Formal Analysis of P802.1CB

IEEE Plenary, Geneva, Jul/2013

Wilfried Steiner, Corporate Scientist
wilfried.steiner@tttech.com

Proposed Solution

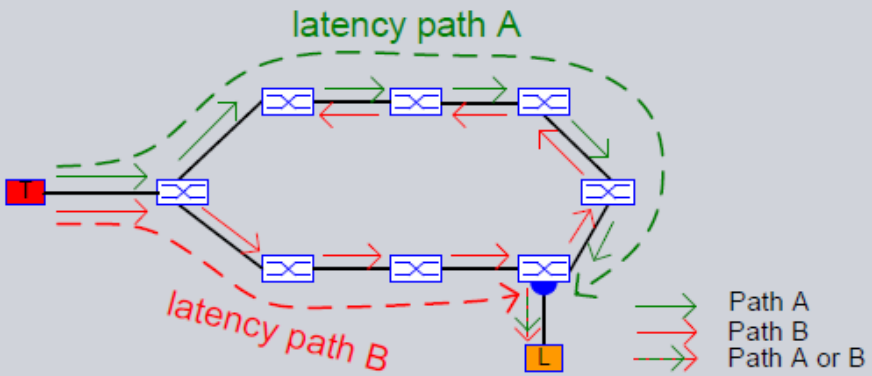
L2 Duplicate Elimination Mechanism Model (1)

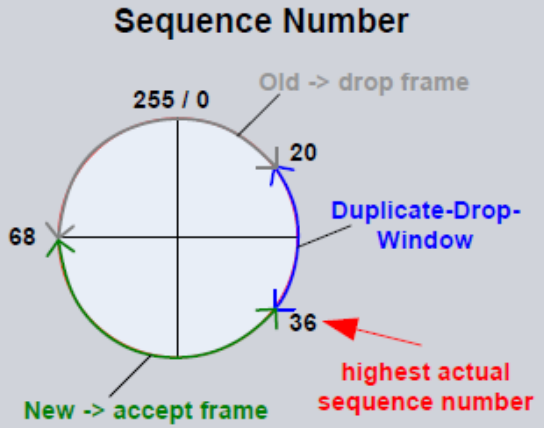
Repetition of our Assumptions:

- Seamless Redundancy only for Reserved- or Scheduled-Traffic (also called stream)
- MAC address (destination address) of Stream is unique
- MAC address & sequence numbers are used for duplicate elimination

Data structure:

- Duplicate-Drop-Window: The size for the window depends on:
  - Latency for path A
  - Latency for path B
  - Transmission period of Reserved- or Scheduled- Stream





Page 28 2012-11-12 IEEE 802.1 Meeting – San Antonio

http://www.ieee802.org/1/files/public/docs2012/new-goetz-jochim-Seamless-Redundancy-1112-v02.pdf

http://www.ieee802.org/1/files/public/docs2013/
new-avb-wsteiner-8021CB-lessons-learned-from-avionics-0513-v01.pdf

Lessons Learned (from ARINC)

Redundancy Management requires precise knowledge of the communication latency and jitter of the messages on the redundant paths through the network.

In certain cases the loss of a frame on one network can cause the loss of its copy on the redundant network.

Sometimes, loss of communication requires to restart the sequence numbering.

The ARINC 664-p7 redundancy mechanism is very well studied by academics and industry due to its importance and criticality for avionics systems.

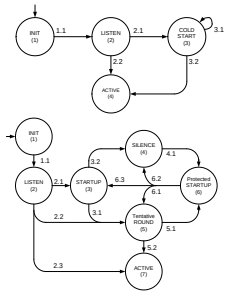
Designed for closed networks.

Formal Analysis by Model Checking

Ensuring Reliable Networks TTTech

Modify

Algorithm Specification



Failure Specification

Properties Specification

?

SAL Model

```
nodes[id:index]: MODULE =
BEGIN

INPUT
 membership_in: ARRAY index
OF BOOLEAN,
 classification_in: CLASSES,
 sender_in: index

OUTPUT
 membership_out: ARRAY index
OF BOOLEAN,
 classification_out: ARRAY index
OF CLASSES

LOCAL
 asymmetry_memb: ARRAY
index OF BOOLEAN,
 asymmetry_clas: ARRAY index
OF BOOLEAN

INITIALIZATION
 (FORALL (i:index):
membership_out[i] = TRUE);
 (FORALL (i:index):
classification_out[i] = accept);


TRANSITION
[
   id=faulty
 -->
   membership_out'=[[j:index] IF
asymmetry_memb[j] THEN
FALSE ELSE TRUE ENDIF];
   classification_out'= [[j:index] IF
asymmetry_clas[j] THEN reject
ELSE flagged ENDIF];
]
END;
```

SAL

Model Checker

ok

no,

because…

Counter Example

```
asymmetry_memb[1][2] = false;
asymmetry_memb[1][3] = false;
asymmetry_memb[1][4] = false;
asymmetry_memb[1][5] = false;
asymmetry_memb[2][1] = false;
asymmetry_memb[2][2] = false;
asymmetry_memb[2][3] = false;
asymmetry_memb[2][4] = false;
asymmetry_memb[2][5] = false;
asymmetry_memb[3][1] = true;
asymmetry_memb[3][2] = false;
asymmetry_memb[3][3] = false;
asymmetry_memb[3][4] = false;
asymmetry_memb[3][5] = true;
```

Proposal – IEEE 802.1Q AVB/TSN Failure Hypothesis

Fault-Containment Regions (FCR):

- Communication Link
- End Station
- Bridge

→ A fault is local to either an end station or a bridge or a communication link.

→ If more than one bridge / one end stations / one link become faulty then we have also more than one fault.

Failure Mode for End Stations and Bridges

- Permanent, Consistent, and Fail-Silent

→ In the case of a failure, a faulty FCR will stop producing output (“Fail-Silent”).

→ A faulty FCR will behave the same on all ports, e.g., a faulty bridge will stop producing output on all ports (“Consistent”).

→ A faulty FCR will be faulty for the remaining mission time (“Permanent”).

Failure Mode for Communication Links

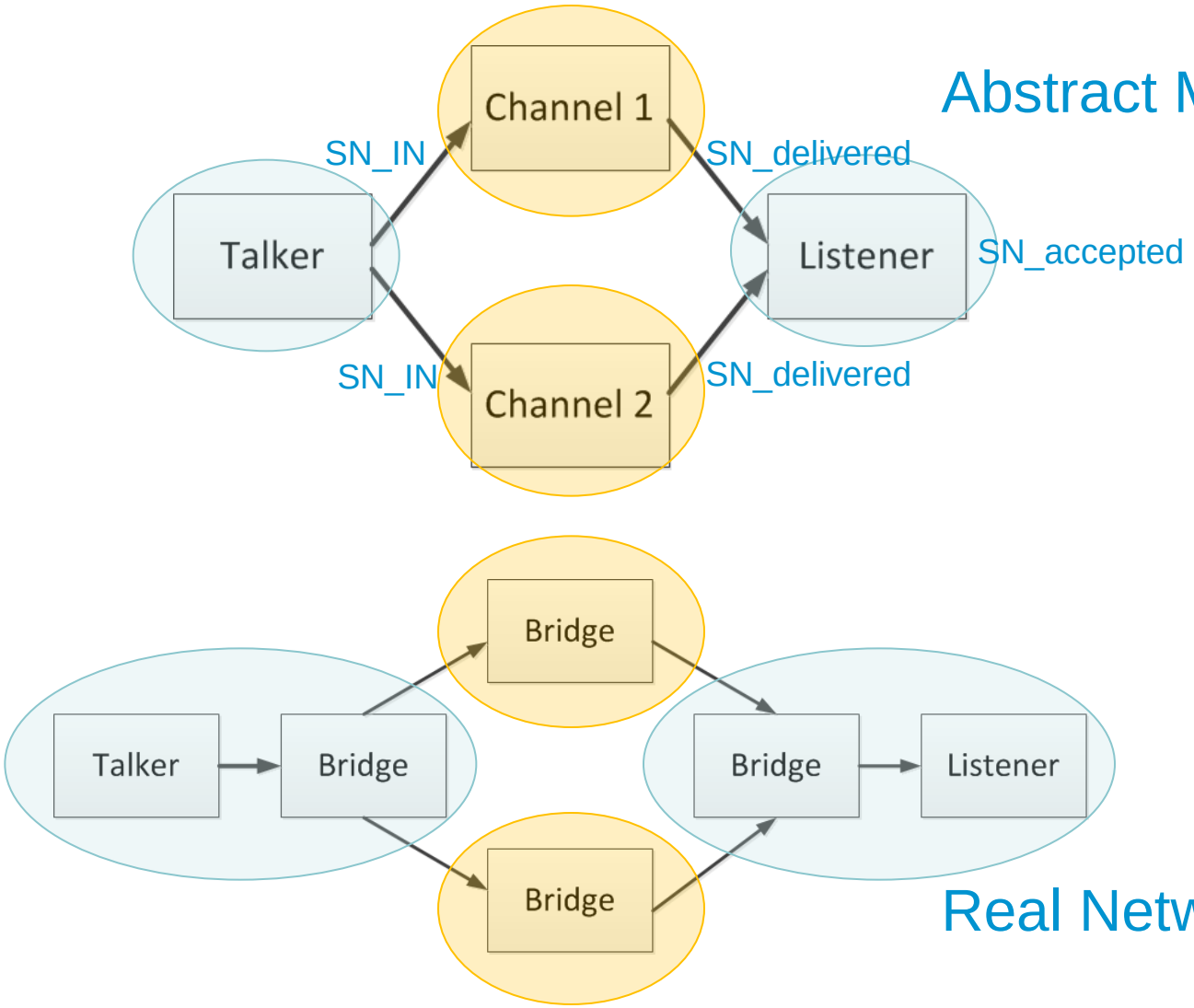
- Transient or Permanent, Detectably Faulty

→ The communication link may drop frames or invalidate the Ethernet FCS on a per frame basis (“Transient”).

→ The communication link may become unavailable for the remaining mission time (“Permanent”).

→ Each failure of the communication link results in either a loss of the frame or an invalidation of the frame’s FCS (“Detectably Faulty“).

802.1CB – Model Structure



Talker Model


```
talker: MODULE =
   BEGIN
   …
   TRANSITION
   [
   talker_state = generate
   -->
   talker_state' = generate;

   []
     talker_state = generate
     AND SN[1]<max_SN
    -->
     talker_state' = generate;
     SN'= [[c: TYPE_channels]
               SN[1]+1];

   []
     talker_state = generate
     AND SN[1]>=max_SN
    -->
     talker_state' = stop;

   []
     talker_state = stop
    -->
     talker_state' = stop;

   ]

   END;
```

if

then

Channel 1/2

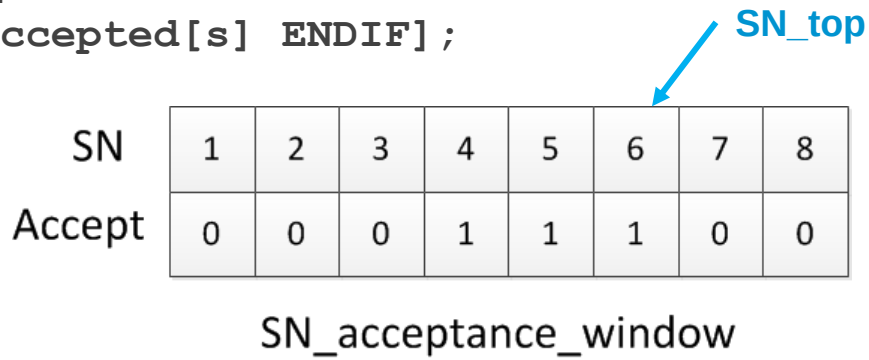

```
ch_state = delay
 -->
  ch_state'  = delay;
  SN_stored' = [[n:TYPE_SN]  IF n=SN_IN AND n/=0
                     THEN TRUE
                     ELSE SN_stored[n]
                     ENDIF];
  SN_delivered' = 0;

[]
  ch_state = delay
 -->
  ch_state'  = forward;
  SN_stored' = [[n:TYPE_SN]  IF n=SN_IN AND n/=0
                     THEN TRUE
                     ELSIF n=nextSN(SN_stored)
                     THEN FALSE
                     ELSE SN_stored[n]
                     ENDIF];
  SN_delivered' = nextSN(SN_stored);


% SN_stored is a bitvector indexed by the SN
% SN_stored[i] will be true if the channel has stored SN i and
   false otherwise
```

Channel 2/2


```
[] ch_state = forward
 -->
  ch_state'  = delay;
  SN_stored' = [[n:TYPE_SN]  IF n=SN_IN AND n/=0
                      THEN TRUE
                      ELSE SN_stored[n]
                      ENDIF];
  SN_delivered' = 0;


[] ch_state = forward
 -->
  ch_state'  = forward;
  SN_stored' = [[n:TYPE_SN]  IF n=SN_IN AND n/=0
                       THEN TRUE
                      ELSIF n=nextSN(SN_stored)
                       THEN FALSE
                      ELSE SN_stored[n]
                      ENDIF];
  SN_delivered' = nextSN(SN_stored);


% SN_stored is a bitvector indexed by the SN
% SN_stored[i] will be true if the channel has stored SN i and
   false otherwise
```

Listener


```
SN_top' = IF list_SN_delivered[1] >  SN_top AND
             list_SN_delivered[1] >= list_SN_delivered[2]
           THEN list_SN_delivered[1]
          ELSIF list_SN_delivered[2] >  SN_top AND
                 list_SN_delivered[2] >= list_SN_delivered[1]
           THEN list_SN_delivered[2]
          ELSE  SN_top
          ENDIF;

  SN_acceptance_window' = [[s:TYPE_SN]
        IF s > SN_top' OR s < SN_top'-ACC_WINDOW
        THEN FALSE ELSE TRUE ENDIF];

  SN_accepted' = [[s:TYPE_SN]
       IF (s=list_SN_delivered[1] OR s=list_SN_delivered[2])
           AND SN_acceptance_window'[s]
       THEN TRUE
       ELSE SN_accepted[s] ENDIF];
```



Correctness Property

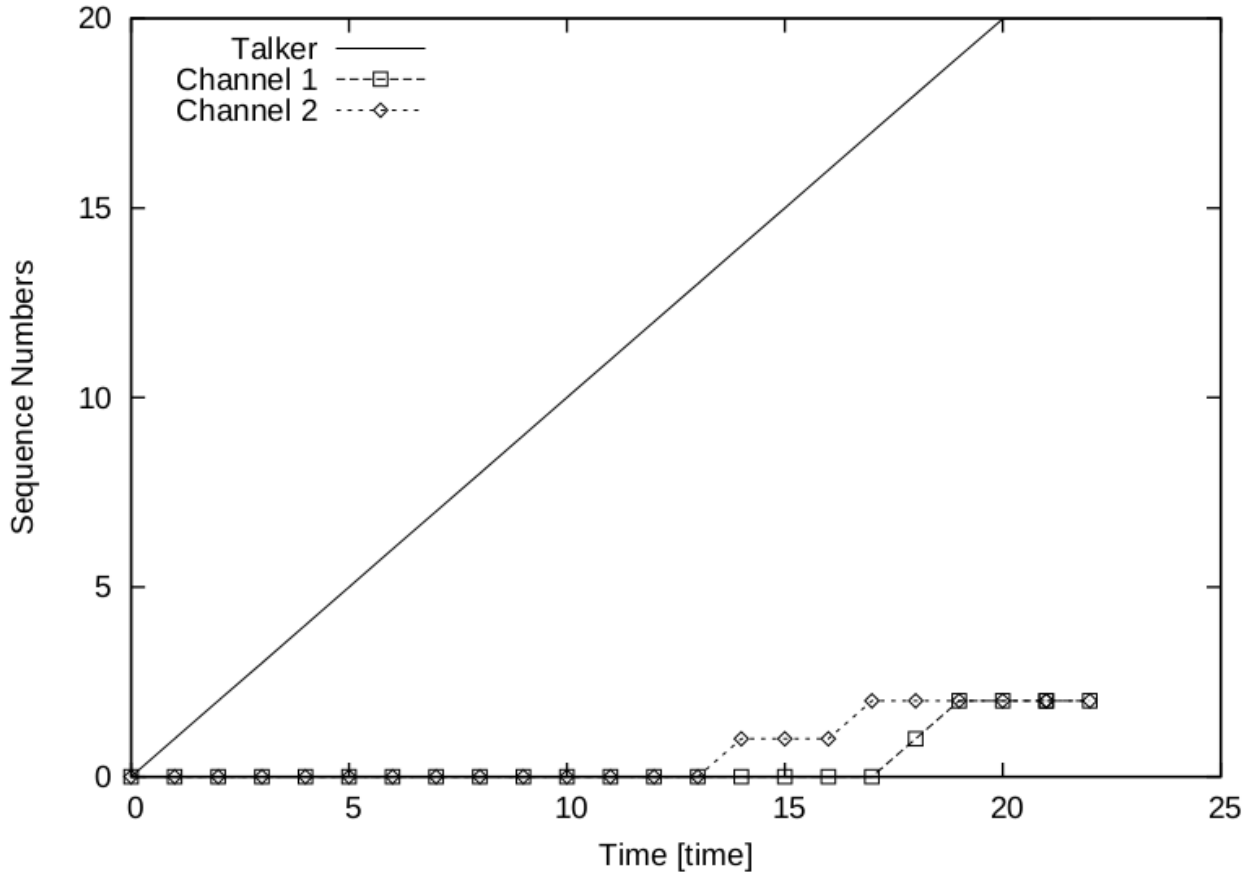

```
all_accepted:
  LEMMA system |- F(FORALL(s:TYPE_SN): SN_accepted[s]);
```

F … in all execution traces, there will be a point in time
(FORALL(s:TYPE_SN): SN_accepted[s]) … all SNs will be accepted

```
%Execution of the model:

> sal-smc network all_accepted
```

Note: this is not a simulation, but rather an exhaustive search.

Counterexample – due to arbitrary delays in the bridges



Channel w. delay upper bound


```
ch_state = delay AND delay_ctr < max_delay
 -->
  ch_state'  = delay;
  SN_stored' = [[n:TYPE_SN]  IF n=SN_IN AND
                              n/=0
                    THEN TRUE
                    ELSE SN_stored[n]
                    ENDIF];
  SN_delivered' = 0;
  delay_ctr' = IF delay_ctr < max_delay
               THEN delay_ctr+1
               ELSE delay_ctr ENDIF;
```

In the model we simply add a delay counter that cannot exceed a particular value.

In reality this imposes a requirement of a known upper bound on the forwarding duration.

→ With this addition the previous counterexample goes away.

Adding a faulty channel

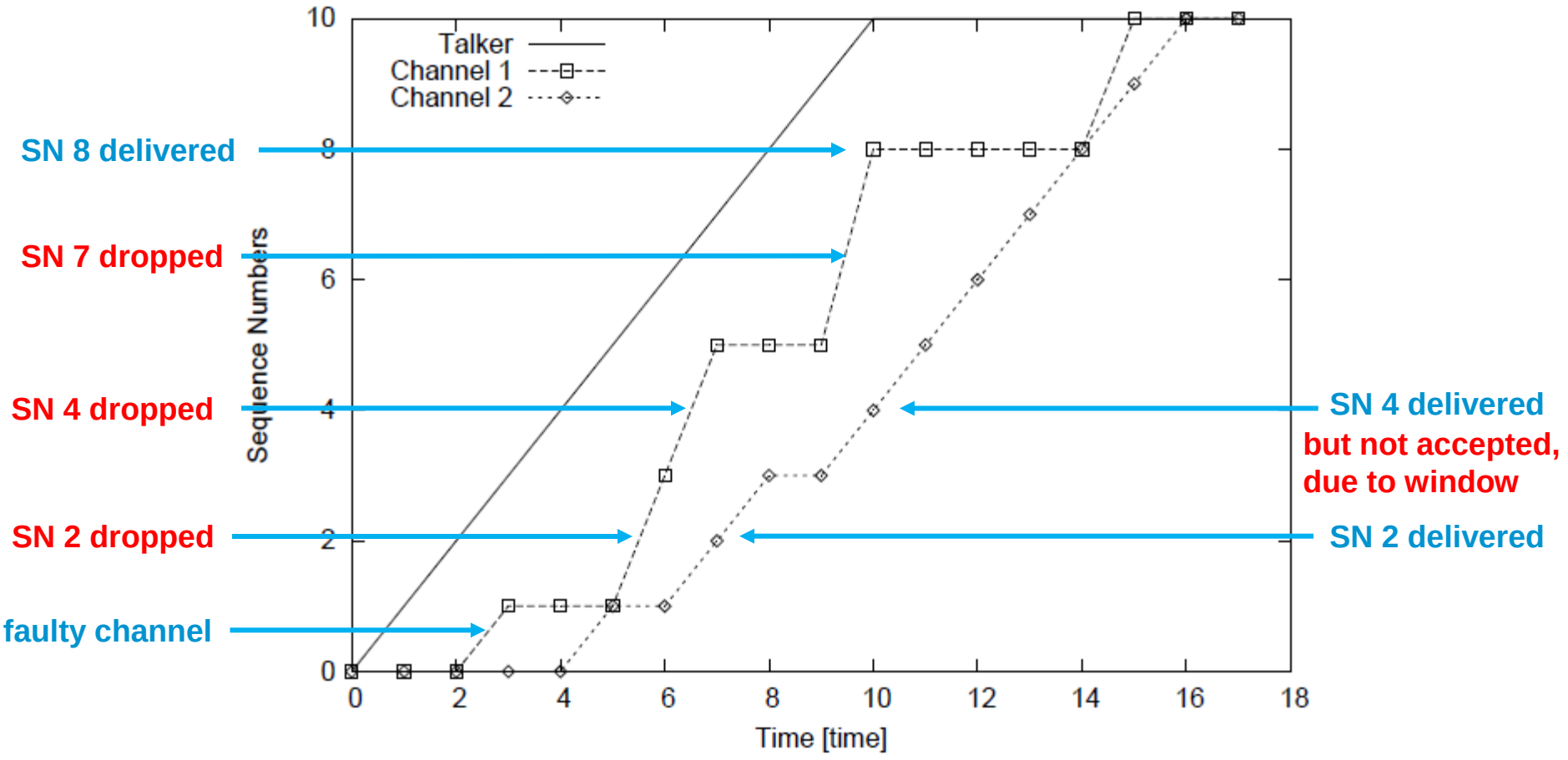

```
[]
  c = FAULTY AND FAULTS_ENABLED
 -->
  SN_stored' IN {x: ARRAY TYPE_SN OF BOOLEAN |
        (FORALL (i:TYPE_SN): NOT SN_stored[i] => NOT x[i])};
```

We simply allow the faulty channel to drop messages.

This is modeled by allowing the faulty channel to set any value in the SN_stored to FALSE.

This behavior results in the following counterexample.

Acceptance Window Size = 3 SN



Proposed Solution

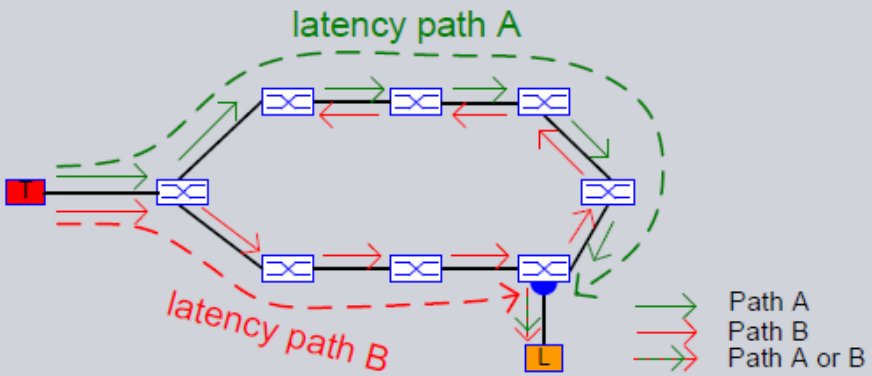
L2 Duplicate Elimination Mechanism Model (1)

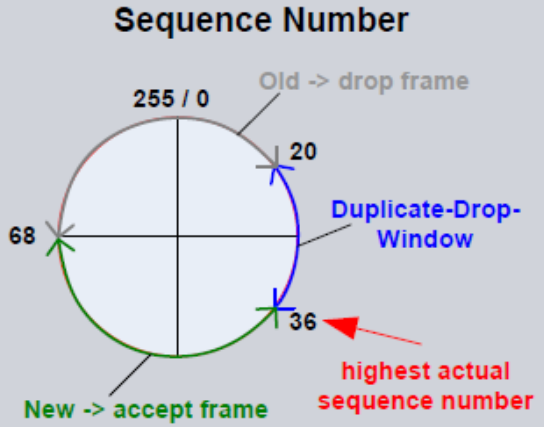
Repetition of our Assumptions:

- Seamless Redundancy only for Reserved- or Scheduled-Traffic (also called stream)
- MAC address (destination address) of Stream is unique
- MAC address & sequence numbers are used for duplicate elimination

Data structure:

- Duplicate-Drop-Window: The size for the window depends on:
  - Latency for path A
  - Latency for path B
  - Transmission period of Reserved- or Scheduled- Stream





Page 28 2012-11-12 IEEE 802.1 Meeting – San Antonio

http://www.ieee802.org/1/files/public/docs2012/new-goetz-jochim-Seamless-Redundancy-1112-v02.pdf

Conclusion

We have analyzed a proposed solution to P802.1CB by means of model checking.

This analysis strengthened the assumptions that,

- the worst-case transmission latencies need to be known
- the failure mode of a faulty channel needs to be taken into account for the configuration of the proposed protocol.

We are currently analyzing how the particular transmission/configuration parameters interrelate, e.g., how large does the acceptance window need to be?

Further Info

In this analysis the SAL model checker developed by SRI International has been used: http://fm.csl.sri.com/

TTTech

Ensuring Reliable Networks

www.tttech.com