Multimedia Conferencing on the Internet

- Mark Handley
  - University College London
- Ian Wakeman
  - University of Sussex

Overview

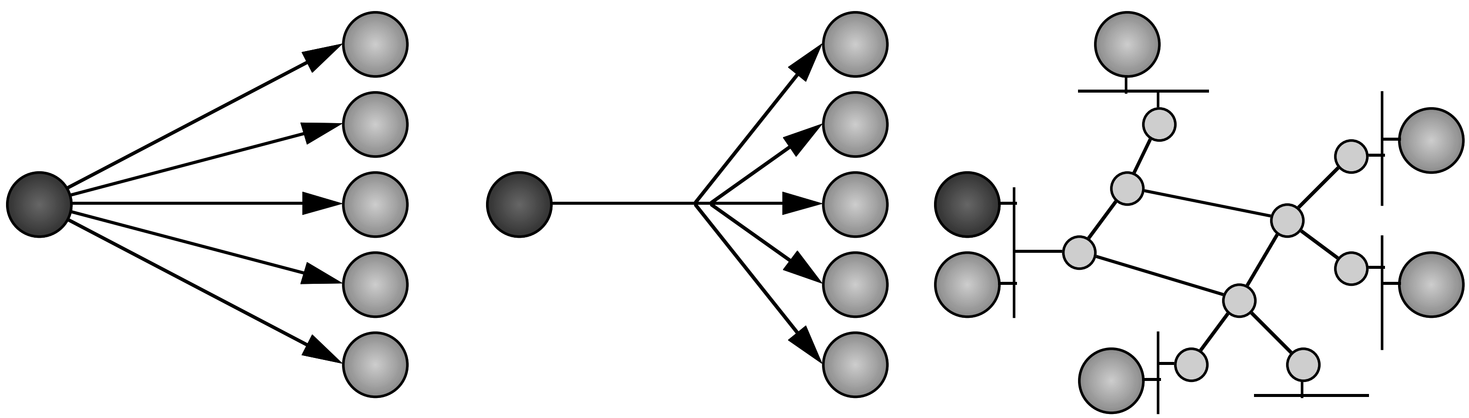
- IP Multicast
  - DVMRP
  - PIM
  - Configuring mrouted
- Security
  - Encryption
  - Scoping
  - Firewalls
- Real-time traffic on the internet
  - Jitter and Adaptive Playout
  - RTP and RTCP

Overview (cont)

- Multimedia Conferencing Applications
  - vat, rat
  - vic, nv, ivs
  - wb
  - sd, sdr
- The Future
  - RSVP
  - Int Serv

Requirements

- Support many to many conferencing
- Scale to large groups
  - Data should only traverse each link once
- Robustness to link failures
  - No fate sharing
  - End systems don't know topology



User's view *Sender's View* *Network Topology*

IP Multicast

- Dynamically Constructs efficient, shortest path distribution trees from senders to receivers
- Service Model
  - Receivers announce interest
  - Senders just send
  - Routers conspire to deliver sender’s data to all interested receivers

Class D addresses

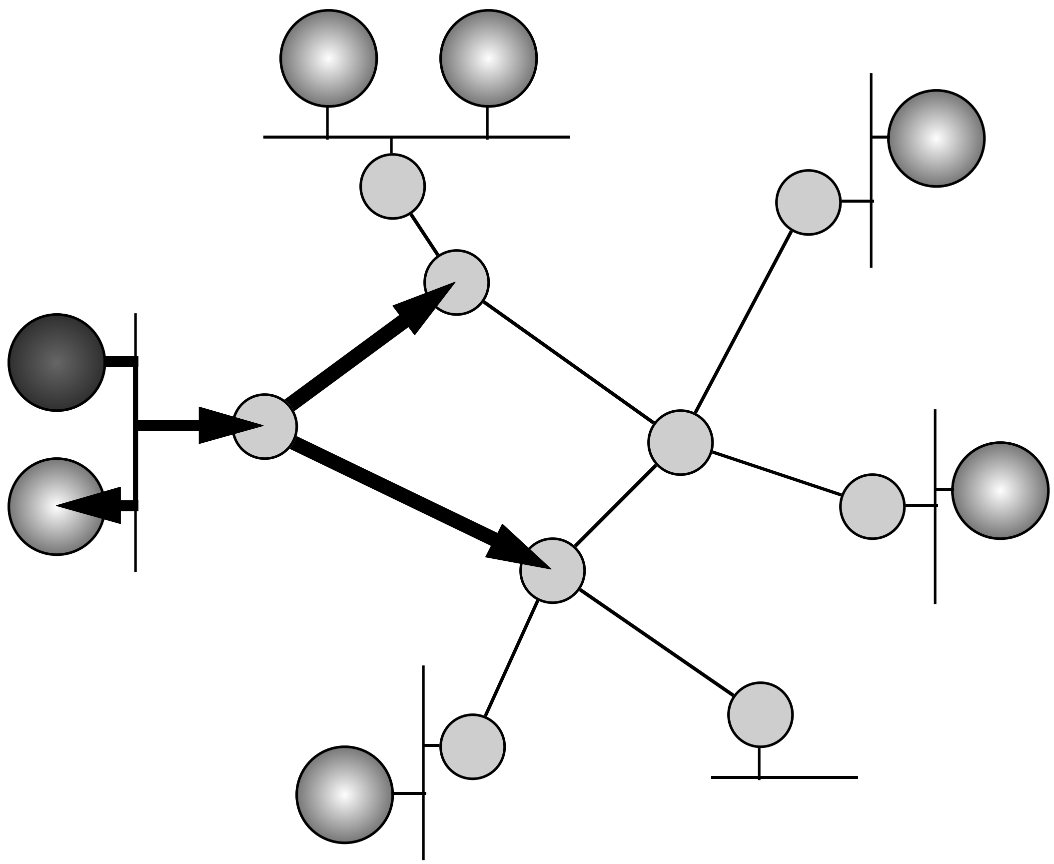
- Traditionally, IP addresses were divided into classes:
  - Class A for companies with millions of hosts
  - Class B for companies with thousands of hosts
  - Class C for companies with hundreds of hosts.
- Class D is multicast group addresses.
  - A multicast sender just sends to a class D multicast address
  - Multicast receivers express an interest in a class D multicast address.
  - If they choose the same address, the network delivers traffic from the senders to the receivers.
  - Class D addresses are in the range 224.0.0.0 to 239.255.255.255, with 224.2.*X.Y* allocated for conferencing.

IP Multicast routing models

- DVMRP
  - Distance Vector Multicast Routing Protocol
  - forms the basis of most of the current Mbone
- PIM
  - Protocol Independent Multicast
  - offered in Cisco router
- MOSPF
  - Multicast OSPF
- CBT
  - Core Based Trees
  - from UCL
  - better scaling properties at the expense of non-shortest path trees.

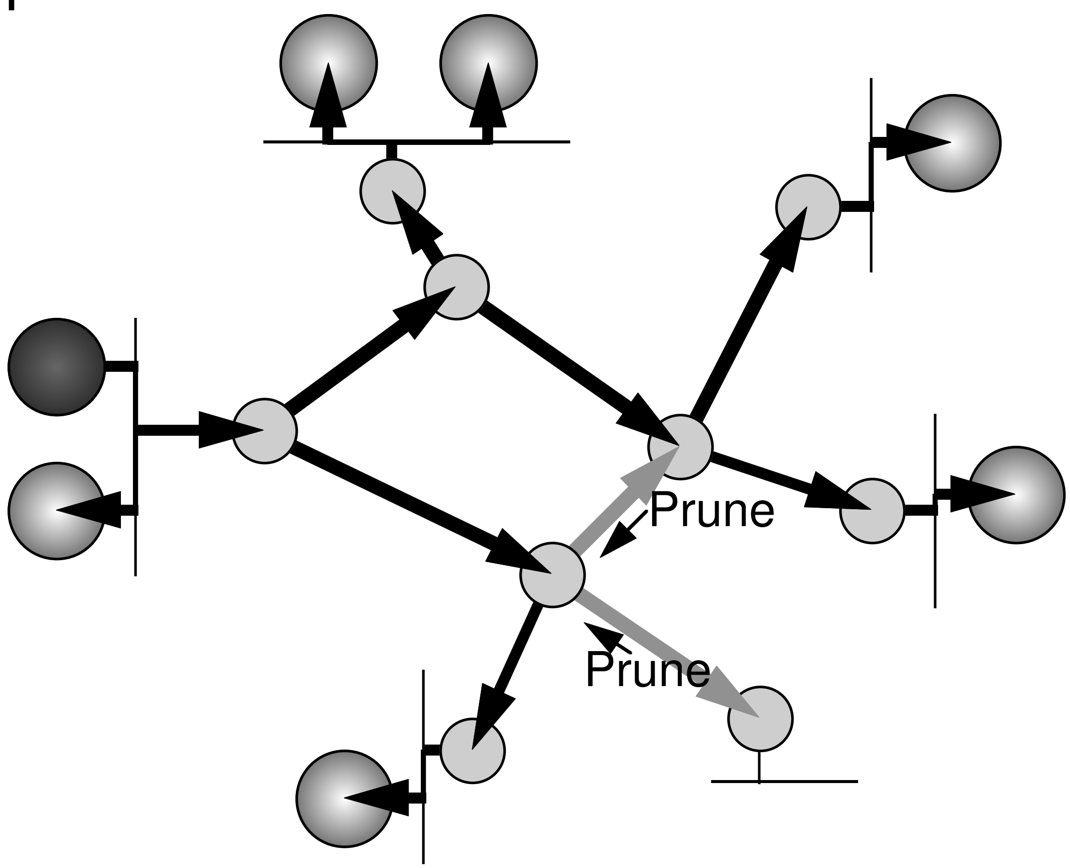
DVMRP (and DM-PIM)

♦ Basic Model is “Flood and Prune”



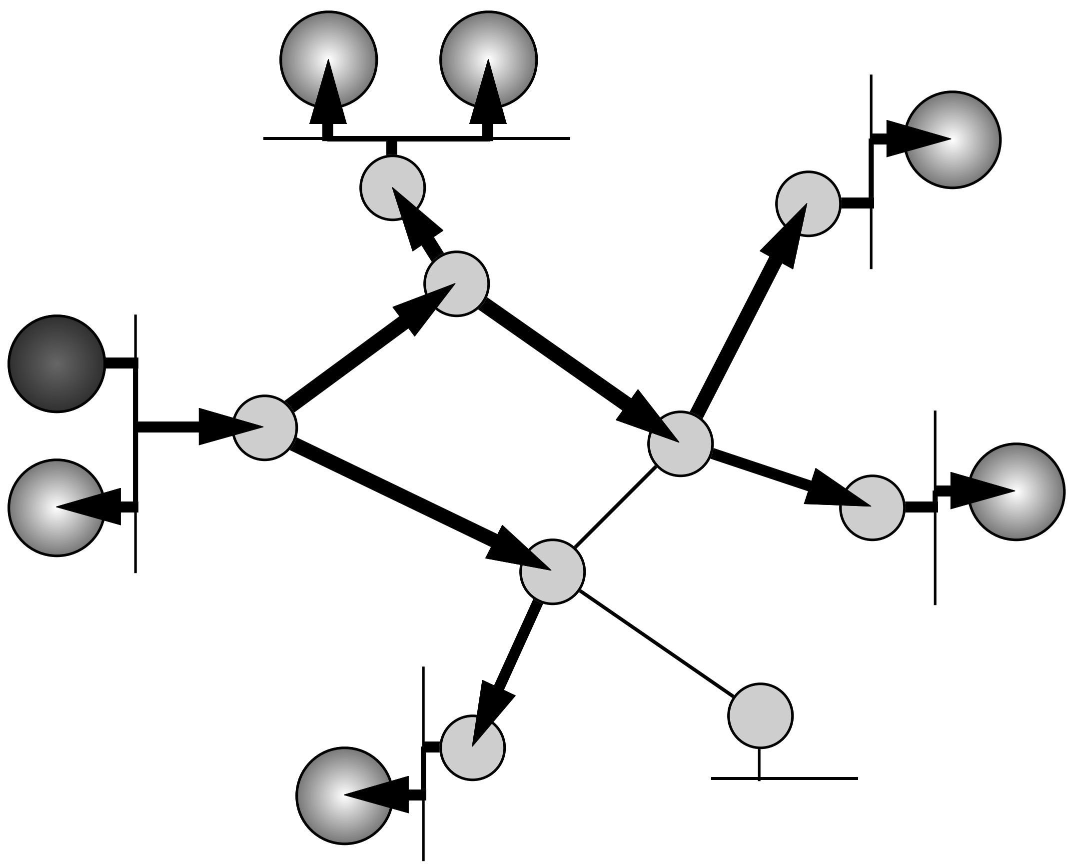
Flood and Prune

Branches not on the shortest path tree to a receiver are pruned off:



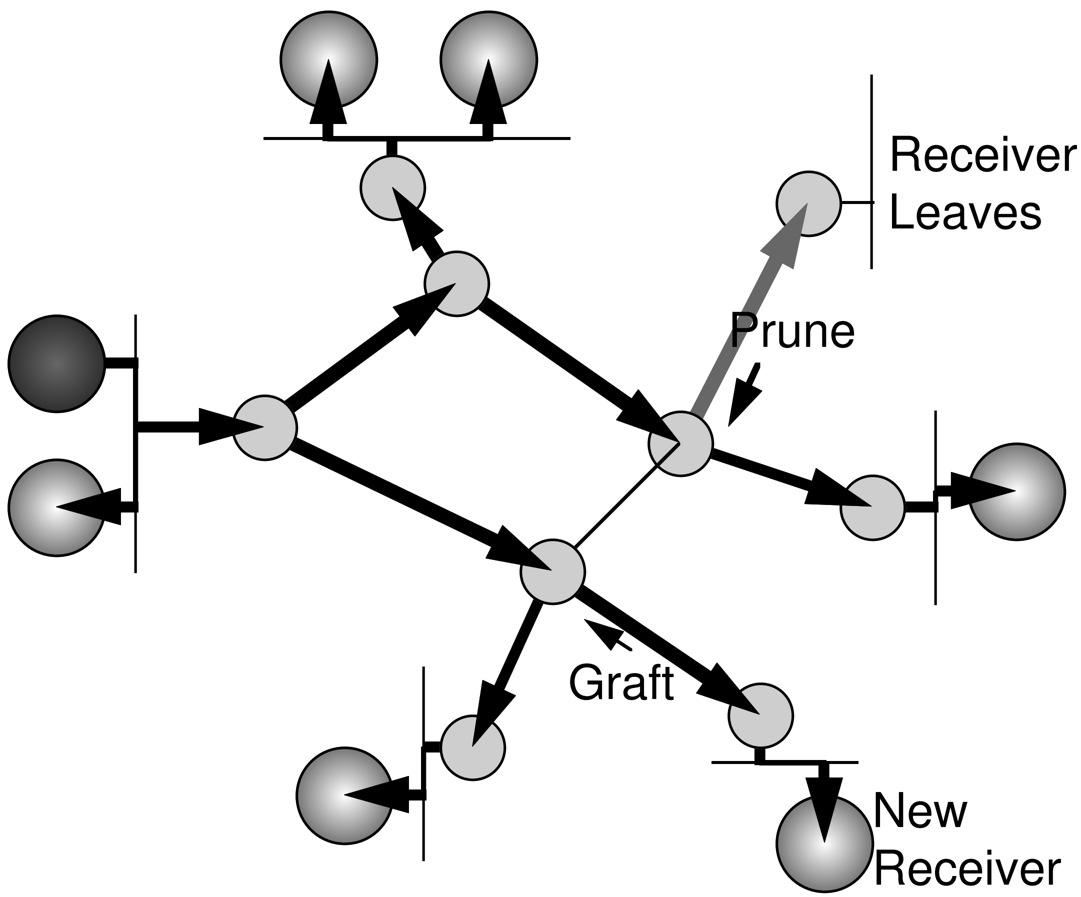
Shortest Path Tree

Resulting in a shortest path distribution tree rooted at the sender:



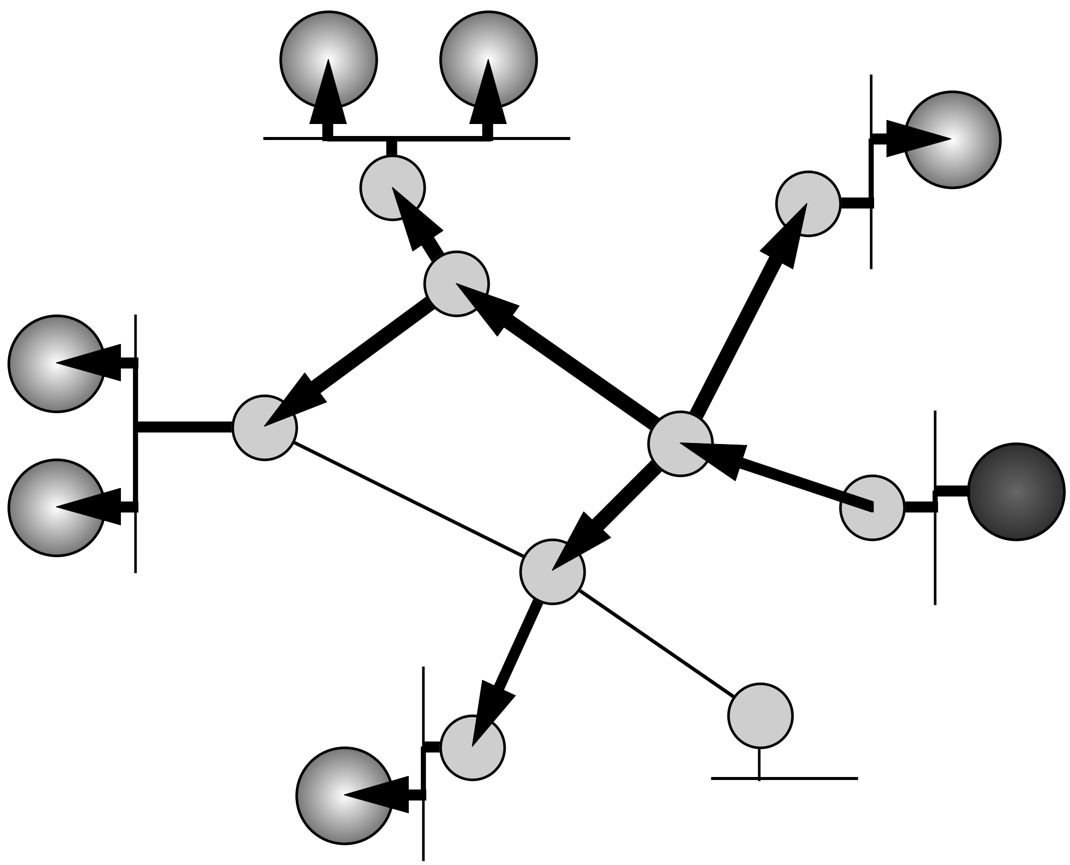
Local Tree Maintenance

Changes in membership are handled efficiently and locally via grafting and pruning:



A Shortest Path Tree per Sender

Distribution trees are per-{sender,group}, and are triggered by data packets:

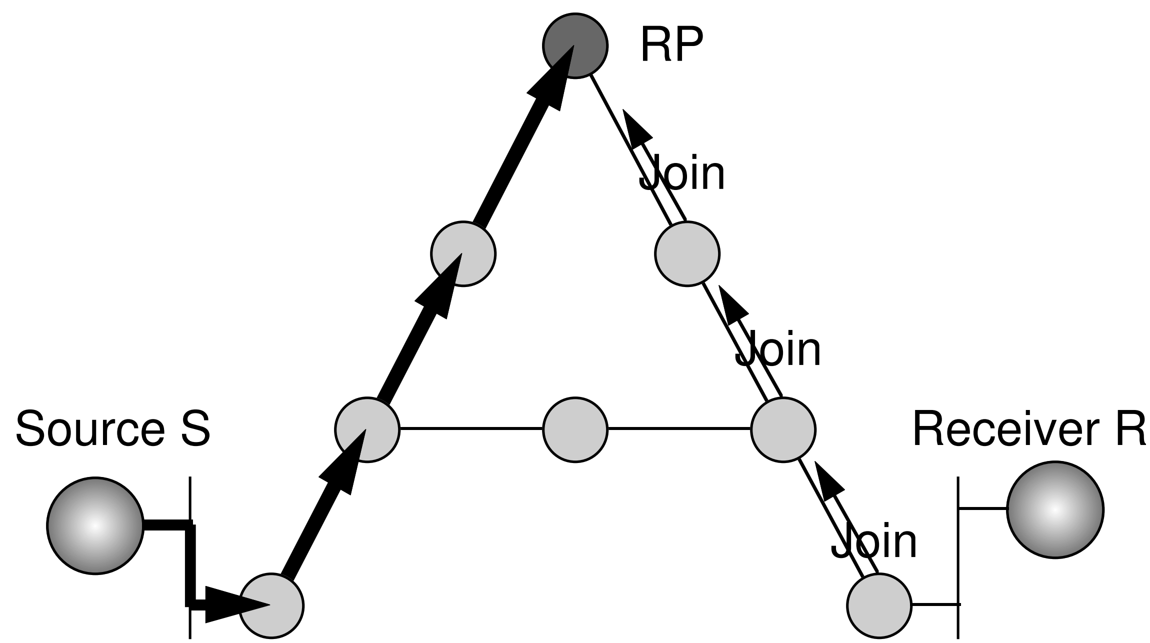


SM-PIM

- Instead of flooding and pruning to directly build a shortest path tree, Sparse-mode PIM initially builds a shared tree.
- The shared tree is built by sending join messages towards a Rendezvous Point (RP)
- Once data is flowing, the shared tree can be converted to a shortest path tree if required.

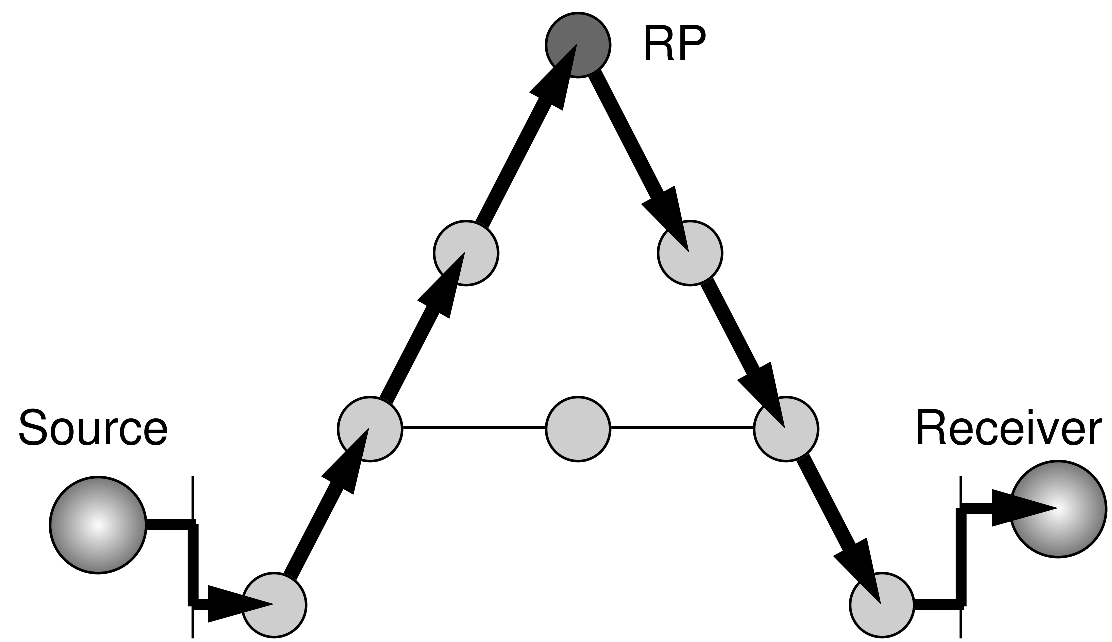
SM-PIM: Building a Shared Tree

- Source S sends traffic to group G
- S's local router forwards group G traffic to the Rendezvous Point (RP) for group G
- R's local router sends a join message for G to RP



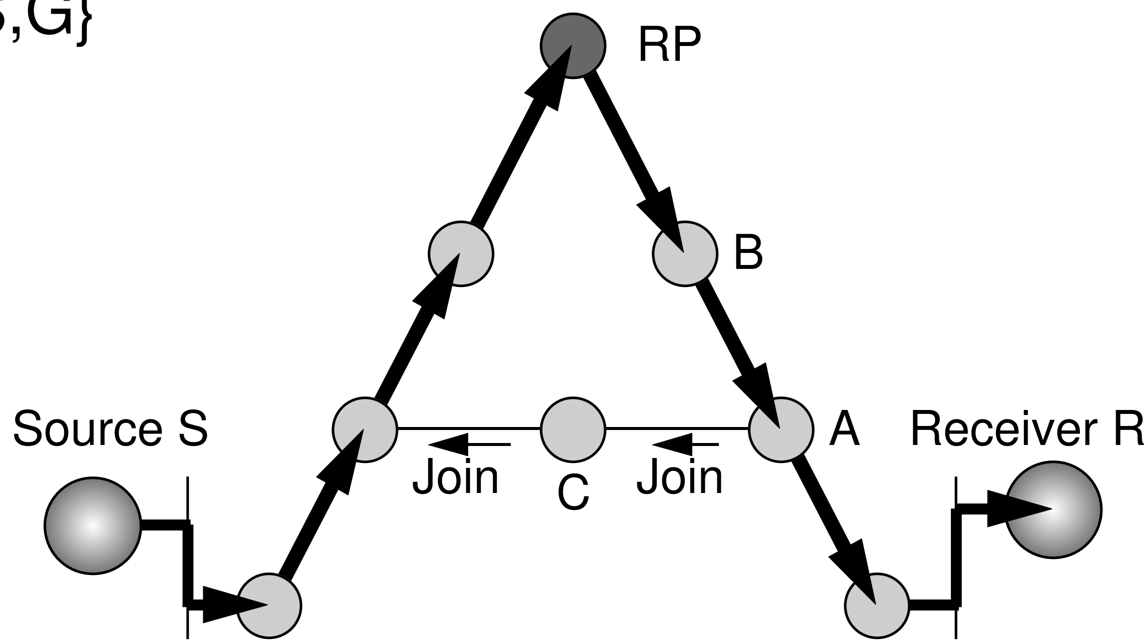
SM-PIM Shared Tree

- Initially data flows over RP tree



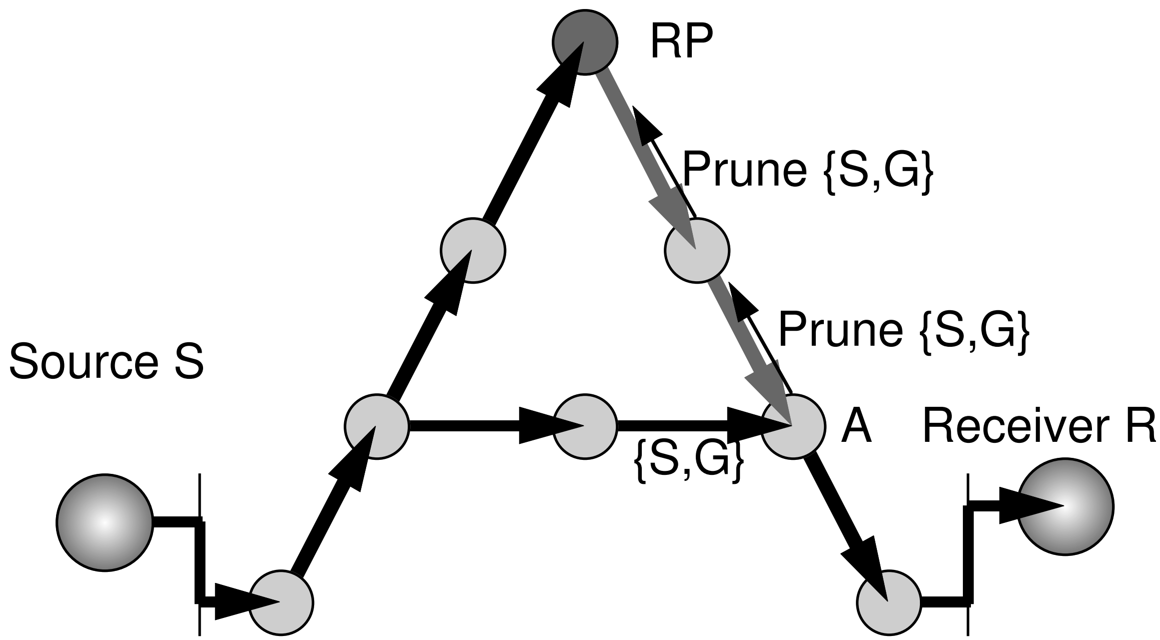
Transfer to SP Tree

- Router A sees traffic from S arriving over its B interface, when its unicast route to S is over its C interface.
- If traffic warrants it, A sends a join towards S for {S,G}



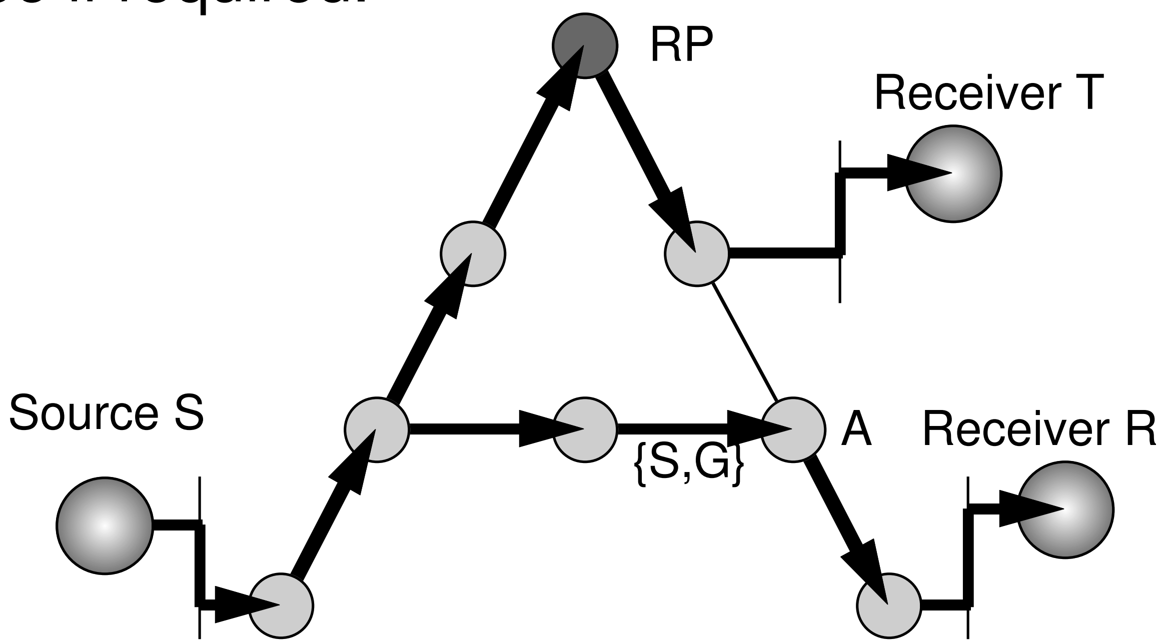
Transfer to SP Tree (cont)

- When the Join towards S is successful, group G traffic from S starts to flow along the shortest path tree. A now receives the traffic twice.
- A then sends a prune for {S,G} towards the RP for group G, and the transfer to SP tree is complete.



Sparse Mode PIM

- Receiver R receives {S,G} on the shortest path tree.
- Receiver T continues to receive {S,G} on the RP tree, although it too can switch to a shortest path tree if required.

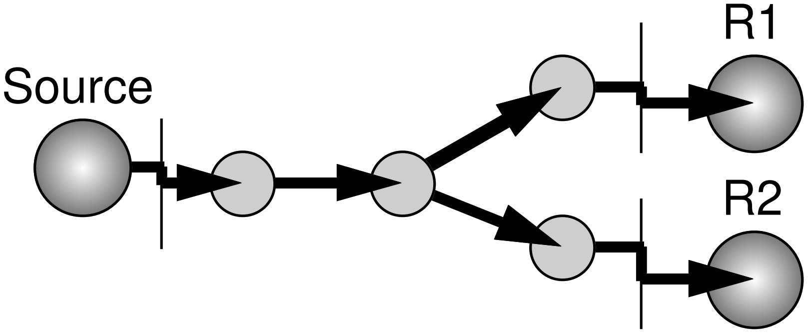


Routing, Tunnelling and the Mbone

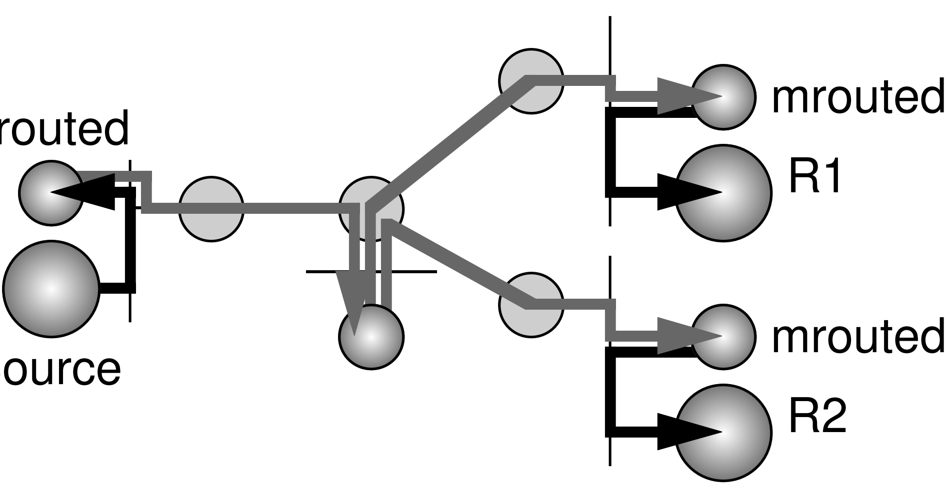
- PIM normally uses the underlying unicast routing to determine its tree formation.
- DVMRP implements its own routing protocol for multicast traffic independently of unicast routing.
- In the current Mbone, DVMRP is usually implemented as an overlay of multicast <u>tunnels</u> because most unicast routers do not yet support multicast.
- In a multicast tunnel, an IP multicast packet is encapsulated in a unicast packet to carry it across multicast unaware links and routers.

Multicast Tunnels

Native Multicast in Routers:

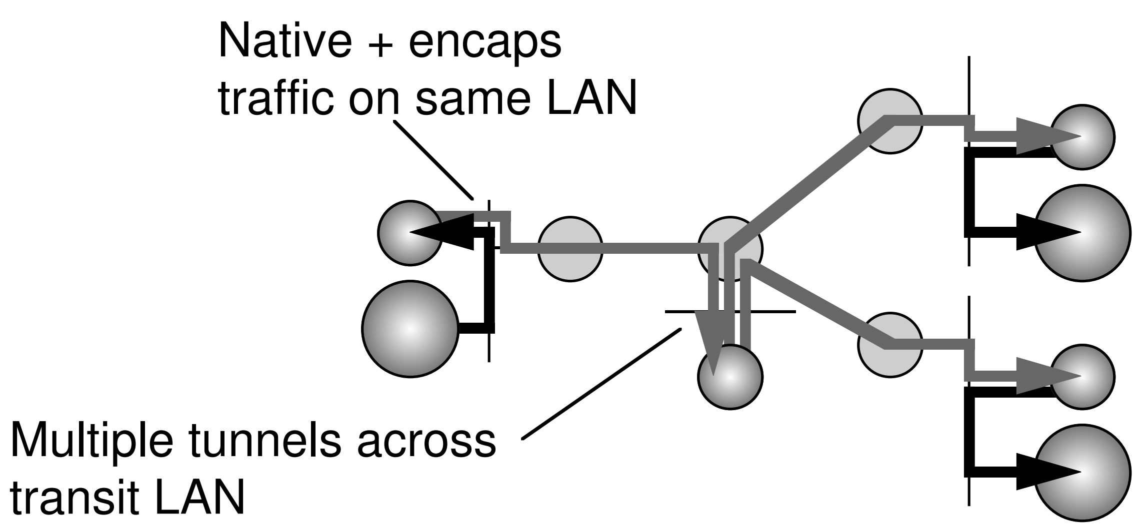


Multicast Tunnelling in workstation mrouters:



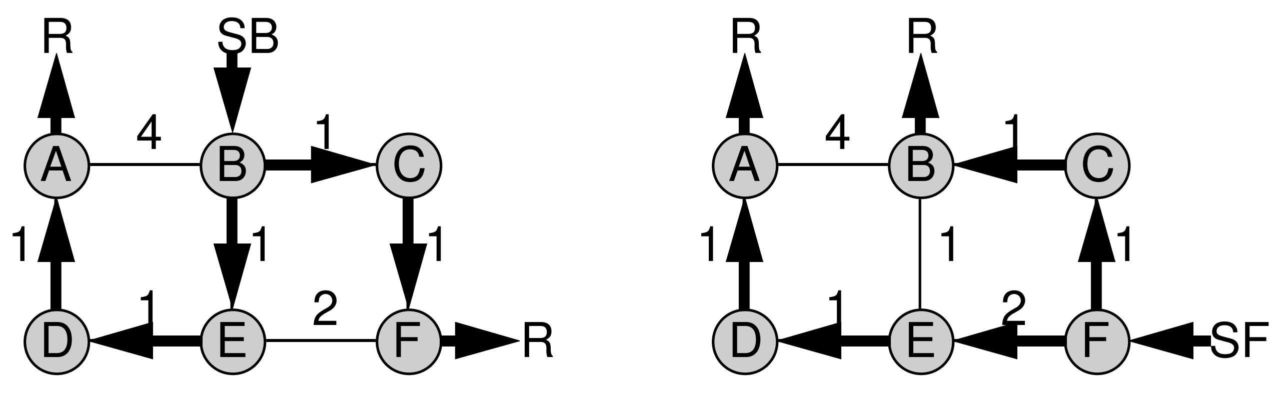
Multicast Tunnelling (cont)

- Multicast can be tunnelled across multicast unaware links, but this can limit performance.
- Workstations are not ideal routers
- At best, multicast traverses your LAN twice
- At worst, many tunnels traverse your LAN
- Tunnels require manual configuration



DVMRP routing

- Each link is given a routing metric (or cost)
- Given two alternative paths from source to a router, the higher cost path is pruned, and the lower cost path used.
- E.g., there are two paths from SB to F. The path via E (cost 3) is pruned and the route via C (cost 2) is retained.



DVMRP routing (cont)

- DVMRP is basically multicast RIP.
  - It was not really intended to be used as one global routing domain (as it is currently in the Mbone)
- DVMRP considers a route with metric 32 to be “infinite”.
  - This helps it prevent loops and achieve convergence.
  - It also is uncomfortably small, so it is advisable to use as small a metric on your links as possible.

DVMRP scoping

- mrouted supports two mechanisms to limit the scope of multicast traffic:
  - **TTL scoping**
  - **Administrative scoping**

Time to Live

- When any IP packet (unicast or multicast) is sent, a field called "Time to Live" (TTL) is set to some positive value.
- Each router the packet passes through decrements the packet by one.
- When the TTL field reaches zero, the packet is dropped. This is supposed to prevent packets looping in the internet forever if a routing loop occurs.
- mrouted leverages the IP TTL field to limit the scope of multicast traffic to particular regions.

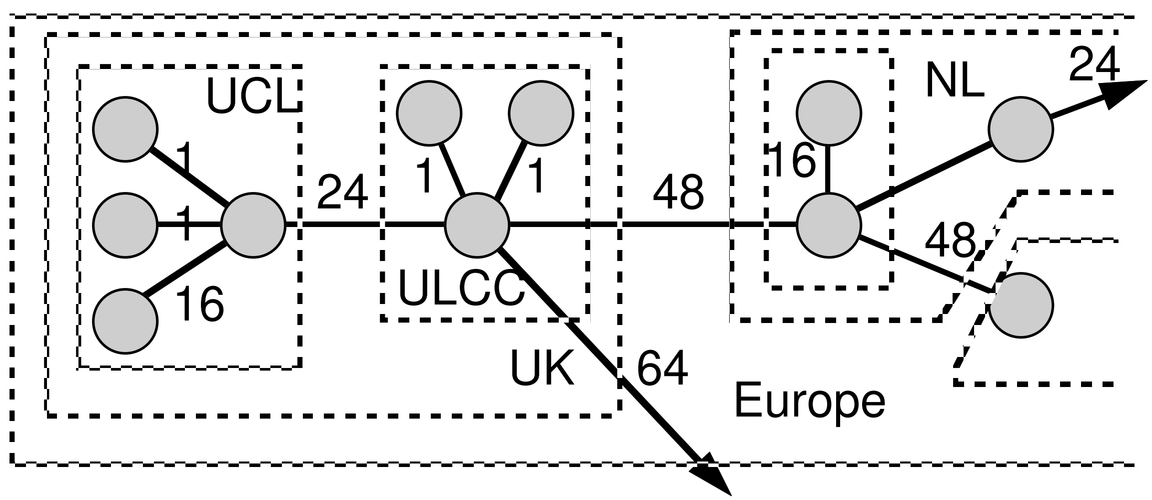
TTL Scoping

- ♦ When an application is started, it is configured to send its data with a particular TTL.
- ♦ Each multicast interface or tunnel is configured with a TTL threshold.
- ♦ Every time a multicast packet passes through a multicast router, its TTL is decremented.
- ♦ If a packet has a TTL less that the threshold of an interface or tunnel, the packet is <u>not</u> forwarded out of that interface or tunnel.
- ♦ If all the tunnels out of a region have the threshold *x*, then sending data with TTL *x*-1 will confine the traffic to that region.

TTL Scopes

The well known thresholds in use in the UK are:

- 1, 16 - links within a site boundary
- 24 - links to high rate Superjanet sites
- 32 - links to lower rate Superjanet sites
- 48 - links to other European countries
- 64 - links to other continents.
- 128 - very low rate links.



Administrative Scoping

- In addition to TTL scoping, mrouted 3.3 or later can be configured so that interfaces or tunnels are administrative boundaries.
- This is done by specifying ranges of multicast addresses which the interface or tunnel will not forward.
- Group addresses in the range 239.0.0.0 to 239.255.255.225 can be administratively scoped.
- Note:
  - Administrative scope boundaries are bi-directional.
  - Administrative scope zones must be concave with respect to routing metrics.
  - It is possible to prune back from an administrative scope boundary but not from a TTL boundary.

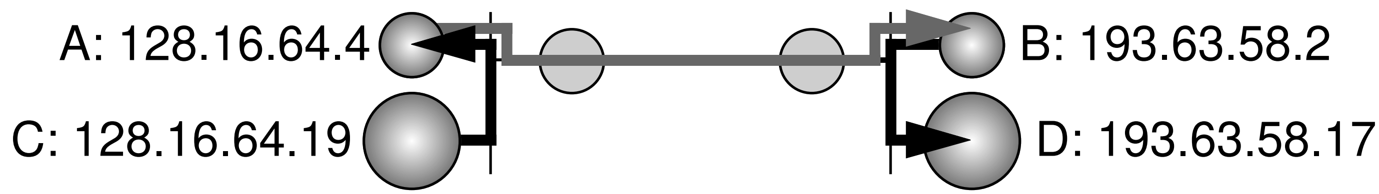
mrouted

- mrouted is the workstation implementation of DVMRP multicast routing.
  - it supports both native multicast (usually on an ethernet) and multicast tunnelling.
  - it runs on many unix platforms, including SunOS 4.1.3, Solaris 2.4, IRIX 5.3, HPUX 9.05, OSF1/2.0, Ultrix 4.4, and NetBSD.
  - not all platforms support all versions of mrouted.
  - mrouted 3.5/3.6 is strongly recommended.
- mrouted 3.x requires kernel modifications on most platforms.
  - these are supplied as part of the IP multicast package.
  - Note: it is vital that the appropriate mrouted is run with the appropriate multicast kernel changes
    - mrouted 3.4 runs with multicast kernel mods 3.3
    - mrouted 3.6 runs with multicast kernel mods 3.5

mrouted configuration

- mrouted is configured from `/etc/mrouted.conf`
  - `mrouted.conf` should contain all your multicast tunnel definitions, plus and traffic rate limiting
  - by default, mrouted will try to do multicast forwarding on all multicast capable interfaces on its host
    - `mrouted.conf` can be used to disable multicast forwarding on a physical interface if required
  - if two multicast routers are on the same physical multicast capable network, they will automatically use that network to communicate and forward multicast traffic - no tunnel is needed.
  - if two multicast routers are not on the same subnet, and you wish them to forward traffic to each other, a multicast tunnel is required, and must be configured in `mrouted.conf`

configuring DVMRP tunnels



- Each tunnel must be configured with:
  - local interface
  - remote interface
  - routing metric
  - TTL threshold
- On host A in mrouted.conf:

  tunnel 128.16.64.4 193.63.58.2 metric 1 threshold 48
- On host B in mrouted.conf:

  tunnel 193.63.58.2 128.16.64.4 metric 1 threshold 48
- Note: asymmetric tunnels are a bad idea!

configuring DVMRP tunnels (2)

- Optionally, a DVMRP tunnel may have:
  - a rate limit, measured in Kbits/second:

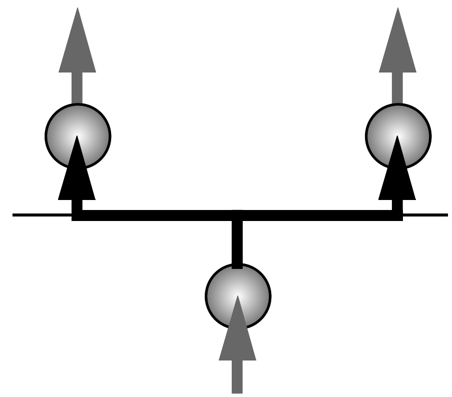
    tunnel *<me> <him>* **metric 1 threshold 48 rate_limit 256**

    - The default rate limit is 512Kbps
  - one or more boundary definitions, defining administrative scope boundaries:

    tunnel *<me> <him> ...* **boundary 239.123.45.0/24**

    - the parameters are of the form <address>/<no of bits>
    - thus this tunnel would be a boundary for traffic on multicast addresses 238.123.45.*X*, for 0<=*X*<=255
    - Note: admin scope boundaries can only be defined for addresses of the form 239.*X.Y.Z,* not for other class D addresses.

DVMRP on broadcast LANs

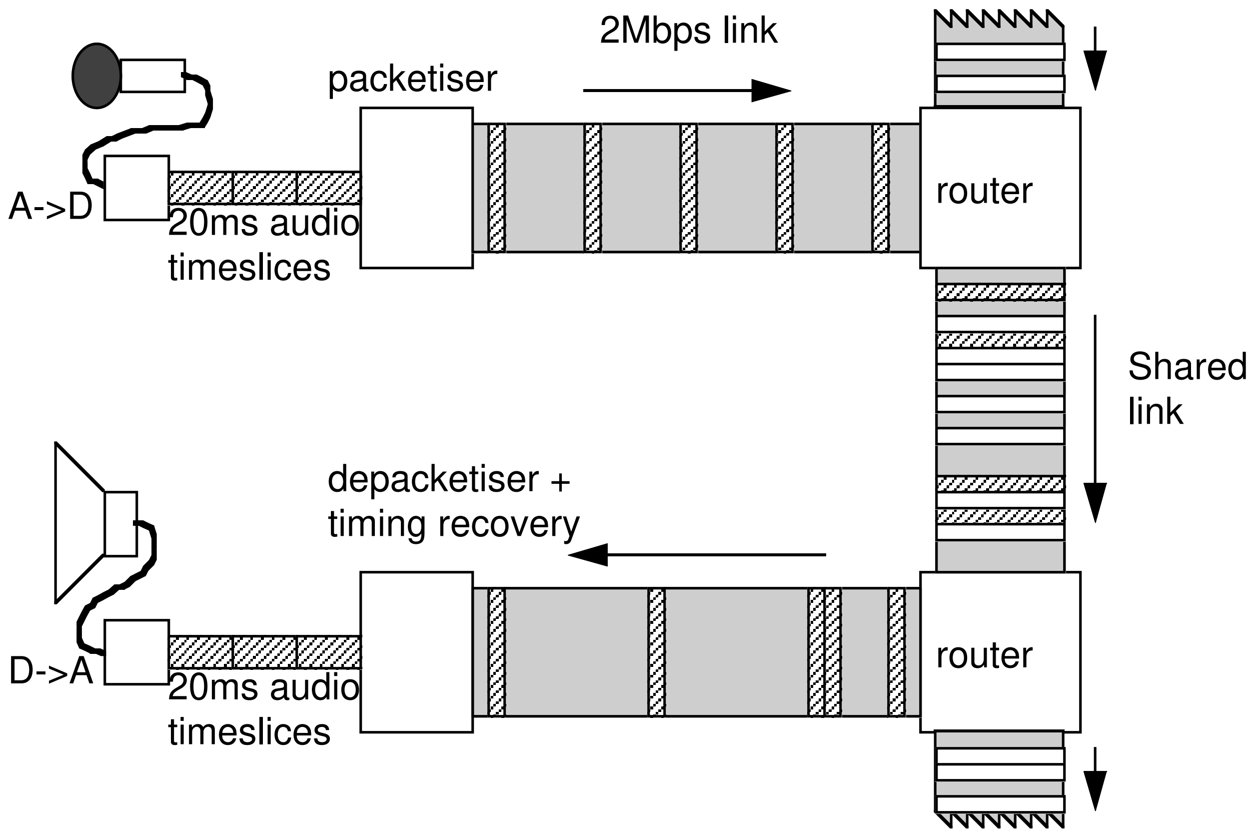


- mrouters connected by a multicast capable LAN (such as an ethernet) do not need a tunnel to connect them - they receive the native multicast packets off the LAN, and forward them out of their other tunnels or interfaces.
- The mrouters elect a querier, who is responsible for periodically querying (via IGMP) whether there are any local receivers.

Real-time Traffic

- Internet links are shared
  - with other users applications
  - with other applications in the same conference
  - with routing protocols
- Timing relationships between your data packets are not preserved
  - because of queuing in routers
  - because of routing changes
- Statistical multiplexing gives efficient utilisation
- Overloads lead to packet loss
  - due to transient congestion
  - due to underprovisioned links

Jitter (delay variance)



– jitter is caused by some packets being delayed for longer than others.

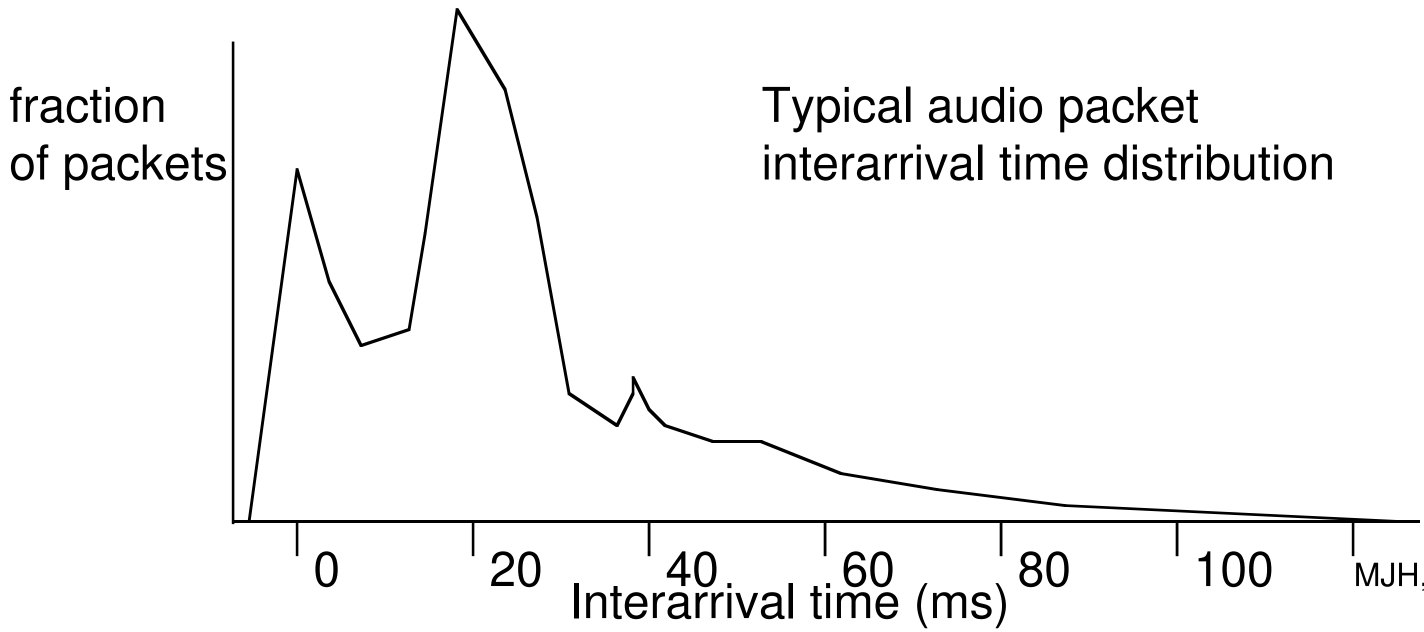
– packets can also be lost or re-ordered.

Adaptive playout

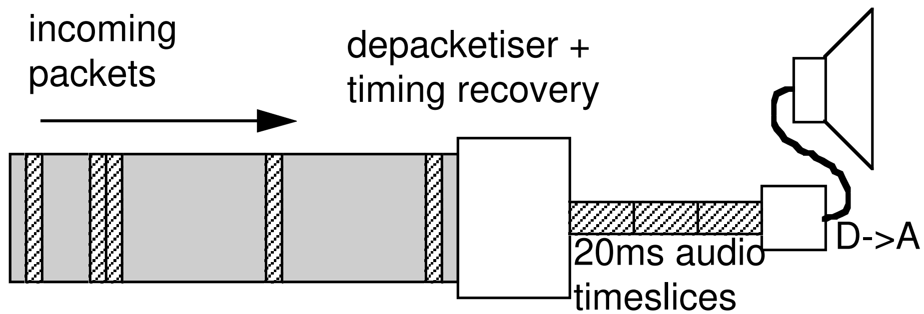
- Note: the average arrival rate at the receiver must equal the departure rate at the sender (otherwise the network doesn't have enough capacity for the data stream)
- As the mean rate is the same at sender and receiver, the receiver only has to adapt to the variance of the arrival rate.
- If each packet carries a sender timestamp, the receiver can reconstruct the original variance by buffering before playout.
- But IP doesn't bound delay, so if the delay isn't bounded, isn't the buffering required unbounded too?

Adaptive playout (2)

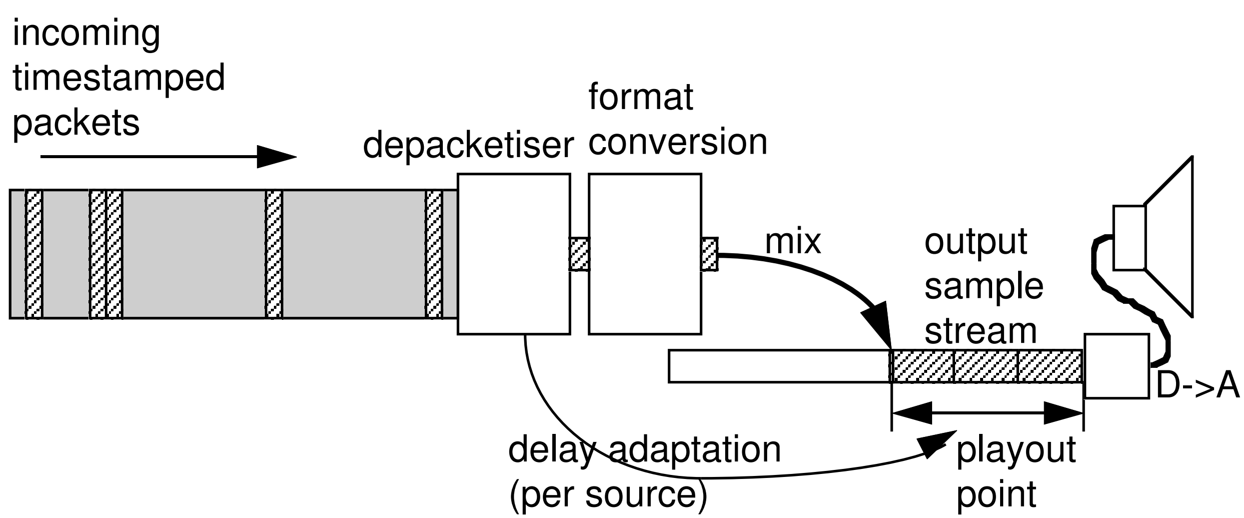
- However, delay is bounded by queues in the routers, and this buffering is bounded.
- Queue size fluctuation and thus jitter is bounded and relatively small.
- Queue sizes along the path also vary, so tend to cancel each other.



Adaptive playout (3)



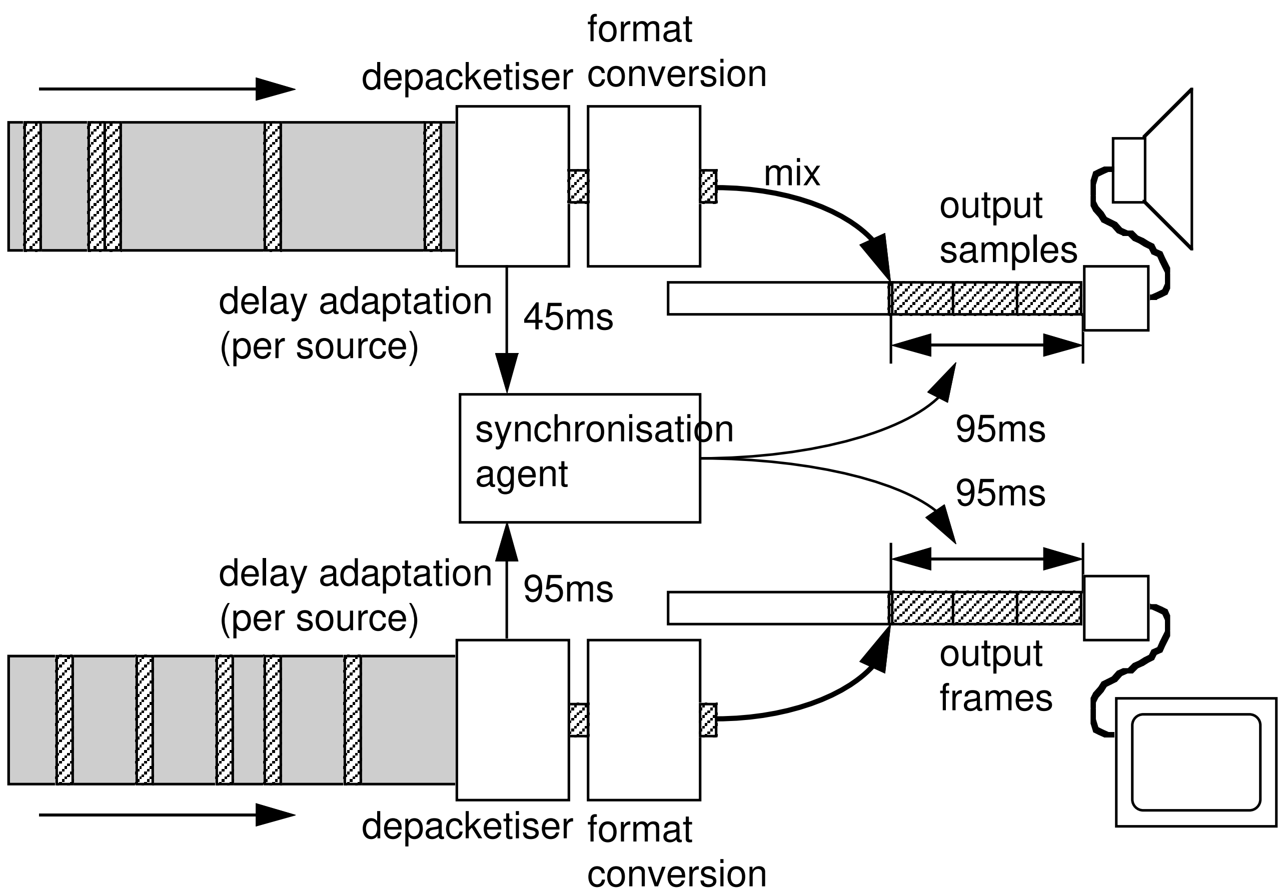
so our adaptive receiver now looks like:



Adaptive playout (4)

- With an audio stream, adapting the playout point can only be done inaudibly during silence (between speech bursts).
  - Disabling silence suppression thus removes the ability to adapt.
  - This can be noticeable as not all audio devices are clocked at exactly the same rate!
- It is possible to trade off quality (low loss) against interactivity (low delay).
  - In an interactive session, the playout delay can be kept small by discarding very late packets.
  - In a seminar (broadcast), interactivity is not important, so the playout delay can adapt to the maximum required.
  - Knowing the purpose of the data stream is important as the delay distribution can have a long tail.

Cross-media synchronisation



Cross-media synchronisation (2)

- Each media tool calculates its own playout point in terms of its own media timestamps.
- Media timestamp based playout points are then translated to real-time playout points, and then communicated to a synchronisation agent.
- If the sync agent decides synchronisation is desirable/possible, it communicates the largest of the media playout points back to all media tools.
- The media tools then translate the real-time playout point back to a media timestamp playout point, and use it to set the playout delay.

Session Packets

- Periodically, each media tool in the conference multicasts a session packet.
- Session packets convey:
  - the participants name (and thus conference membership)
  - information to match up different media streams from the same source
  - the relationship between media timestamps and real time
  - reception reports (loss, jitter, etc)
- The total session packet rate is kept constant
  - Each participant can see how many other participants are in the session, and reduces its session packet rate as the number of participants increases.
  - The total session packet rate is chosen to be a small fraction (typically 1-5%) of the data rate.

The Standards

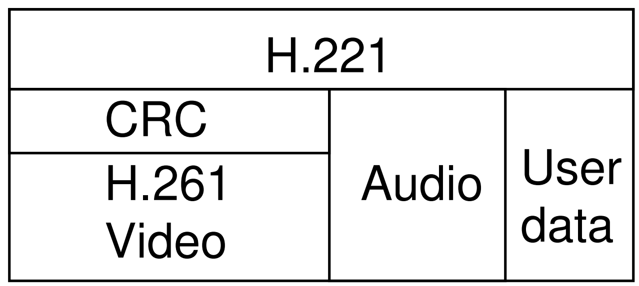
- RTP (Realtime Transport Protocol)
  - This is the IETF's proposed standard for real-time data traffic on the internet (now stable after 3 years of evolution)
  - It's a packet header format, giving payload type, sequence numbers, timestamps, and source identifiers.
  - It's also a set of conventions for how to use the format.
- RTCP (Realtime Transport Control Protocol)
  - part of the RTP proposed standard, it specifies packet formats and usage convention for session packets in multiparty multimedia sessions.
  - a number of RTCP packet types are defined:
    - sender report
    - receiver report
    - membership report
    - bye, etc

Usage of the standards

- Associated with the RTP standard is the RTP Audio/Video Profile, and a set of Payload Format documents.
- These specify how RTP is used to convey particular encoding types.
- These include (amongst others):
  - Audio: PCM, ADPCM and GSM
  - Video: H.261, Motion JPEG and MPEG video
- Some of these Payload Formats are still evolving.
  - In particular, after 2 years of usage, the H.261 payload format was recently re-written (for good reasons), meaning not all H.261 tools will currently interoperate.

A note on ITU standards

- The basic ITU standard for videophones is the H.320 family for px64Kbps circuit switched lines.
- This includes H.261 video, various audio coding schemes, a user data raw channel, and H.221 serial line framing:



- H.221 is totally inappropriate for packet switched networks.
- H.261 on its own is OK for packet switched networks when used with RTP as transport.

A note on ITU standards (2)

- The T.120 family is the ITU's model for multiparty conferencing and conference control.
- T.120 assumes reliable circuits and Multipoint Control Units as its infrastructure .
- Mbone based conferencing is decentralised with no MCU functionality:
  - direct multicast is more flexible than MCU mixing.
- T.GCC (the ITU's Generic Conference Control) will probably be applied in the internet for tightly coupled conferences, although it is not so generic!

A note on ITU standards (3)

- The ITU is currently working on putting T.120 over packet switched LANs (ethernets, etc).
- Their model is long-distance traffic is circuit switched, but local distribution may be on your company LAN.
- They are currently working on standards for this, and are talking to the IETF about using (or modifying) RTP as a H.221 replacement for the LAN.
- The jury is still out on this, and on conference control in general.

Mbone Conferencing Tools

The most widely used tools are:

Audio:

- **vat**, Lawrence Berkeley Labs' *Visual Audio Tool*

Video:

- **nv**, Xerox PARC's *network video*
- **vic**, Lawrence Berkeley Labs' *video conference* tool
- **ivs**, INRIA's *INRIA videoconferencing system*

Whiteboard:

- **wb**, Lawrence Berkeley Labs' *whiteboard*

Session Directory

- **sd**, Lawrence Berkeley Labs' *session directory*

vat - sending

- Simplest way to use is push-to-talk
  - press the right mouse button anywhere in the vat window to transmit. release the mouse button to stop transmitting.
- An alternative is to use the built-in silence suppression
  - unmute the mike, then vat will only transmit when you talk.
  - the silence suppression is not perfect!
- Echo suppression :
  - vat has three modes:
    - net mutes mike
      - a good default
    - mike mutes net
      - should only be used by a chairman , and only then with push-to-talk
    - full duplex
      - only really effective with headsets

vat - sending

- Setting your mike volume correctly is the most important point
  - aim for about 80% on the mike bargraph.
  - if it's often in the red, it's too loud.
  - you can set your default mike gain from your X resources
- You're only sending when your name is highlighted!
  - many people start talking without unmuting their mike!
- Check your mike
  - enable loopback to see what the quality is like
  - do this whenever you plug in a new mike

vat - encoding

- vat has 8 encoding modes
  - PCM - 64Kbps pulse code modulation in 20ms packets
  - PCM2 - 64Kbps pulse code modulation in 40ms packets
  - PCM4 - 64Kbps pulse code modulation in 80ms packets
  - IDVI - 32Kbps adaptive delta modulation in 20ms packets
  - DVI2 - 32Kbps adaptive delta modulation in 40ms packets
  - DVI4 - 32Kbps adaptive delta modulation in 80ms packets
  - GSM - 14Kbps Group Speciale Mobile (cellphone) in 80ms packets
  - LPC - 4Kbps Linear Predictive Coding in 80ms packets
- note: the above data rates don't include packet headers
- pcm2 or dvi2 are good defaults

vat - receiving

- The vat packet headers describe the encoding
  - vat will decode any of the encoding formats automatically
- The names of people sending are highlighted
  - the small boxes next to the names indicate the last few people to talk
  - clicking in these small boxes mutes the individual sender (so you can ignore someone sending garbage)
- Clicking on the loudspeaker logo switches between headphones, speaker, or line out.
  - except on SGIs where it is dependent on whether you have headphones plugged in!
- Click shift-left on a person’s name pops up receiver statistics.

vat - receiving

- You can run multiple vats on one machine at once:
  - but only one of them can hold the audio device at a time.
  - vats communicate so that if only one of them has incoming audio, that vat will grab the device and raise its window.
  - if more than one vat has incoming audio, you can determine which one gets the audio device by setting the priority.
  - more frequently, you only want to pay attention to one at once, so simply press “Keep Audio” to ensure that vat keeps the audio device.
- Vat has two playout buffer modes:
  - conference mode
    - try to keep the playout delay low (interactive)
  - lecture mode
    - let the playout delay increase (higher quality)

vat - versions

- The current version of vat is 3.4 (in Oct '95)
  - This uses the "vat" packet format, not RTP
- An alpha version of vat 4.0 has just been released.
  - This can use either the vat protocol or RTP.
  - The default is still the vat protocol, but increasingly people will start to use RTP for audio.

video compression

- ♦ video compression schemes fall into a number of categories:
  - – Discrete Cosine Transform based *
    - + loses high frequency components
  - – Wavelet encoding based *
  - – Fractal based
  - – Model based

    “head position 42, mouth position 7, eye position 3”
  - – others:

    cell B is basically video compression by colour dither lookup table *.

* - realistic in software on today’s fast workstations.

video display

- video framebuffers generally fall into one of four categories:
  - black and white
  - greyscale
  - 8 bit colour
  - 24 bit colour
- 8 bit colour requires colour dithering:
  - your only have 256 colours available
  - choose a standard colour table
  - take a block of 4 pixels - average their colour
  - choose the colours from your table that when mixed together best match the average colour of your 4 pixel block.
- 24 bit colour requires transferring 3 times as much data to the display hardware

video compression (DCT)

- DCT based schemes divide the image up into blocks.
- Each block in the new frame is checked against the decoded block from the previous frame.
  - only blocks that have changed more than some threshold are compressed and sent.
- To compress a block:
  - transform the block into frequency space using a Discrete Cosine Transform.
  - quantise and run-length encode the discrete cosine transform.
  - as most of the power in the DCT space of a typical image is in the low frequency components, the high frequency components tend to get lost in this quantisation.

video compression (DCT)

- Most of the common schemes (Motion JPEG, H.261, MPEG) are DCT based schemes.
- Motion JPEG does not do frame differencing
  - every block in the image is sent in every frame.
- H.261 and MPEG can also encode the *differences* between one block and another, rather than the block itself
  - this is called inter frame compression (vs intra frame).
  - inter-frame compression can get better compression, but is more succeptable to packet loss.
- H.261 and MPEG and can also encode motion as a *motion vector* plus a DCT of the difference.
  - this is really terrible with packet loss!

vic - sending

- to send video, you need a camera and a video framegrabber
  - a framegrabber takes analog video and digitises it.
  - some framegrabbers can provide a degree of compression.
  - some framegrabbers can digitise the image at various different resolutions.
  - note: 752 pixels * 625 lines * 25 fps * 12 bits = 140 Mbps.
  - normally we work at lower resolutions and lower frame rates, but it really helps if the hardware helps you do so.
- video tools are complicated by the need to support many different video framegrabbers.
  - not all copies of vic are the same - some have different framegrabber support built in.
  - ensure you have an appropriate binary for your hardware!

vic - encodings

- Vic 2.6 supports a number of encodings:
  - **nv** - Xerox PARC network video format
    - wavelet encoding compatible with nv tool with the `-Anv` flag
  - **cellb** - Sun Microsystems cell-b format
  - **scr** - nv with no compression (for screen grabbing, etc)
  - **jpeg** - motion JPEG format
    - uses hardware compression on some video framegrabber cards
  - **h261** - the ITU's H.261 video compression format
    - vic's implementation does intra-coding to make it more resilient to loss.
    - this is probably the best bandwidth/quality tradeoff for normal videoconferencing
    - needs the `-Aivs` flag to be compatible with IVS

vic - rate control

- Vic lets you control both the frame rate and the bandwidth sent.
  - Generally speaking, it is considered impolite to send more than 128Kbps at a TTL that takes it offsite.
  - On an SS20/61 with a Sunvideo card, around 15fps of CIF (352x288 pixels) H.261head-and-shoulders video should easily be possible at around 100 - 200Kbps.
  - Not all machines can *decode* this, especially if they have to display the video at CIF size or larger.
  - Data/frame rates depend on the compression scheme, the quality parameters, and on the images being compressed.
  - Generally speaking, a camera with no auto-focus and no auto-iris is a good idea.
  - If a monitor is in the field of view of the camera, compressing the screen scan can waste a lot of bandwidth.

vic - decoding problems

- Vic displays a small sized image of each video stream being received. Clicking on the small image displays that stream.
- If the vic reports no loss, fine. If it reports loss, then this often happens due to the receiving machine being overloaded.
- There are three main costs associated with receiving video:
  - decoding the compression scheme
  - colour dithering the image (on a 8 bit display)
  - transferring the image to the framebuffer

vic - decoding solutions

Any of these three main costs can be a bottleneck.

- “muting” video streams (including yourself) you’re not viewing stops vic having to decode them for the small thumbnail images
- switching to greyscale saves vic having to colour dither (as does buying a 24 bit display)
- displaying less important images small saves cuts the cut of transferring to the display
- asking the sender to reduce their frame rate is often the best option!

video - general

- sometimes it's worth getting those spec sheets out and doing a back of the envelope calculation:
  - if I want to decode 2 CIF (352x288) video streams to a 24 bit display, and encode video:
    - 352 rows * 288 lines * 15 fps * 3 bytes * 2 streams = 10Mbytes/sec (mean) just to transfer the data from memory to display card.
    - you probably want to spend 50% of your time decompressing data, 30% compressing data, 10% grabbing images from your framegrabber, and only 10% of your time displaying data.
    - this implies a bus speed to your display of 100Mbytes per second.
    - note: an 8 bit display implies 30% more time decoding whilst you do colour dithering.
- lesson: don't expect the impossible!

wb: whiteboard

- wb's model is that anyone can draw at any time.
- all drawing objects are owned by their creator:
  - only their creator can modify or delete them
  - you can locally "mute" all the objects of a particular user if they're being disruptive.
- wb's data model is entirely distributed
  - every user keeps a copy of all drawing objects.
  - you can leave the conference and rejoin - all you drawing (and everyone else's) will be refreshed from one of the other participants.
  - this means everyone must leave the conference to clear the whiteboard

wb: whiteboard

- wb has a concept of pages
  - normally, different people can see different pages at any time, but if someone draws, everyone will switch to the page they drew on.
  - this can be overridden by clicking on the page number, which disables the auto-page-changing.
- new pages can:
  - be blank
  - contain text loaded from a file
  - contain postscript images
    - by default, there's a maximum of 32KBytes per postscript page (multicast shouldn't really be used for file transfer!)
    - first, try compressing your postscript with *lzps*
    - as a last resort, the size limit can be overridden with the `-p` *<size>* flag.

sd: session directory

- sd is used to publicly announce multicast sessions
- it provides information about:
  - the name and purpose of the session
  - the time of the session
  - the media tools required
  - the multicast addresses and ports to be used
- it does this by periodically “broadcasting” a session description.
- sd keeps a local cache of sessions so it doesn’t have to wait a long time when you start it up.
- sessions simply time out when they’re no longer annouced
- this means to keep a session advertised, you must keep your sd running (on the same machineyou orginally announced it from)
- a next generation sd is under development at UCL

Another note on displays

- Most workstations have 8 bit displays
- This means you can only display up to 256 colours at a time.
- Multimedia is very colour intensive
  - the order you start the tools is important
  - generally, start the video first, then the others
  - if you already have tools running that use a lot of colours (netscape is an example of a colour hungry application) then quit those tools before starting your video.
- colour dithering is expensive for a video tool
  - think about buying a fast 24 bit colour display, and then you can ignore this slide!