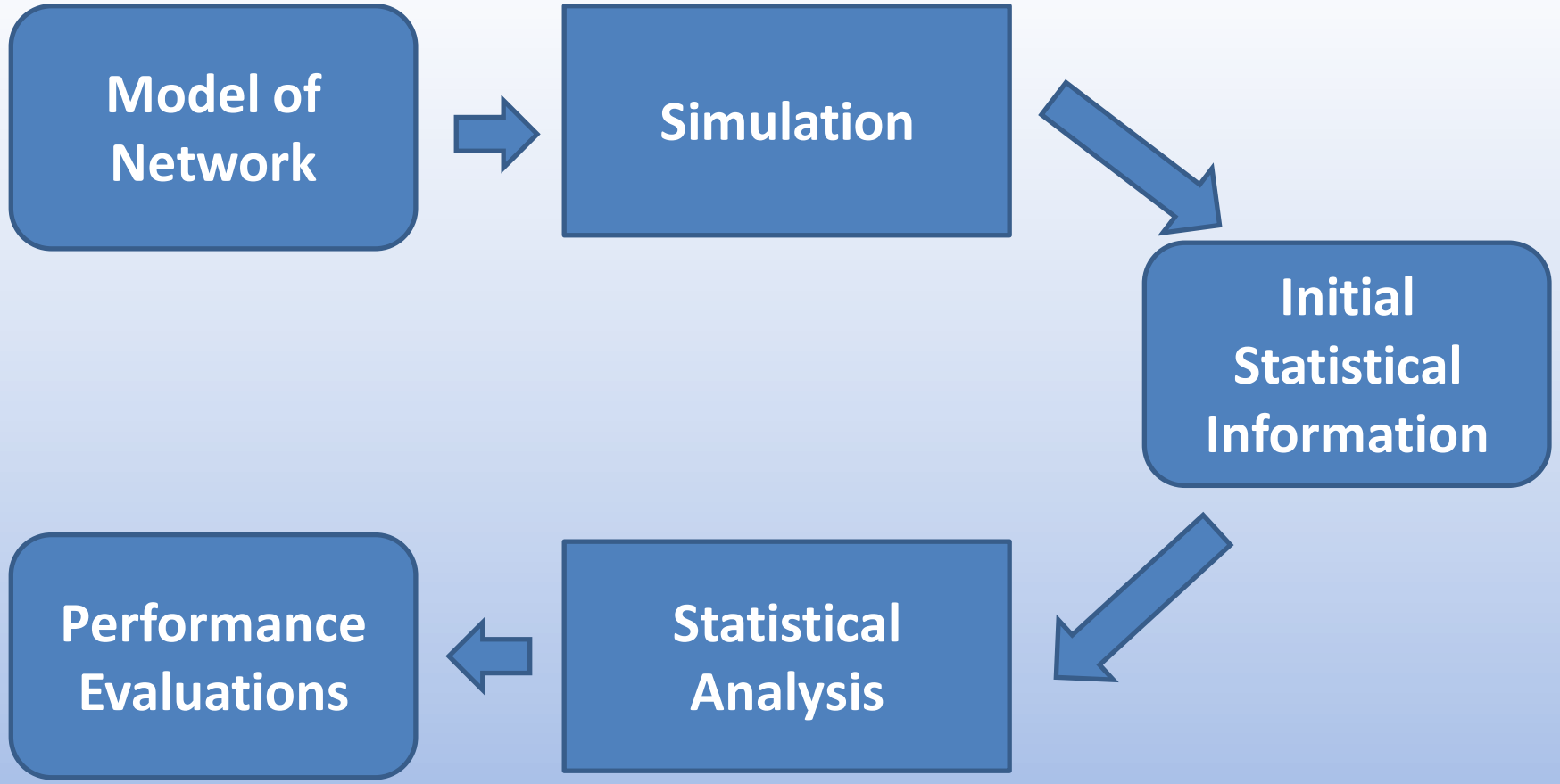
26th Telecommunications Forum TELFOR 2018

Reenterable Colored Petri Net Models of Networks, Grids, and Clouds: Case Study for Provider Backbone Bridge

http://daze.ho.ua

Dmitry A. Zaitsev, Senior Member, IEEE,
Tatiana R. Shmeleva, and
Anatoly I. Sleptsov

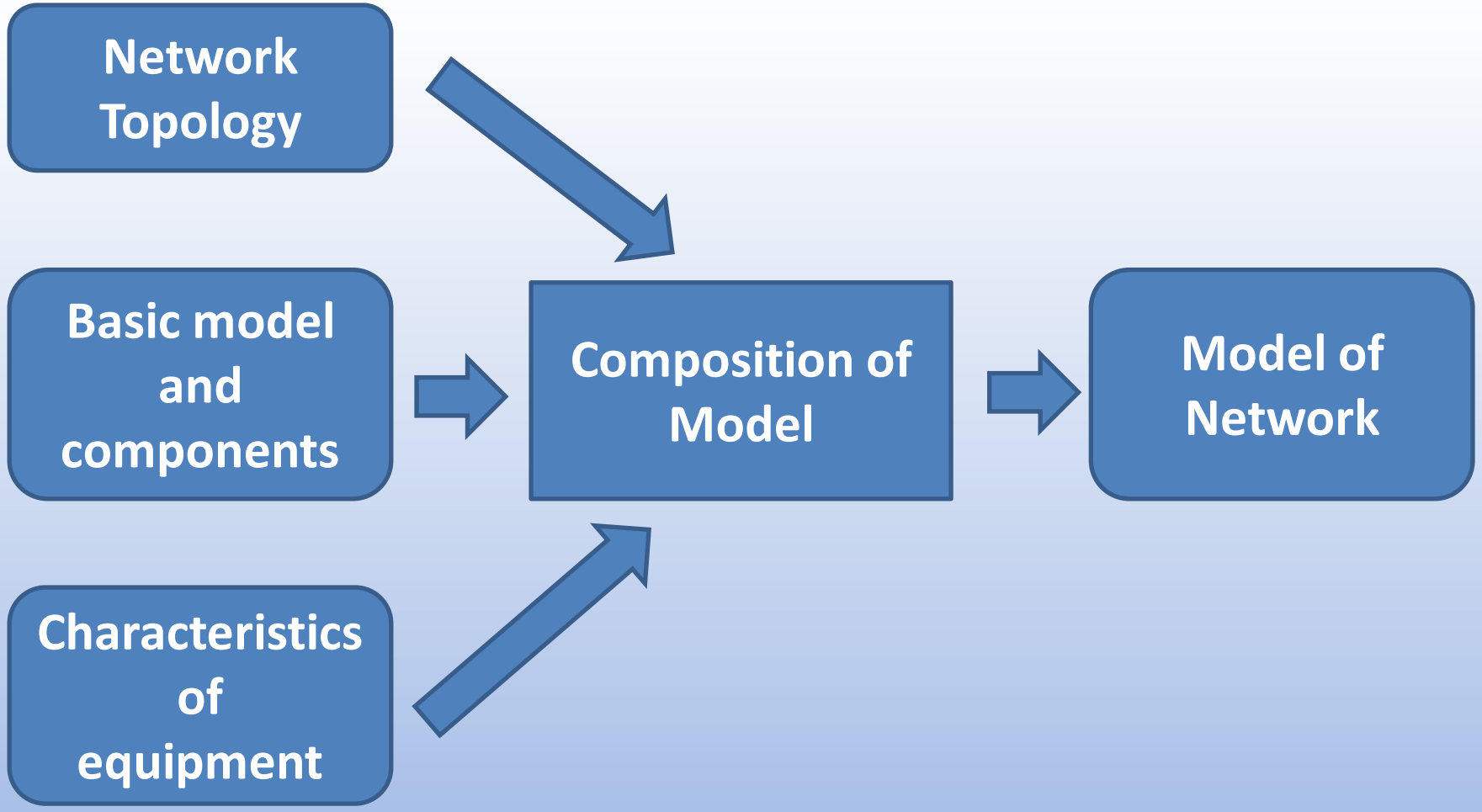
Performance Evaluation of Networks via Simulation



Basic Characteristics of Performance

- Network/component bandwidth
- Packet delivery time (average, maximal)
- Percent of dropped packets
- Jitter
- Response time (for client-server applications)

Composing Model of a Given Network



Network Simulation Systems

- **ns3** https://www.nsnam.org
- **NetSim** https://www.tetcos.com
- **Opnet** https://www.riverbed.com
- **OMNeT++** https://www.omnetpp.org
- **GNS3** https://gns3.com
- **Packet Tracer** https://www.netacad.com
- **QualNet** https://web.scalable-networks.com

Modeling with Colored Petri Nets

Colored Petri Net
Model of Network

CPN Tools
http://cpntools.org

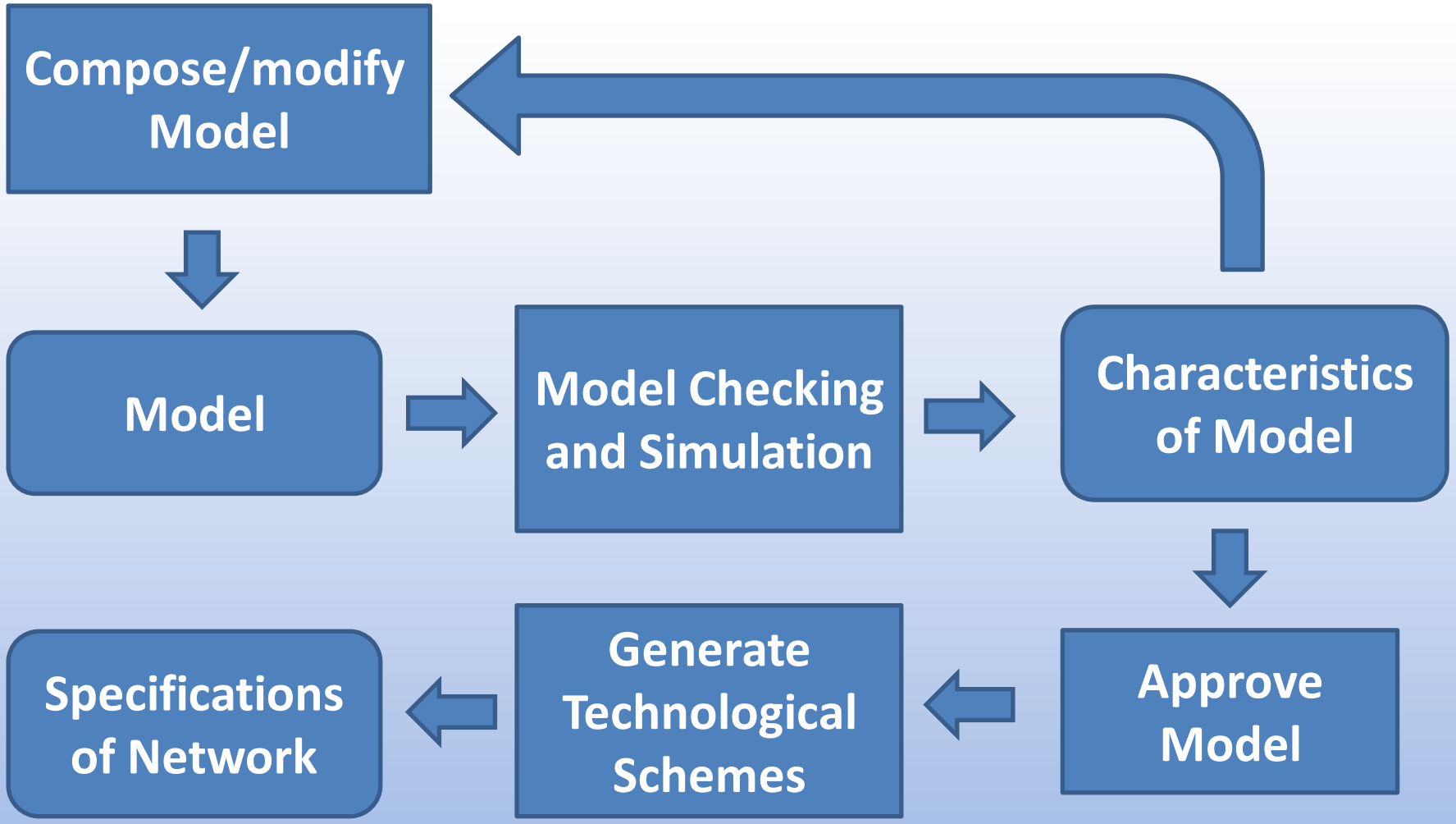
Model Checking - Verification of Networking Protocols

Simulation - Performance Evaluation of Networks

Model – a Piece of Art

- Reflect basic features of object significant in the present research
- Not too abstract (not simple)
- Not too detailed (not complex)
- Allow obtaining evaluations of characteristics close to actual measurements on object

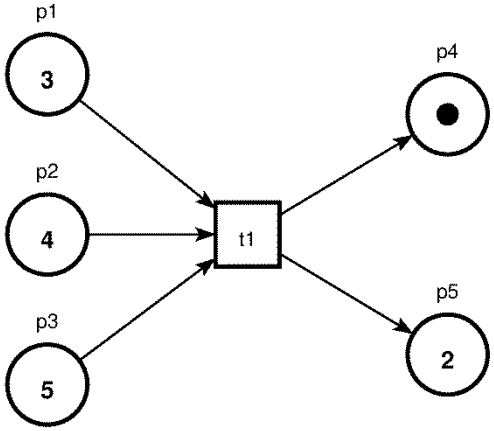
Model-Driven Design of Networks

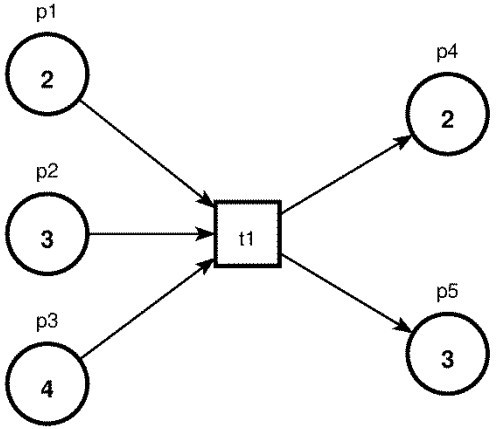


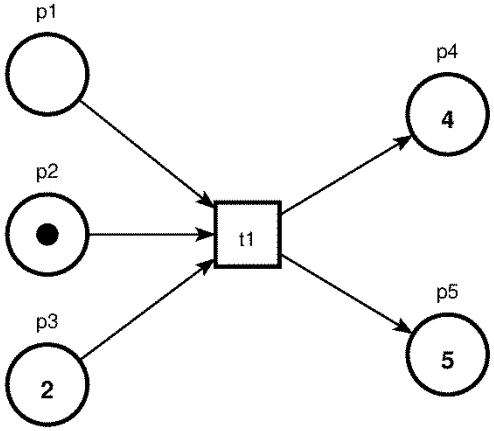
Petri Net

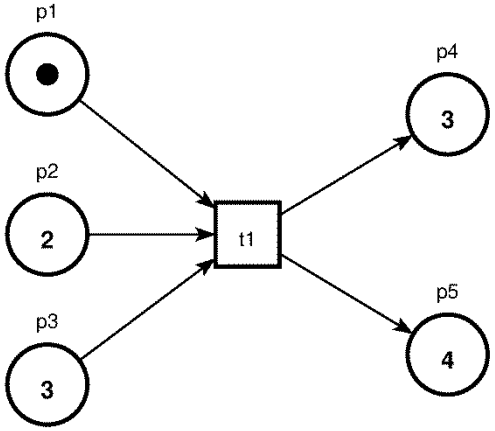
- Directed bipartite graph
- Two parts of vertices: places (circles, ovals), transitions (squares, rectangles)
- Dynamic elements – tokens (dots) – inside places
- Consuming and producing tokens as a result of transition firing (moving tokens)

Firing a Transition





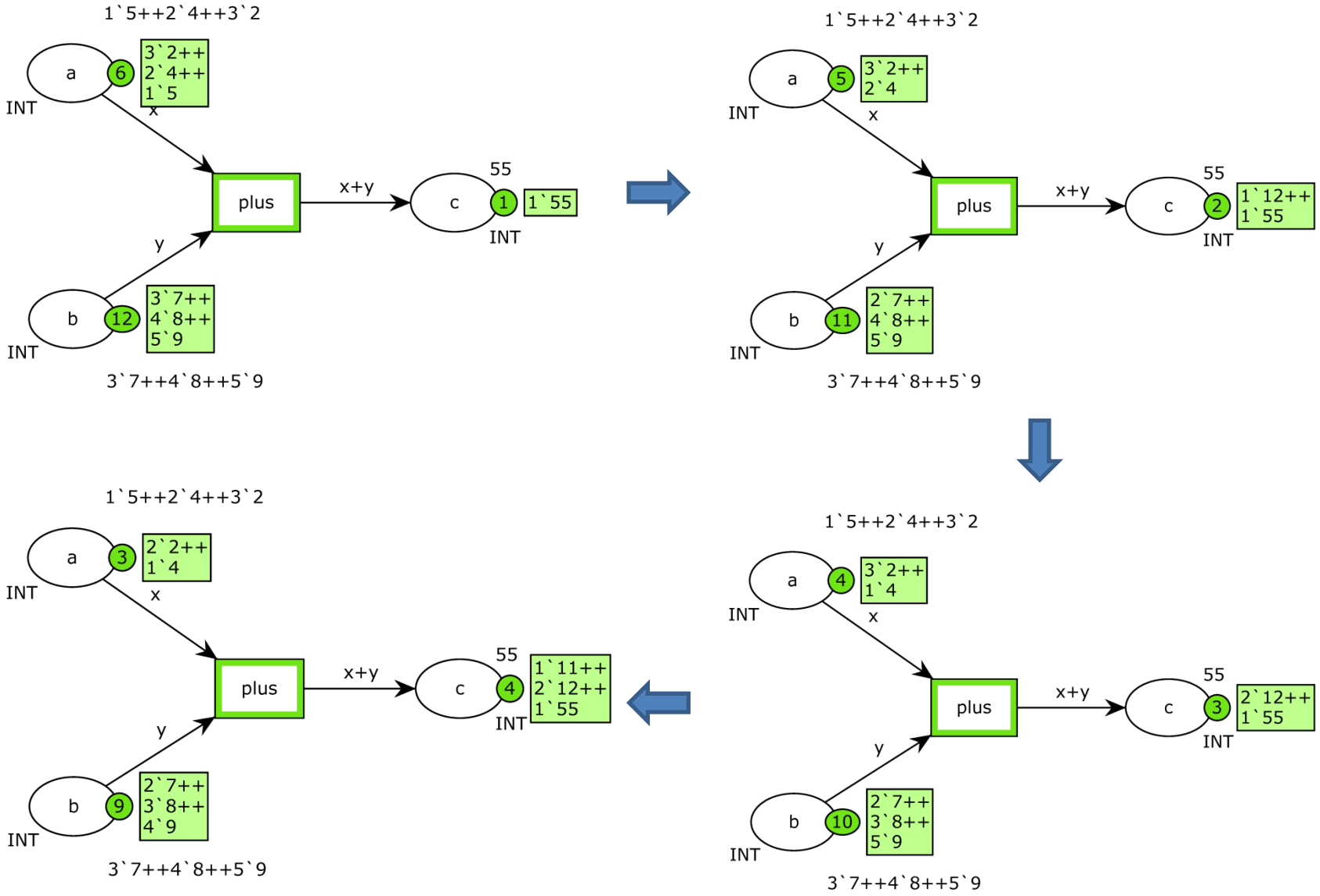




Colored Petri Net of CPN Tools

- Petri Net: places, transitions, arcs, tokens
- Functional Programming Language ML for inscriptions of Petri net elements: color sets, variables, constants, functions
- Timed characteristics – timestamps of tokens and delays of transitions/arcs
- Hierarchical composition – substitution of a transition by a subnet

Firing Transition of CPN



Provider Backbone Bridge (PBB)

"IEEE Standard for Local and Metropolitan Area Net-works—Virtual Bridged Local Area Networks, Amendment 7: Provider Backbone Bridges," IEEE Std 802.1ah™, 12 June 2008.

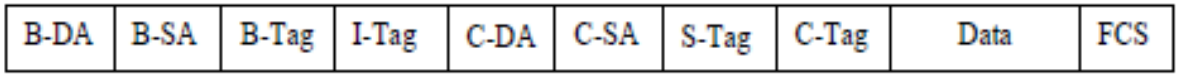
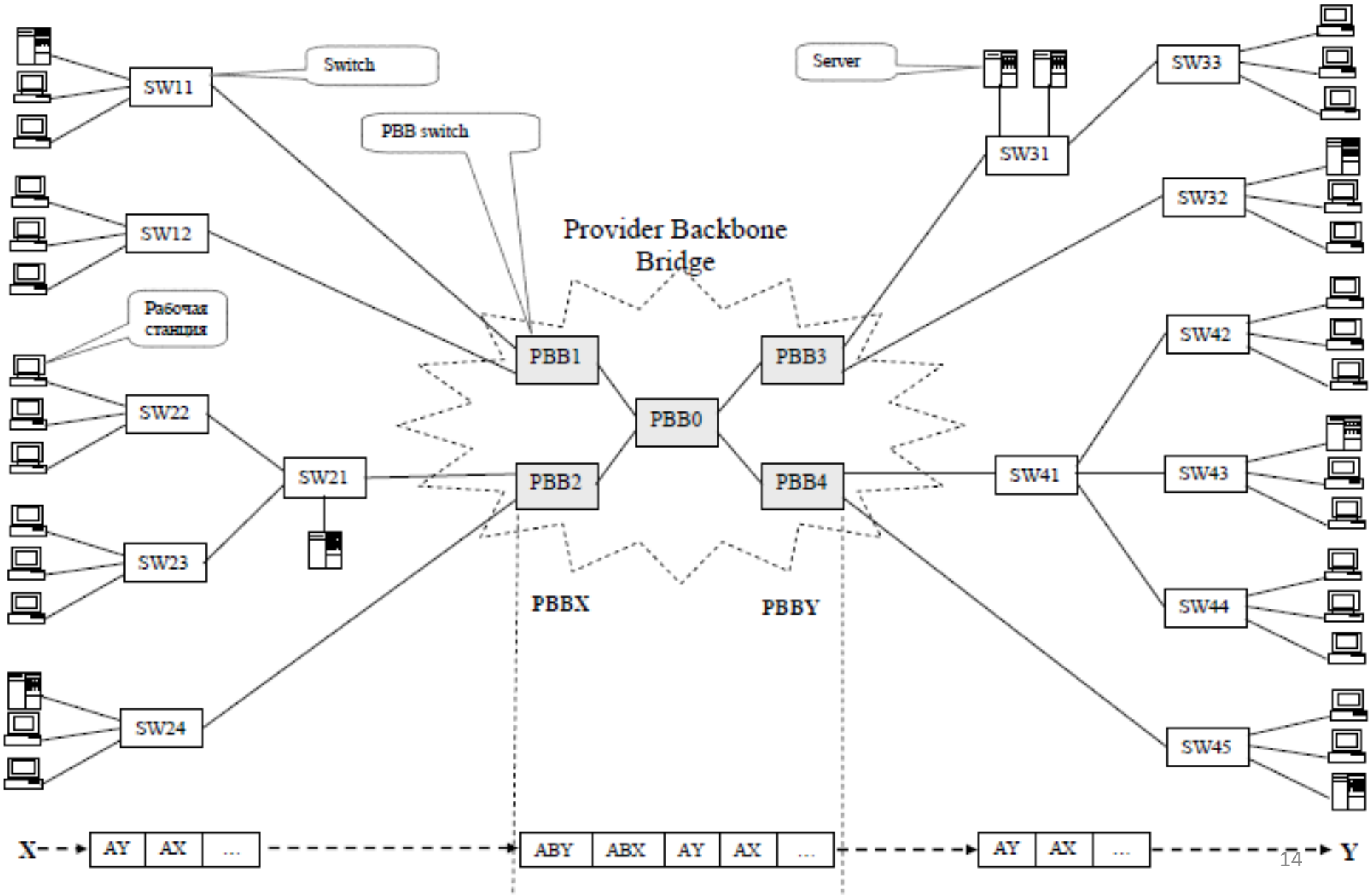


Figure 1. Format of IEEE 802.1ah frame header.

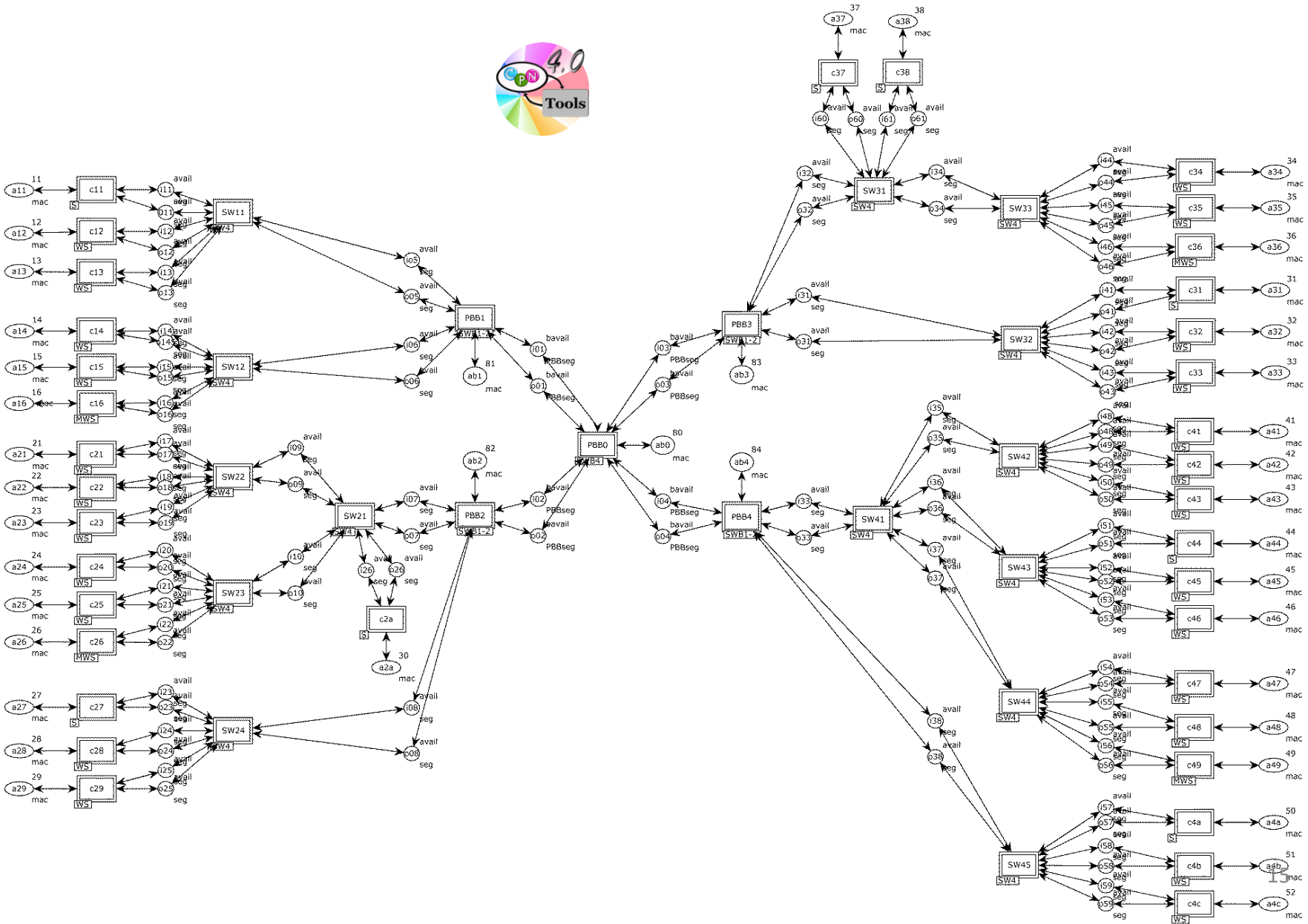
Table 1. Description of IEEE 802.1ah frame header fields.

Notation	Description
B-DA	Backbone destination address
B-SA	Backbone source address
B-Tag	Backbone VLAN tag
I-Tag	Service instance tag
C-DA	Customer destination address
C-SA	Customer source address
S-Tag	Service provider VLAN tag
C-Tag	Customer VLAN tag
Data	Data
FCS	Frame check sequence

PBB Network Example



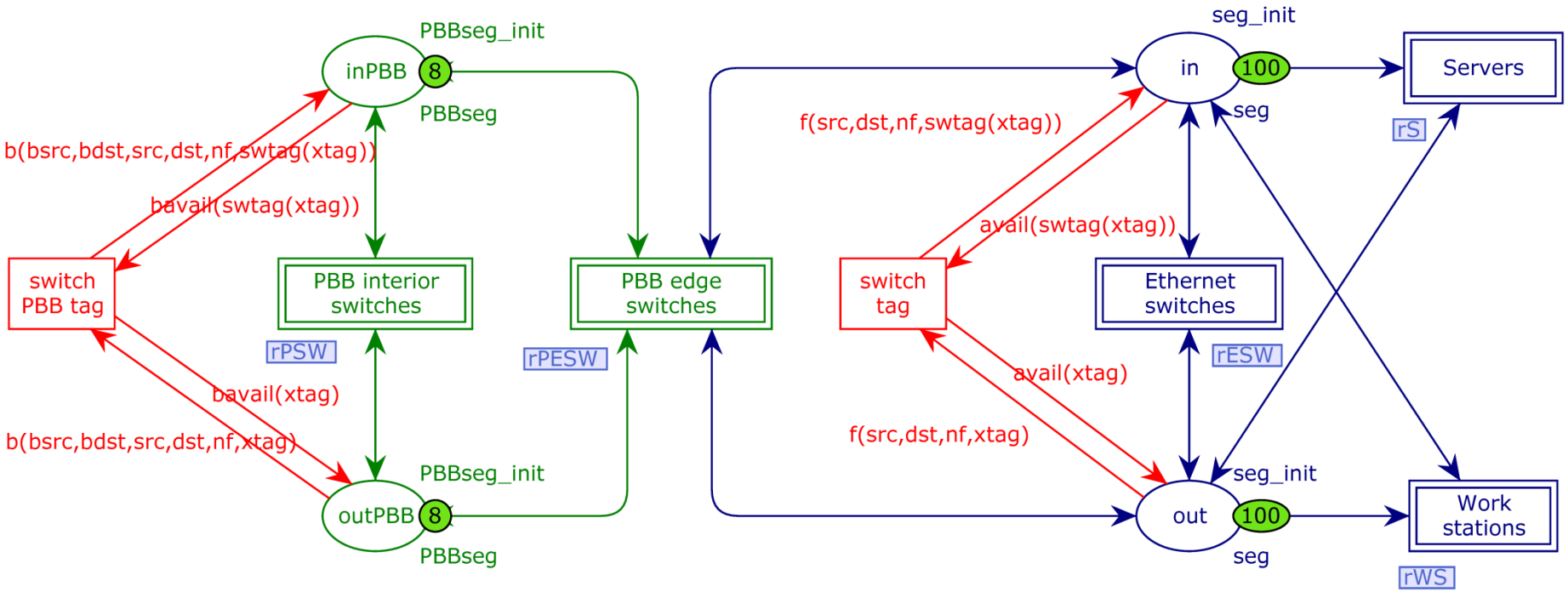
Conventional CPN Tools model of PBB



Principles of Reenterable Models

1. Represent each component (device) in a single copy.
2. Represent the network topology as a parameter (marking of a place or a constant).
3. Supply data of components (devices) and moving elements (packets) with a special topology location tag (TLT).
4. Represent movements of elements with respect to topology via switching TLTs.
5. Using data of components choose the instance having required TLT.

Main Page of Reenterable PBB Model

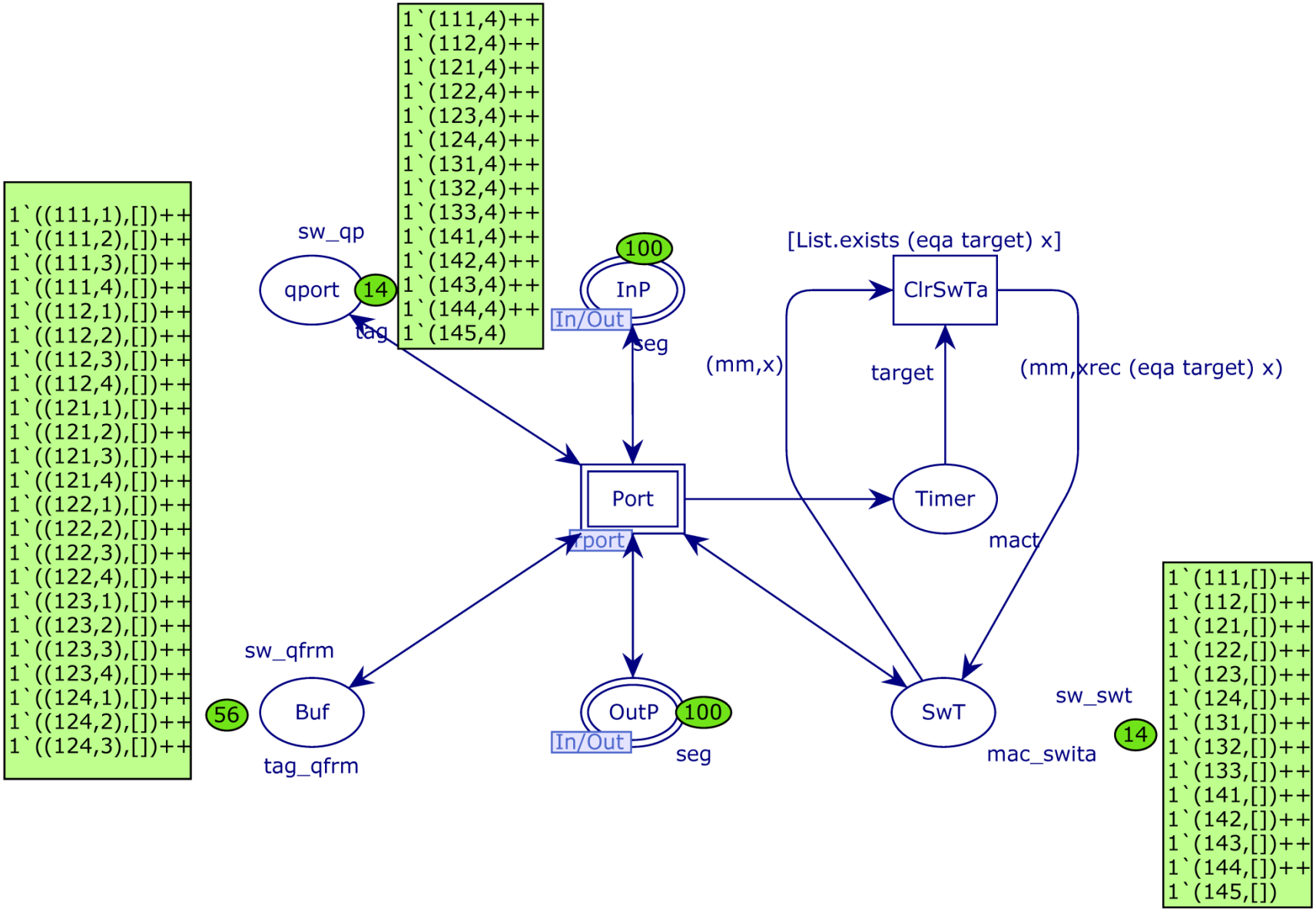



```
colset tag=product mac*portnum;
colset frm = product mac * mac * tstamp * tag timed;
colset seg = union f:frm + avail:tag timed;
colset PBBfrm = product mac * mac * mac * mac * tstamp * tag timed;
colset PBBseg = union b:PBBfrm + bavail:tag timed;
```

Switching TLTs

```
colset topo=product tag*tag;

fun eqtag1 t (rr:topo) = ((#1 rr)=t);

fun eqtag2 t (rr:topo) = ((#2 rr)=t);

fun gtag prd [] = ((0,0),(0,0)) |

gtag prd (q::r) = if prd(q) then q else gtag prd r;

fun swtag(x:tag) = if (List.exists (eqtag1 x) nwtopo)
then (#2 (gtag (eqtag1 x) nwtopo)) else (if
(List.exists (eqtag2 x) nwtopo) then
(#1 (gtag (eqtag2 x) nwtopo)) else (0,0));
```

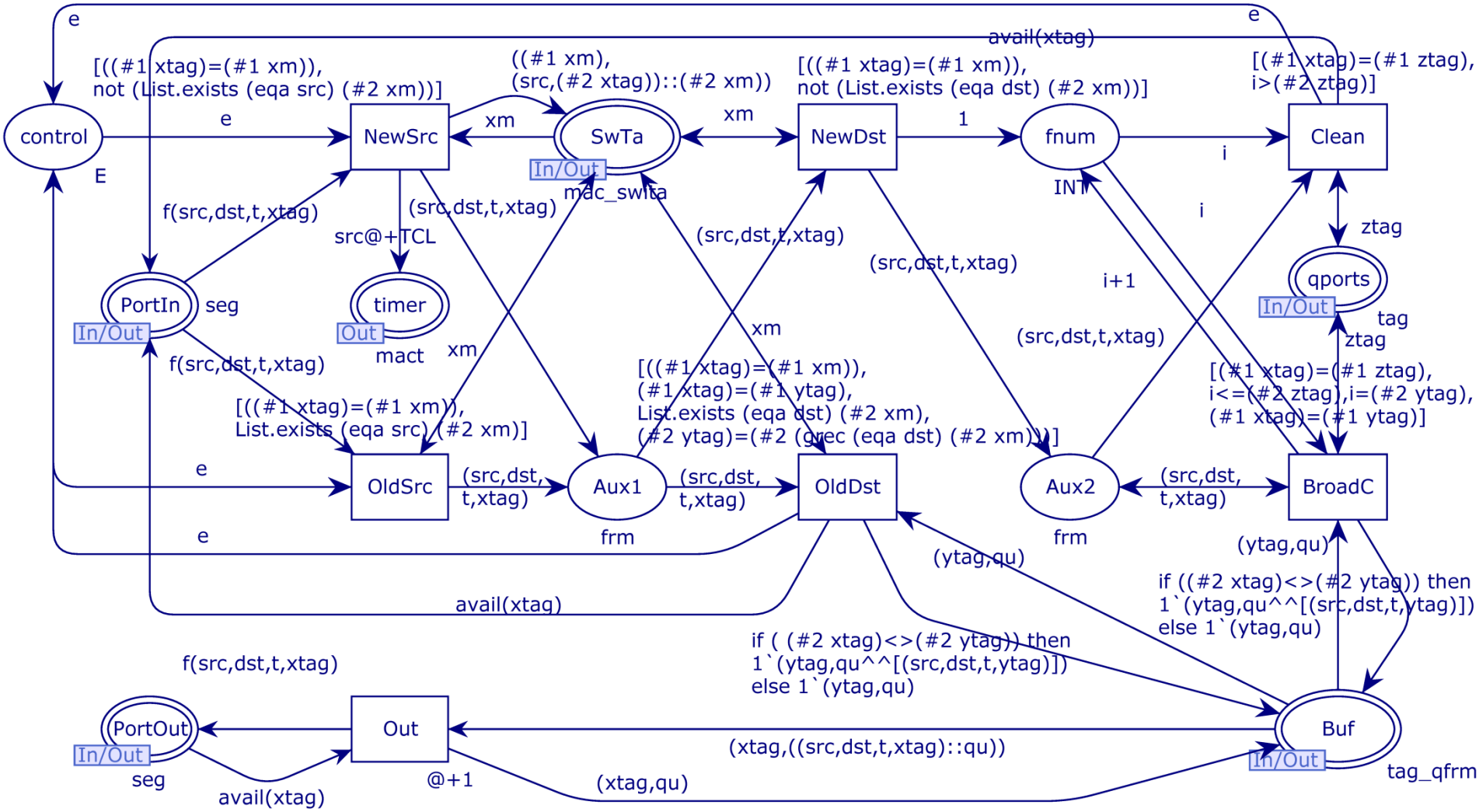
Model of Ethernet/PBB Switch



Work with Switching Tables

```
colset swi = product mac * portnum;
colset swita=list swi;
colset mac_swita=product mac*swita;
fun eqa a (rr:swi)=((#1 rr)=a);
fun grec prd [] = (0,0) | grec prd (q::r) =
        if prd(q) then q else grec prd r;
fun xrec prd [] = [] | xrec prd (q::r) =
        if prd(q) then r else q::(xrec prd r);
```

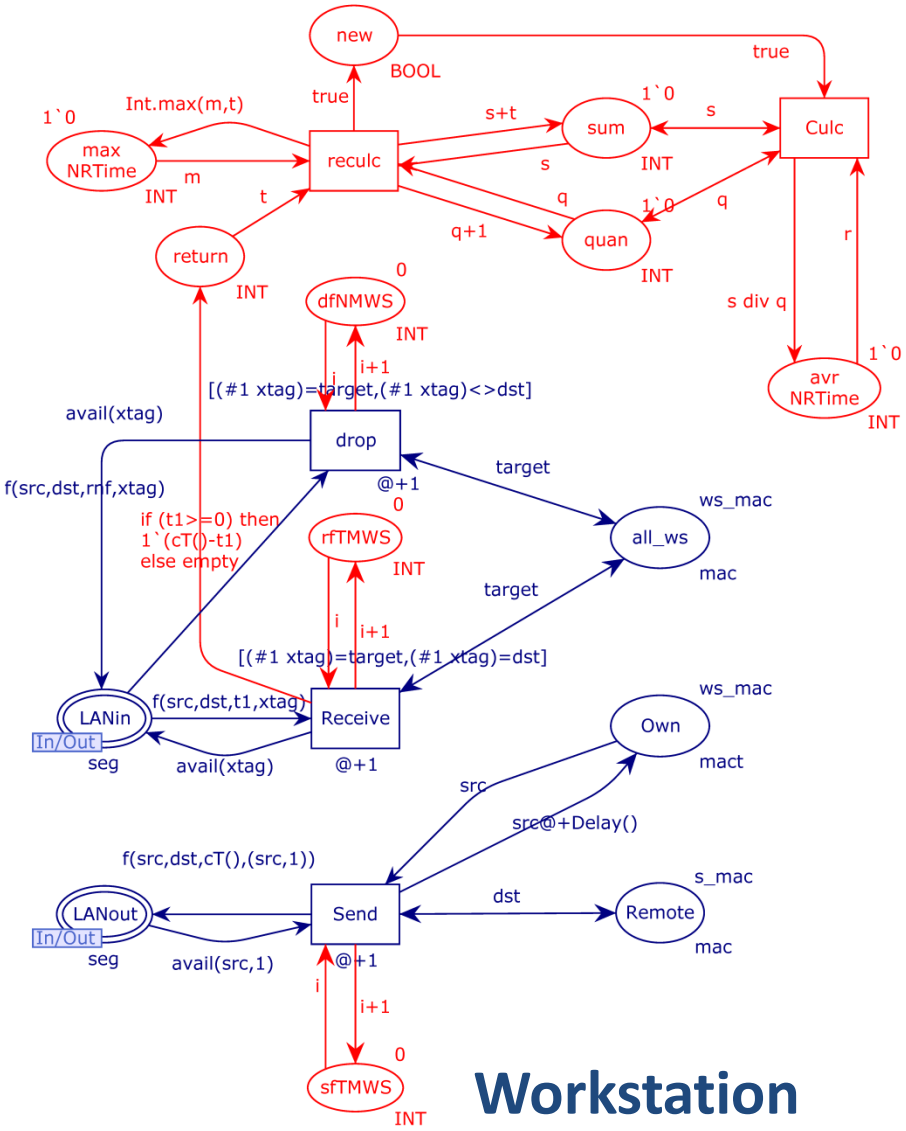
Model of Ethernet/PBB Switch Port



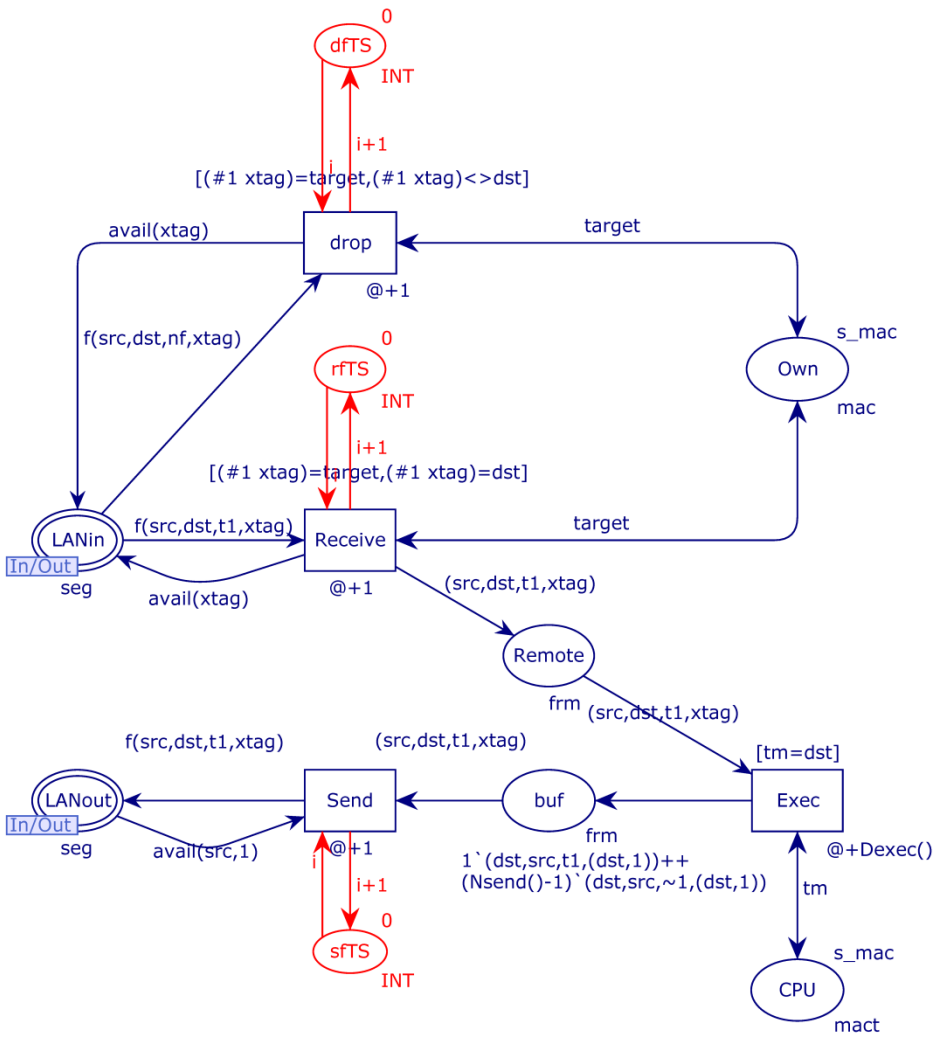
Queues of Frames to Ports

```
colset qfrm = list frm;
colset pqfrm = product portnum *qfrm;
colset tag_qfrm=product tag*qfrm;
colset xfrm = union cf:frm + bf:PBBfrm;
colset qxfrm = list xfrm;
colset pqxfrm = product portnum *qxfrm;
colset tag_qxfrm = product tag *qxfrm;
```

Models of Terminal Devices



Workstation

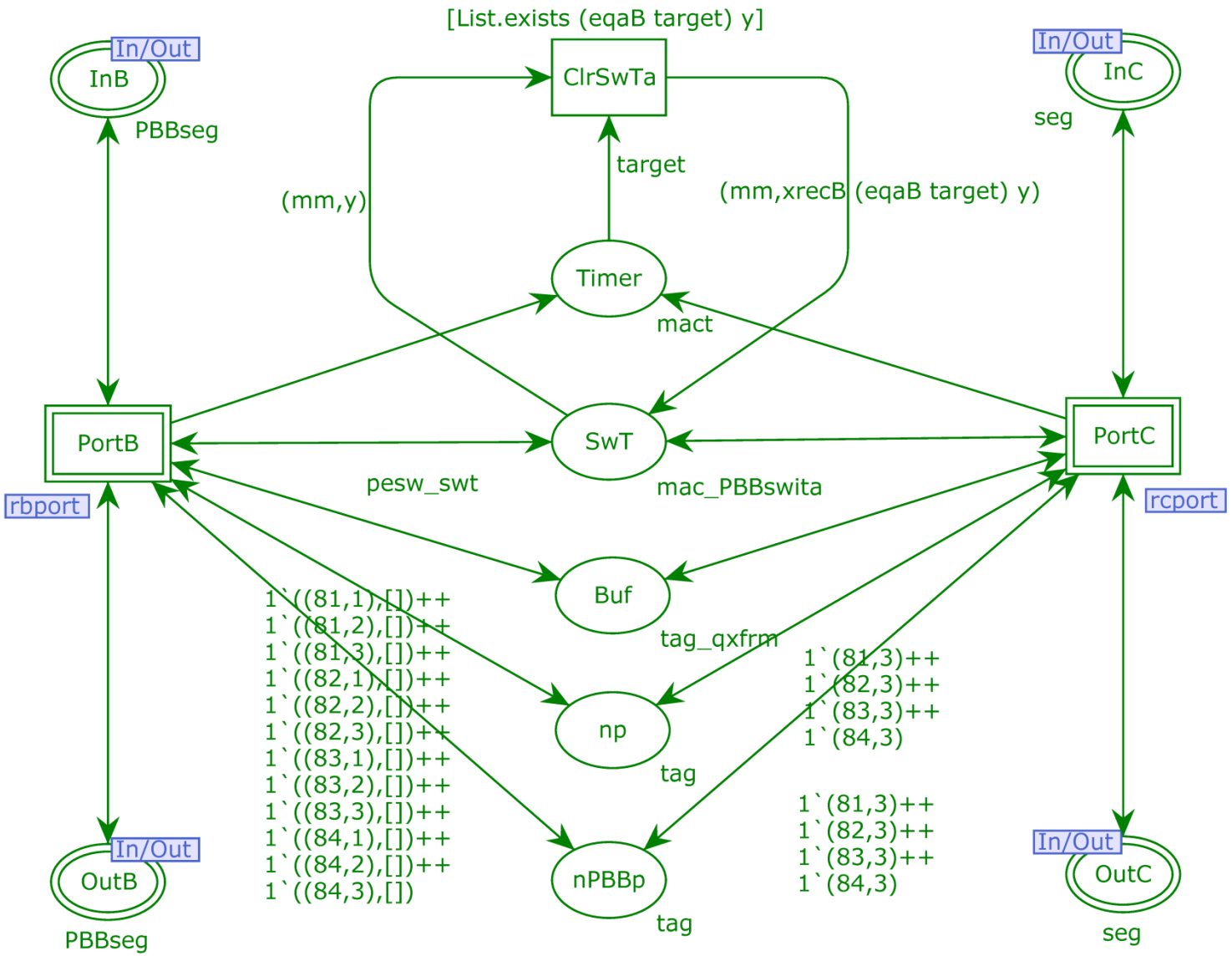


Server

Measuring Subnets

- CPN is a universal algorithmic system
- Specify algorithms of computing network performance characteristics by CPN
- Use timestamps and counters of packets/frames
- Use algorithms to compute average and dispersion in a single passage
- Performance evaluation on-fly without storing bulky initial statistical information

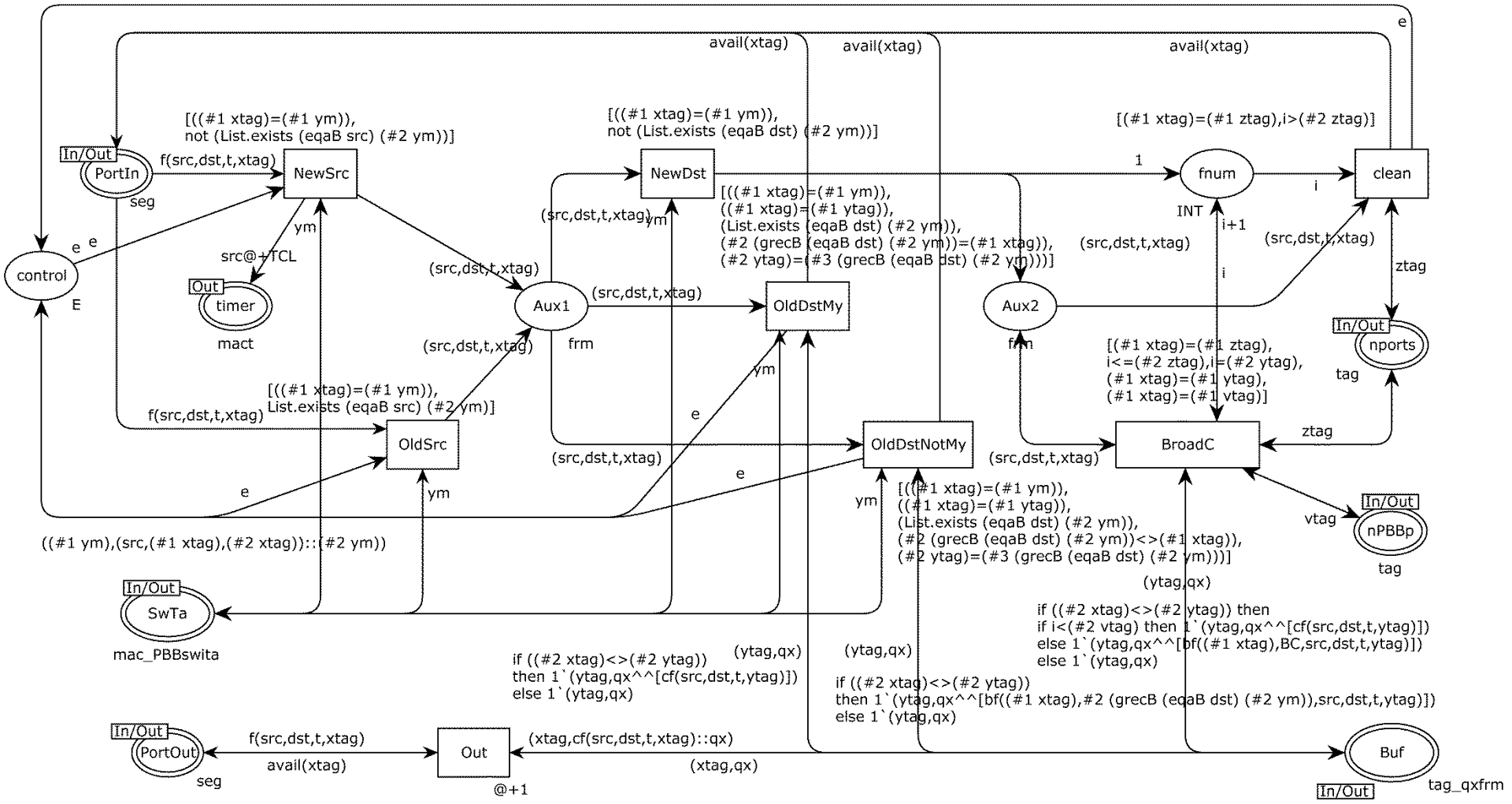
Model of PBB Edge Switch



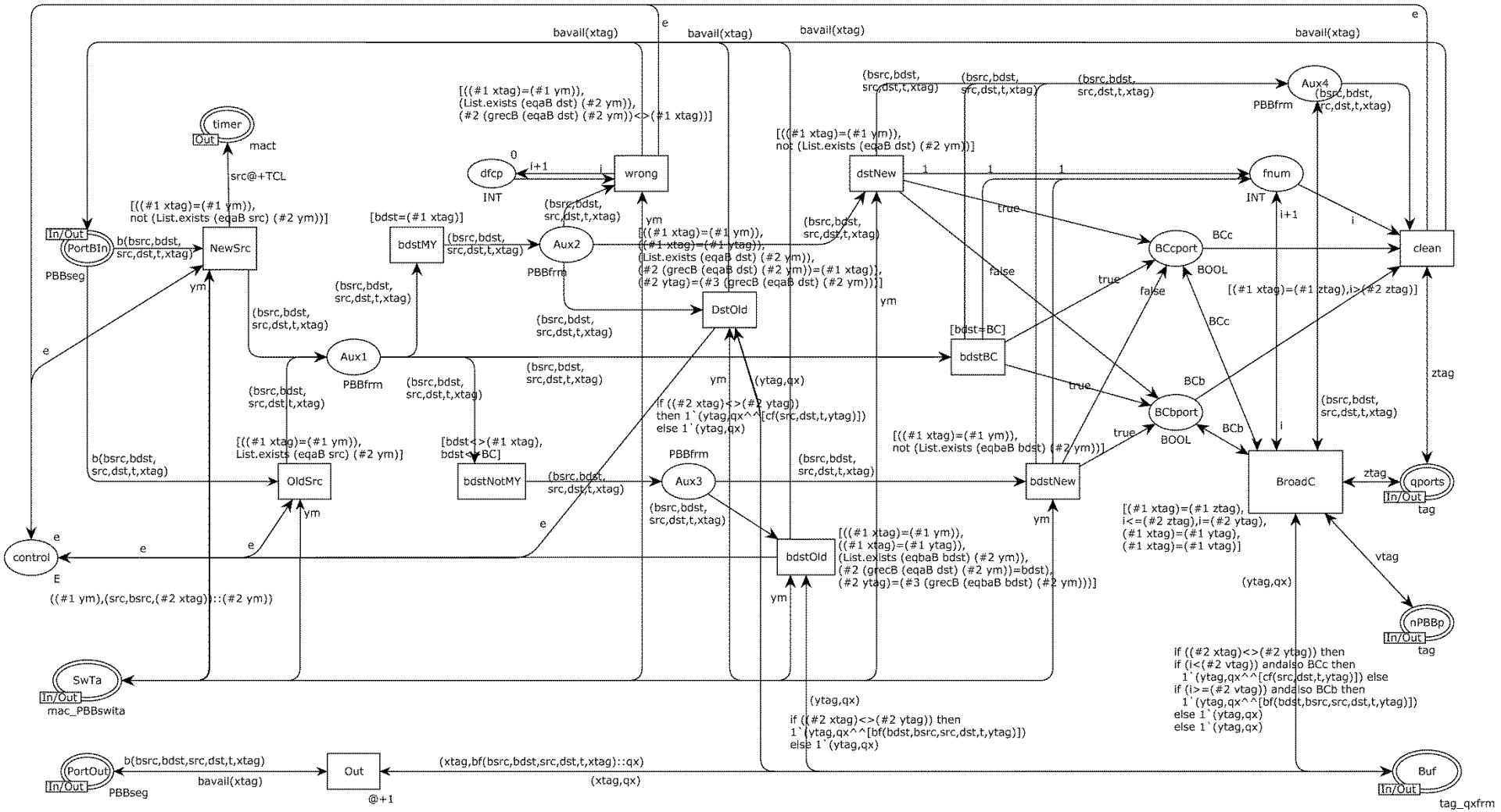
PBB Edge Switching Tables

```
colset PBBswi = product mac * mac * portnum;
colset PBBswita = list PBBswi;
colset mac_PBBswita = product mac * PBBswita;
fun eqaB a (rr:PBBswi)=((#1 rr)=a);
fun eqbaB a (rr:PBBswi)=((#2 rr)=a);
fun grecB prd [] = (0,0,0)       |    grecB prd (q::r) =
          if prd(q) then q else grecB prd r;
fun xrecB prd [] = []       |    xrecB prd (q::r) =
          if prd(q) then r else q::(xrecB prd r);
```

Model of PBB Edge Switch C-Port



Model of PBB Edge Switch B-port



Parameters of Model

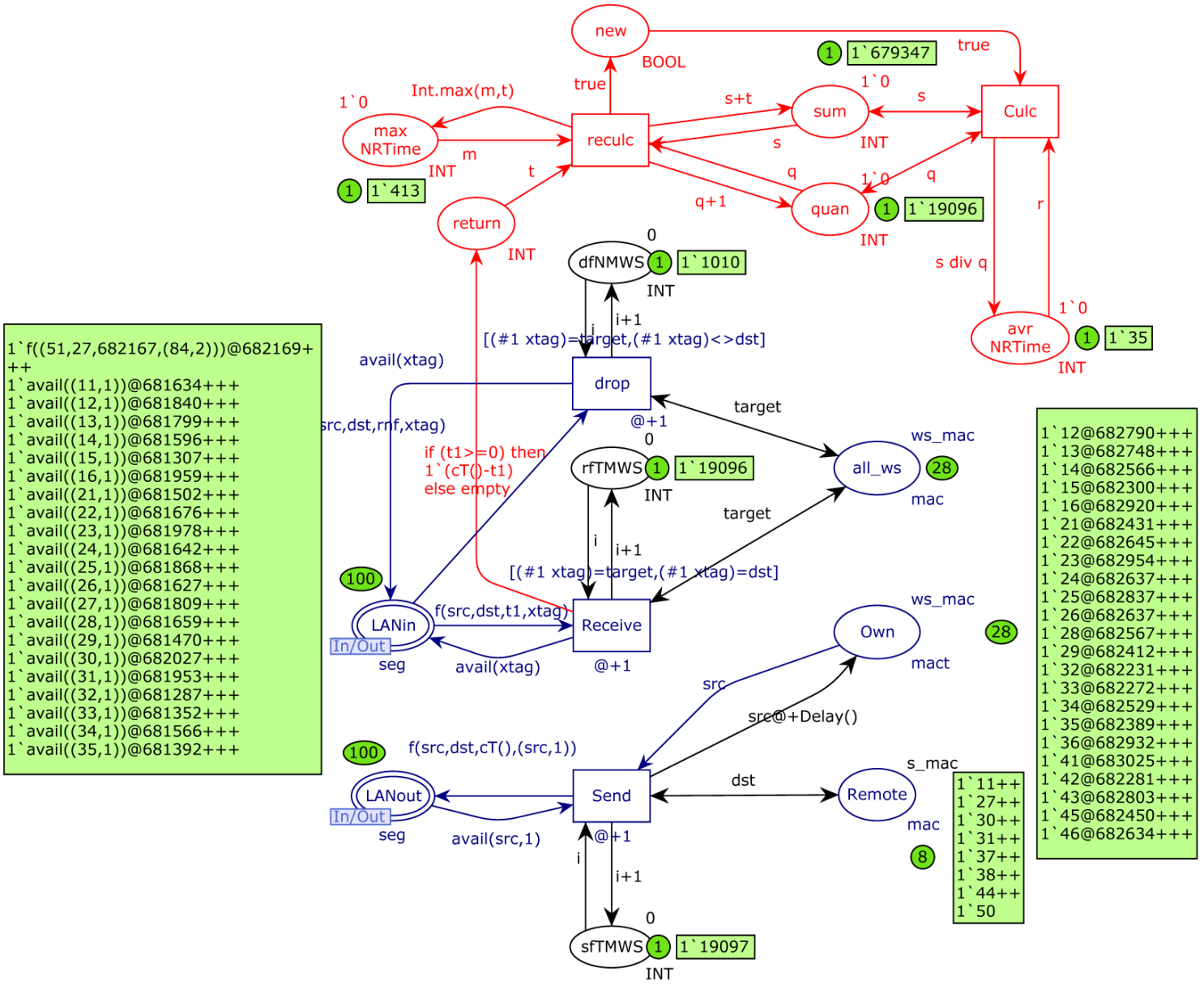
```
val nwtopo=[
((11,1),(111,1)),((12,1),(111,2)),((13,1),(111,3)),
((14,1),(112,1)),((15,1),(112,2)),((16,1),(112,3)),
((21,1),(122,1)),((22,1),(122,2)),((23,1),(122,3)),
((24,1),(123,1)),((25,1),(123,2)),((26,1),(123,3)),
((30,1),(121,3)),((122,4),(121,1)),((123,4),(121,2)),
((27,1),(124,1)),((28,1),(124,2)),((29,1),(124,3)),
((31,1),(132,1)),((32,1),(132,2)),((33,1),(132,3)),
((34,1),(133,1)),((35,1),(133,2)),((36,1),(133,3)),
((37,1),(131,1)),((38,1),(131,2)),((133,4),(131,3)),
((41,1),(142,1)),((42,1),(142,2)),((43,1),(142,3)),
((44,1),(143,1)),((45,1),(143,2)),((46,1),(143,3)),
((47,1),(144,1)),((48,1),(144,2)),((49,1),(144,3)),
((142,4),(141,1)),((143,4),(141,2)),((144,4),(141,3)),
((50,1),(145,1)),((51,1),(145,2)),((52,1),(145,3)),
((80,1),(81,3)),((80,2),(82,3)),((80,3),(83,3)),
((80,4),(84,3)),((81,1),(111,4)),((81,2),(112,4)),
((82,1),(121,4)),((82,2),(124,4)),((83,1),(131,4)),
((83,2),(132,4)),((84,1),(141,4)),((84,2),(145,4))];
```

```
val ws_mac=1`12++1`13++1`14++1`15++1`16++
1`21++1`22++1`23++1`24++1`25++1`26++1`28++1`29++
1`32++1`33++1`34++1`35++1`36++1`41++1`42++1`43++
1`45++1`46++1`47++1`48++1`49++1`51++1`52;

val s_mac=1`11++1`27++1`30++1`31++1`37++1`38++
1`44++1`50;

val sw_mac=1`111++1`112++1`121++1`122++1`123++
1`124++1`131++1`132++1`133++1`141++1`142++
1`143++1`144++1`145;

val be_mac=1`81++1`82++1`83++1`84;

val sw_swt=mac_swita.mult(sw_mac,1`[]);

val sw_qp=tag.mult(sw_mac,1`4);

val sw_qfrm=tag_qfrm.mult(tag.mult(sw_mac,
1`1++1`2++1`3++1`4),1`[]);

val pesw_swt=mac_PBBswita.mult(be_mac,1`[]);
```

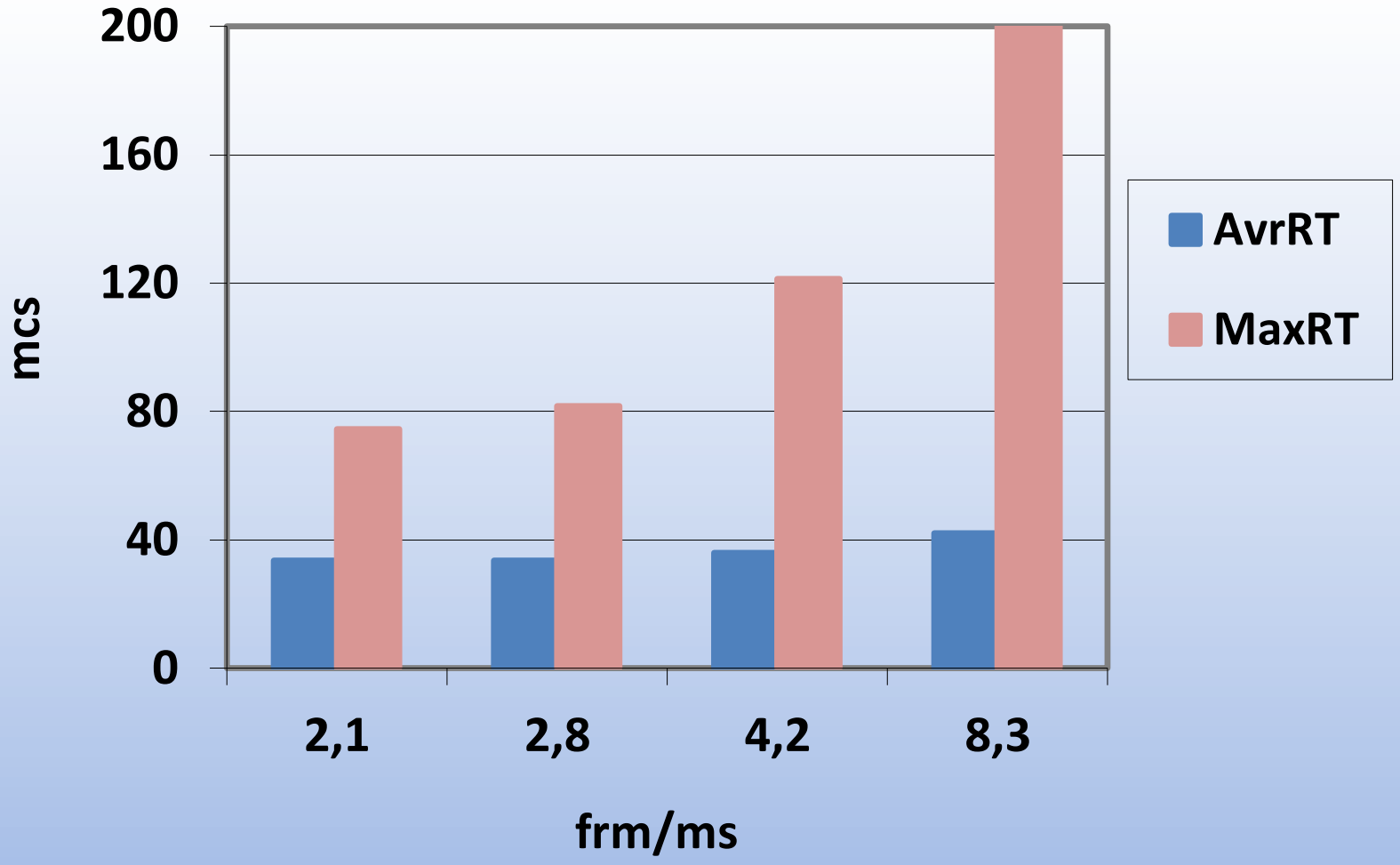
Examples of Switching Tables

- Ethernet:
  1`(123,[(37,4),(11,4),(30,4),(44,4),(50,4),(31,4),(27,4),(38,4),(43,4),(49,4),(45,4),(41,4),(48,4),(34,4),(42,4),(16,4),(35,4),(13,4),(46,4),(36,4),(14,4),(51,4),(47,4),(12,4),(33,4),(52,4),(32,4),(15,4),(22,4),(29,4),(21,4),(28,4),(23,4),(24,1),(26,3),(25,2)])
- Edge PBB:
  1`(82,[(11,81,3),(44,84,3),(37,83,3),(31,83,3),(50,84,3),(27,82,2),(30,82,1),(38,83,3),(43,84,3),(49,84,3),(45,84,3),(41,84,3),(48,84,3),(34,83,3),(42,84,3),(16,81,3),(35,83,3),(13,81,3),(46,84,3),(36,83,3),(14,81,3),(51,84,3),(47,84,3),(12,81,3),(24,82,1),(33,83,3),(22,82,1),(21,82,1),(52,84,3),(32,83,3),(26,82,1),(23,82,1),(15,81,3),(29,82,2),(25,82,1),(28,82,2)])
- Interior PBB:
  1`(80,[(84,4),(82,2),(83,3),(81,1)])

Performance Evaluation



Network Response Time Evaluation



Conclusions

- Constructed a reenterable model of PBB
- Model has the same structure for any given network
- Network topology and equipment characteristics are parameters of the model
- Performance evaluation results coincide with the conventional model
- Devised for model-driven design of networks

http://daze.ho.ua

Basic References

- D.A. Zaitsev, Clans of Petri Nets: Verification of Protocols and Performance Evaluation of Networks, LAP, Saarbrucken, 2013.
- Zaitsev D.A., Shmeleva T.R. “Parametric Petri Net Model for Ethernet Performance and QoS Evaluation,” in *Proc. of 16th Workshop on Algorithms and Tools for Petri Nets*, September 25-26, 2009, University of Karlsruhe, Germany, pp. 15-28.
- P.P. Vorbiyenko, K.D. Guliaiev, D.A. Zaitsev, T.R. Shmeleva, “PBB Efficiency Evaluation via Colored Petri Net Models.” Communications and Network, vol. 2, 2010, pp. 113-124.
- T. Shmeleva, “Reenterable model of communication grid with cut-through nodes: Performance evaluation,” in *4th International Scientific-Practical Conference Problems of Infocommunications: Science and Technology (PIC S&T)*, 10-13 Oct. 2017, pp. 223 – 227.

Co-Authors

Dr. Tatiana R. Shmeleva received the Eng. degree in applied mathematics from Moscow Railroad Engineer Institute, Moscow, Russia, in 1990; the Ph.D. degree in telecommunications from Odessa National Academy of Telecommunications, Odessa, Ukraine, in 2008. T. R. Shmeleva is author of more than 30 papers and chapters published in journals, books, and conference proceedings. Her research interest is Petri net theory and its application in networking, especially in verifications of protocols, and performance evaluation. She developed the analysis of infinite Petri nets with regular structure and parametric colored Petri net models of networks. Since 2005, she has been with the Odessa National Academy of Telecommunications, where she is currently an Associate Professor of Switched systems department.

Dr. Anatolii Illich Sleptsov received his Ph.D. in automation & telecommunications from Donetsk Polytechnic Institute (Ukraine) in 1974 and a doctor of sciences degree in information technologies in 1989 at the Institute of Cybernetics of National Academy of Sciences of Ukraine. Prof. Sleptsov taught undergraduate and graduate information system courses for computer and software engineering students in Donetsk Polytechnic Institute and Donetsk National University for more than 30 years. Since 2007, he has been teaching data mining courses for management students in the National Pedagogical Dragomanov University. He is also the author of more than 50 articles and six books. The developed software system Opera-Topaz worked for years in the production of very sophisticated articles.