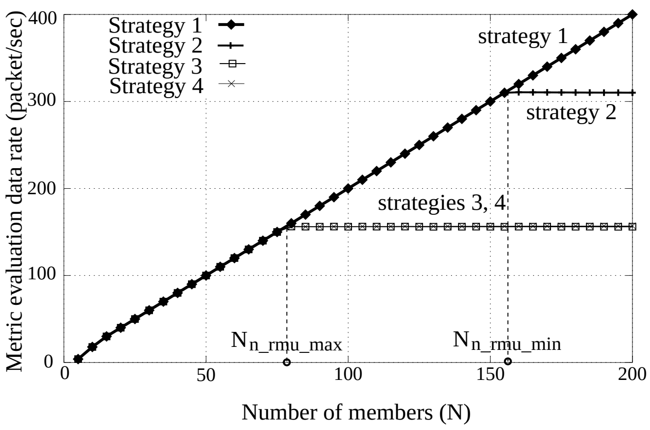
(c) Number of records, $n_{rmu}(N)$, in a metric update message.



(d) $\frac{n_{rrm}(N)}{(N-1)}$ ratio, i.e. representativity of a MU message.



(e) Average number of metric update message per topology update.



(f) Metric evaluation overhead (with the ping command).

Figure 2: Theoretical results for the various strategies.

$$T_{mu}(N) = \begin{cases} max(T_{mu}, T_{m\_eval} * (N-1)) \\ \qquad \text{if } N \le N_{n\_rmu\_max} \\ T_{m\_eval} * n_{rmu\_max} * \frac{s_{mu}(N)}{s_{mu}(N_{n\_rmu\_max})} \\ \qquad \text{if } N > N_{n\_rmu\_max} \end{cases}$$

$$T_{tu}(N) = \begin{cases} T_{tu} & \text{if } N \le N_{n\_rmu\_max} \\ T_{tu} * \frac{s_{tu}(N)}{s_{tu}(N_{n\_rmu\_max})}) & \text{if } N > N_{n\_rmu\_max} \end{cases}$$

In this strategy, the number of records in each metric update message keeps increasing, even with large values of N, but in that case more slowly (we define a $1/\beta < 1$ slope). Consequently, the $T_{mu}(N)$ period increases much faster than with the previous solutions. This strategy goes further in the idea of strategy 3, i.e. have larger but less frequent metric update messages.

3.3 Metric Evaluation Overhead

HBM uses the `ping` command (by lack of a better tool) to evaluate the communication costs (i.e. RTT and losses). It keeps the default `ping` parameters (56 byte payload, one request per second), but limits each evaluation to 6 `echo requests`. The corresponding bit rate overhead (including ICMP/IP), for N members, is:

$$ping\_rate(N) = N * \frac{6 * 2 * (20 + 8 + 56) * 8 * n_{rmu}(N)}{T_{mu}(N)} \quad (5)$$

where the $n_{rmu}(N)$ and $T_{mu}(N)$ parameters depend on the chosen strategy. Therefore $T_{m\_eval} \simeq 6$ seconds. Figure 2-(f) shows the metric evaluation overhead (in packet/s) and proves that strategies 3 and 4, and to a lesser extent 2, also limit this overhead.

4 Experimental Evaluation

4.1 Experimental Setup

The HBM protocol and all the strategies defined previously have been implemented in a dedicated C++ Group Communication Service Library. Experiments have been carried out with two hosts (PIII-1 GHz/Linux), one of them running the RP and the other one the N members. Since we only focus on the control traffic overhead, having all the members on the same host is not a problem. The parameters are set as follows: $T_{mu} = 60.5$ seconds, $T_{tu} = 120.5$ seconds, $T_{m\_eval} = 6$ seconds, $R_{data} = 128$ kbps, $R_{ctrl\_max} = 6.74$ kbps, i.e. $\alpha = 5\%$ of $(R_{ctrl}(N) + R_{data})$. The various control messages exchanged are sent as uncompressed, plain text. Each point in the figures is the average rate obtained after 30 minutes. In a second step we carried the same experiments with higher R_{data}/R_{ctrl_max} parameters in order to evaluate their impacts.

4.2 Experimental Results with $R_{ctrl\_max} = 6.74$ kbps

Figure 3-(a) shows the control overhead for strategy 1 and serves as a reference. It shows that $R_{ctrl}(N)$ increases linearly with N (and not in $O(N^2)$ because of the reason explained in section 3.2.2). Figure 3-(b)-(c)-(d) are for strategies 2, 3 and 4 respectively. $R_{mu}(N)$, $R_{tu}(N)$ and $R_{ctrl}(N)$ (the sum) increase progressively until $N = N_{n_{rrm-max}} = 79$ members are present. Above $N_{n_{rrm-max}}$, $R_{mu}(N)$, $R_{tu}(N)$ and $R_{ctrl}(N)$ are constant, around 6.75 kbps, 0.75 kbps, and 7.5 kbps respectively.

We can say that *all strategies do achieve their goal of limiting the total control overhead rather well.* The total rate is in practice slightly larger than expected because of simplifications hypothesis taken during the theoretical analysis (e.g. the fixed size of the various messages fields of figure 1).

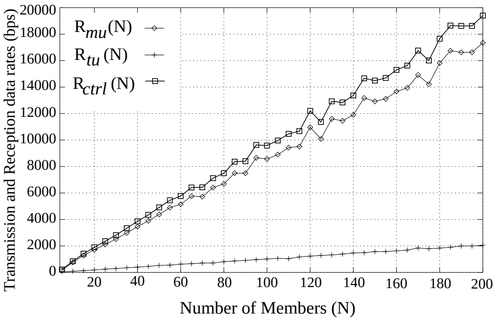
4.3 Results with higher R_{ctrl_max} values

The previous results largely depend on the R_{data}/R_{ctrl_max} parameters, since they determine the maximum possible control overhead. We performed the same experiments with higher values and summarized the results for N=200 members in figures 4-(a) and (b). Non-surprisingly the HBM responsiveness is improved by increasing the bandwidth allocated to control messages (up to a minimum of $T_{m\_eval} * (N-1) = 1194s$. time required to evaluate the metrics for T_{mu}). An exception is the T_{mu} period for strategy 2 which increases with R_{ctrl_max}. At the same time the number of records in a MU message, n_{rmu}, also increases more rapidly than with other strategies (from 38 records to 199 records, not shown) and MU messages become more representative.

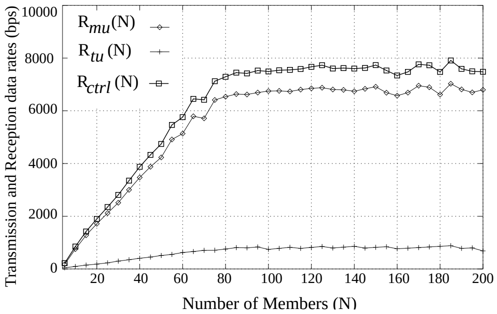
5 Discussion

If the three strategies do achieve their goal, they do it differently. The question is thus: "what is the best solution to achieve that goal"?

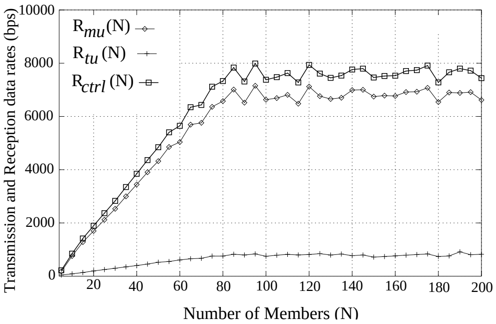
With strategy 2 the $T_{mu}(N)$ period slowly increases. This is an advantage since the metrics with the direct neighbors are frequently updated (figure 2-(a)) which enables a fast discovery of congestion problems. But they are also less representative (figure 2-(d)), which reduces the probability of finding better neighbors in the overlay topology. Another drawback is the higher metric evaluation overhead (section 3.3).



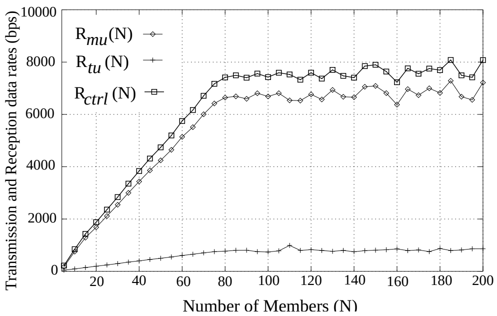
(a) Strategy 1



(b) Strategy 2



(c) Strategy 3



(d) Strategy 4

Figure 3: Experimental total control overhead for the various strategies.

In our opinion *Strategy 3 offers a better compromise*. The metric evaluation overhead is reduced compared to strategy 2 and many members are probed during the metric update processes (figure 2-(c)), thereby offering a better opportunity to improve the overlay topology. The price to pay is a lower metric update frequency.

Finally strategy 4 goes to far in this direction, and the $T_{mu}(N)$ period for very large values of N is by far too high, leading to a bad adaptability.

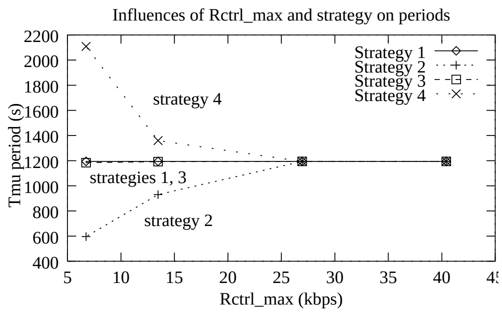
Note that results are largely impacted by the various protocol parameters (parameter $max(T_{mu}, T_{m\_eval} * (N-1))$ in the $T_{mu}(N)$ equations). For instance this is the reason why the T_{mu} period is only constant for $N << 1$ and keeps increasing afterward (at least up to $N_{n\_rmu\_max}$, and sometimes after this value depending on the strategy chosen). Having fast (i.e. not in $O(N)$) yet reliable metric evaluation techniques between the various members would largely modify our results.

6 Conclusions

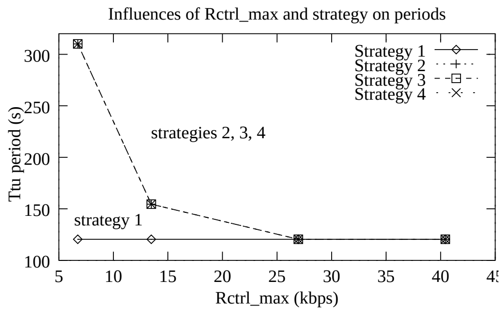
This work discusses the scalability of a control protocol, called HBM, that provides an application level group communication service. HBM is a centralized solution, where everything, including group membership management and overlay topology creation, is under the control of a single Rendez-vous Point (RP). This feature naturally leads to scalability problems. After a detailed modeling of the protocol behavior, this paper explains how the scalability can be largely improved, with a few simple protocol parameter adjustments: the number of records in a metric update message, the metric update generation period, and the topology update period. Finally an appropriate solution, strategy 3, that in our opinion offers a good compromise between the various aspects, is identified. This paper also highlights the practical limitations raised by the metric evaluation process which largely impacts our results.

References

[1] M. Castro, P. Druschel, A-M. Kermarrec, and A. Rowstron. Scribe: a large-scale and decentralised application-level multicast infrastructure. *IEEE Journal on Selected Areas in Communication (JSAC)*, 2002.



(a) T_{mu} period



(b) T_{tu} period

Figure 4: Influences of R_{ctrl_max} on the various periods for $N = 200$ members.

[2] C. Diot, B. Levine, B. Lyles, H. Kassem, and D. Balensiefen. Deployment issues for the ip multicast service and architecture. *IEEE Network*, January 2000.

[3] A. Elsayed, V. Roca, and L. Mathy. A survey of proposals for an alternative group communication service. *IEEE Network, special issue on Multicasting: an enabling technology*, January 2003.

[4] Yang hua Chu, S. Rao, and H. Zhang. A case for end system multicast. In *ACM SIGMETRICS*, June 2000.

[5] Laurent Mathy, Roberto Canonico, and David Hutchison. An overlay tree building control protocol. In *In proceeding of the third International COST264 Workshop, Networked Group Communication (NGC 2001), London, UK*, November 2001.

[6] Vincent Roca and Ayman El-sayed. A host-based multicast (hbm) solution for group communications. In *First IEEE International Conference on Networking (ICN'01), Colmar, France*, July 2001.

[7] Beichuan Zhang, Sugih Jamin, and Lixia Zhang. Host multicast: A framework for delivering multicast to end users. In *Proceedings of IEEE INFOCOM'02*, June 2002.