Video Streaming using Icecast/P2PSP

Cristóbal Medina López cristobalmedinalopez@gmail.c
Juan Pablo García Ortiz jp.garcia.ortiz@gmail.c
Leocadio González Casado leo@ual.
Vicente González Ruiz vruiz@ual.

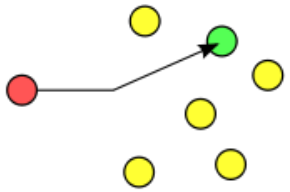
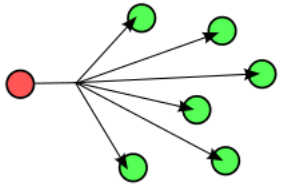
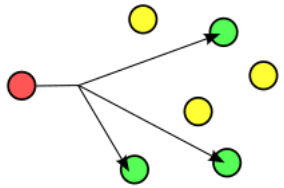
Elche

Abstract

P2PSP (https://p2psp.github.io) is an application-layer protocol that provides real-time broadcasting, also known as Application Layer Multicast (ALM), of a media stream on the Internet. Peers collaborate to diseminate the stream that is generated by a single source, minimizing the latency and the protocol overhead. P2PSP overlays are push-based (topology-driven) meshes. To minimize the transmission latency, P2PSP establishes a communication between nearby peers, and chunks of data are flooded without explicit requests. P2PSP has a modular design organized in *sets of rules*, where each module is especialized in implementing a different functionality.

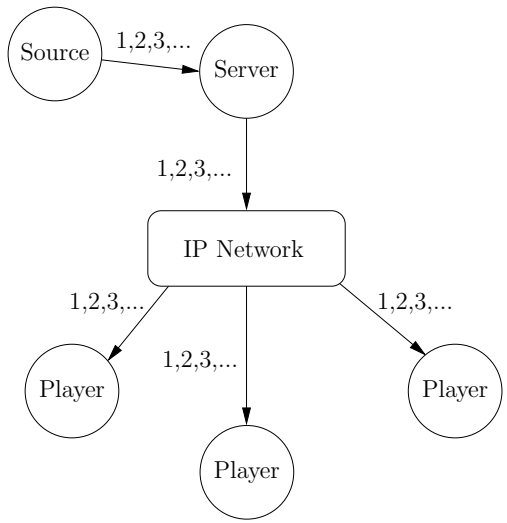
Contents

1. Internet transmission models

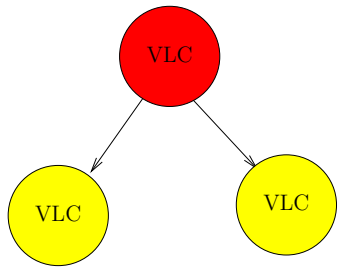
MODE	TOPOLOGY
Unicast	
Broadcast	
Multicast	

2. Streaming models

IP Multicast

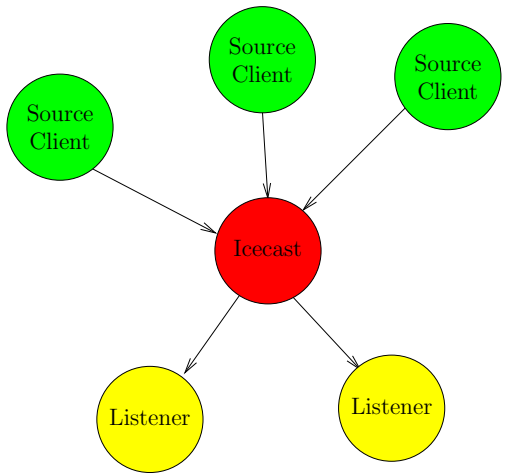


Lab 1: Streaming with VLC




```
echo "Killing all VLC instances (sources and listeners)
killall vlc
sleep 1

echo "Creating the source"
cvlc ~/Videos/LBig_Buck_Bunny_small.ogv --sout "#duplic
sleep 5

echo "Create two listeners"
cvlc http://localhost:8080/BBB.ogv 2> /dev/null &
cvlc http://localhost:8080/BBB.ogv 2> /dev/null &
```

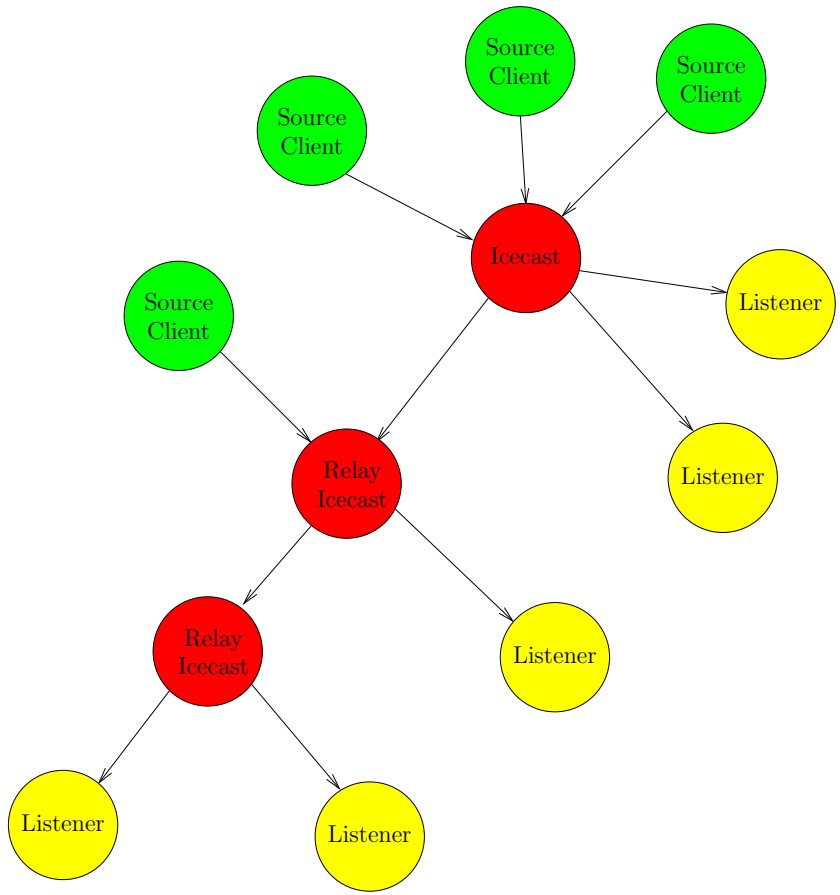
Lab 2: Streaming with VLC and Icecast




```
set -x

echo "Killing all VLC instances (sources and listeners)
killall vlc
sleep 1

echo "Create two sources"
#cvlc ~/Videos/LBig_Buck_Bunny_small.ogv --sout "#dupli
cvlc ~/Videos/LBig_Buck_Bunny_small.ogv --sout "#std{ac
```

```
sleep 1
#cvlc  ~/Videos/Lchi84_14_m4.ogv --sout "#duplicate{dst
cvlc ~/Videos/Lchi84_14_m4.ogv --sout "#std{access=shou
sleep 1

echo "Check the infrastructure"
firefox http://localhost:8000 2> /dev/null &
sleep 5

echo "Create three listeners"
mplayer http://localhost:8000/BBB.ogv 2> /dev/null &
mplayer http://localhost:8000/BBB.ogv 2> /dev/null &
cvlc http://localhost:8000/chi.ogv 2> /dev/null &

set +x
```

Lab 3: Relaying



- Icecast servers can be connected following a tree structure to increase scalability.
- All or a subset of the streams (channels) can be relayed between

servers.

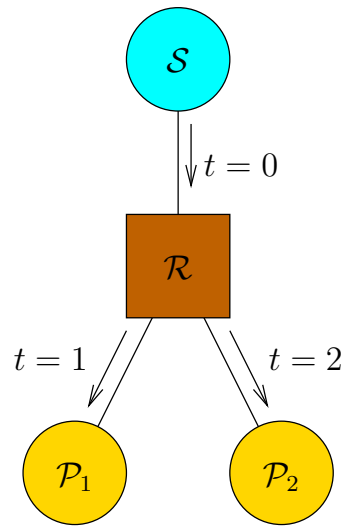
- Clients, the DNS (IP Anycast) or an intermediate server (which performs HTTP redirection) are in charge of selecting the most suitable server.

```
set -x

echo "Killing all VLC instances"
killall vlc
sleep 1

echo "Run a second Icecast2 server listening at port 90
killall icecast2
sleep 1
# The file ~/icecast/icecast.xml must be configured to
# 9000 and to relay all the master's channels
/usr/bin/icecast2 -b -c ~/icecast/icecast.xml
sleep 5
```

```
echo "Feed the fist (8000) icecast server"
cvlc ~/Videos/LBig_Buck_Bunny_small.ogv --sout "#std{ac
sleep 1
cvlc ~/Videos/Lchi84_14_m4.ogv --sout "#std{access=shou
sleep 1

echo "Feed the second (9000) icecast server"
cvlc ~/Videos/Lhcil2003_01.ogv --sout "#std{access=shou
sleep 1

echo "Check the infrastructure"
firefox http://localhost:8000 2> /dev/null &
sleep 10
firefox http://localhost:9000  2> /dev/null
sleep 2
echo "Plase, push <enter> to continue"
read
```

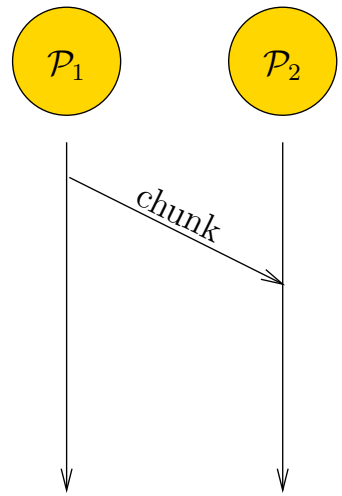
```
echo "Run the listeners, one for the 8000 and two for t
cvlc http://localhost:8000/BBB.ogv 2> /dev/null &
sleep 1
cvlc http://localhost:9000/chi.ogv 2> /dev/null &
sleep 1
cvlc http://localhost:9000/hcil.ogv 2> /dev/null &

set +x
```

3. ALM (Application-Layer Multicast) versus NLM (Network-Layer Multicast)



Network Layer Multicast

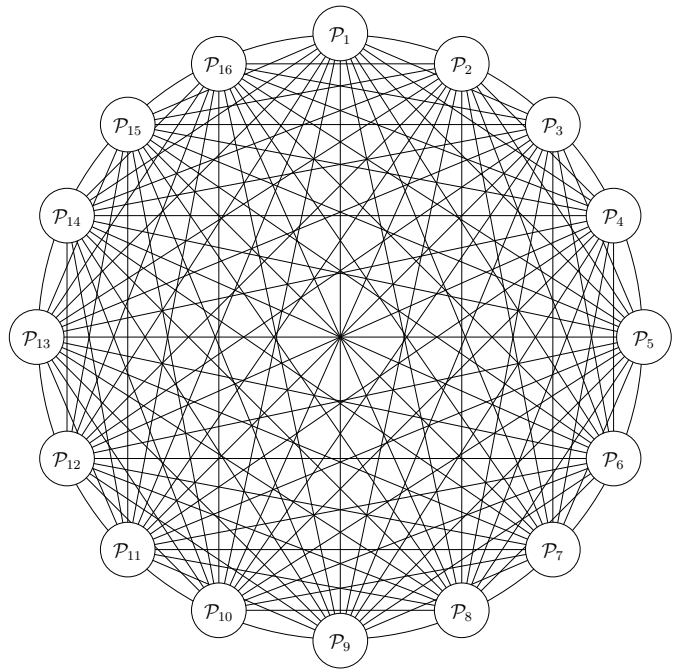
4. Push-based versus Pull-based



Push-based Protocol

5. P2PSP

An Application-layer Multicast protocol which uses a push-based fully-connected mesh (when possible) overlay for real-time streaming of media content between networked entities.



5.1. DBS (Data Broadcasting Set of rules)

DBS provides ALM [1] of a media stream in unicast environments [3].

1. The media is sent by a streaming server, and received by a *splitter*.
2. The splitter divides the stream into a sequence of chunks of data, and relay them to its *team* using a round-robing schema.
3. Each *peer* gathers the chunks from the splitter and the rest of peers of the team, and sends them to a *player*.
4. Incoming peers receive from the splitter the list of peers of the team, and send to they a [hello] message.
5. When a peer receives a [hello], incorporates the sender to a table forward. If peer P_i has an entry $P_i : [P_j, P_k]$ in their forwarding table, each chunk originated at P_i will be forwarded to P_j and P_k.
6. To perform data flow-control, peers write down in a table pending all chunks that must be sent to each neighbor, which are selected using a round-robing scheduler when a new chunk is received.

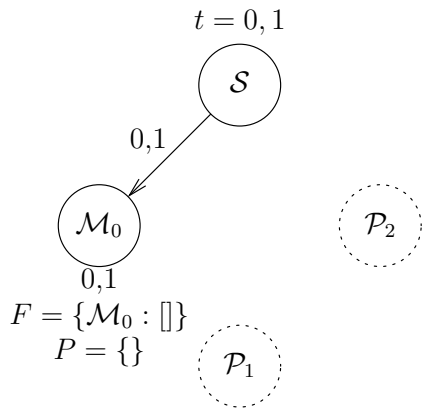


Figure 1: A team has been created with a single monitor M_0 ([`hello`] messages are not shown). Chunks with numbers 0 and 1 (the time t is measured in chunks-time) have been transmitted from the splitter S to M_0. F and P represents the forward[] and the pending[] structures, respectively. The chunks stored in the buffer is shown under the entity.

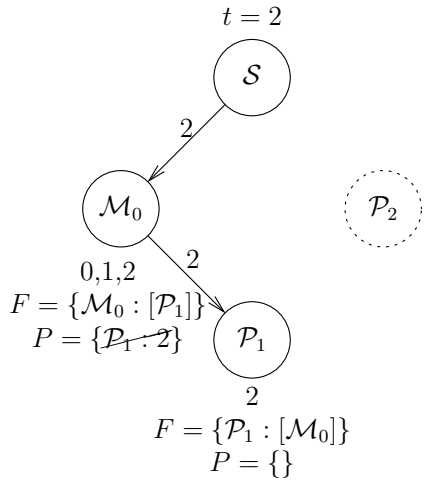


Figure 2: Peer P_1 joins the team (the [hello]'s are not shown). In M_0, $F = \{M_0 : [P_1]\}$ because when M_0 receives the [hello] from P_1, M_0 is the origin peer for all chunks received from S and P_1 is its neighbor. P_1 includes an entry $P_1 : [M_0]$ in its forwarding table because M_0 is in the list of peers received from the splitter. After that, when the chunk number 2 arrives to M_0 from S, an entry $P_1 : 2$ is created in $P\{\}$ for that chunk, and this entry is deleted when the chunk 2 is sent to P_1. Notice that P_1 does not receive chunks 0 and 1 because these chunks belong to a old (already played) section of the stream.

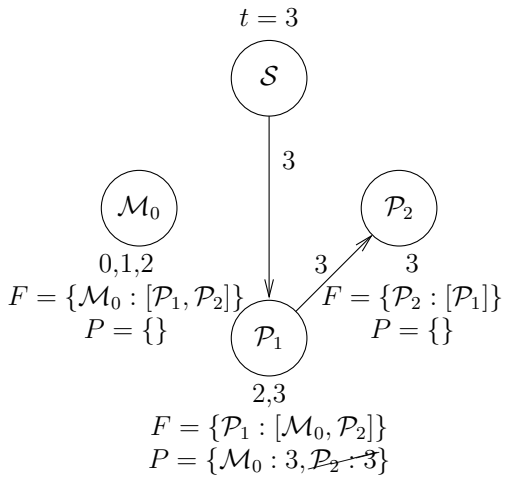


Figure 3: P_2 joins the team (sending the [`hello`]'s not shown). M_0 and P_1 includes P_2 in their forwarding tables. The chunk 3 is received by P_1 which decides to send it to P_2. Chunk 3 remains as pending to be sent to M_0 when the next chunk is received by P_1. P_2 is the origin peer for M_0 and P_1 because both of them are in the list of peers that P_2 receives from S.

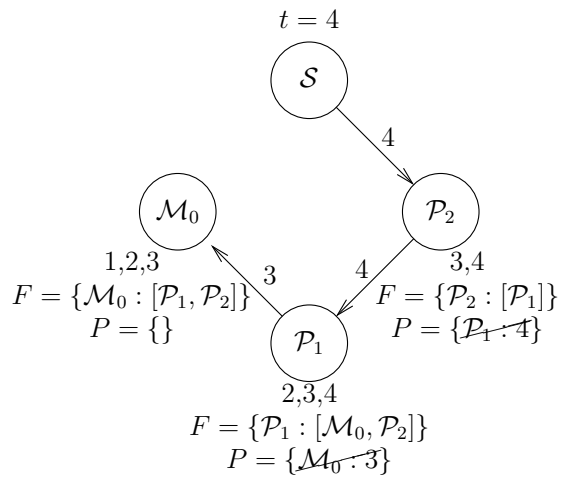


Figure 4: Chunk 4 is received by P_2 which relays it to P_1, what relays chunk 3 to M_0.

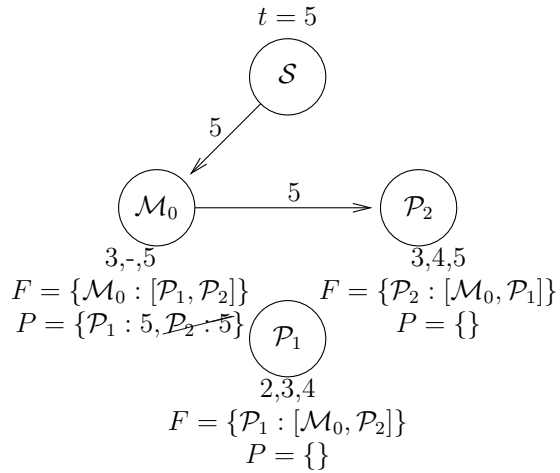
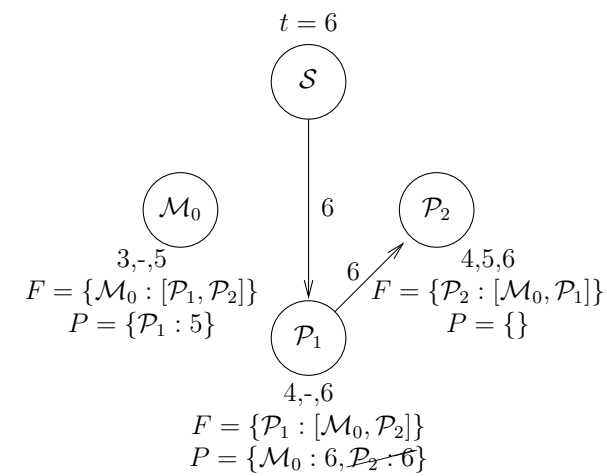
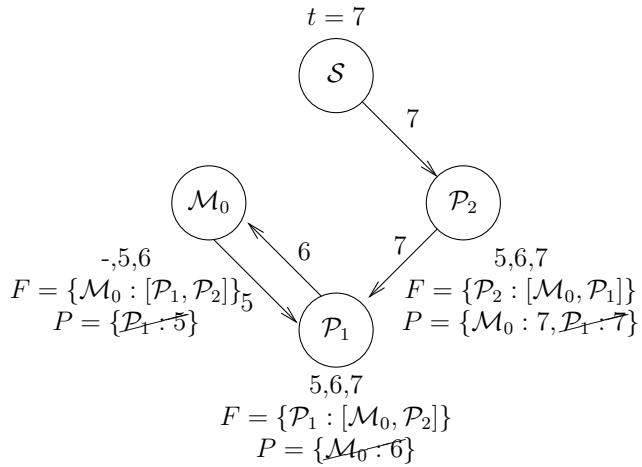
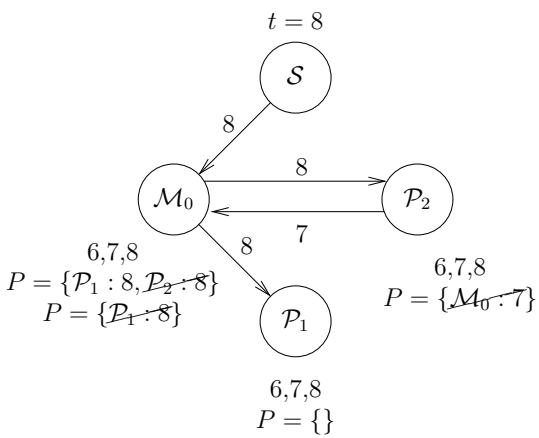
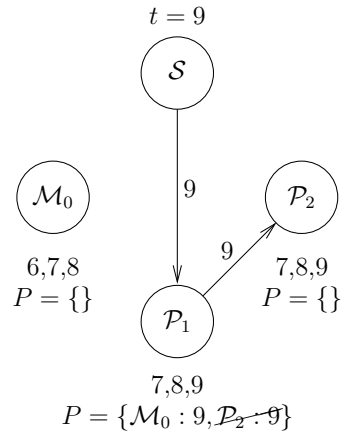


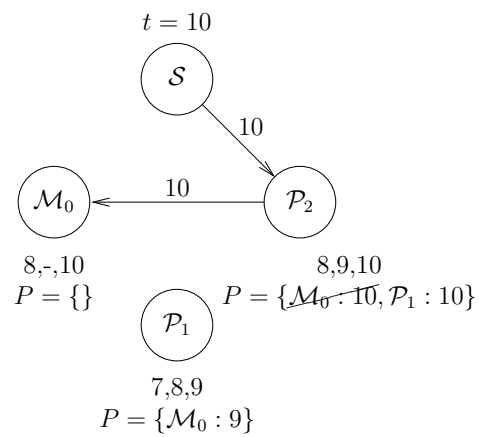
Figure 5: Chunk 5 is received by M_0 which relays it to P_2.

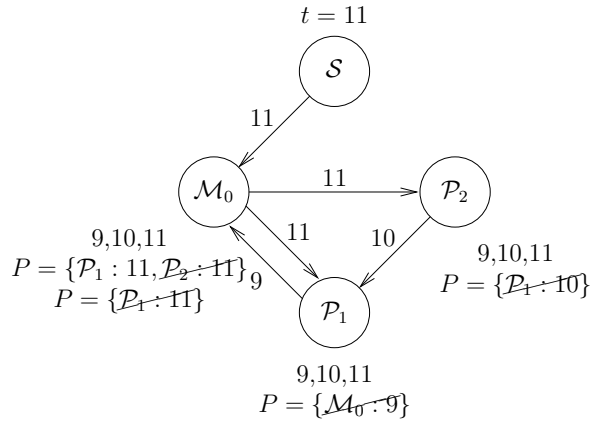


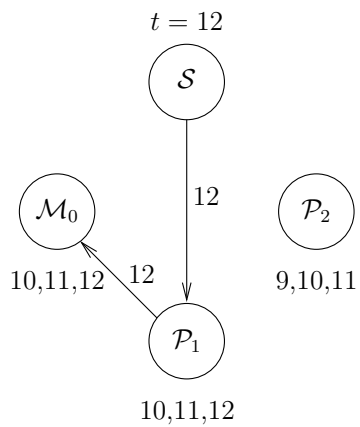

$t = 7$
$\mathcal{S}$
7
$\mathcal{M}_0$
$\mathcal{P}_2$
-,5,6
$F = \{\mathcal{M}_0 : [\mathcal{P}_1, \mathcal{P}_2]\}$
$P = \{\cancel{\mathcal{P}_1 : 5}\}$
6
5
7
$\mathcal{P}_1$
5,6,7
$F = \{\mathcal{P}_2 : [\mathcal{M}_0, \mathcal{P}_1]\}$
$P = \{\mathcal{M}_0 : 7, \cancel{\mathcal{P}_1 : 7}\}$
5,6,7
$F = \{\mathcal{P}_1 : [\mathcal{M}_0, \mathcal{P}_2]\}$
$P = \{\cancel{\mathcal{M}_0 : 6}\}$

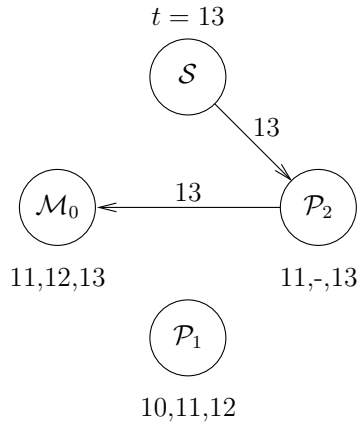

$t = 8$
$\mathcal{S}$
8
8
$\mathcal{M}_0$
$\mathcal{P}_2$
7
8
6,7,8
$P = \{\mathcal{P}_1 : 8, \cancel{\mathcal{P}_2 : 8}\}$
$P = \{\cancel{\mathcal{P}_1 : 8}\}$
6,7,8
$P = \{\cancel{\mathcal{M}_0 : 7}\}$
$\mathcal{P}_1$
6,7,8
$P = \{\}$

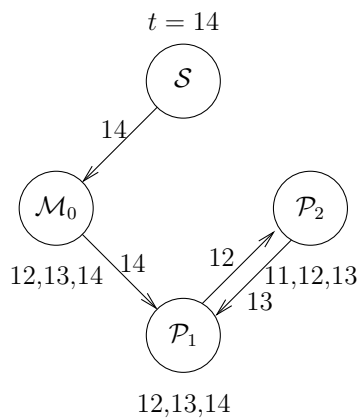

$t = 9$
$\mathcal{S}$
9
$\mathcal{M}_0$
$\mathcal{P}_2$
9
6,7,8
$P = \{\}$
7,8,9
$P = \{\}$
$\mathcal{P}_1$
7,8,9
$P = \{\mathcal{M}_0 : 9, \cancel{\mathcal{P}_2 : 9}\}$


$t = 10$
$\mathcal{S}$
10
$\mathcal{M}_0$
10
$\mathcal{P}_2$
8,-,10
$P = \{\}$
8,9,10
$P = \{\cancel{\mathcal{M}_0 : 10}, \mathcal{P}_1 : 10\}$
$\mathcal{P}_1$
7,8,9
$P = \{\mathcal{M}_0 : 9\}$


$t = 11$
$\mathcal{S}$
11
$\mathcal{M}_0$
11
$\mathcal{P}_2$
9,10,11
$P = \{\mathcal{P}_1 : 11, \cancel{\mathcal{P}_2 : 11}\}$
$P = \{\cancel{\mathcal{P}_1 : 11}\}$
11
9
10
9,10,11
$P = \{\cancel{\mathcal{P}_1 : 10}\}$
$\mathcal{P}_1$
9,10,11
$P = \{\cancel{\mathcal{M}_0 : 9}\}$


$t = 12$
$\mathcal{S}$
12
$\mathcal{M}_0$
$\mathcal{P}_2$
10,11,12
12
9,10,11
$\mathcal{P}_1$
10,11,12


$t = 13$
$\mathcal{S}$
13
$\mathcal{M}_0$
13
$\mathcal{P}_2$
11,12,13
11,-,13
$\mathcal{P}_1$
10,11,12


$t = 14$
$\mathcal{S}$
14
$\mathcal{M}_0$
$\mathcal{P}_2$
12,13,14
14
12
11,12,13
13
$\mathcal{P}_1$
12,13,14

5.2. IMS (Ip Multicast Set of rules)

IPM is available by default in LANs (Local Are Network)s and VLANs (Virtual LANs) [4], but not in the Internet [2]. IMS runs on the top of DBS and provides efficient native IPM, where available.

1. The idea of IMS is simple: all peers in the same LAN or VLAN have the same network address. When a joining peer P_i receives the list of peers from its splitter, first checks if there are neighbors in the same subnet (all of them share the same private network address). For all those peers, P_i uses the IP address 224.0.0.1 (all systems on this subnet), (default) port 1234, to multicast (only) the chunks received from the splitter.

References

[1] Suman Banerjee, Bobby Bhattacharjee, and Christopher Kommareddy. *Scalable application layer multicast*, volume 32. ACM, 2002.

[2] Douglas E. Comer. *Internetworking with TCP/IP. Principles, Protocols, and Architectures (4th Edition)*, volume 1. Prentice Hall, 2000.

[3] Douglas E Comer and Ralph E Droms. *Computer networks and internets*. Prentice-Hall, Inc., 2003.

[4] Lior Shabtay and Benny Rodrig. Ip multicast in vlan environment, April 12 2011. US Patent 7,924,837.