INTERNET DRAFT
MALLOC(?) Working Group
Expiration: 26 May 1998

R Briscoe
M Tatham
BT

21 Nov 1997

End to End Aggregation of Multicast Addresses

draft-briscoe-ama-00.ps,.txt [28]

Status of this Memo

Abstract

This paper presents an approach for solving the inherent problem with multicast routing scalability - by co-operation between end-systems and the network. We introduce an extremely efficient, elegant way to name arbitrary sized inter-meshed aggregations of multicast addresses. This is done in such a way that it is easy to calculate how to change the name to encompass many more related names. We describe how these aggregate names could be used anywhere in place of the set of addresses to which they refer, not by resolving them into multiple operations, but by a single bulk action throughout the routing tree, and in session descriptions potentially including those for reservations. Initial aggregation in end-systems might only reduce the problem by an order of magnitude, but it is believed that this will provide sufficient structure for routers to be able to recognise further aggregation potential. To improve the chances of router aggregation, address set allocation schemes must fulfil certain criteria that are laid down in this paper.

Keywords

Multicast, address, aggregation, end to end, scalable, Internet.

Contents

1. Introduction & Requirements

The addressing scheme used for Internet multicast has been recognised as unscalable since its inception. Every multicast group has to have a separate entry in the forwarding tables of every router on its path. Because multicast groups, being logical entities, have no direct relationship with physical topology, they cannot directly be matched to the hierarchical design of the Internet. In this paper we present an approach for solving this problem indirectly, in the proven tradition of the Internet; end to end - by co-operation between end-systems and the network.

The primary problem is that multicast addresses are fixed by session initiators, but aggregation relies on a clustering pattern emerging from the demography of the receivers. Further, the initiator usually doesn't know who the receivers will be until after the addresses to be used have been fixed. Therefore, any clustering beyond small-scale aggregation within applications will have to be achieved on a longer time-scale by session initiators predicting the likely demography of their session (e.g. based on past experience of similar sessions). The job of these predictive systems will be much simpler if they have some degree of pre-existing aggregation to nucleate around, rather than having to crystallise aggregation from complete chaos without any bootstrap process.

A large class of applications utilises multiple multicast addresses internally. Further, many such applications consist of varying sets of multiple multicast groups that are all sub-sets of a bigger set (e.g. news, stock-feeds, network games, virtual worlds, distributed simulations, all applications using layered multicast [1]). Any solution should ensure that such "nucleating applications" use aggregations of addresses that can be identified as such.

Related work is described later. Most is in the area of straightforward multicast address allocation, which is, perhaps surprisingly, far from being sorted out. It is very difficult, indeed probably reckless, to comment on proposals that have only been hinted at in the odd mailing list posting, or conference presentation. However, it appears that most schemes are assuming that aggregation will be possible if addresses are allocated based on the topological position in the Internet of the root of the multicast tree. Although this may well be the case, it is only true if most multicast applications will tend to be used by groups of users that happen to use the same ISP (Internet Service Provider), or they use ISPs that have a common parent ISP. No evidence that this is the case has been presented. In some cases the tree root even seems to be (erroneously) assumed to be in the same place as the session initiator (the party requesting the address). Certainly, nowhere does it seem to be taken into account that the likely receiver topology is just as important in determining whether multicast trees can be aggregated. Some of the proposals that are available also seem to implicitly assume that routing state will be aggregated in the same way unicast routing is aggregated - by discarding the right-hand bits of addresses which have common left-hand bit fields, despite there being no greater significance to any bit in a multicast address. These criticisms may prove to be unfounded when the work in question is published in detail.

The primary thesis of this paper is that it will never be possible to aggregate multicast state in routers if there is no means to *name* aggregations of multicast addresses. In unicast, an address prefix is an aggregation name, but for multicast the prefix means nothing. A similar concept is required, but with radically different properties and requirements.

These names must take up significantly less space than would the list of addresses themselves. Further, if only some sizes of aggregations can be named, this will lead to wastage of addresses. Therefore, any solution must offer a way to name arbitrary sized aggregations. Further, the naming of aggregations must not make any one type of address more in demand than others (e.g. addresses with trailing zeroes, or with sequences of zeros) or encourage the hoarding of certain addresses. It should also be possible to grow or shrink the size of the set identified by the name, while avoiding clashes with addresses in use by other sessions. An informal list of further requirements of address allocation schemes, which are not directly relevant to the discussion here, is at [26].

These aggregation names should themselves be open to aggregation, implying the naming should be recursive. Otherwise the small-scale clustering that the "nucleating applications" will be able to engender, will simply result in these small clusters collecting in routers deeper into the network. Where two names might be aggregated, it should be possible to test this possibility, based on the structure of the names, rather than by trial and error or exhaustive expansion to the lists of addresses that the names resolve to.

To encourage their use, these aggregation names should offer convenience and efficiency for the programmer. Preferably they should enable bulk joins to and leaves from sets of multicast addresses. The "nucleating applications" would then naturally assist the network in aggregating together multicast addresses, not out of any sense of altruism, but because using an atomic name for an aggregation is convenient for the programmer and efficient for the system it runs on.

This approach could be accused of moving away from random address allocation and therefore possibly encouraging address hoarding. Certainly, once a pattern emerges, the aim should be to bias address allocation to re-inforce the pattern. However, this is still originally based on random allocation, so shouldn't lead to hoarding as long as we don't aim to re-inforce patterns to such an extent that they become entrenched beyond their useful life.

2. Scheme Description

The proposed scheme for identifying aggregates of multicast addresses builds directly on IP multicasting [2]. The scheme assigns a new meaning to (a number that looks like) any existing multicast address, but only when it is in in a context associated with (at least) two other parameters that define aggregation size. Multiple sets of these aggregation size parameters can be associated with the same single multicast address to signify a list of aggregates of multicast addresses affording further aggregation. These aggregated addresses would be suitable replacements even for single discrete multicast addresses both in hard multicast routing state on the router and soft state in messages initiating and refreshing it, as well as in session descriptions. Thus, in the context of sending data to or receiving data from a multicast address, there is no change from the existing meaning for the address - it doesn't represent an aggregate, just a single address. In the context of joining or leaving a multicast group and in the context of describing a session, the address means an aggregation (although possibly of size 1) if associated with size parameters. This switch of meaning dependent on context is consistent with the fact that the aggregation parameters aren't associated with sending and receiving, only with allocation, joining, leaving and describing.

For conciseness, we shall refer to this tuple of multicast address and aggregation parameters as an aggregated multicast address (AMA[i]).

Thus, it is intended that AMAs are used in future versions of Internet Group Management Protocol (IGMP) [3, 4] and the various multicast routing protocols (DVMRP [5, 6], CBT [7, 8, 9] & PIM [10, 11, 12]) to replace multicast addresses wherever they are used. It would also be highly advantageous for session directory, invitation and announcement protocols (e.g. SDP [13], SAP [14] & SIP [15]) to evolve to use AMAs in place of multicast addresses. Evolution from the current version of IGMP and multicast routing protocols and from current session description protocols is discussed under the "Evolution" section.

Further, it would be natural for evolving multicast address allocation schemes ([20, 21]) to use AMAs to define allocations concisely. Just as sets of related AMAs can be aggregated together into a single AMA, the reverse is also possible. Thus parts of large allocations can be "sub-allocated" to further parties if required. However, allocation of multicast addresses (and hence AMAs) is not as straightforward as for unicast addresses. Allocation issues are discussed under the section on "Probability of address aggregation".

Firstly we shall define how an AMA is constructed and how its construction affects its meaning. Then we shall follow through a typical life-cycle of an AMA:

1. how an application would assign an AMA
2. how a sending application would understand an AMA
3. how a receiving host would join or leave an AMA
4. how a receiving application would understand an AMA
5. how a router would aggregate AMAs
6. how an application would expand or contract an AMA if required

2.1 Construction of an AMA

The construction of an AMA is based on two architectural principles:

1. to identify a range of addresses by starting from a base, using an offset (`d` or `v`) to the minimum and a range value (`n` or `s`) to identify the maximum
2. to use two levels of this structure - the first based on bit-field widths to home in on a rough area using very few bits, the second using actual values to achieve the arbitrary aggregation size requirement

```
----------2^d---------->--2^n------>
                      |-v-->-s->
```

The motivation for having a maximum for the rougher range as well as the fine-grained one is explained later.

The address-like component of an AMA is defined to be a concatenation of bit-fields as shown (with recommended values of bit widths shown for IPv4 - see discussion later):

```
Description:        mcast  mask        remainder
                    addr   offset
                    flags
Value:                     d                r
Width:                4    w_d              w_r
Recommended width:         4                24
                     <--><-->.<-------.--------.------->
                     1110XXXX.XXXXXXXX.XXXXXXXX.XXXXXXXX
```

The three associated aggregation parameters are defined below. The recommended values of bit widths are shown for IPv4, but later a scheme for optimising these is proposed (see Further storage optimisation). Introducing this now would just confuse the explanation:

```
Description:        mask          aggr'n           aggr'n
                   width          size             offset
Value:               m             s                t
Width:               w_m           w_s              w_s
```

```
Recommended width:   4           16                    16
                   <-->  <-------.------->  <-------.------->
                   XXXX  XXXXXXXX.XXXXXXXX  XXXXXXXX.XXXXXXXX
```

The value in the "mask width" bit field, m, (recall this is associated with an address, not part of it) determines the width, n, of a bit-mask overlaid on the remainder (as described next), where:

$$n = m + m_0 \qquad \text{--------(1)}$$

and where m_0 is a constant integer (set for this scheme) that determines the minimum size of an AMA (see later for discussion). It is recommended that:

$$m_0 = 0$$

The value in the mask offset field, d, determines the offset in bits from the right-hand end of the address to the right-hand end of the mask, for example, if $m_0 = 0$:

```
                            d              r                m    s
Example pseudo-dotted     13 .    221  .  147   .  93       3    5
 decimal value:         <--><-->.<-------.--------.------->  <--><-...
                        11101101.11011101.10010011.01011101  0011
Mask (XXX):             11101101.11011101.XXX10011.01011101
                                             <----d--------
                                          <-n
```

Note: in these diagrams, the label "d" is used to denote where the value of d is stored and the length of the offset set by this value.

If $n + d > w_r$, the bit mask simply wraps round into the right-hand end of the remainder, r.

The value within the masked bits, v_{min} (100_b in this example) determines the base or minimum address of the AMA[(ii)].
The value of the aggregate size, s, (recall this is associated with an address, not part of it) determines the upper or maximum address of the AMA by setting the maximum value in the masked bits, v_{max}, where:

$$v_{max} = v_{min} + s - 1$$

The AMA is defined as the set of addresses that result when v is varied between its minimum and maximum values. For example, the AMA defined by our example address and example aggregation size parameters follows:

```
                   <v>                  dotted decimal
11101101.11011101.10010011.01011101   237.221.147.93   a0
11101101.11011101.10110011.01011101   237.221.179.93   a1
11101101.11011101.11010011.01011101   237.221.211.93   a2
11101101.11011101.11110011.01011101   237.221.243.93   a3
11101101.11011101.00010011.01011101   237.221. 19.93   a4
```

Note that if $v_{max} > 2^n$, v may wrap within the bit-field (without carrying outside the bit-field).
Note that $s=0$ means no values of v at all (useful later).
Note that meanings of $s > 2^n$ are reserved but undefined. ------(2)

For convenience, we might denote an AMA as so far described by the tuple:

(a, m, s)

which for the example above would be:

(237.221.147.93, 3, 5)

Where two or more AMAs are disjoint sets that share a common a (even if the masked bits are different) and m, they may be represented by the more general form that is the correct full definition of an AMA:

$$(a,\ m,\ s_0,\ [(t_1,\ s_1),\ \ldots\ (t_i,\ s_i),\ \ldots\ ])$$

where t_i is an offset from v_{min}, and s_i is the size of the set of addresses based on v_{mini} where:

$$v_{mini} = v_{min} + t_i.$$

Thus, continuing the above example, all three of the following sets of addresses or AMAs represent the same thing:

$a_0,\ a_1,\qquad a_3,\ a_4$
$(a_0,\ m,\ 2),\ (a_3,\ m,\ 2)$
$(a_0,\ m,\ 2,\ 3,2)$

This allows re-use of one address by associating it with multiple aggregate size pairs with minimal extra storage.

Further, as the sequence of pairs along the AMA is processed,

if $t_i >= 2^n$, m is incremented until $t_i < 2^n$ --------(3)

m remains at its higher value for subsequent pairs, unless it is incremented further later.

Again continuing the above example to illustrate this point, if we denote a_8 as the address related to a_0 like so:

```
                <-v->                  dotted decimal
11101101.11011101.10010011.01011101    237.221.147.93   a0
11101101.11011100.10010011.01011101    237.220.147.93   a8
```

then these two sets represent the same thing:

$(a_0,\ 3,\ 2),\ (a_3,\ 3,\ 2),\ (a_8,\ 4,\ 7)$
$(a_0,\ 3,\ 2,\ 3,2,\ 8,7)$

The motivation for this particular design of scheme should start to become apparent; AMAs can be fully manipulated without caring about which multicast address was randomly chosen to start with as the base. This ensures there is no more demand for any one multicast address than another. v_{min} doesn't have to be all zeroes to ensure all 2^n combinations of bits under the mask are used. This ensures that the task of aggregating multicast addresses into AMAs and small AMAs into big AMAs should be achievable with just bit-wise logic operations.

2.2 Probability of address aggregation

Fig 1. attempts to represent multicast trees of various topologies across the Internet as various three-sided shapes. The Internet is represented by the oval with end-systems around the perimeter. The root of each tree is indicated by a dot. The receivers that have joined the tree are spread across the side of the "triangle" opposite the tree root. The shaded areas are an attempt to represent the aggregation potential of a representative topology by representing an even density of receivers around the edge and an even density of routers across the oval, rather than being a representation of the physical density of hosts and routers. Trees "***a***", "***d***" and "***e***" are core-based trees, with tree roots within the network rather than on end-systems. Trees "***b***" and "***c***" are source-based with roots on end-systems. It should be noted that these trees are probably not typical, mainly because it is difficult in such a diagram to draw trees that have receivers spread all over the Internet. However the trees have been chosen to illustrate various points.

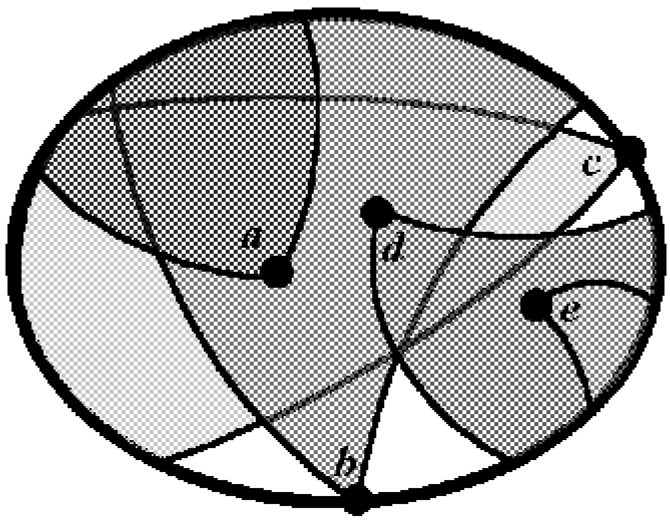

Fig 1 - Various multicast tree topologies across the Internet

Although trees "***a***" and "***d***" have roots that could well be in the same domain (autonomous system), their receiver sets have caused them to grow in completely opposite directions so no aggregation would be possible. Tree "***e***" is completely contained within tree "***d***" although their roots could well be in completely unrelated domains. Thus it should be possible to aggregate the routing information they need. Tree "***a***" is nearly contained within tree "***b***", so it would be desirable to be able to aggregate their routing at least where they overlap. Trees "***a***" and "***c***" have a similar relationship to that between "***a***" and "***b***". Trees "***b***" and "***c***" cross eachother at the network core so some aggregation may be possible where the directions of the trees coincide on certain links. There could even be occaisions where the routing of "***a***", "***b***" and "***c***" can all be aggregated together.

It should now be possible to see that, although it is more likely that two trees with close roots will have routing information that is condusive to aggregation, this is by no means a necessary or sufficient condition. In fact it appears that receiver distribution is a better indication of aggregation potential.

A multicast address needs to be allocated when a session is initiated, at which time the likely receiver topology may be difficult to predict. For certain types of tree (e.g. source-based) it is much easier to be certain where the root should be at this early stage. For core-based trees, the positioning of the core is ideally dependent on the prediction of the receiver topology (which can be sketchy, as we have already said).

Receiver topology is difficult but not impossible to predict. This usually reduces to a marketing-type problem. What is clear is that address allocation schemes that are based exclusively on the position of the root of the multicast tree (or worse the position of the session initiator, who may not even be taking part once the session is going) will not realise anything close to the full aggregation potential of any conceivable topologies of multicast trees that are likely on the Internet. A good scheme must allow addresses to be allocated to sessions based on a combination of root and receiver topology. It is outside the scope of this paper to solve how this allocation would be done, but it is clear that a scheme that disallows a future solution to this problem is a bad scheme.

The scheme proposed for AMAs allows just such a combination of allocation between root and receiver topology. Taking the example AMA used already,

```
                              d            r
Example pseudo-              13 .   221  .  147   .  93
 dotted decimal value:  <--><-->.<-------.--------.------->
                        11101101.11011101.10010011.01011101
                                          |<----d--------
                                          |
                              d           |    f
root-based allocation:     <-->.          |<----.-->
receiver-based allocation:      --------.-->        <----
                                |_________g________________|
```

The 16 bits, labelled `g`, to the left of the offset (to which `d` points) are those that the largest possible mask could cover so they might conceivably all be aggregated together. Therefore these bits should be allocated purely on likely receiver positioning. In the example this field wraps after 11 bits with the remaining 5 bits being at the far right hand end of the address. On the other hand, it is proposed that `d` and `f` would be allocated based on the position within the Internet of the root of their multicast tree (possibly combined with the likely direction from the root from which most receivers would join, to take account of trees like "***d***" and "***e***" above). `f` is defined as the 8 bits to the right of the offset (wrapping round the length of `r` if necessary). The mask can never cover `f` which is why it is available for allocation based on the root position.

A simple short term solution for an address allocation scheme might be to generate `g`, `d` and `f` from an algorithm seeded by a unicast address prefix (or a set of a few representative prefixes) that represents the majority of likely receivers and the unicast address of the tree root. The latter would be necessary if the tree were source-based, but it may be possible for the allocation service (or some other service) to suggest the best tree root position if using core-based multicast.

2.3 Further storage optimisation (for IPv4)

Optimisations considered but rejected are recorded elsewhere [25].

2.3.1 Mask width, `m`

In the contexts where AMAs are used, the first four bits of the address are redundant, as a non-multicast address wouldn't make sense. `m` is the natural candidate to store in place of this `1110` bit pattern which is fixed for IPv4 (class D) addresses (but see below). When the AMA is read from storage, m can be read into another variable and this bit pattern can be re-instated.

```
Description:        mask  mask        remainder
                    width offset
Value:                m   d               r
Width:                w_m w_d             w_r
Recommended width:    4   4               24
                     <--><-->.<-------.--------.------->
                     XXXXXXXX.XXXXXXXX.XXXXXXXX.XXXXXXXX
```

However, there is no guarantee that the multicast address range will always be confined to this fixed first four bits. For instance, class E addresses (starting `1111`) are currently reserved, and might conceivably form an extension to the IPv4 multicast address space in the future. Thus this optimisation is suggested but if it were adopted, the implications for the future use of other ranges as an extension to the multicast range would need serious consideration. However, as this is an optimisation, implementations could be designed so that if other address ranges ever became valid for multicast, the optimisation could be removed.

2.3.2 Aggregation size, s

Many applications will have sessions with less than 16 multicast addresses and most will have less than 256. Thus it seems wasteful to use 16 bits for `s` (and `t`) when most of the time, most of their leading bits will be zeroes. Also we must not forget that there will be many sessions that only use one multicast address.

Therefore, it is recommended that w_s be dependent on the value of `m` at least in the environments indicated:

			Rule applies in: router storage	Rule applies in: protocol fields
If m = 0	(0000_b),	w_s = 0	Y	N?
If 0 < m <= 1	(0001_b),	w_s = 2	N??	N
If 1 < m <= 3	(0011_b),	w_s = 4	N?	N
If 3 < m <= 7	(0111_b),	w_s = 8	Y	N?
If 7 < m <= 15	(1111_b),	w_s = 16	Y	Y

\--------(4)

The question marks indicate where the rule is open for discussion after implementation experience.

It is recommended, at least for router storage, that m = 0 is used to indicate an AMA that is actually just a single discrete multicast address, where associating a size of 1 with it would be a waste of space.

However, this appears to make the widest possible mask 15 bits, because it precludes using `m = 0` to mean `n = 16`. Although the need for a 16 bit wide mask is likely to be rare, it is still possible to force the mask to be this wide by exploiting the equivalence of the two AMAs in the following example (due to the ability of `t` to increment `m`, given in formula `(3)`):

`(a, 16, s)` (impossible value of m)
`(a, 15, 1,` $>2^{15}$`, s)`

The two conditions that lead to sub-byte storage requirements are not recommended as they are likely to be more trouble than they are worth, even for router storage.
It has also been assumed that saving a byte or two is not worth it for protocols (like IGMP) as opposed to router storage. The hassle of a conditional width field is probably not worth the small reduction in message size that would result.

Note that the width of `t` would always follow the same rule as that for `s`. That is:

$w_{ti} = w_{si}$

Note that in all cases (except `m = 0`) `t` can always be large enough, within the available field-width, to be capable of incrementing m into the next range, by formula `(3)`. For example, these two AMAs are equivalent :

(a, 8,0)
(a, m=7, s_0=0, t_1=128, s_1=128)

In the second AMA, the starting mask width is 7, so s_0, t_1 and s_1 are 8 bits wide. But, by formula (3), because $t_1 >= 2^7$ (has the first bit set) m is incremented to 8.

Note that if t forces m to increment into a range that would alter the width of both t and s (formula (4)), the increase in width of t doesn't happen until the next pair of t and s, if at all, while the s in the current pair should be widened immediately. The reasoning, is that the width of t mustn't be affected by its own value.

2.4 Life-cycle of an Aggregate Multicast Address

2.4.1 How an application would assign an AMA

First, the application initiating (or expanding) the session must decide how many multicast addresses it needs (bearing in mind the scheme allows it to expand or shrink from this initial decision later). This we will call s_{max}. (As discussed above, even if s_{max} is 1, it may still make sense to use AMAs.)

We assume some smart address allocation scheme has been worked out that meets the requirements laid down in the section on the Probability of address aggregation and may indeed be like the simple one outlined in that section. It should be noted that a sub-optimal, tactical address allocation service wouldn't stop this scheme working, but as the art of allocation improved, aggregation results would improve.

The application (or session initiator) now predicts the likely receiver topology for the session it is initiating. From this, it decides the best position for the root of the multicast tree(s) that make up the session (or some other service does this at its request). The application would then pass all these constraining parameters, including the size of AMA it required to the address allocation service. The address allocation service would return an AMA of the required size.

To arrive at an AMA, it would internally (probably) allocate g based on the receiver topology and d and f based on the root and receiver topology given to it by the application.

Also, internally, it would have to calculate n such that $2^{n-1} < s_{max}$, but $2^n >= s_{max}$.
In other words, it would decide integer n where no more than 2^n multicast addresses are needed in the session. m follows rather straightforwardly from formula (1).

For example, if the session needed 6 multicast addresses and $m_0 = 0$, it would use $m = n = 3$.

We shall assume the address allocated is the example address used above.

It then joins 6 multicast groups simultaneously by sending one message to join the AMA:

```
(237.220.147.93, 3, 6)
```

If the allocation scheme is light-weight it may give no guarantees that the any of the addresses haven't been simultaneously allocated to other sessions. Therefore, the application may have to monitor whether any of the multicasts are in use. If they are free, it holds them for itself (as is done with discrete multicast addresses today), then goes ahead and advertises the session (or invites users in) using this AMA.

In fact, if there is a finite possibility of simultaneous allocation, the above is not necessarily the best opening strategy for grabbing a set of free multicast addresses. Which strategy is best depends on the size of the set required and the level of utilisation of the address space (which will oscillate daily and increase into the future, decreasing the probability of finding a free set in a single attempt). This is a large enough problem space to become a topic for study in its own right (see Further Work), so it is not gone into in any depth here, save to make some broad generalisations.

If the probability of finding a free set of multicast addresses first time is high, the obvious strategy would be to

just try another address set if part of the first was busy. If the allocation service couldn't guarantee all its allocations were not in use, it would have to be possible to receive a slightly different response to a repeated identical request.
Otherwise it may be best to test a larger set than is required, then drop the busy ones. AMAs make this easy, as it is often possible to derive one smaller AMA from a larger one to deliberately avoid some of the addresses. Yet a third method would be to try a number of contiguous or overlapping AMAs, then drop those that tested busy and amalgamate the remaining ones into one AMA (again often but not always possible).

The comments in this section on allocation of AMAs for applications apply equally well to allocation of AMAs to allocation sub-authorities in an allocation hierarchy. However, it is made clear in the section on Probability of aggregation that allocation hierarchies *per se* are not a panacea where multicast is concerned.

2.4.2 How a sending application would understand an AMA

This section is included to remind the reader that sending to an AMA probably isn't a valid activity. As explained earlier, the concept of an AMA doesn't exist in the context of sending. If the sending part of an application includes understanding of sessions it may well utilise AMAs (this is also a sure sign it hasn't been written in a structured way!). However, the raw sending aspects will be direct to discrete multicast addresses, and shouldn't involve AMAs, unless they are modelled as an array of multicast addresses.

It might be desirable to send the same information to multiple multicast group addresses using only one socket and one flow of packets through the network. If a clear need for this were identified, it would be sensible to use the same addressing scheme as AMAs provide. However, no clear need is immediately apparent to the author, so it would not be sensible to update APIs for sending to multicast sockets and the router code that disseminates multicast packets so that they act on this extra parameter.

2.4.3 How a receiving host would join or leave an AMA

This is the other main use for AMAs besides in session initiation. An application's request to join an AMA given in a session announcement or invitation would be translated directly into a (future) IGMP call to join that AMA as a single atomic action. If the AMA was a superset of another AMA already having its soft-state joins regularly refreshed by the stack, the stack would have to merge the two AMAs into one expanded AMA (if it didn't the router would, before forwarding the joins).

Similarly, a call to leave an AMA would translate directly to a (future) IGMP call to leave. If the call to leave an AMA resolved to a sub-set of the multicast addresses previously joined, the router would be designed to be able to handle contracting the size of the AMA it considered was still of interest on that interface (or layer 2 address) down to the AMA that mapped to the remaining addresses. The host stack would also have to be able to contract its AMA for it to regularly refresh the soft-state of the remaining joined addresses.

The stack would have to ensure the frequency of join refreshes remained sufficient while it amalgamated or contracted down any out of phase AMA refreshes.

AMAs can be used as a consistent computational type for addressing any number of multicast addresses, whether the AMA resolves to many or just a single multicast address. In fact, APIs (application programming interfaces) could pre-empt the introduction of AMAs into the network, by presenting AMAs to the programmer but having middleware or the stack convert AMAs into sets of discrete multicast addresses until the network is upgraded. However, it would be sensible for routers and protocols to signify an AMA of size 1 by not storing or passing the aggregation size parameters at all (see Further storage optimisation - this would help backward compatibility too). Otherwise the efficiency benefits of the scheme would be offset by the 25% increase in storage required for each non-aggregated, isolated address. On the other hand, to hide AMAs from those programmers not interested in sessions with multiple multicast addresses, it may well be best to implement an API for AMAs by overloading a similar one for discrete multicast addresses[iii].

2.4.4 How a receiving application would understand an AMA

All the comments above on the irrelevance (and possible relevance) of AMAs to sending data apply equally to receiving.

2.4.5 How a router would aggregate AMAs

As a router received the equivalent of "join" and "leave" requests and refreshed "joins" it would continually be looking for matches at two levels:

1. on the one hand, the router would continually attempt to merge incoming AMAs on a per interface basis to reduce the state being held in its tables.
2. on the other, the router would attempt to aggregate the outgoing "join" refresh messages it forwarded upstream on a per-interface basis too.

AMAs can be aggregated at two levels (whether by a router, or by a host).

1. where two or more AMAs share a common a (even if the masked bits are different) and m, overlapping s ranges can be merged
2. where a whole set of combinations under a mask is present (s = 0), m can be incremented and this can be represented as half a full range at the wider level, which is then subject to further aggregation using the first technique again

Where multicast trees cross eachother (e.g. "***b***" and "**c**" in Fig 1), it would be necessary to "de-aggregate" routing at the point where the routes to the two roots diverge. As stated before, because AMAs can be split as easily as they can be amalgamated, this is not a problem.

The section on Router Implementation should be referred to for more detail on these matters.

2.4.6 How an application would expand or contract an AMA if required

As long as the overall application session was utilising an AMA with $s = s_{max}$, any one receiver could vary v_{min} and s around to utilise any subset AMA of the larger AMA set (useful for many simulation applications). Thus contraction and re-expansion of individual joins is straightforward.

To expand a session beyond the original s_{max} but still with a single AMA sometimes requires time for preparation of the ground and a degree of luck. In all cases this is because one is attempting to acquire use of extra multicast addresses without altering the addresses already being used. This limits the sets of interesting multicast addresses to AMAs that are related to the one in use. This is a limitation of the AMA scheme when compared to schemes based on arbitrary masks such as [19] (which has different limitations discussed under Related work). However, this was a concious compromise to avoid the larger storage needed for arbitrary masks as opposed to contiguous ones.

The following operations would (probably) be enacted by the address allocation service in response to a request to increase the size of an existing allocation (the uncertainty is because the detailed design of such a service is outside the scope of this paper).

Firstly, if not already there, s_{max} should be increased to 2^n and the new addresses tested for prior allocation.

If enough addresses still aren't free, n can be incremented without harm, then s can be doubled for every increment and the new addresses tested.

If this doesn't find enough free addresses, keeping n incremented , v_{min} can be changed while s is increased to try addresses related to the current range which will still give the network a chance to aggregate addresses as long as other receivers are co-operating within the same rules.

If this doesn't work, one can look for a close but disjoint AMA (or AMAs) and watch for a gap between the current AMA and the new one until it can be closed by a superset AMA later.

Beyond that, the only solution is to give up and use more than one AMA, but this is unlikely to be necessary.

2.5 Router implementation

The discussion in this section shouldn't be taken to imply that this is the best way to implement multicast routing. It is simply necessary to establish at high level that it would theoretically be possible to modify multicast routing implementations to take advantage of AMAs.

Currently, the most general form of multicast routing table is a database with (a usually large number of) rows for each unique address/source pair as follows:

$$a,\ S,\ f_I,\ [f_{Oj},\ \ldots\ f_{Ok},]\ misc$$

where the variables are defined as:

a	: multicast address
S	: source
f_I	: incoming interface
f_{Oj} ... f_{Ok}	: list of outgoing interfaces to which to duplicate and forward packets
misc	: other miscellaneous information not relevant to this paper (e.g. MTUs, prune state)

To take advantage of the aggregation and deaggregation potential of AMAs, the most general modification to this would be for each row to contain the following:

$$a,\ m,\ S,\ [(t_m,\ s_m,\ f_{Ih},),\ \ldots\ (t_n,\ s_n,\ f_{Ii},)],\ [(t_p,\ s_p,\ f_{Oj},),\ \ldots\ (t_q,\ s_q,\ f_{Ok},)],\ misc$$

Here one row represents the routing for all the related AMAs that are being aggregated and de-aggregated at this router with respect to one source. We define related AMAs as AMAs that can be represented with the same base address and mask, only differing in their offset and size. This is illustrated in Fig 2. It should be noted that the three trees, A_1, A_2 & A_3 represent related AMAs, not discrete multicast trees (although they could be AMAs of size one), because, as noted before, routing is one of the contexts where AMAs can completely replace discrete multicast addresses. Thus, for these related AMAs, only one set of routing information would be passed in or out of each interface (e.g. because all three AMAs "join" f_4, only one routing message would enter this interface for them all). Another way of defining related AMAs is to say that it is possible to represent their union as a single AMA (we will call this the super-AMA). The only distinction between the three trees illustrated is that they represent the different sub-sets (of the super-AMA) that share identical routing at this router. Each row such as that presented above represents the complete routing definition for each super-AMA at this router.

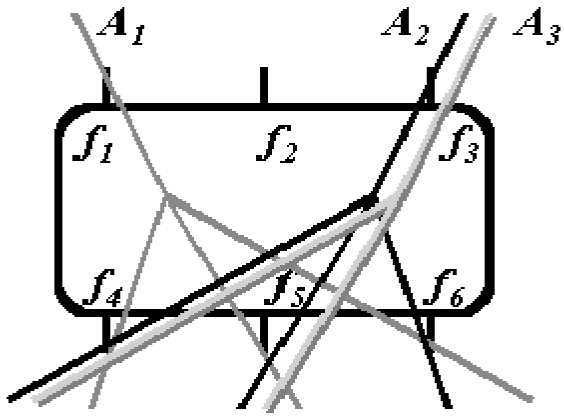

Fig 2 - Aggregation and de-aggregation of multicast trees

The routing row for this super-AMA is similar to the previous row structure, except that:

- the multicast address field has subtly different semantics. It doesn't mean a single address, but it is taken to mean the base address with which all the offset/size pairs in the row are associated - the base address of the super-AMA.
- the mask is included, which for IPv4, could be overlaid over the front of the base address as described under Further storage optimisation
- each outgoing interface is associated with an offset/size pair(s) (`t` and `s` are as already defined under Construction of an AMA above) which defines the outgoing AMA on that interface when associated with the base address
- there may me more than one incoming interface associated with all the related AMAs being de-aggregated at this router, each of which is listed with an offset/size pair(s) as for the outgoing interfaces.

For example, if A_1, A_2 & A_3 in Fig 2 were the three AMAs respectively:

```
(a,m,17), (a,m,0, 31,42), (a,m,0, 20,37)
```

then the routing table entry for the super-AMA `(a,m,42)` with respect to source `S` would be:

$$a,\ m,\ S,\ [(0,\ 17,\ f_1,),\ (20,\ 42,\ f_3,)],$$
$$[(0,\ 42,\ f_4,),\ (0,\ 42,\ f_5,),\ ((0,\ 17,\ 31,\ 42,\ f_6,)],\ misc$$

When a packet arrives at the incoming interface, it's destination is matched against the AMA associated with each outgoing interface. A match causes it to be duplicated and forwarded out of that interface.

Obviously, the row structure above isn't the most efficient for routing look-ups. It is more suited for routing updates. This suggests that a read-optimised data-structure should be built from this write-optimised structure and both held by the router, with the former being regularly refreshed from the latter. This is similar to the way most unicast routing is implemented. Whether there is one read-optimised table per router or a number of sub-sets of the main table specific to each interface is beyond the scope of this paper.

2.6 Derivation of recommended values for constants

The principles for choosing the constants for IPv4 have been laid out in detail, so that they can be re-applied judiciously for IPv6 once (or while) its multicast addressing scheme [18] is finalised.

w_d = 4

It is an important design feature that the mask offset is contained within the address, rather than added as another parameter outside the address. This is because, as a packet arrives at the router, this offset can be read to find the point at which any mask would be applied. This should greatly speed the process of matching a packet against the AMAs in the routing table. It also crams as much meaning into the address as possible, reducing the extra state held in routers and passed in protocols. The offset doesn't need to be varied as aggregation progresses, so it makes sense for it to be a fixed value, and therefore it might as well be part of the address.

For IPv4, a width of 4 for the offset was a compromise. Ideally the width would have allowed any offset value across the remainder ($28 - w_d$). Note that, if the offset did allow the mask to start anywhere in the remainder, this does not mean there would be no space in the remainder that was guaranteed to always be outside the mask (e.g. for address allocation related to the position of the tree root - see "How an application would assign an AMA" above). This depends on the mask width, not the offset size. Therefore, w_d could have been set to 5 leaving the remainder 23 bits wide. However, this would only have used half the value of the fifth bit, so 4 was chosen as a compromise between availability of more descriptions for AMAs and utilisation of bits. It should be noted that, because AMAs overlap in their use of addresses, there are diminishing returns in having more ways to split the description of sub-groups of the address space. Also a 5/23 split was "binarily awkward" compared to a 4/24 split.

$\mathbf{w_m = 4}$

Some thought was given to limiting the mask to 3 bits wide. This would have led to a maximum AMA size of 2^8, which would probably have been adequate if just considering application aggregation. However, we hope that higher level techniques will be found to enable considerable router aggregation based on the foundations laid by AMAs. Limiting routers to aggregation of just two orders of magnitude seemed short-sighted, when using one more bit would raise the maximum aggregation limit to 2^{16} (and align on a half-byte boundary). The importance of keeping the size of m down, is that it is stored in the router for each AMA.

The bit mask is deliberately fixed on its right-hand end so that when m is varied, the LSB of the mask stays in the same place otherwise large-scale aggregation would be terribly hard.

$\mathbf{w_s = 0, 8 \text{ or } 16}$

The discussion of this value's potential dependence on m is well rehearsed under Further storage optimisation . The motivation for keeping its size down is again router storage. It is expected that a large number of values of s will be under 16 (at least in edge routers), so it would be profligate to allocate a whole byte, let alone two just because such bit-widths will be needed for higher level aggregation or applications with more complex sessions.

$\mathbf{m_0 = 0}$

We could have made $m_0 = 1$ but this would have led to an extra increment operation every time the AMA was interpreted. Whether this is more efficient than testing for $m = 0$ is debatable (machine instruction set dependent). $n = 0$ could be made illegal rather than mapping it to 16, which would only lose a few ridiculously large (?) aggregates. Alternatively, we could even have made $m_0 = 2$, on the grounds that AMAs of size 2 are uninteresting, and AMAs of size 1 can be indicated by no value of s. We decided against this on grounds of caution over lack of elegance. If we put discontinuities in the maths, it makes it much more difficult to build higher order mechanisms that are simple.

2.6.1 Statistics

Below are listed the number of distinct AMAs of a selection of sizes (note they are not confined to powers of

2):
`AMA size    distinct AMAs`
{left as an exercise for the reader or until the author gets round to working out the formula...}

2.7 Evolution

It is a simple matter to convert an AMA to the list of addresses it refers to. It is also possible to convert lists of addresses into AMAs or lists of AMAs.

AMA-enabled hosts and routers shouldn't send or forward joins or leaves to routers that don't understand AMAs, they should convert the AMAs to lists of discrete multicast addresses.
This assumes routers can determine which version of a multicast routing protocol their upstream router for each interface is using. This should be straightforward as version stamped routing will also be arriving at the interface down that link (multi-host links will cause complications for some routing protocols). This also assumes hosts can determine which version of IGMP is being used by their upstream router which should be possible, but is probably difficult.

Session descriptions would either have to use both forms for an interim period, or parallel descriptions in the two versions would have to be transmitted on different channels.

IGMP, multicast routing protocols and the session description protocols could evolve independently as long as applications and AMA-enabled routers had the capability installed to convert AMAs into lists of discrete multicast addresses when necessary.

3. Related Work

The initial schemes for allocation of multicast addresses involve three distinct classes of address:

- addresses that are permanently assigned for specific purposes [23]
- address ranges that are permanently assigned for certain uses [23], from which single addresses are intended to be temporarily assigned
- an address range reserved for assignment within an administrative scope which may therefore safely have multiple assignments within multiple administrative domains [17]

For several years, multicast group addresses for public Mbone (experimental Internet multicast backbone overlay) sessions have been allocated using the sd (session directory) or sdr tools. In addition to advertising multicast sessions based on the SDP [13], sd and sdr encompass a proprietary mechanism for allocating multicast addresses to sessions from an IANA (Internet Assigned Numbers Authority) defined address range 224.2.128.0-224.2.255.255 [23]. The algorithm used has not been described publicly in detail, but is essentially random assignment from the available addresses to ensure very efficient use of the address space, but taking account of existing allocations to minimise the probability of collisions. There is also a collision detection algorithm included for the cases where an address is chosen simultaneously by multiple parties. Thus there is currently no standard mechanism for address allocation, but the author of the sd tools recently agreed to publish the address allocation mechanism from sdr as a separate protocol - Address Allocation Protocol (AAP) [20] - which could be incorporated into other address allocators.

Microsoft created an alternative means for allocation of multicast addresses by extending their Dynamic Host Configuration Protocol, originally designed to allocate temporary unicast addresses. Multicast DHCP (M-DHCP) [22] will allocate multicast addresses, consistent with the requested Administrative Scope (Link Local, Organisation Local, Global, etc.) and with a `lease period', the effectiveness of which is unproven. This lease period can be renewed if desired. The M-DHCP authors recently removed all aspects of hierarchy from the M-DHCP protocol, to confine it to allocating multicast addresses to hosts initiating sessions, rather than also using it to allocate ranges of addresses down allocation hierarchies. This was in response to criticism over its static hierarchy and static allocation wasting the address space. All aspects of multicast address hierarchy, and allocation policy are expected to be handled by a separate protocol, such as AAP (see above).

Where allocation of ranges of addresses to organisations is concerned there is concensus among the known proposals that these should not be static. Range allocations should be over one (longish) timescale with the allocation of addresses from within that range for the duration of individual sessions. The intention is to avoid long term "ownership" of addresses ranges by assuring organisations that they will be able to have addresses on demand as long as they co-operate with everyone else in returning unused address space. Thus the multicast address allocation proposals are distinct from unicast in the following aspects:

- No fixed assignments of address ranges to IP domains - address ranges are claimed as needed
- Random allocation of addresses to sessions within a domain
- Allocations of addresses have finite lifetimes

The MASC (Multicast Address Set Claim) [21] is one such proposal. This is work in progress but the outline of the mechanism as described by the authors is as follows:

1. The ISP (Internet Service Provider) claims an address set (in general, the ISP's next level provider will be in control of address allocation to the ISP)
2. Address ranges will be allocated for a set lifetime and in such a way that they may be aggregated. Allocation will take account of administratively scoped multicast addresses.
3. The ISP then advertises this address range to other domains using BGP4++ [21] (or similar mechanism). All border routers will thus hear these announcements. It waits a certain time, however, (typ. 3 days) before using the address range, to detect collisions
4. Address allocation mechanisms (such as AAP or M-DHCP (see below)) will listen to announcements and allocate addresses to sessions appropriately.

A possibility being investigated for MASC address range allocation is "Kampai style" addressing [19] which uses a non-continuous mask to define the allocated range so is more flexible when range sizes need to be arbitrarily increased or decreased. However, Kampai-style addressing allocates ranges in sizes that are powers of two and hence could be very wasteful where large ranges are concerned. Although there is brief consideration of a way to remove this restriction, it is admitted it will be more complex and hasn't been thought through.

The BGMP (Border Gateway Multicast Protocol) draft [24], proposes a new architecture for Inter-Domain IP multicast:

1. Address ranges are associated (at least temporarily) with domains, allocated using the MASC mechanism described above.
2. These address ranges will be advertised globally. The proposal is that BGP4 will be used for this, given the proposed Multiprotocol Extensions [21]. Thus, multicast address ranges will be advertised in much the same way as unicast IPv4 NLRIs are advertised in BGP4 now. They will be advertised at least a day in advance of use, to enable any clashes of address allocation to be resolved.
3. A multicast session initiated within a domain will be allocated an address from within the domain's multicast address assignment.
4. BGMP then assumes an inter-domain shared tree, for which the `root domain' (the focal domain of the shared tree) is the domain owning the address.

Some alternative ideas generated while preparing this paper (but rejected in favour of the scheme presented) are recorded elsewhere [25]

4. Limitations and Further Work

The ability to do large-scale aggregation is based on hope that higher level organisation will be achieved. This scheme "feels" as if it is a good foundation for solutions to these problems, but the author can't give any guarantees without more work. This implies it will be necessary to simulate, build, test & refine.

Aggregation of multicast addresses by this scheme will probably mean it is more efficient for routers to store multicast routing state keyed on interface than on multicast address. In other words, a table of aggregated multicast addresses would be held for each interface, rather than a table of interfaces for each address. The implications of this need investigation.

The problem of how a router would efficiently look up each multicast packet in a table of AMAs has been deliberately left to one side. Because AMAs are logically similar to unicast address prefixes, similar techniques should be appropriate. This may not appear obvious, because an AMA is more obfuscated than a unicast routing prefix. Briefly, the first item to be extracted from an incoming packet would be `d`. This would point to where the mask started. The eight bits to the right of this mask would then be guaranteed to be unmasked (invariant), so `d` and this octet could be looked up in a Patricia trie or similar but more efficient data-structure [16]. The two potentially masked octets would then have to be built into another similar look-up table. Finding any one address in this structure would again be akin to the longest prefix (shortest mask) problem.

It is possible that `m` is redundant, but it is believed that keeping it will make the aggregation maths a lot easier .

Applicability of AMAs as a solution to reliable multicast clustering/layering needs assessment.

Applicability of AMAs for aggregation in RSVP (reservation protocol) [27] when used for multicast needs assessment.

Further work is needed on the potential for using AMAs to insulate upstream routers from high join/leave churn by introducing pessimistic inertia in the aggregation. The effect on leave latency (particularly where used for congestion control in layered multicast) would need careful study.

The assertion that the weak capability for allocation growth (as compared to kampai-style addressing) is offset by more efficient storage needs more justification.

It is believed that, due to topological realities, aggregation in the network will never approach the aggregation potential of applications that use sessions with multiple multicast addresses. This would be a strong argument against aggregation schemes that are not end to end, but this needs proving.

Evolution from current versions of protocols needs more careful analysis.

The design of an AMA scheme for IPv6 [18] needs to be done.

ATM (asynchronous transfer mode) multicast may benefit from a scheme based on similar thinking?

5. Security Considerations

Phasing of AMA aggregation in routers must be designed carefully so that it is not possible for one user to join to an AMA that overlaps a neighbouring join, then leave the AMA and cause the neighbours to lose their join before their soft-state refresh re-instates it.

It is possible that the aggregation of multicast addresses into sets for use in the description of complex sessions will cause service providers to hoard multicast addresses more than when they are allocated singly. The scheme has been carefully designed to avoid such a tendency on technical grounds, but predicting how selfish people adapt based on their understanding of a mechanism moves us into the realms of psychology where anything goes. In other words, address hoarding shouldn't be any worse than the situation was without this proposal as long as people don't misunderstand.

This proposal is considered independent of all aspects of the security of (encrypted, tamper-proofed, authenticated etc.) multicasts.

6. Conclusions

A strong case has been made that, for multicast addresses to be open to aggregation, there must be a standard way to name arbitrary size groups of addresses. An efficient, elegant technique for doing this has been presented which gives equal value to each point of the multicast address space, thus preserving the principle of randomness designed to prevent address hoarding. These names have been called aggregated multicast addresses (AMAs).

For IPv4, AMAs look like a 32 bit IP address coupled with, typically, an extra octet (but more generally with two octets) representing the size of the aggregation of addresses. The list of addresses that form the aggregation are defined by identifying a string of bits (of defined width) within the address, which are incremented as if they were a binary number in their own right until the aggregation size is reached. All but the first four bits of the address-like component represents the base address of the aggregation from which the incrementation starts. Because multicast addresses always(iv) start with the same four bits in IPv4 ($\mathtt{1110}_b$), the first four bits of an AMA can be used to store another value in transit and storage, but replaced by $\mathtt{1110}_b$ to derive the base multicast address of the aggregation when required. The value stored in the first four bits of an AMA represents the width of the field within the AMA which varies to define the list of addresses. Related but disjoint AMAs can also be represented efficiently by using the most general AMA form: an AMA followed by a sequence of alternating pairs of numbers which represent respectively a further jump from the based address then a further size of aggregation on top of this.

In the scheme recommended for IPv4, potentially an arbitrary-sized aggregation of any size up to 2^{16} - 1 multicast addresses could be represented by a field the size of an IPv4 address (4B) plus 2B (256kB reduced to 6B). However, it should be understood, that such ultra-large-scale aggregation has a low probability of happening without higher-level organisation of the address space by end-systems. First or second order aggregation will occur naturally under this scheme due to the large class of applications that build sessions from multiple multicast addresses. Medium-scale aggregation will be possible where routers can identify overlap or concatenation within multicast routing tables, which was not possible before without a way to describe aggregations of multicast addresses. This is because the AMA scheme is inherently recursive, so that it is possible to merge certain sets of AMAs into one AMA. To improve the chances of aggregation in multicast routing tables, address set allocation schemes must fulfil certain criteria that are laid down in this paper.

The proposal is that AMAs will completely replace multicast addresses in contexts where aggregation makes sense, that is when describing, joining, leaving or updating the routing of multicast addresses. For sending data to and receiving data from multicast addresses, aggregation, and therefore AMAs, are not relevant. This implies protocols for describing multicast sessions (such as SDP, SAP, SIP, RSVP etc.) and protocols for updating the routing of multicast addresses (such as IGMP, DVMRP, CBT, PIM) will all need to be updated to handle AMAs in place of discrete multicast addresses. Independent evolution of all these protocols is considered to be reasonably straightforward. Although this represents a major round of protocol upgrades, all these protocols are experimental, and it is a commonly held view that multicast as it stands is not sufficiently scalable for wide-area deployment.

It should be clarified that, although joining and leaving aggregates of multicast addresses can be achieved in single bulk operations, AMAs deliberately overlap in their use of individual addresses. Thus, allocation remains on a per address basis. In other words, when joining an AMA, it may be found that some of the addresses within it are in use if a strong address allocation scheme is not in use. An AMA allocation procedure is described in the text for mutating the AMA to cover an overlapping set of addresses to avoid those addresses in use while testing different addresses, until a full set of unused addresses is obtained without losing the addresses in the original AMA that were free.

To summarise, we have presented a new paradigm(v) for multicasting, such that describing, joining, leaving and updating multicast routing can and should all be discussed in terms of aggregates of multicast addresses (even if they are of size unity) rather than discrete multicast addresses. We have introduced an efficient, elegant way to name such aggregates that preserves all the architectural principles on which multicast addressing is founded, and which will allow potentially large-scale aggregation of multicast addressing by both end-systems and routers in end to end co-operation.

7. Acknowledgements

Peter Bagnall, BT, for reviews and suggestions.

8. References

[1] S. McCanne, V. Jacobson, and M. Vetterli, (LBNL/UCB) "Receiver-driven Layered Multicast", In *Proceedings of the ACM SIGCOMM'96*, Stanford, California, pp.1-14. Aug 1996, <URL:ftp://ftp.ee.lbl.gov/papers/mccanne-sigcomm96.ps.gz>

[2] S.E.Deering, "Multicast Routing in Internetworks and Extended LANs", In *Proceedings of the ACM SIGCOMM'88*, Stanford, California, Aug 1988

[3] S.E.Deering (Stanford U.), "Host Extensions for IP Multicasting", *IETF RFC*, Aug 1989 <rfc1112.txt>

[4] W. Fenner (Xerox PARC), "Internet Group Management Protocol, Version 2", *IETF Internet Draft*, 01 Nov 1997, <draft-ietf-idmr-igmp-v2-08.txt>

[5] D. Waitzman, C. Partridge, S. Deering, "Distance Vector Multicast Routing Protocol" v1, *IETF RFC*, Nov 1988, <rfc1075.txt>

[6] T. Pusateri (Juniper Networks), "Distance Vector Multicast Routing Protocol", *IETF Internet Draft*, 30 Oct 1997, <draft-ietf-idmr-dvmrp-v3-05.txt>

[7] A.Ballardie, J.Crowcroft, "Core Based Trees: An Architecture for Scalable Inter-Domain Multicast Routing", In *Proceedings of the ACM SIGCOMM'93*, San Francisco, Sep 1993

[8] A. Ballardie, "Core Based Trees (CBT) Multicast Routing Architecture", *IETF RFC*, Sep 1997, <rfc2201.txt>

[9] A. Ballardie, "Core Based Trees (CBT version 2) Multicast Routing", *IETF RFC*, Sep 1997, <rfc2189.txt>

[10] S.E.Deering, D.Estrin, D.Farinacci, V.Jacobsen, L.Ching-Gung and L.Wei, "An Architecture for Wide-Area Multicasting". In *Proceedings of the ACM SIGCOMM'94*, London, Sep 1994

[11] Deering, S, et. al., "Protocol Independent Multicast Version 2, Dense Mode Specification", *IETF Internet Draft*, 28 May 1997 <draft-ietf-idmr-pim-dm-spec-05.txt>

[12] Estrin, D, et. al., "Protocol Independent Multicast Sparse-Mode (PIM-SM): Protocol Specification", *IETF Internet Draft*, 9 Sep 1997, <draft-ietf-idmr-pim-sm-specv2-00.ps>

[13] M. Handley (ISI) and V. Jacobson (LBNL), "SDP: Session Description Protocol", *IETF Internet Draft*, 02 Sep 1997, <draft-ietf-mmusic-sdp-04.ps>

[14] M. Handley (ISI), "SAP - Session Announcement Protocol", *IETF Internet Draft* (currently expired, see Multiparty Multimedia Session Control IETF working group charter for status <http://www.cs.utk.edu/~moore/san-jose/wg/mmusic/charter.html>)

[15] M. Handley (ISI), H. Schulzrinne (Columbia U.), E. Schooler (Caltech), "SIP: Session Initiation Protocol", *IETF Internet Draft,* 13 Nov 1997, <draft-ietf-mmusic-sip-04.ps>

[16] M.Degermark, A.Brodnik, S.Carlsson, S.Pink, "Small Forwarding Tables for Fast Routing Look-ups", In *Proceedings of the ACM SIGCOMM'97*, Cannes, France, Sep 1997 <http://www.acm.org/sigcomm/sigcomm97/program.html#ab192>

[17] D. Meyer, University of Oregon, "Administratively Scoped IP Multicast", *IETF Internet Draft,* Nov 1997, <draft-ietf-mboned-admin-ip-space-04.txt>

[18] R. Hinden, Ipsilon Networks and S. Deering, Xerox PARC, "IP Version 6 Addressing Architecture", *IETF RFC* <rfc1884.txt>

[19] P. Tsuchiya, Bellcore, "Efficient and flexible Hierarchical Address Assignment", to be published - author's address: <tsuchiya@thumper.bellcore.com>

[20] M. Handley (ISI), "AAP: Address Allocation Protocol" (presentation), In *Proceedings of the 39th IETF* (mmusic working group) <http://www.ietf.org/proceedings/97aug/toc-97aug.html>

[21] D. Estrin (ISI), M. Handley (ISI) and D. Thaler (Merit), "MASC: Multicast Address Set Claim" (presentation), In *Proceedings of the 39th IETF* (idmr working group) <http://www.ietf.org/proceedings/97aug/toc-97aug.html>

[21] Tony Bates et al., Cisco Systems, Juniper Networks, "Multiprotocol Extensions for BGP-4", 26 Sep 1997, <draft-ietf-idr-bgp4-multiprotocol-01.txt>

[22] Baiju V. Patel, Intel Corp, Munil Shah, Microsoft Corp. "Multicast address allocation extensions to the Dynamic Host Configuration Protocol", *IETF Internet Draft,* 24 Nov 1997, <draft-ietf-dhc-mdhcp-03.txt>

[23] Internet Assigned Numbers Authority, "Internet Multicast Addresses", <ftp://ftp.isi.edu/in-notes/iana/assignments/multicast-addresses>

[24] D. Thaler (U. Michigan), D. Estrin (USC/ISI) and D. Meyer (U. Oregon) , "Border Gateway Multicast Protocol (BGMP): Protocol Specification", *IETF Internet Draft,* 30 Oct 1997, <draft-ietf-idmr-gum-01.txt>

[25] R. Briscoe, "Rejected ideas for End to End Aggregation of Multicast Addresses", 13 Nov 1997, <http://www.labs.bt.com/people/briscorj/projects/lsma/e2ama_rejects.html>

[26] M. Handley, "Multicast address allocation", in *archive of postings to ipmulticast mailing list*, 19 Jun 1997, <http://www.ipmulticast.com/hypermail/ipmulticast/0144.html>

[27] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, "RSVP: A New Resource ReSerVation Protocol" In *IEEE Network*, Sep 1993, <http://www.isi.edu/div7/rsvp/pub.html>

[28] This paper itself, in HTML format <http://www.labs.bt.com/people/briscorj/projects/lsma/e2ama.html>

9. Authors' Addresses

Bob Briscoe

B54/74 BT Labs
Martlesham Heath
Ipswich, IP5 3RE
England
Phone: +44 1473 645196
Fax: +44 1473 640929
EMail: briscorj@boat.bt.com

Home page: http://www.labs.bt.com/people/briscorj/

Dr Martin Tatham

B29/129 BT Labs
Martlesham Heath
Ipswich, IP5 3RE
England
Phone: +44 1473 642498
Fax: +44 1473 640709
EMail: martin.tatham@bt-sys.bt.co.uk

Notes

i) Pronounced as in "If I 'ad an AMA" (American civil rights song in a cockney accent).

ii) The multicast addresses that are within the bit mask defined by the width m, but not between v_{min} & v_{max} (defined by v_{min} and `s`) are not "wasted" by using an AMA. In the example used above, the addresses with `v = 001, 010 & 011` are not reserved by the AMA. They can be used by a completely different application and different set of users. The bit mask is just a way of narrowing the context of the `s` parameter (to define when it wraps). It causes no direct effect on address space utilisation, but without it, it is believed aggregation would be a lot more difficult.

iii) Application programs and even routers should never need to know the list of multicast addresses that an AMA resolves to (other than for evolution from today's protocols). They will not need to bit shift along multicast addresses to find the value of `d`, then bit shift `d` bits back from the end to find the mask etc. The AMA tuple can be used by applications in all cases in place of listing the set of addresses it resolves to {this statement needs proving, mind - I haven't worked through AMA aggregation, expansion and contraction maths yet, but we live in hope!}.

iv) A discussion of the issues if this turns out to not always be the case is under the section on "Further storage optimisation".

v) There are some places where the "p word" really is appropriate.