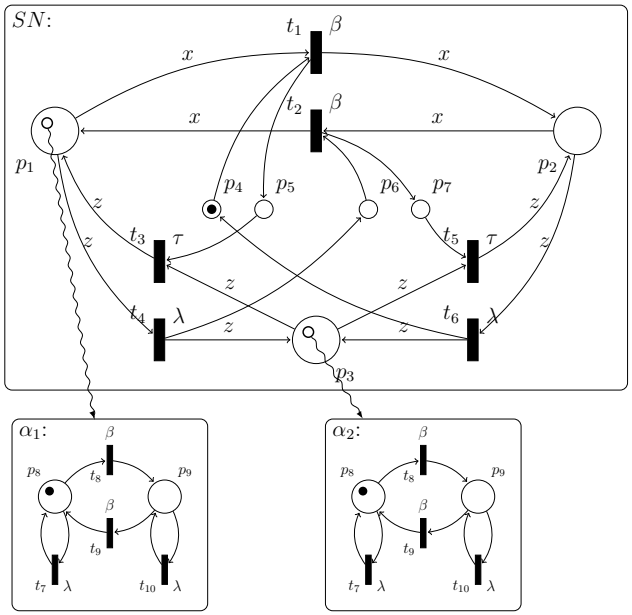
Methods of Analysis for Nested Petri Nets (NP-nets)



Speaker: L. Dworzanski (leo@mathtech.ru)
Sci. Advisor: I. A. Lomazova (ilomazova@hse.ru)
National Research University Higher School of Economics

NP-nets — the first impression

An instance of “nets-within-nets” paradigm (cf. R.Valk 1998)

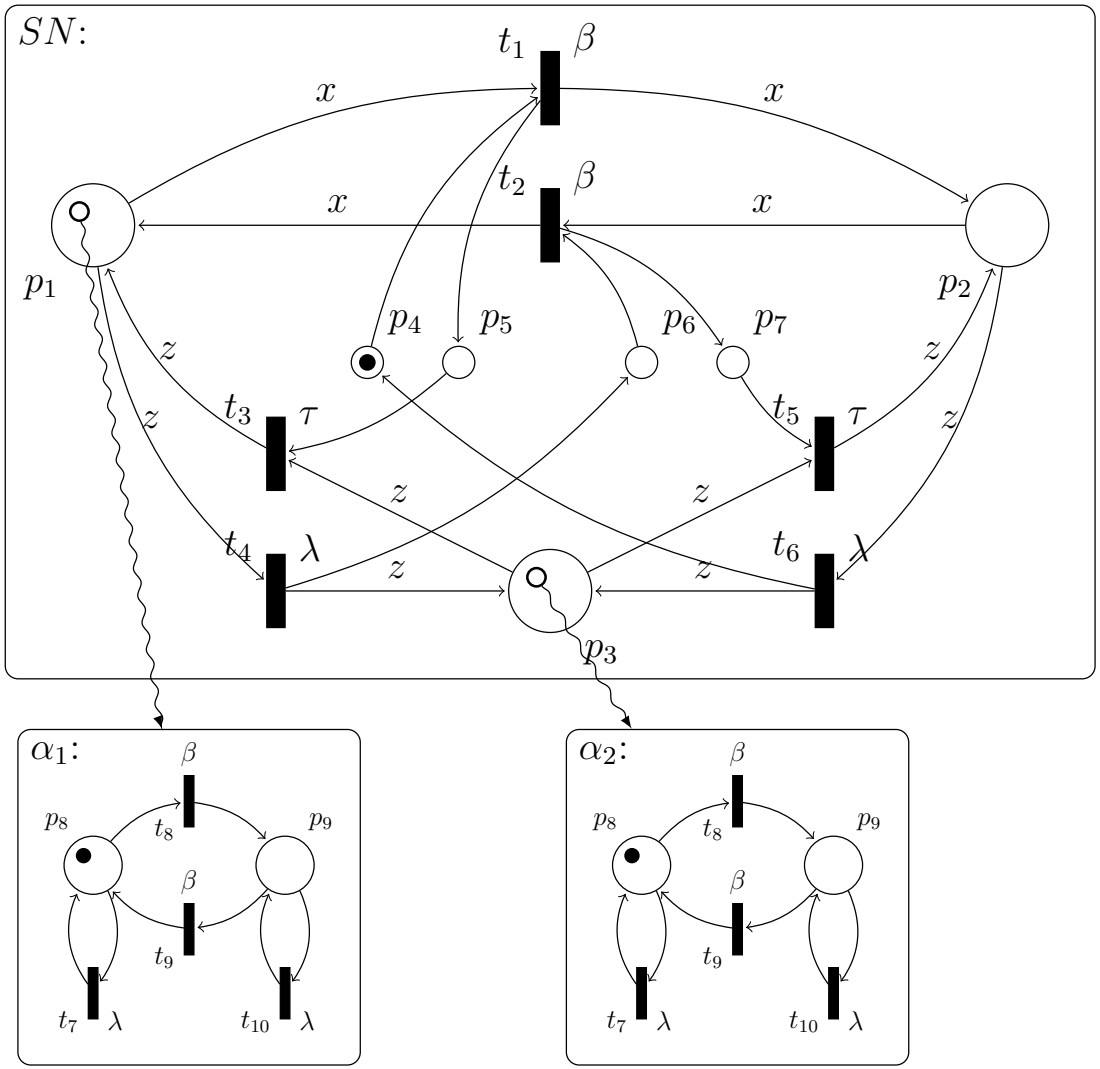
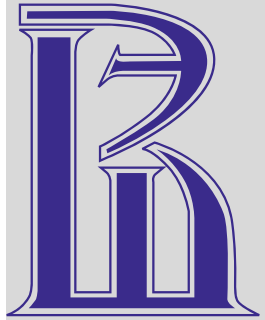


Figure : An NP-net — a caught sample.

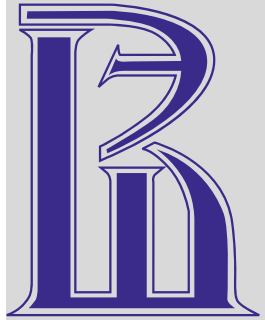
Whats the Plan?

- **Prerequisites of NP-nets**
- Ostensive definition of NP-nets: Simple model of P2P protocol
- Formal definition of NP-nets
- Need for analysis methods
- Compositionality of NP-net properties
- Technical achievements
- Ongoing research
- Conclusions and further directions



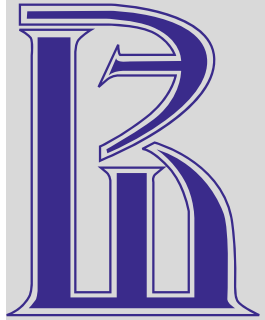
Prerequisites of NP-nets

- Via NP-nets we can model/execute adaptive workflow nets and handle exceptions. For details:
  K. M. van Hee, I. A. Lomazova, O. Oanea, A. Serebrenik, N. Sidorova, M. Voorhoeve: *Nested Nets for Adaptive Systems*. ICATPN 2006: 241-260
- "Flat" Petri nets is an inconvenient tool to model systems with distributed agents
  - **P2P protocols**: system net (SN) — orchestration protocol, tokens — peers with protocols implementations (cf. L. Dworzanski, I.A.Lomazova, PSSV'11);
  - **Wireless sensor networks**: SN — physical topology and orchestration protocol, tokens — sensors (cf. N. Buchina paper);
  - **Social systems**: SN — software process, tokens — stuff (developers, testers, project managers et cetera) (A. Phillipov model).
  - **Swarm systems**: SN — orchestration protocol, tokens — attacking drones (D. Frumin model),
  - other applications: unix daemons, mobile code, mobile robot system etc.



Whats the Plan?

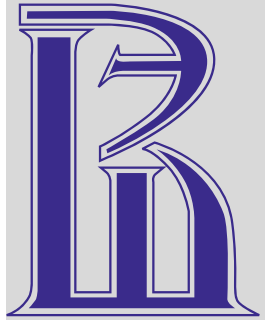
- Prerequisites of NP-nets
- **Ostensive definition of NP-nets: Simple model of P2P protocol**
- Formal definition of NP-nets
- Need for analysis methods
- Compositionality of NP-net properties
- Technical achievements
- Ongoing research
- Conclusions and further directions



Case study: P2P protocol

P2P protocol (system) consists of the participants of three kinds

- Seeds — sources of datum;
- Peers — consumers of datum;
- Tracking server — coordinator of seeds and peers interaction.



Seed

A **seed** is a source of data. In the initial state it waits for a request from a peer to upload the data to the peer. Then it clears its upload buffer and returns to the initial state.

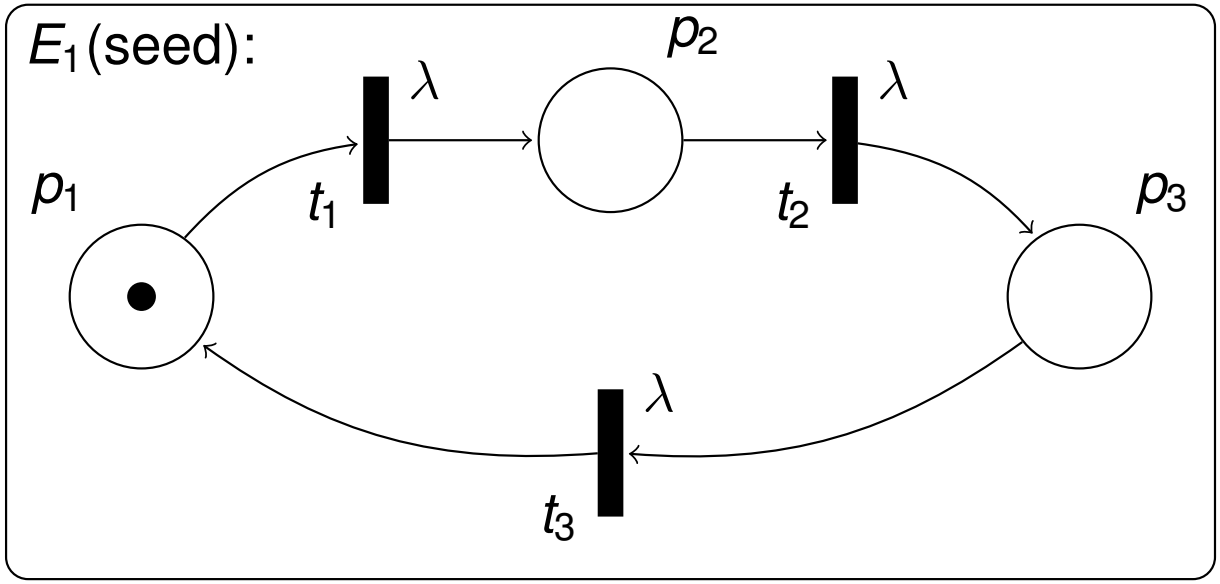
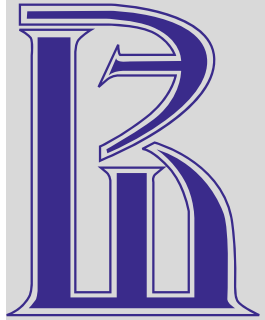


Figure : Seed net.



Peer

A **peer** is just a consumer of data. In the initial state it waits for a seed with a piece of datum. Then it downloads the data and returns to the initial state.

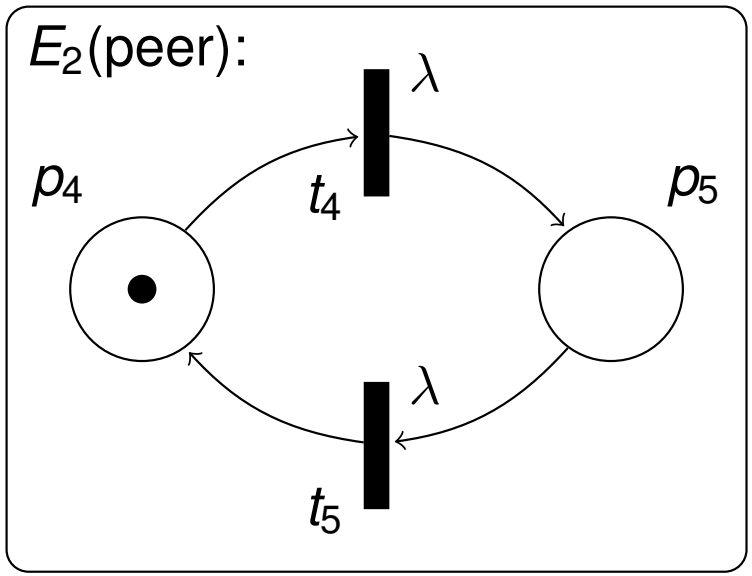


Figure : Peer net.

Tracking server

A **tracking server** is a coordinator of the seeds and peers interaction. It has the pools for ready-to-transmit peers and seeds and the pool for reinitializing seeds.

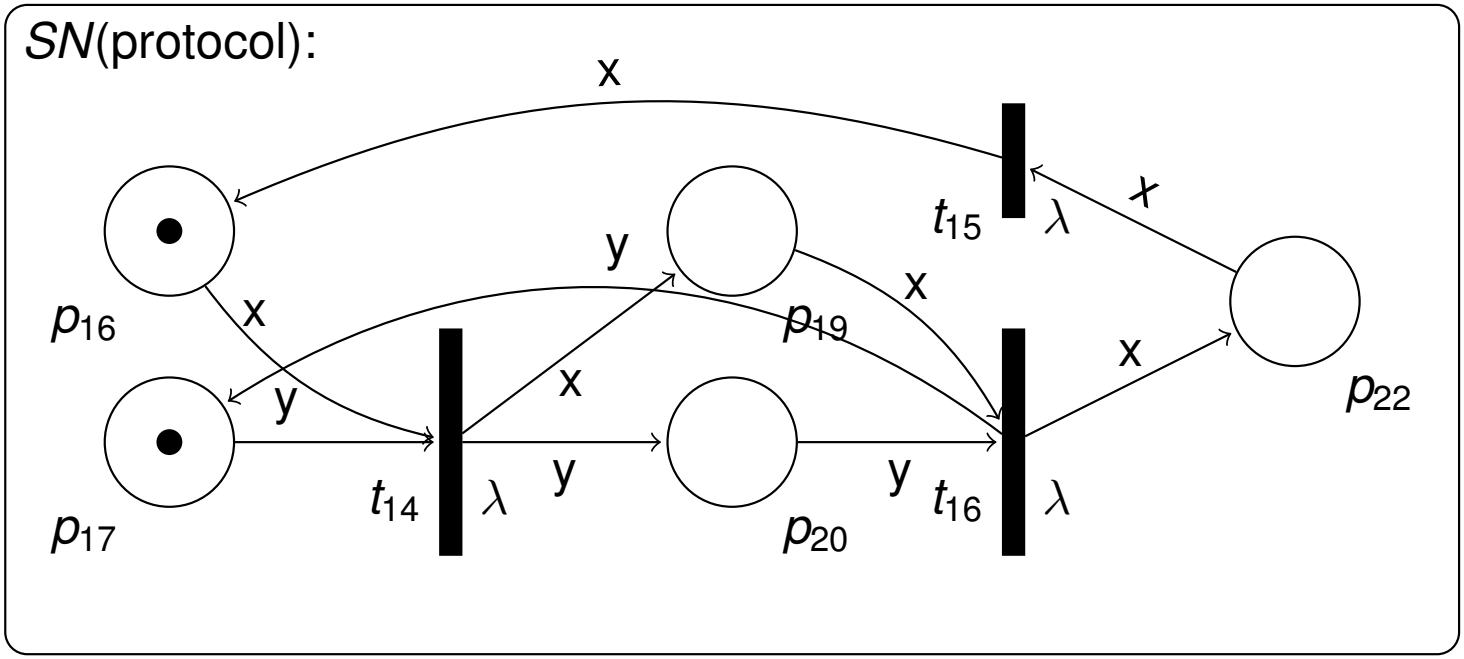


Figure : Tracking server.

Pipe

Tracking server coordinates such interactions by introducing a new entity — **pipe**. In the initial state a pipe waits for a seed and a peer, then it coordinates their interaction. Then the pipe reinitializates its internal buffers and returns to the initial state. A pipe can handle such aspects of a transmition as security, reliability, anonimousity etc.

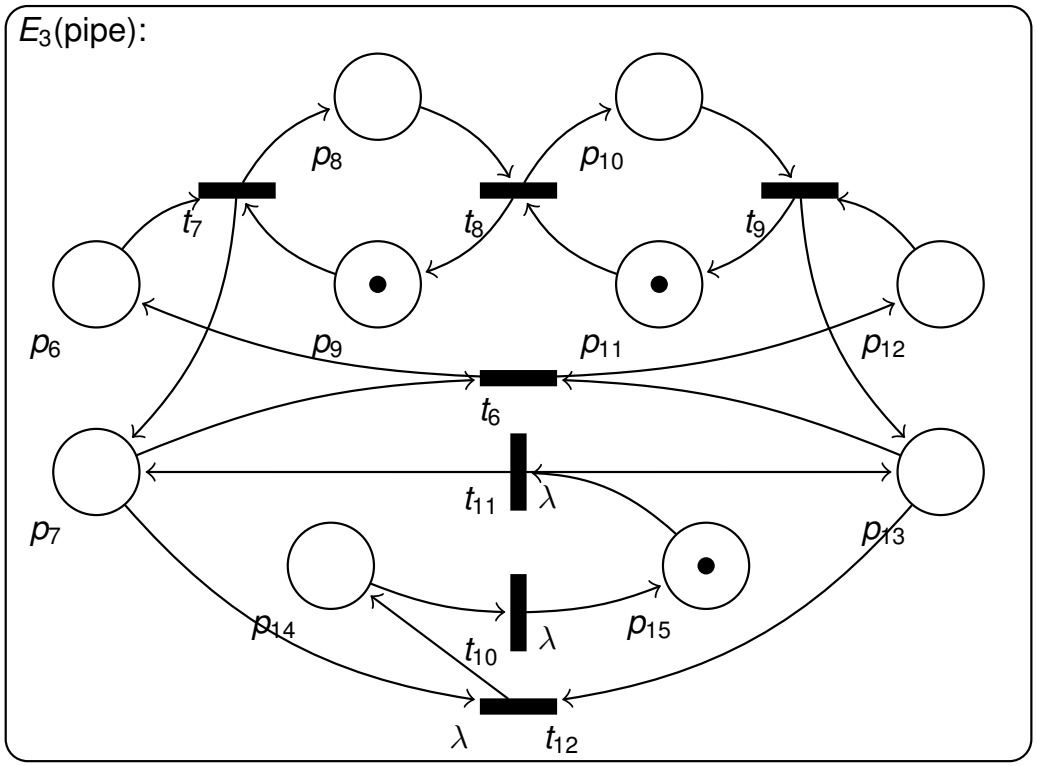
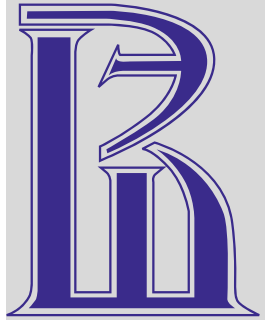


Figure : Pipe net.



Tracking server

Returning to our tracking server. There are no pipes.

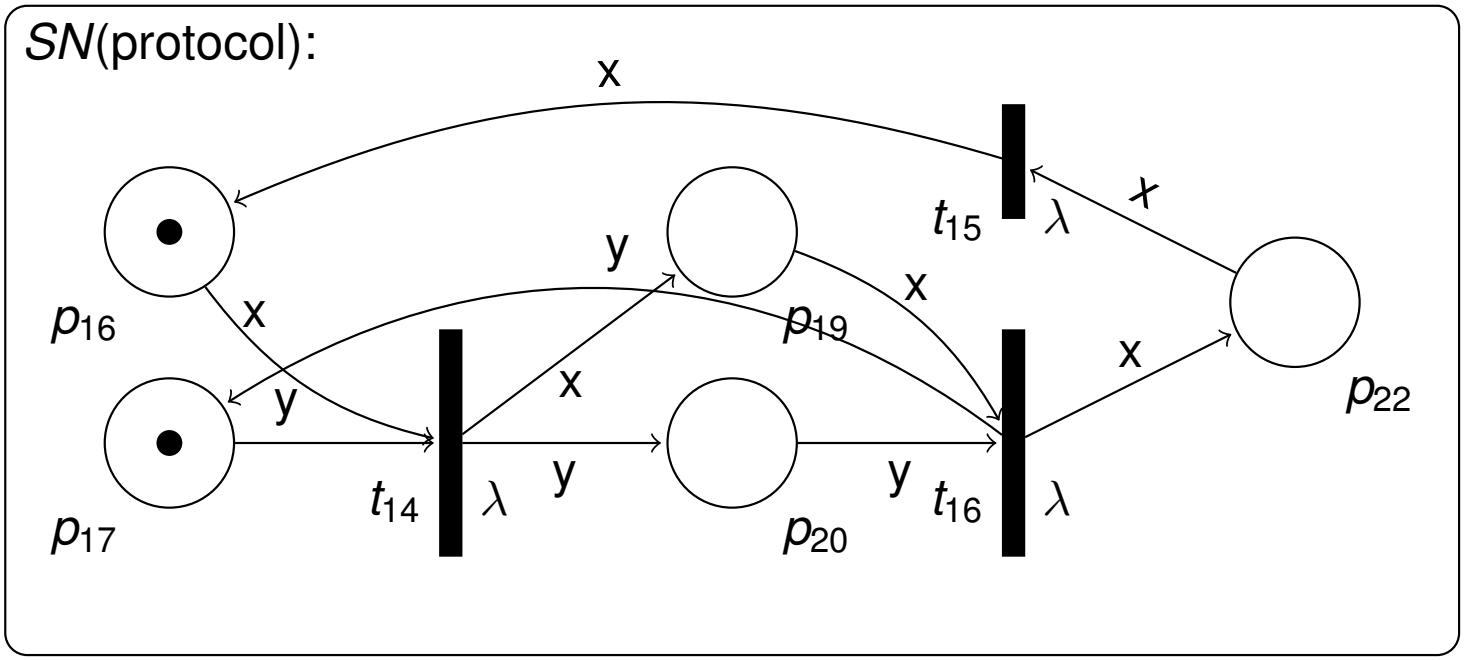


Figure : Tracking server.

Tracking server

But we can add them.

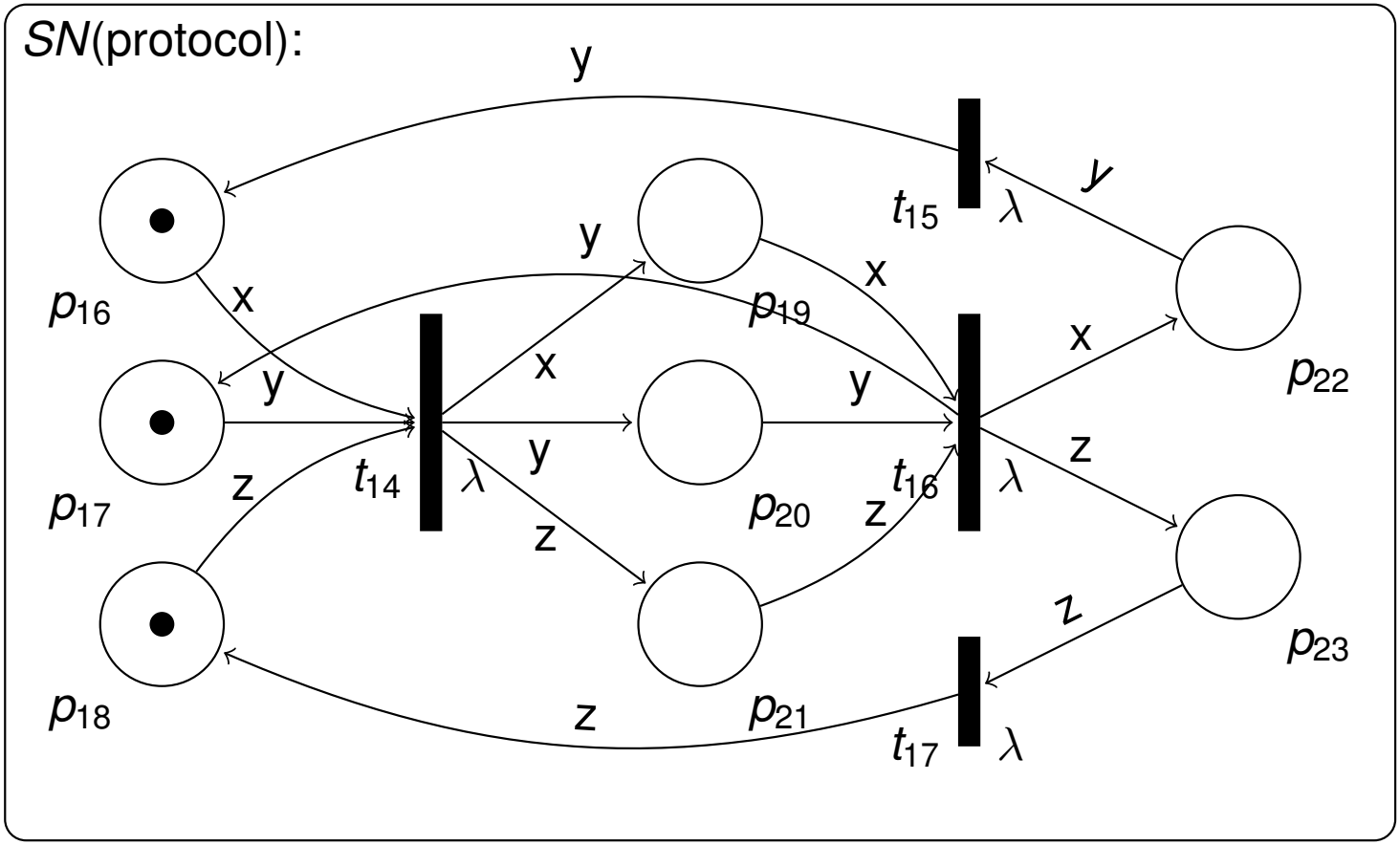
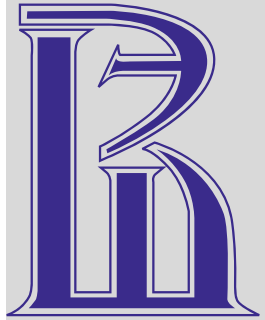
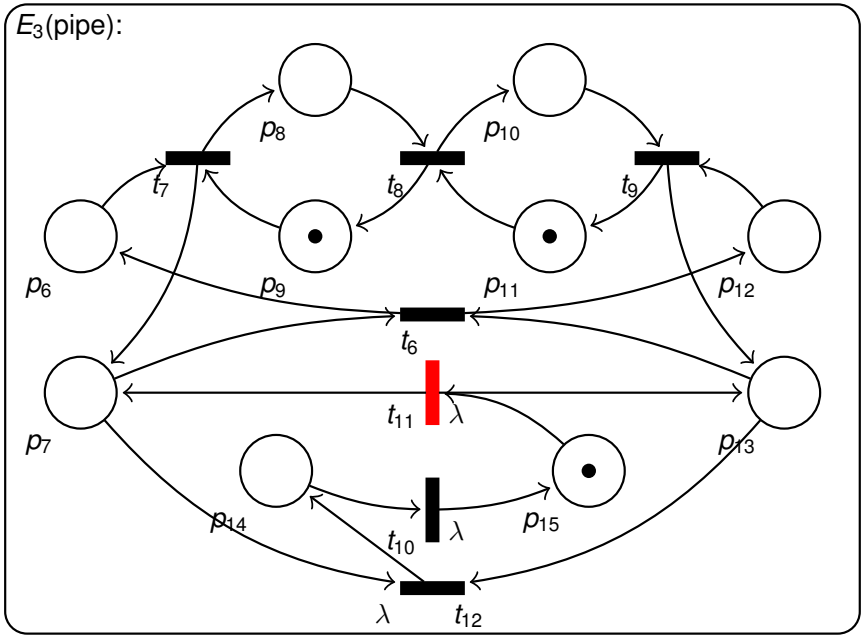
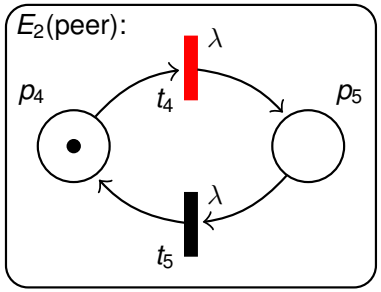


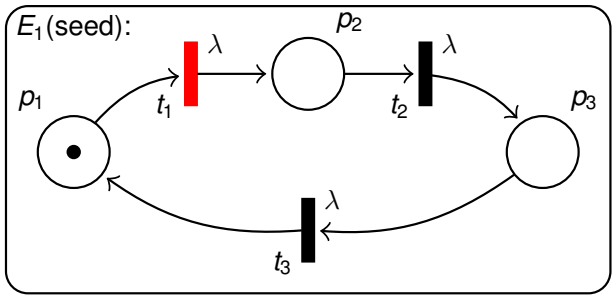
Figure : Tracking server.

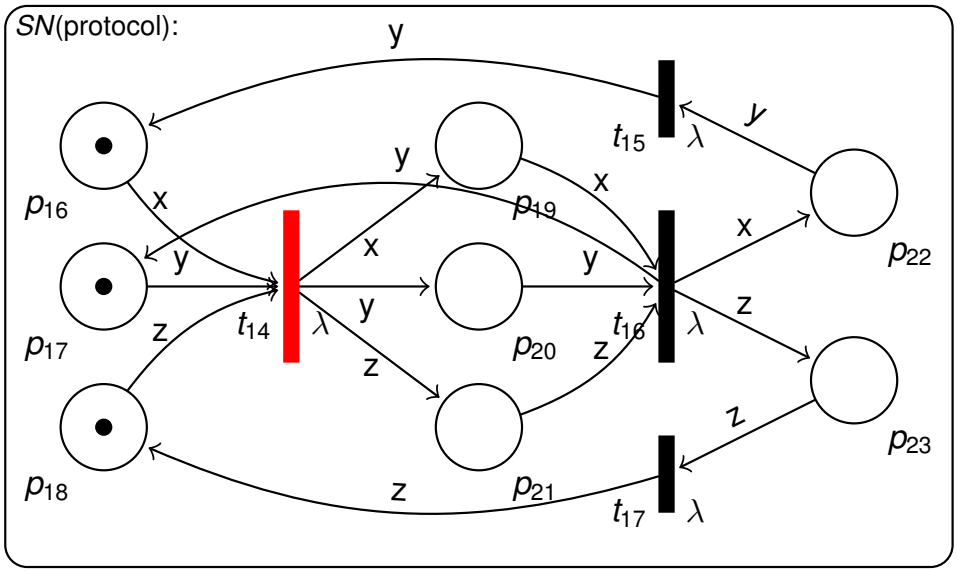


The whole picture

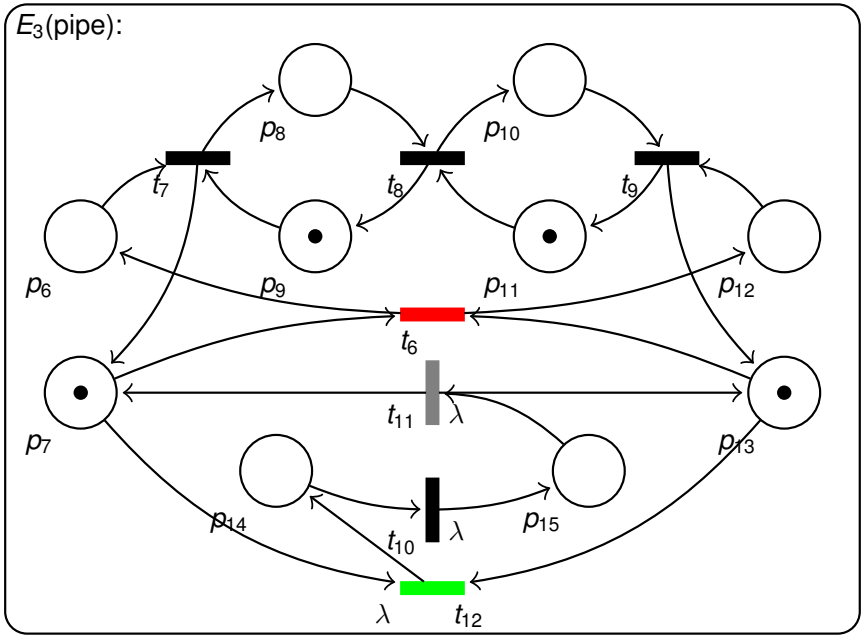


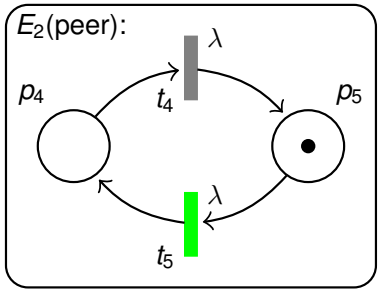


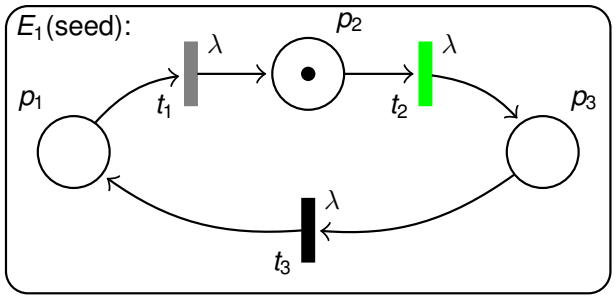


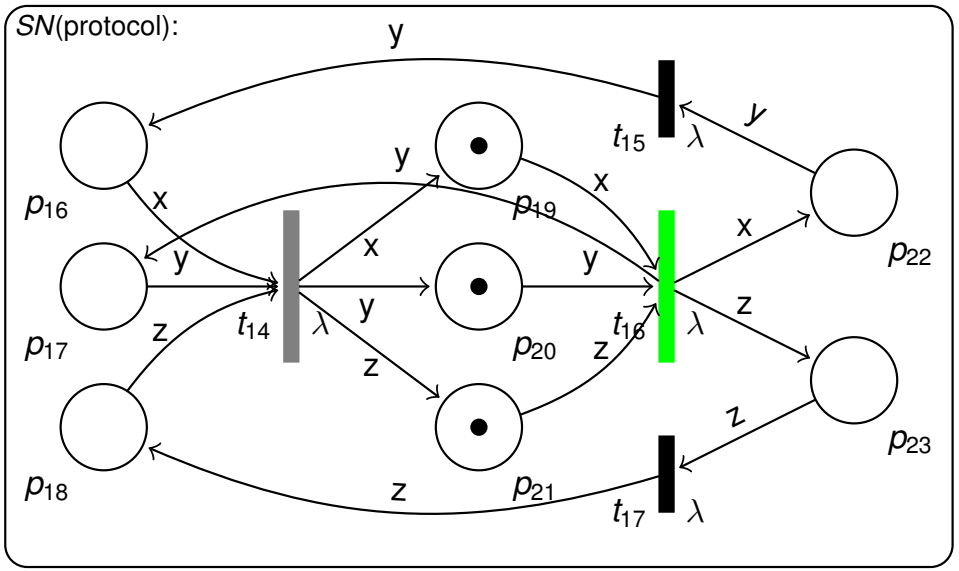


NP-net

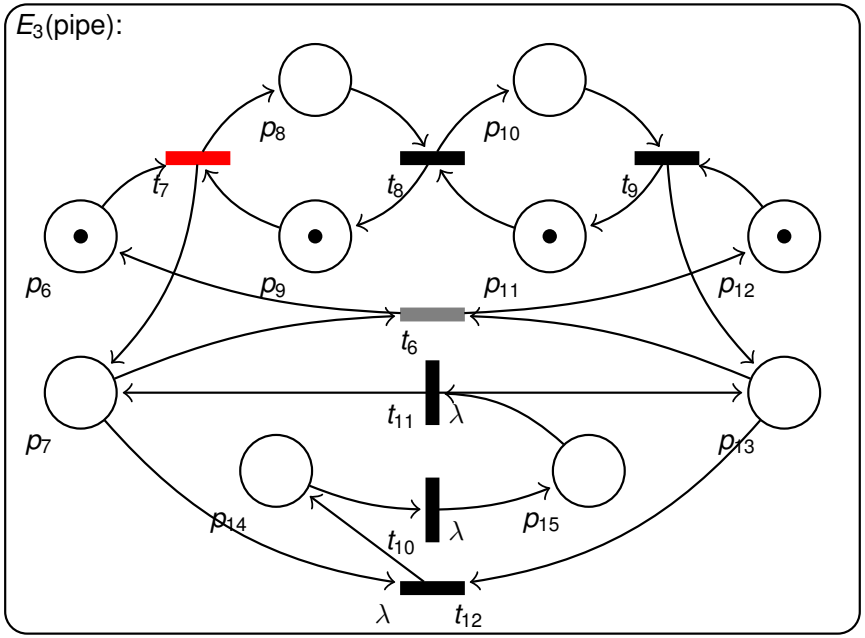


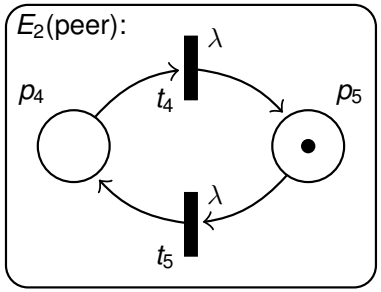


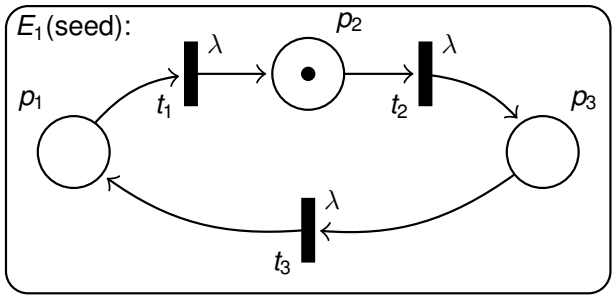


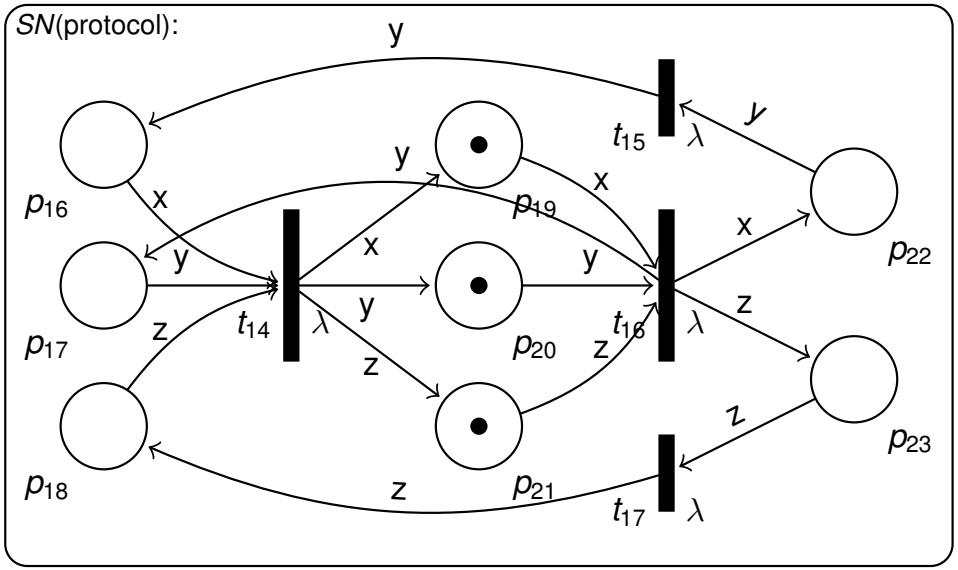


NP-net

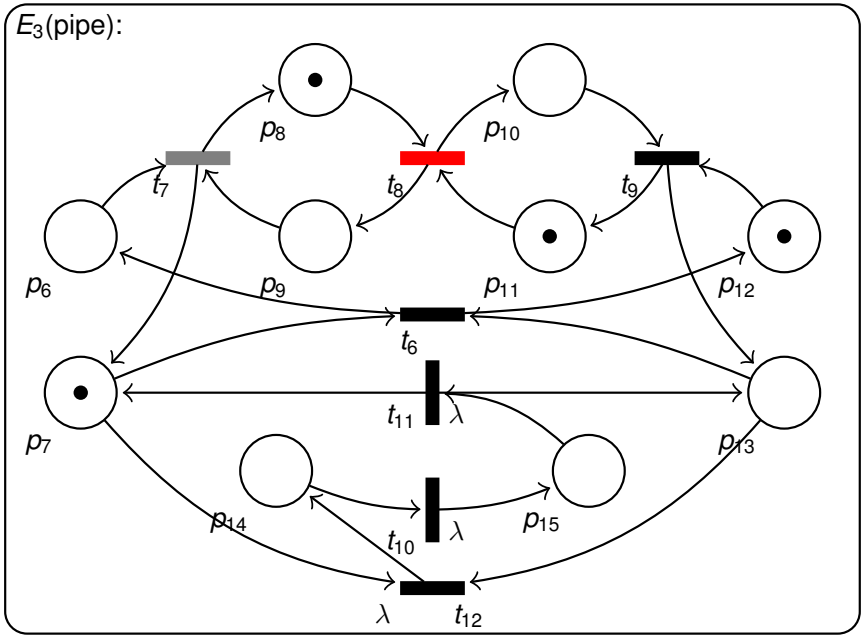


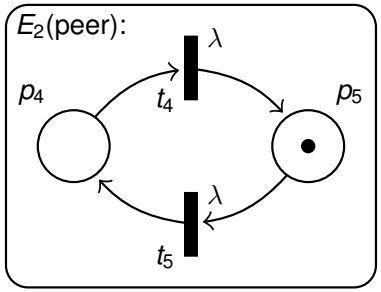


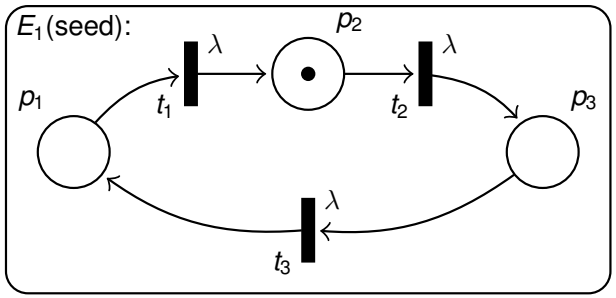


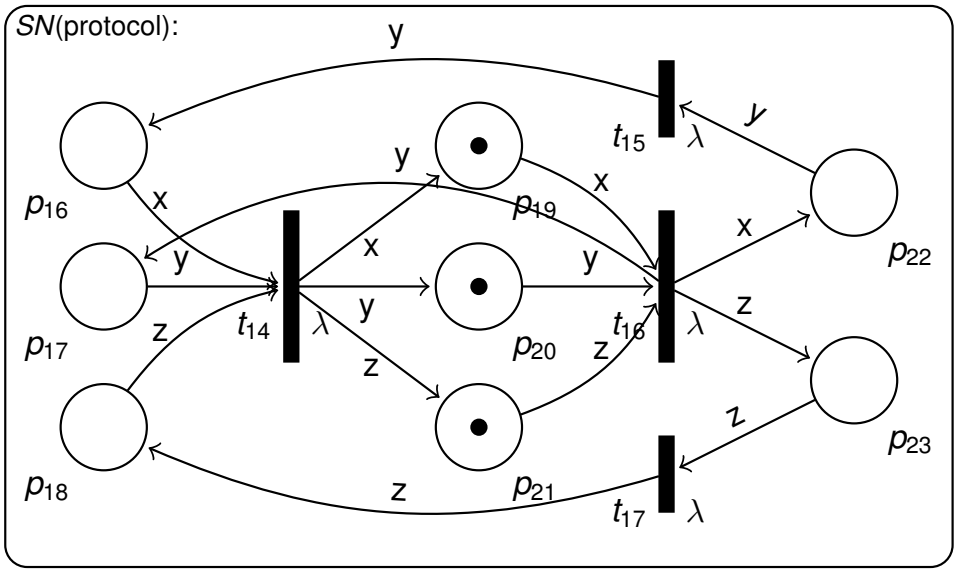


NP-net

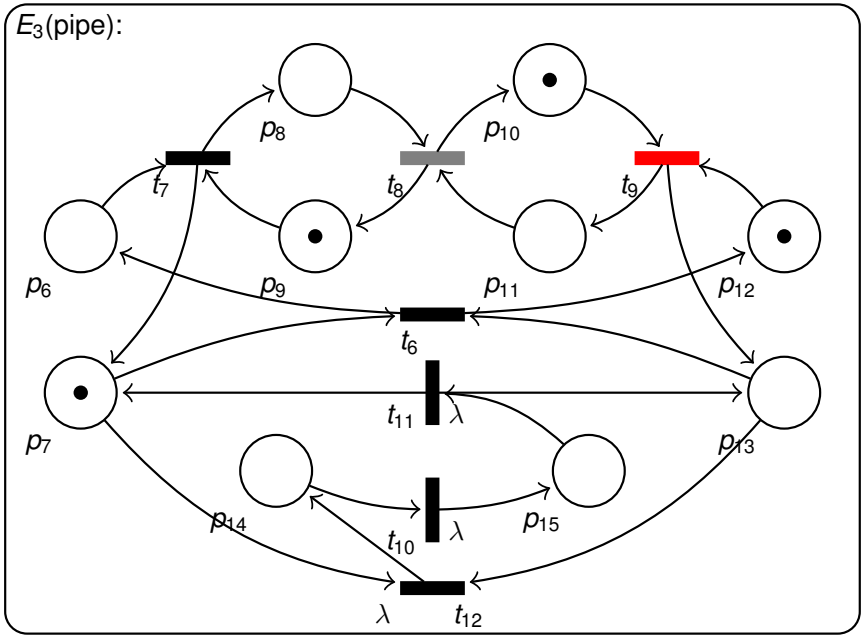


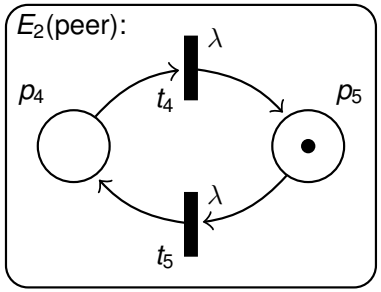


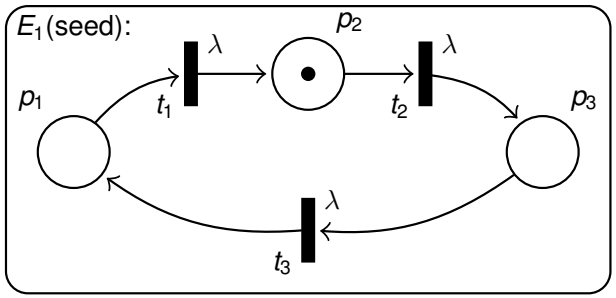


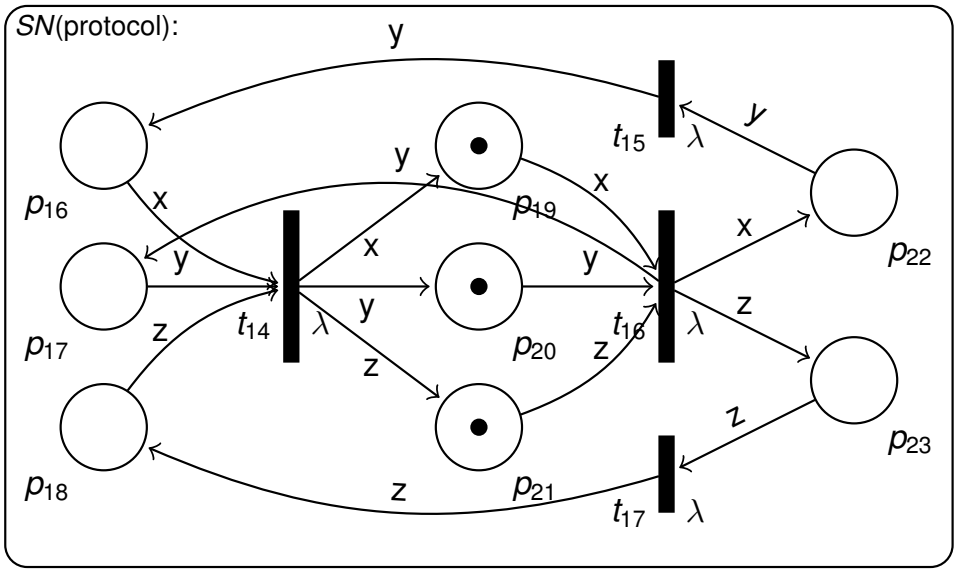


NP-net

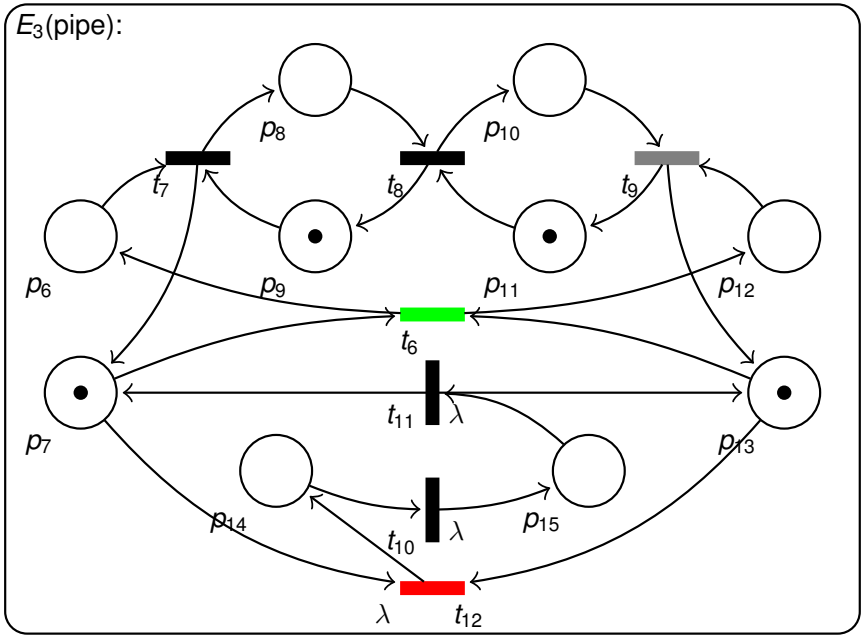


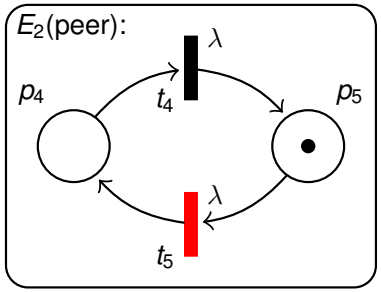


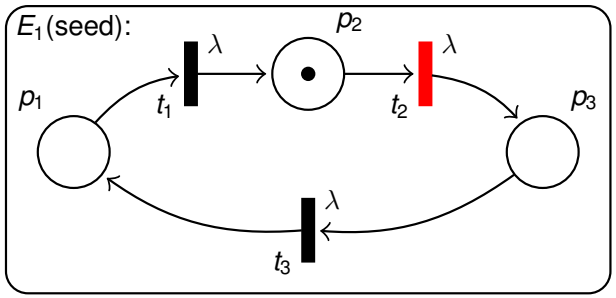


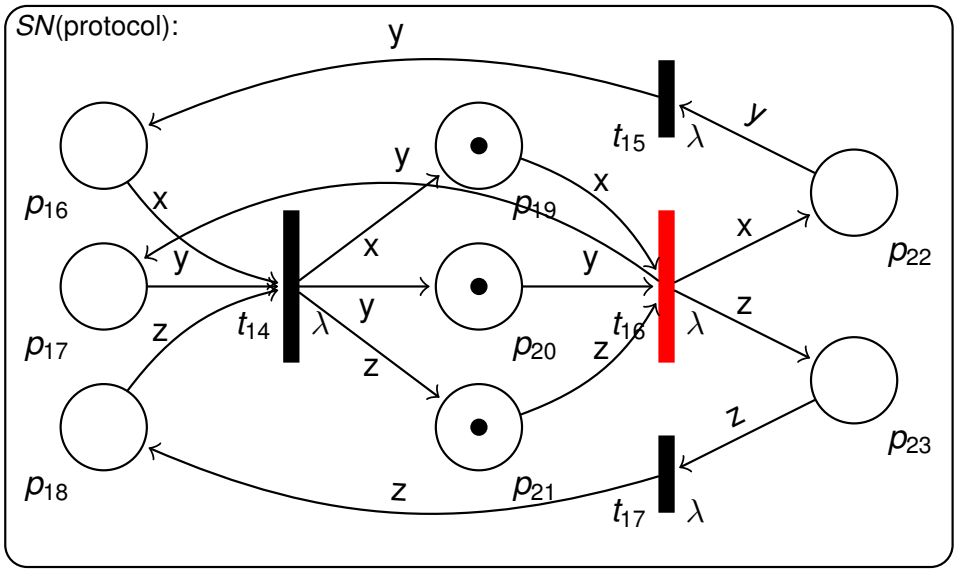


NP-net

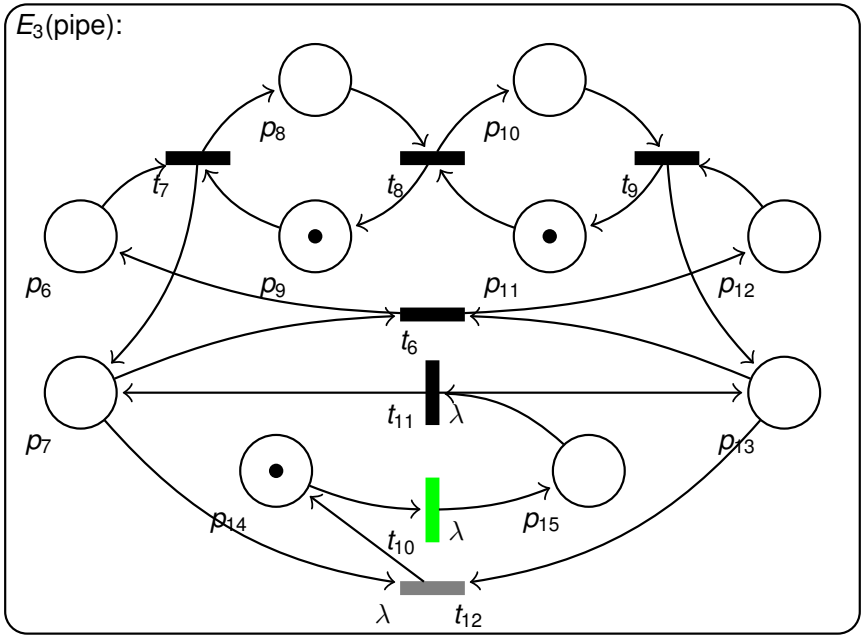


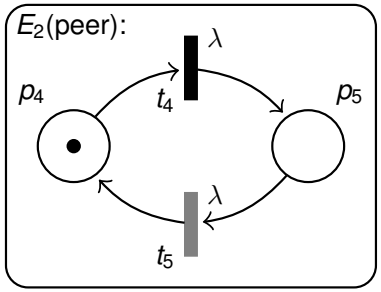


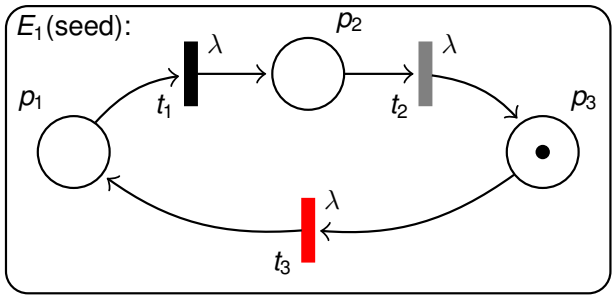


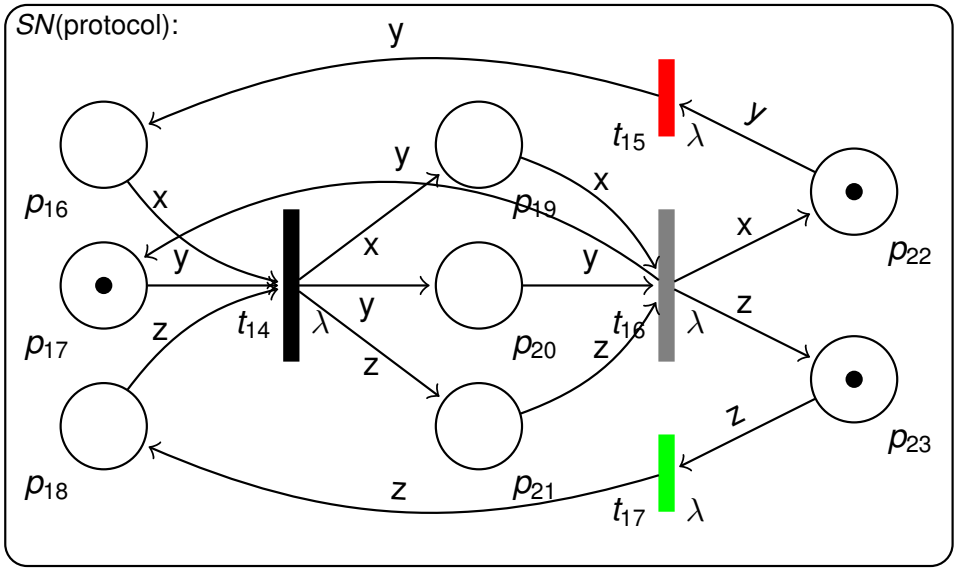


NP-net

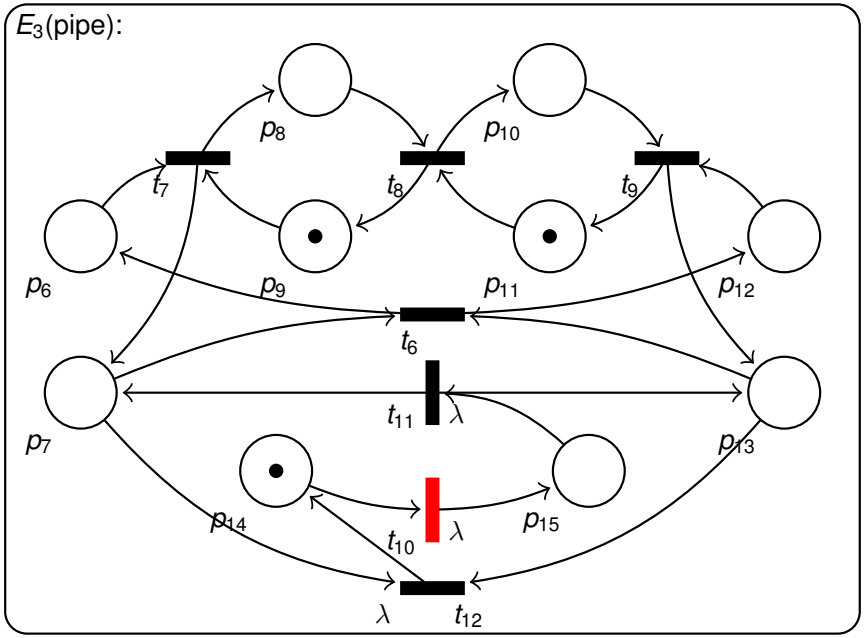


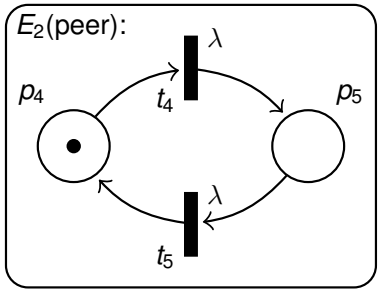


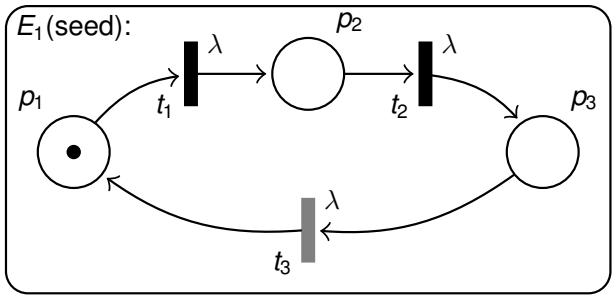


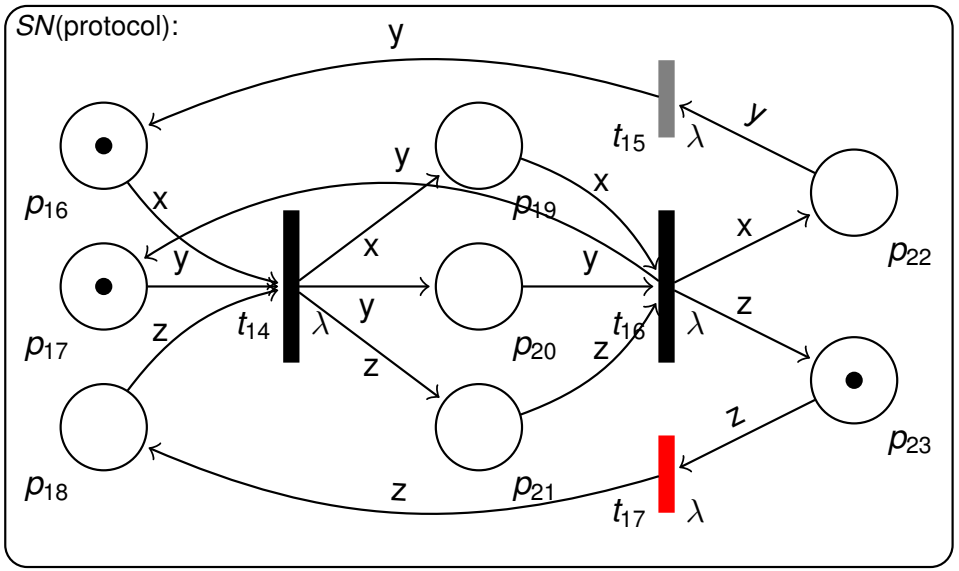


NP-net

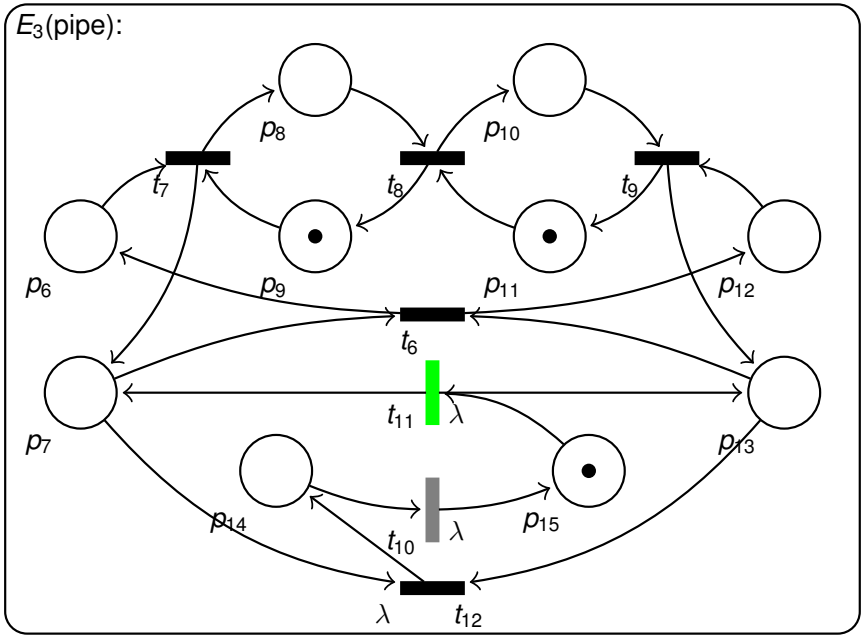


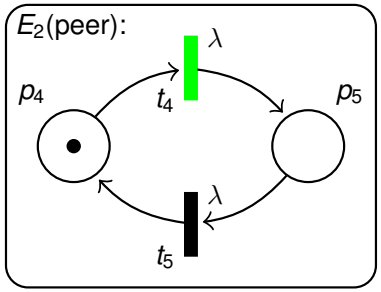


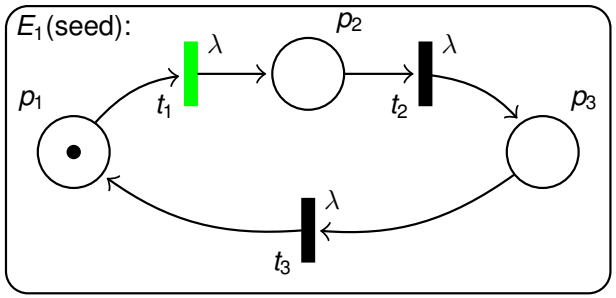


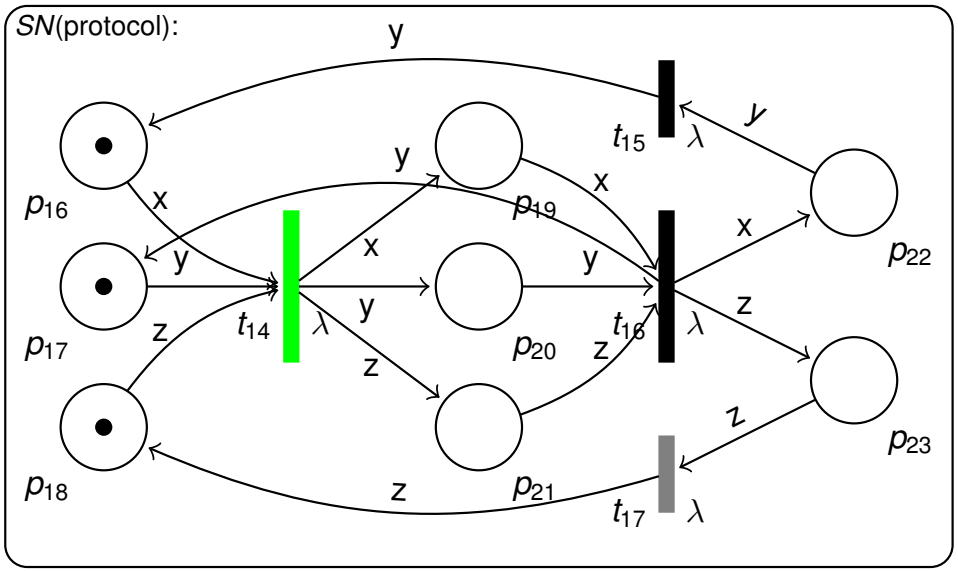


NP-net

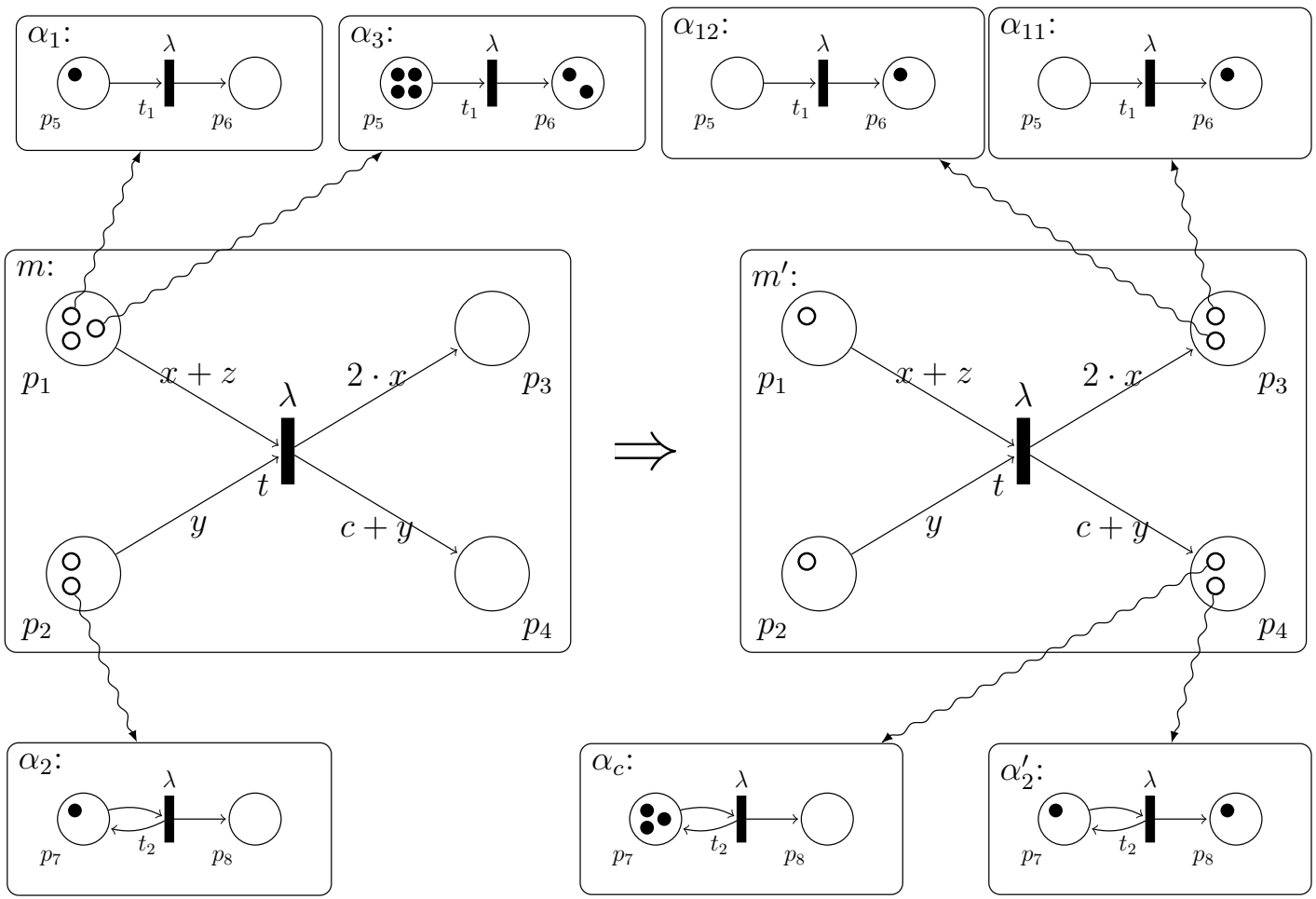






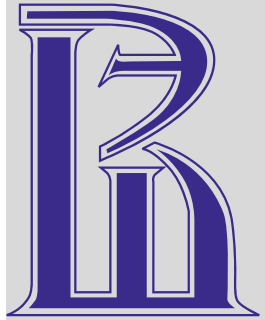


NP-net: net constants, net token elimination



Whats the Plan?

- Prerequisites of NP-nets
- Ostensive definition of NP-nets: Simple model of P2P protocol
- **Formal definition of NP-nets**
- Need for analysis methods
- Compositionality of NP-net properties
- Technical achievements
- Ongoing research
- Conclusions and further directions

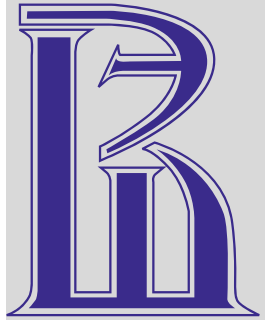


NP-net

Definition

Two-level NP-net is a tuple
$NP = \langle Lab, SN, \mathbb{E} \rangle$ where

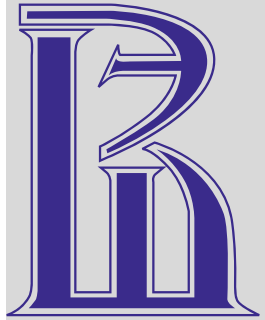
- Lab — set of labels for synchronization of transitions;
- SN — system net of the NP-net;
- $\mathbb{E}$ – finite set of element nets.



Definition

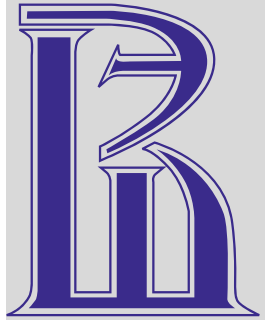
System net is a tuple
$SN = \langle N, \mathcal{L}, \mathcal{U}, W, m_0 \rangle$ where

- $N = \langle P_{SN}, T_{SN}, F_{SN} \rangle$ — high-level PN of System net;
- P_{SN} — set of typed places;
- $\mathcal{L} = Expr$ — arc expression language (sums of constants and variables);
- $\mathcal{U} = \langle A, \mathcal{I} \rangle$ — model of $\mathcal{L}$ with domain $A = A_{net} \cup A_{atom}$;
- A_{net} – set of marked element nets (net tokens);
- A_{atom} — set of plain colored tokens;
- $\mathcal{I} : Con \to A$ — interpretation function;
- $W : F_{SN} \to \mathcal{L}$ — arc expression function;
- $\Lambda : T_{SN} \to Lab$ — transition labeling function.



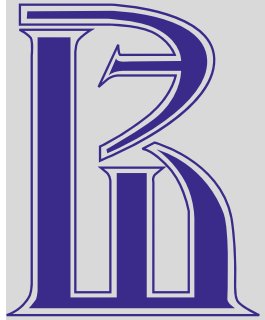
Restrictions

1. constants or multiple instances of the same variable are not allowed in input arc expressions of t;
2. each variable in an output arc expression for t occurs in one of the input arc expressions of t.



Whats the Plan?

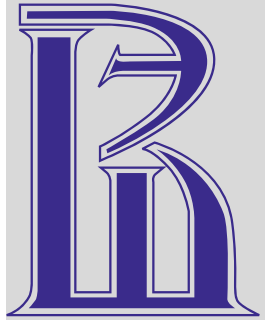
- Prerequisites of NP-nets
- Ostensive definition of NP-nets: Simple model of P2P protocol
- Formal definition of NP-nets
- **Need for analysis methods**
- Compositionality of NP-net properties
- Technical achievements
- Ongoing research
- Conclusions and further directions



Need for analysis methods

Common sense:

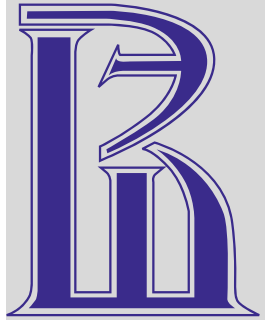
- Vertical synchronization, additional levels, net tokens creation and destruction, net tokens constants;
- Distributed concurrent systems are complex and non-tractable for human understanding;
- Distributed concurrent systems with dynamical hierarchical structure are even more complex;



Need for analysis methods

Theoretical issues:

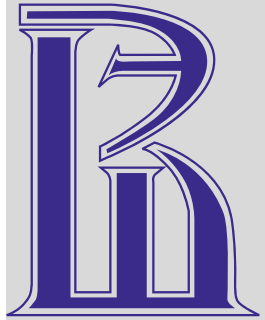
- Covering problem is decidable



Need for analysis methods

Theoretical issues:

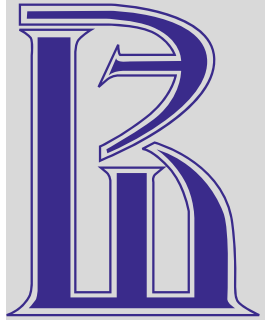
- Covering problem is decidable
- NP-nets strictly more expressive than Petri nets
- It is possible to model Petri nets w/ reset arcs via NP-nets so...



Need for analysis methods

Theoretical issues:

- Covering problem is decidable
- NP-nets strictly more expressive than Petri nets
- It is possible to model Petri nets w/ reset arcs via NP-nets so...
- Boundedness is undecidable
- Reachability is undecidable
- Liveness is undecidable

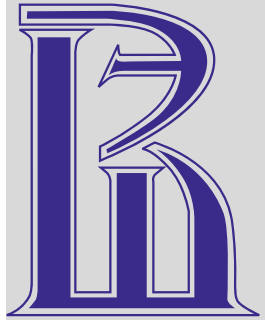


Need for analysis methods

Theoretical issues:

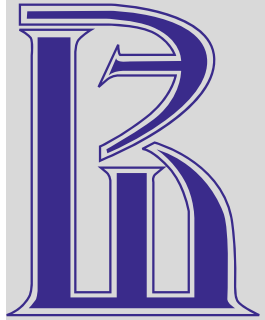
- Covering problem is decidable
- NP-nets strictly more expressive than Petri nets
- It is possible to model Petri nets w/ reset arcs via NP-nets so...
- Boundedness is undecidable
- Reachability is undecidable
- Liveness is undecidable

What can we do?



Whats the Plan?

- **Prerequisites of NP-nets**
- Ostensive definition of NP-nets: Simple model of P2P protocol
- Formal definition of NP-nets
- Need for analysis methods
- **Compositionality of NP-net properties**
- Technical achievements
- Ongoing research
- Conclusions and further directions



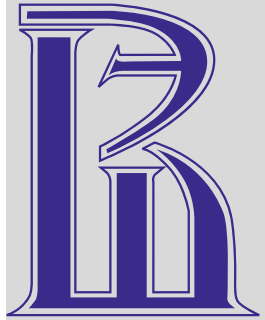
Compositionality and modularity

Modularity allows

- describe system's components independently.

Compositionality enables

- derive system properties from properties of its components;
- reuse components' analysis results for systems with same components (library of verified components);

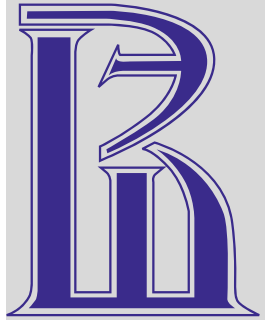


Compositionality and modularity

Prof. Joseph Sifakis, *A Dialogue with Professor Joseph Sifakis about Concurrent Systems Specification and Verification*, Bulletin of the European Association for Theoretical Computer Science, 104, 2011

We need theory, models and tools for cost-effectively building complex systems by assembling heterogeneous components

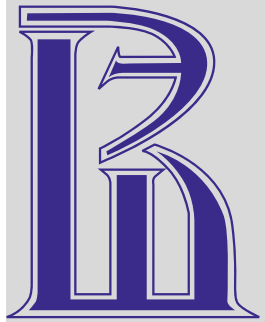
- *As system synthesis from requirements is intractable for complex systems, we should study principles for building correct systems from components. The aim is to avoid a posteriori monolithic verification as much as possible*;



Compositionality and modularity

Prof. Joseph Sifakis, *A Dialogue with Professor Joseph Sifakis about Concurrent Systems Specification and Verification* (contd.),

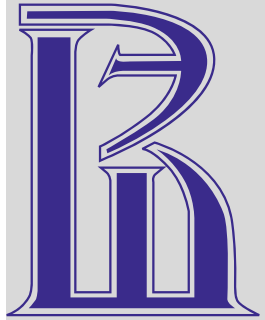
- *(Complexity problem) As system synthesis from requirements is intractable for complex systems, we should study principles for building correct systems from components. The aim is to avoid a posteriori monolithic verification as much as possible*;



Compositionality and modularity

Prof. Joseph Sifakis, *A Dialogue with Professor Joseph Sifakis about Concurrent Systems Specification and Verification* (contd.),

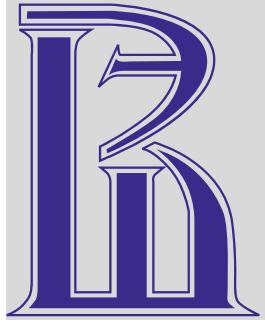
- *(Complexity problem) As system synthesis from requirements is intractable for complex systems, we should study principles for building correct systems from components. The aim is to avoid a posteriori monolithic verification as much as possible*;
- *(Constructive correctness) How can the global properties of a composite system be effectively inferred from the properties of its constituents? This remains an old open problem that urgently needs answers*.



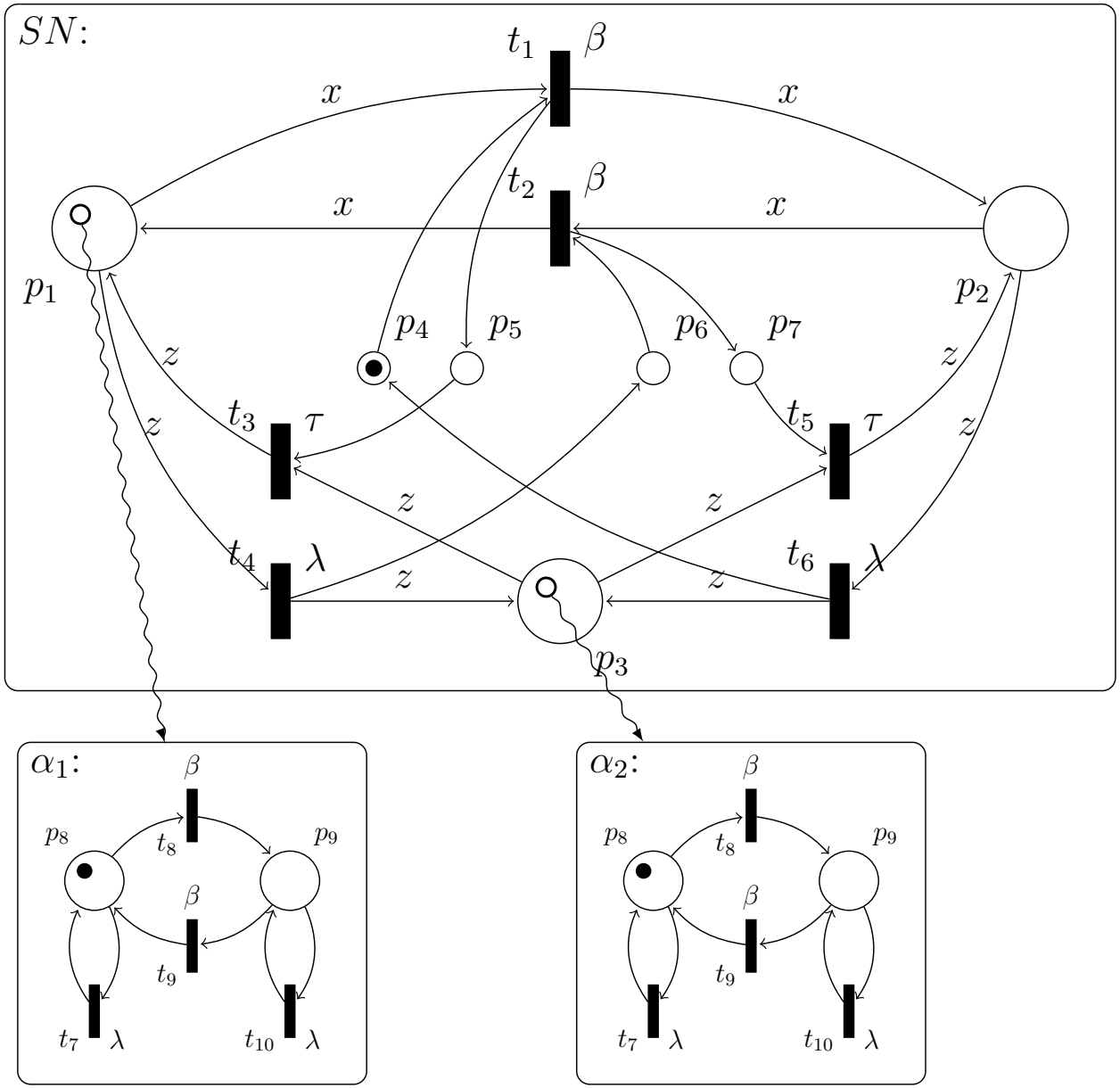
Compositionality and modularity

Prof. Joseph Sifakis, *A Dialogue with Professor Joseph Sifakis about Concurrent Systems Specification and Verification* (contd.),

- *(Complexity problem) As system synthesis from requirements is intractable for complex systems, we should study principles for building correct systems from components. The aim is to avoid a posteriori monolithic verification as much as possible*;
- *(Constructive correctness) How can the global properties of a composite system be effectively inferred from the properties of its constituents? This remains an old open problem that urgently needs answers*.
- *Compositionality rules for inferring global properties of composite components from the properties of constituent components e.g. the composition of deadlock-free components is - under some conditions - a deadlock-free component*;



Compositionality and modularity

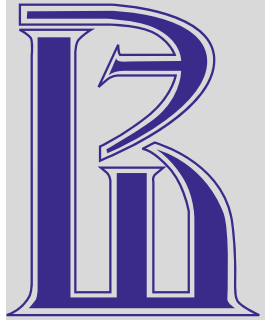


Compositionality and modularity

Compositionality rules for inferring global properties of composite components from the properties of constituent components e.g. the composition of deadlock-free components is - under some conditions - a deadlock-free component.

For a given NP-net *N*:

- The element nets of *N* are classical Petri nets;

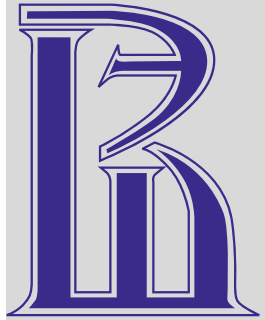


Compositionality and modularity

Compositionality rules for inferring global properties of composite components from the properties of constituent components e.g. the composition of deadlock-free components is - under some conditions - a deadlock-free component.

For a given NP-net *N*:

- The element nets of *N* are classical Petri nets;
- The net-tokens and net constants of *N* are instances of the element nets;

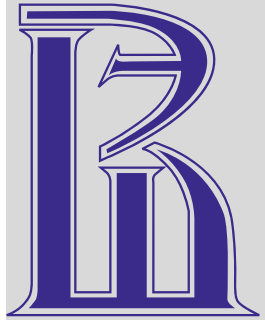


Compositionality and modularity

Compositionality rules for inferring global properties of composite components from the properties of constituent components e.g. the composition of deadlock-free components is - under some conditions - a deadlock-free component.

For a given NP-net *N*:

- The element nets of *N* are classical Petri nets;
- The net-tokens and net constants of *N* are instances of the element nets;
- The system net of *N* can be considered as classical PN, if we abstract from internal contents of net-tokens;

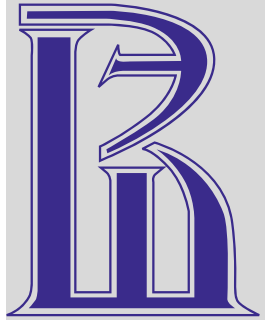


Compositionality and modularity

Compositionality rules for inferring global properties of composite components from the properties of constituent components e.g. the composition of deadlock-free components is - under some conditions - a deadlock-free component.

For a given NP-net *N*:

- The element nets of *N* are classical Petri nets;
- The net-tokens and net constants of *N* are instances of the element nets;
- The system net of *N* can be considered as classical PN, if we abstract from internal contents of net-tokens;
- The whole NP-net *N* can be considered as the composition of the system net and the element nets.

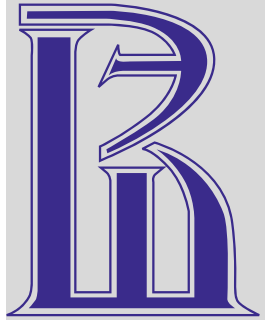


Compositionality and modularity

Compositionality rules for inferring global properties of composite components from the properties of constituent components e.g. the composition of deadlock-free components is - under some conditions - a deadlock-free component.

For a given NP-net *N*:

- The components of *N* satisfy some property;

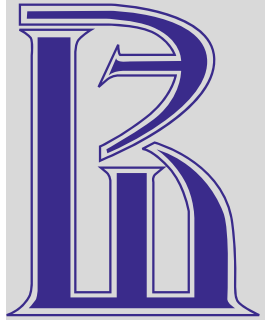


Compositionality and modularity

Compositionality rules for inferring global properties of composite components from the properties of constituent components e.g. the composition of deadlock-free components is - under some conditions - a deadlock-free component.

For a given NP-net *N*:

- The components of *N* satisfy some property;
- Some structural/behavioural conditions are satisfied

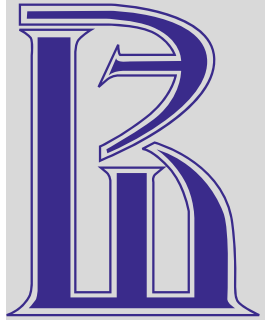


Compositionality and modularity

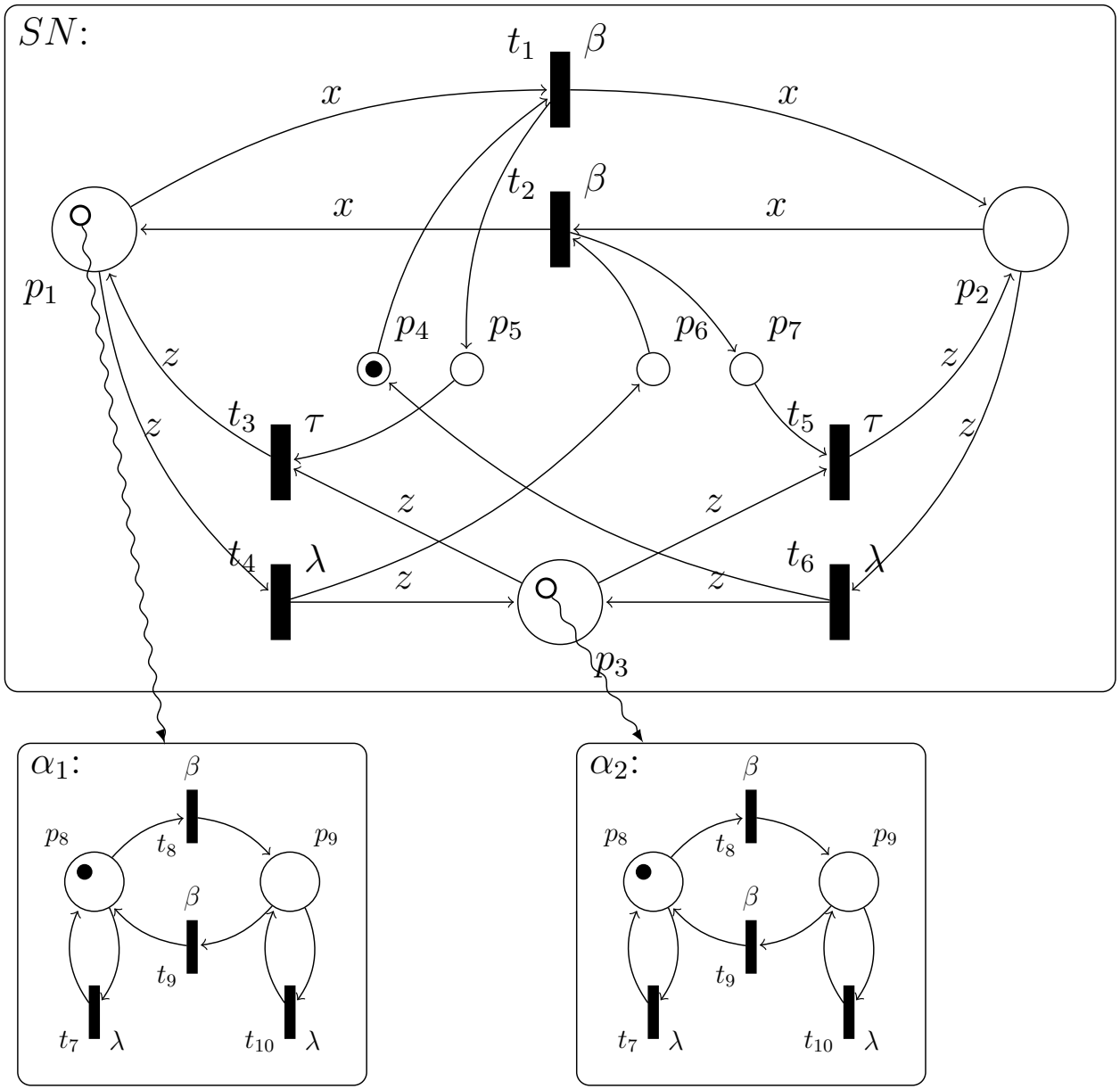
Compositionality rules for inferring global properties of composite components from the properties of constituent components e.g. the composition of deadlock-free components is - under some conditions - a deadlock-free component.

For a given NP-net *N*:

- The components of *N* satisfy some property;
- Some structural/behavioural conditions are satisfied
- The whole NP-net *N* satisfies some property.



Compositionality and modularity



Boundedness compositionality

Definition

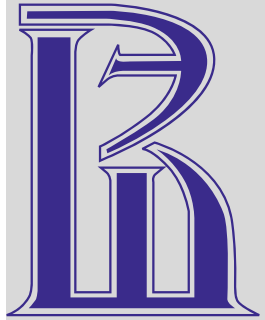
Marked NP-net *NP* is *bounded* iff its reachability set $R_{NP}(m_0)$ is finite.

Theorem

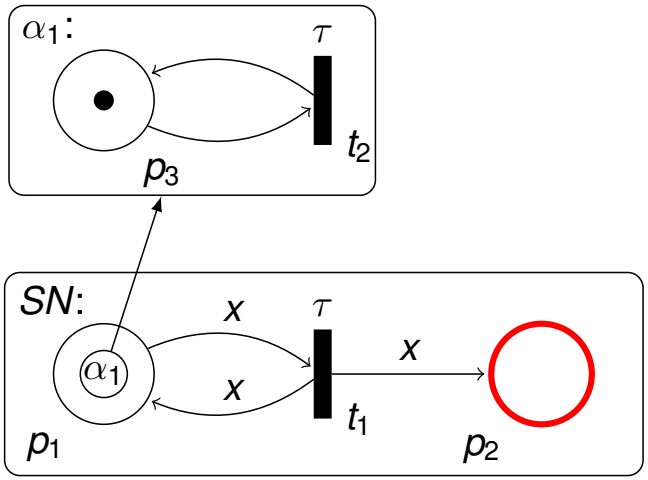
Let NP be a marked NP-net. If

1. *the system net in NP is bounded (as a separate component, i.e. a flat PN)*
2. *all net tokens in the initial marking are bounded (as separate components)*
3. *all net constants in arc expressions in NP are bounded (as separate components)*

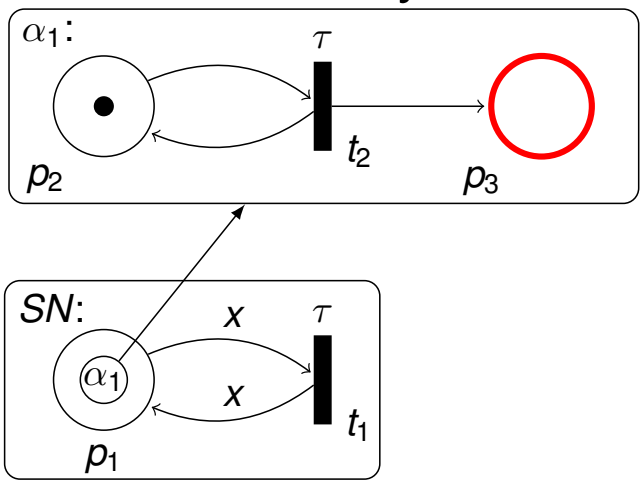
then NP is bounded.



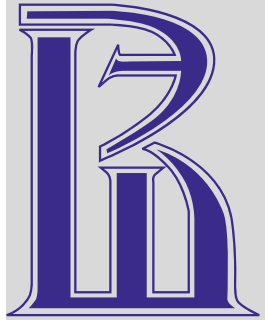
Counterexamples for boundedness (1)



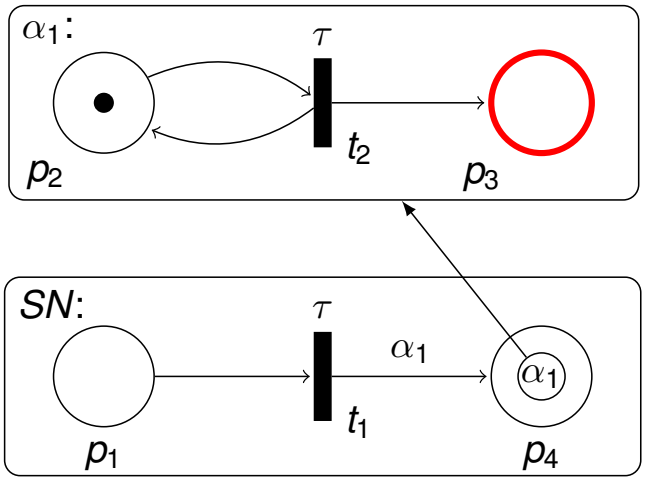
Condition: “the system net in *NP* is bounded”



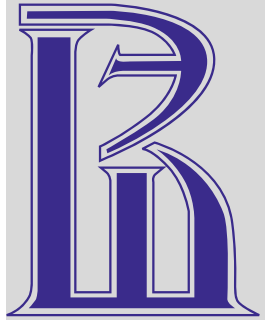
Condition: “all net tokens in the initial marking are bounded”



Counterexamples for boundedness (2)



Condition: all net constants in arc expressions are bounded”



Single label case: Liveness compositionality (0L-live)

Definition

Let *NP* be a marked two-level NP-net. *NP* is called 0L-live iff every transition in its system net is live.

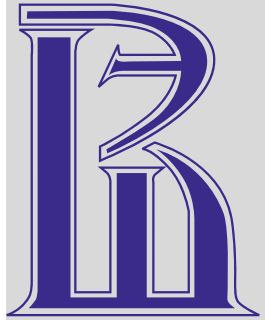
Theorem

Let NP be a marked two-level NP-net. Let also NP satisfy the following conditions:

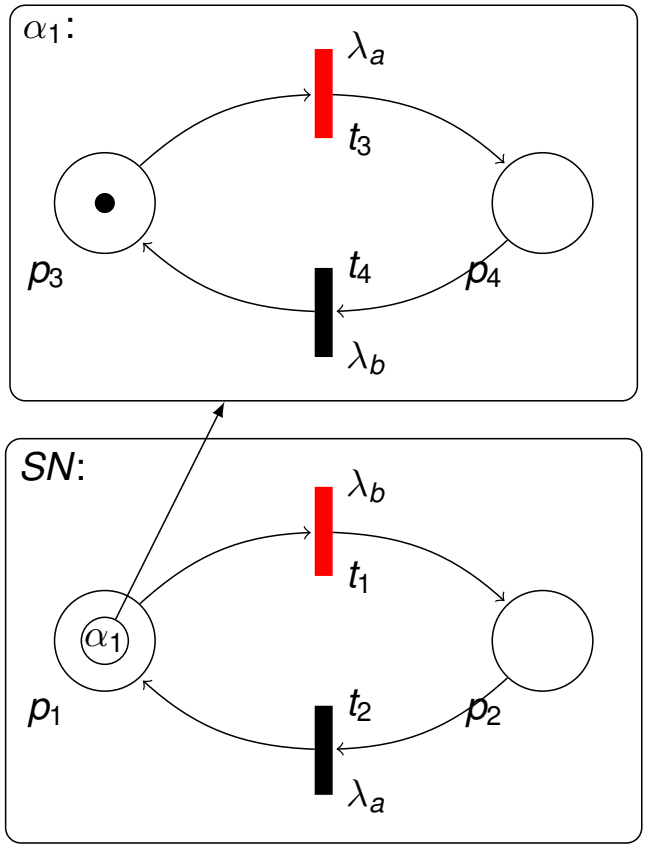
1. *the system net in NP is live (as a separate component, i.e. a flat PN);*
2. *all net tokens in the initial marking and all net constants in every arc expression are live (as separate components);*
3. *NP has only one label of vertical synchronization λ;*
4. *if t is a system net transition in NP labeled with λ, then for any $p \in {}^\bullet t$ the type of p is an element net, containing a transition labeled with λ.*

Then NP is 0L-live.

1,2 — counterexamples are similiar to boundedness property,
3,4 — counterexamples are given below

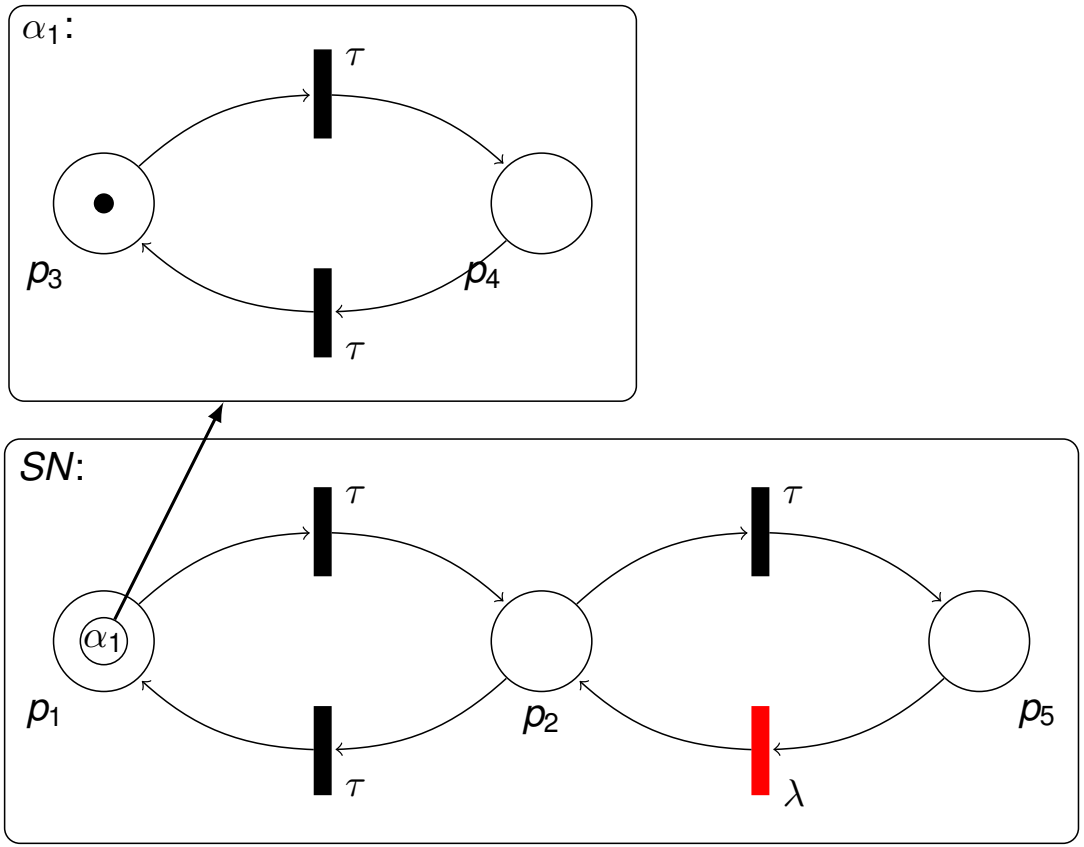


Counterexamples for 0L-liveness (1)



Condition: "*NP* has only one label of vertical synchronization λ".

Counterexamples for 0L-liveness (2)



Condition: “if t is a system net transition labeled with λ, then for any $p \in {}^\bullet t$ the type of p is an element net, containing a transition labeled with λ”.

Single label case: Liveness compositionality (1L-live) (1)

Definition

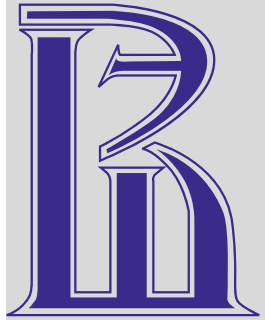
Let *NP* be a marked NP-net. *NP* is said to be 1L-live iff every transition of its system net and every transition in each net token from the initial marking in *NP* are live.

Theorem

Let NP be a marked NP-net.

1. *the system net in NP is live (as a separate component);*
2. *all net tokens in the initial marking and all net constants in every arc expression are live (as separate components);*
3. *NP has only one label for vertical synchronization λ;*
4. *if t is a system net transition in NP labeled by λ, then for any $p \in {}^\bullet t$ an element net corresponding to p contains a transition labeled with λ;*
5. *NP is conservative (on the next slide)*
6. *each SCC of the system net contains at least one transition labeled with λ.*

Then NP is 1L-live.

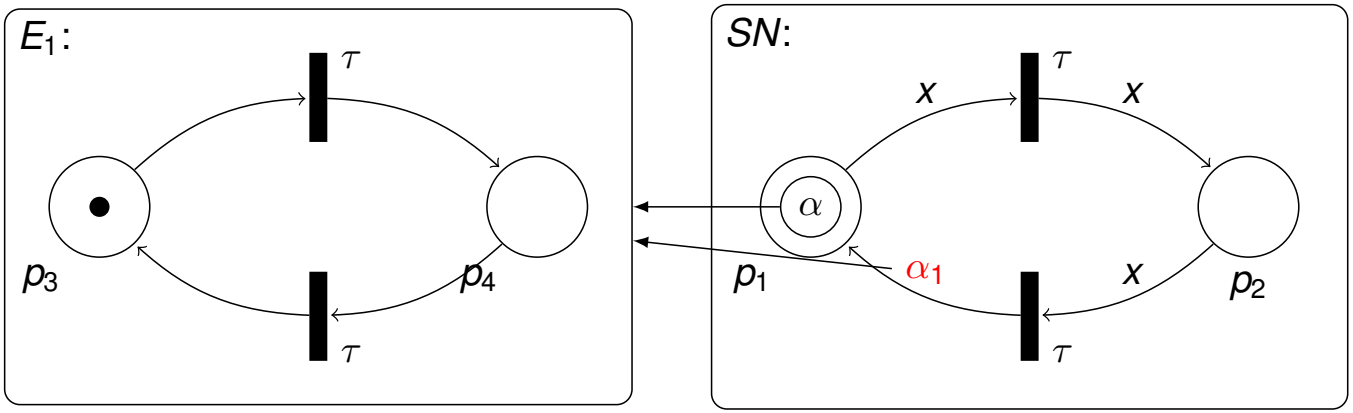


Conservative NP-nets

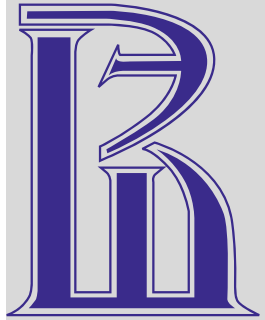
Definition

Let *NP* be an NP-net. *NP* is called conservative iff for each transition *t* in the system net of *NP* the set of all variables in the input arc expressions for *t* is a subset of all variables in its output arc expressions.

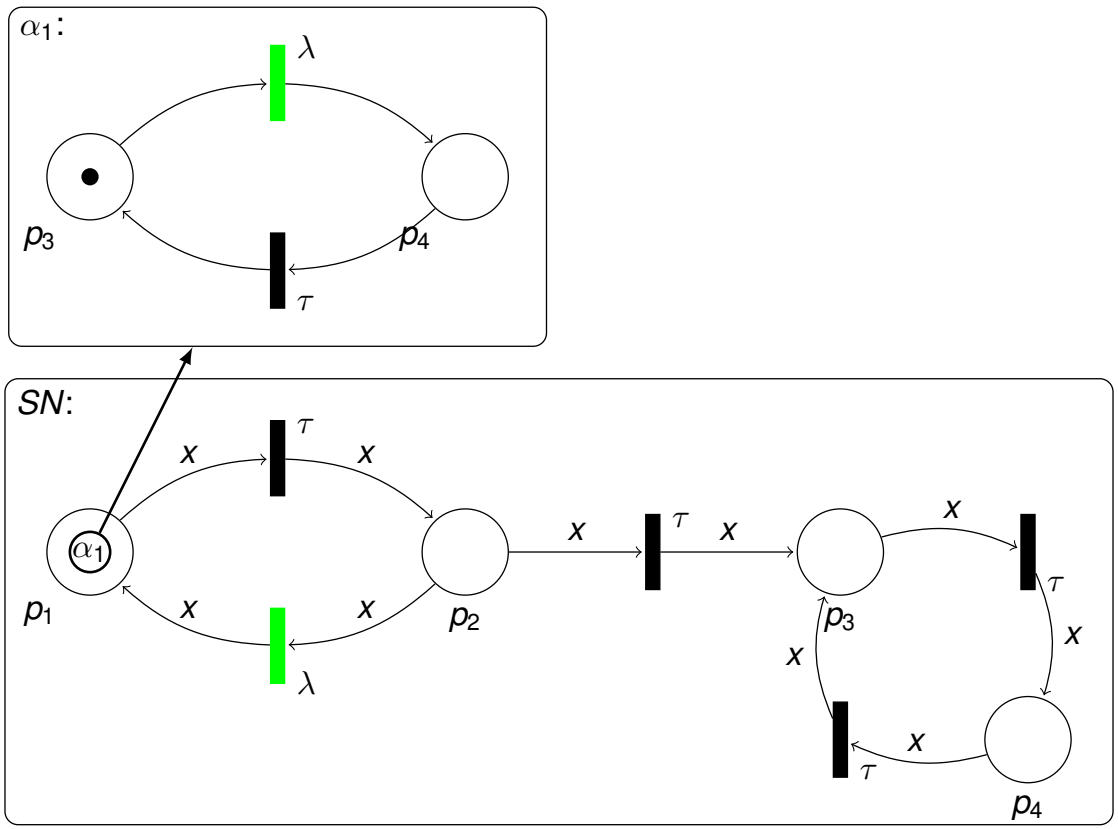
Conservativeness is needed as for 1L-liveness we consider liveness of net-tokens. In exiled net-tokens transitions are "dead":



However, if we don't bother about eliminated net-tokens, we can exclude the condition about convervativeness.

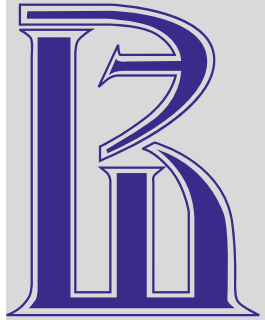


Counterexamples for 1L-liveness (1)



Condition: each SCC of the system net contains at least one transition labeled with λ.
Note: SCCs in SN are cycles in α-trail nets (defined below)

Multiple labels case: Liveness compositionality (1L-live) (1)



Theorem

Let NP be a conservative marked NP-net with a system net SN and initial marking m_0. Let also NP satisfy the following conditions:

1. *the system net in NP is live (as a separate component);*
2. *all net tokens in the initial marking m_0 and all net constants in all arc expression in SN are live (as separate components);*
3. *for each net token α in m_0: α is m-bisimilar to the α-trail net for NP;*
4. *for each arc (t, p) with arc expression e in SN: if e contains a net constant with a value α, then α is m-bisimular to the α-trail net for NP with the initial marking m_α, where $m_\alpha(p) = \alpha$, i.e. α resides in p.*

Then NP is 1L-live.

α-trail net extraction (1)

How to construct α-trail net for the position p_α of the system net *SN* of an NP-net:

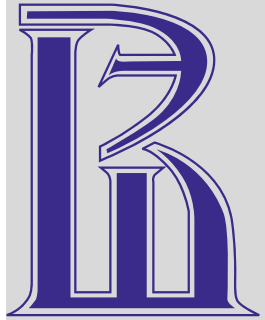
Definition

Step 0. We start building the α-trail net with a new place p', corresponding to p_α, and consider p_α as a current place in *SN*.

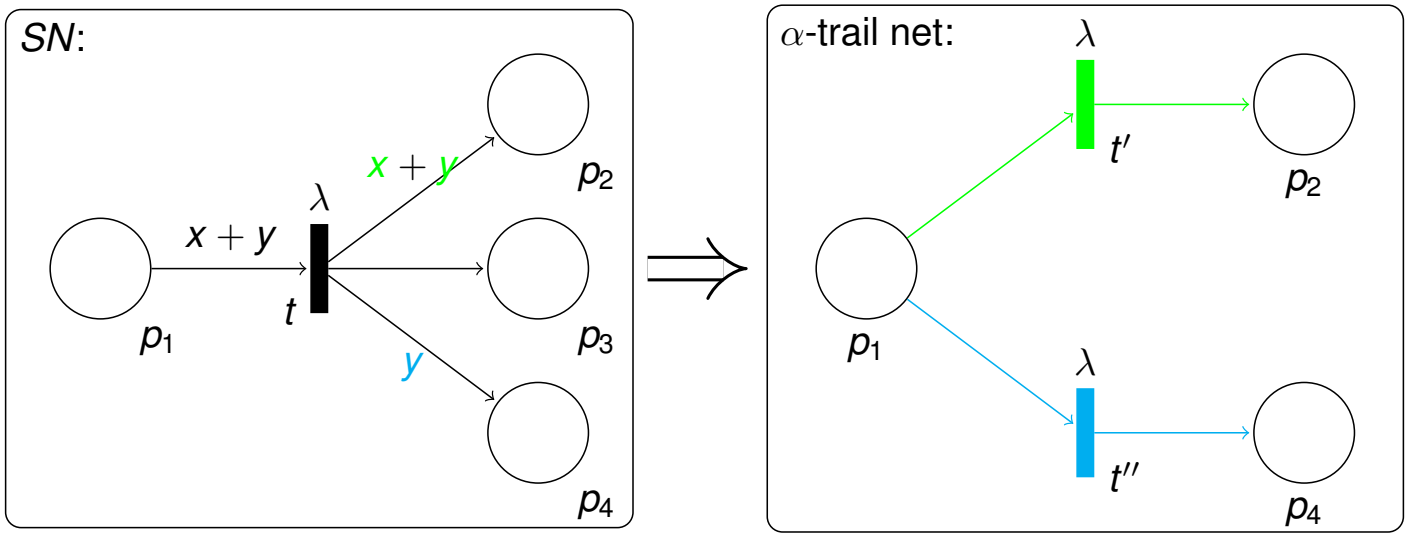
Step 1. Let p be a current place, and let t be a transition in *SN*, for which p is an income place, i.e. there is an arc from p to t with an arc expression *expr*. Let V be the set of variables occurring in *expr*. For each $v \in$ V, s.t. there is an arc from t to some place with an expression containing v, we build a new transition t' and a new place p': ${}^\bullet t' = p$, $t'^\bullet = p'$ and $\lambda(t') = \lambda(t)$.

Step 2. Repeat Step 1 with every new place as a current place.

Illustrations are on the next slide.

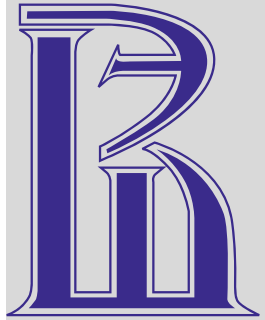


α-trail net extraction (2)

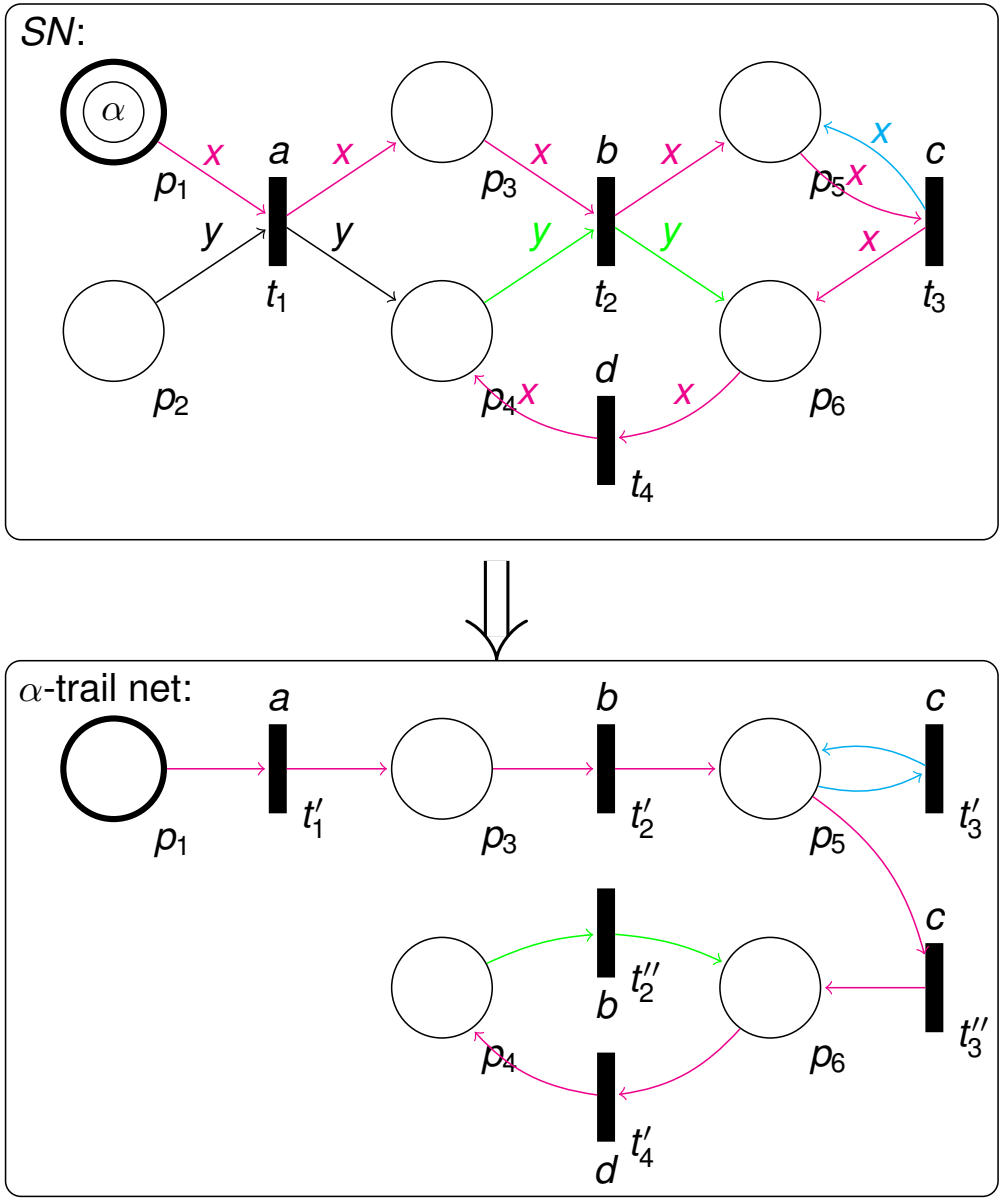


- path $p_1 \to t' \to p_2$ — is a trajectory of a net token under x or y variables from $x + y$ expression on the arc $\langle t, p_2 \rangle$.
- path $p_1 \to t'' \to p_4$ — is a trajectory of a net token under y variable from the arc $\langle t, p_4 \rangle$

No net token can traverse through the t-p_3 arc, so there is no corresponding arc in the constructed α-trail net.



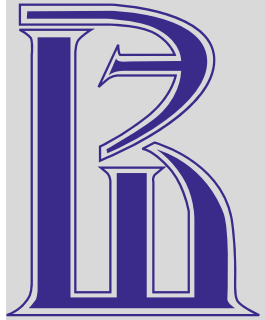
α-trail net extraction (3)



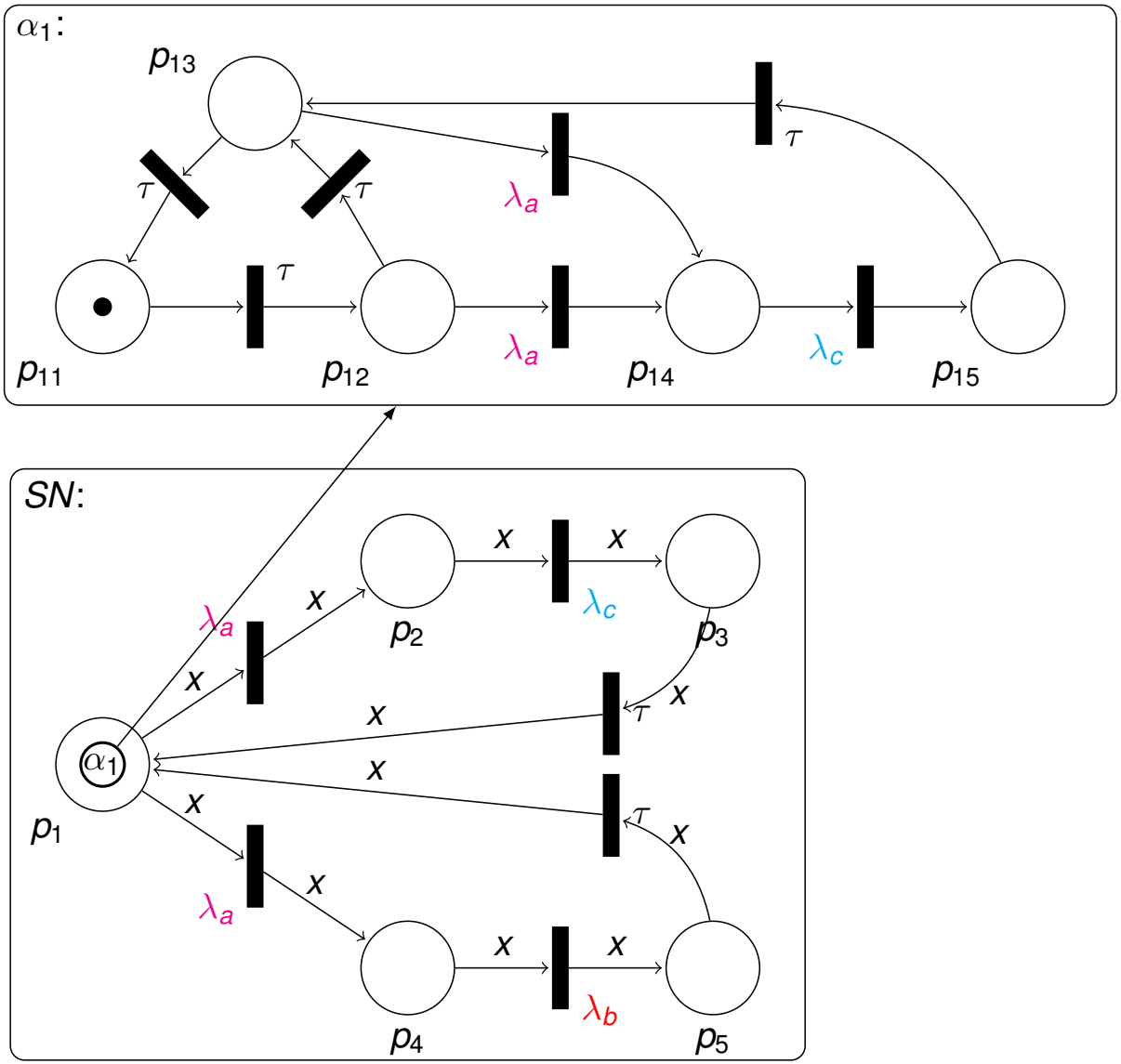
Here is a more complex case of α-trail net extraction.

α-trail net extraction (4)

α-trail net extraction is just a specialization of an extraction of NDA from high level Petri net with transitive inclusion of a token place propagation.



1L-liveness for several labels NP-nets



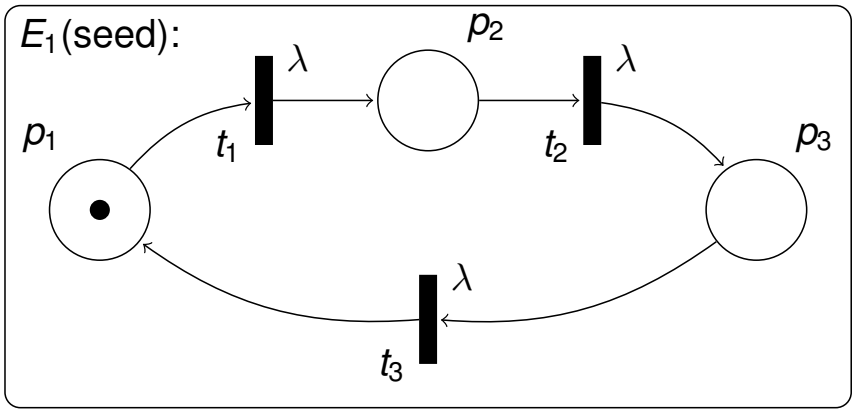
This NP-net has infinite executions, but it's not 1L-live, due to break of m-bisimulation.

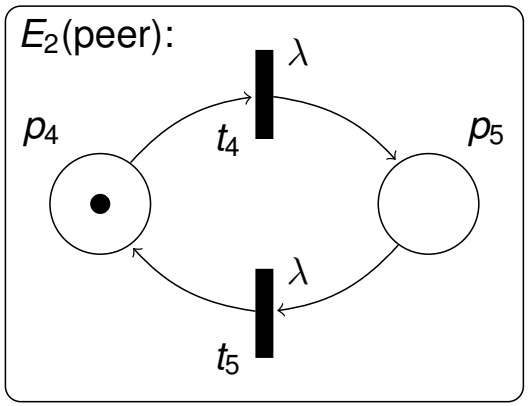
Properties of the components

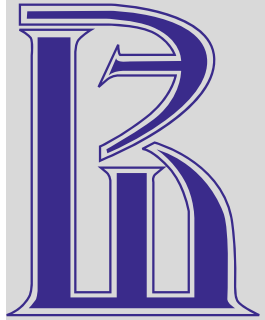
All components are

- free-choice (syntactically)
- bounded
- live

For Seed and Peer nets this is trivial

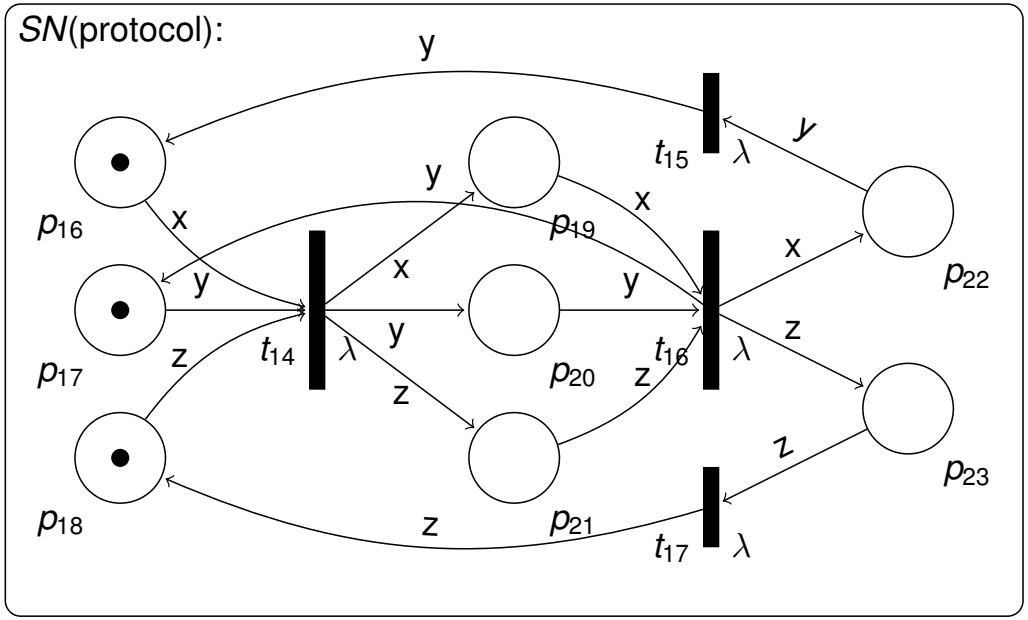




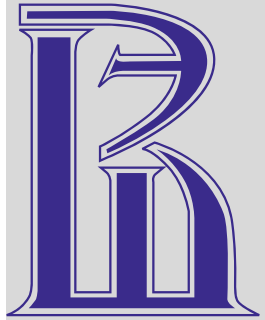


Properties of the components

For protocol net (System Net)

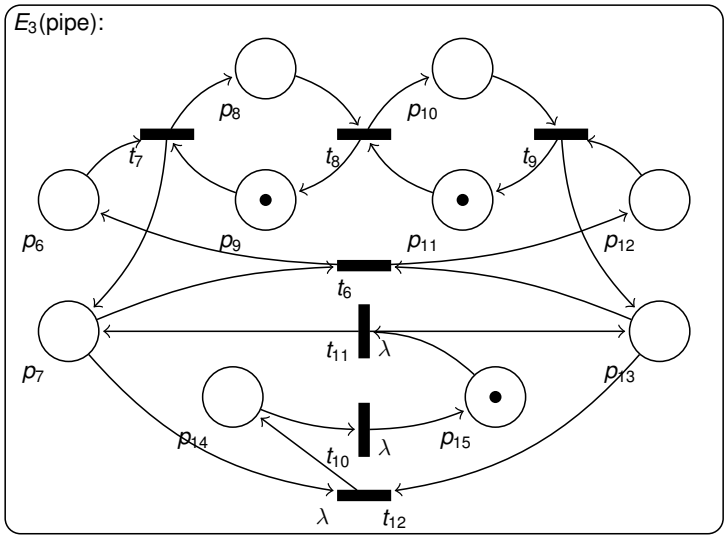


- Free-choiceness — syntactically
- Boundedness — as preset and postset powers are equal for each transition
- Liveness — by simple T-invariant $\langle 1, 1, 1, 1 \rangle$ (transitions are enumerated by their index)



Properties of the components

For Pipe net (E_3)



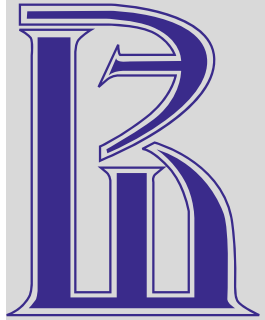
- Free-choice — syntactically
- Boundedness — by simple S-invariant
  $\langle 1, 1, 1, 1, 1, 1, 1, 1, 2, 2 \rangle$
  (places are enumerated by their index)
- Liveness — by simple T-invariant $\langle 1, 1, 1, 1, 1, 1, 1 \rangle$

Invariants are simple as we artificially constructed our example to be simple.

Properties of the system

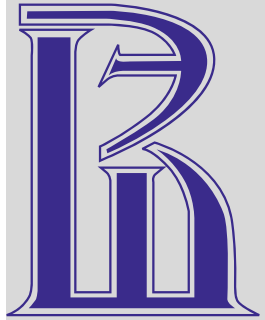
We can conclude the next properties of the whole system

- Boundedness — as “boundedness compositionality” conditions hold;
- “1L-live” property — as “1L-live” conditions hold.



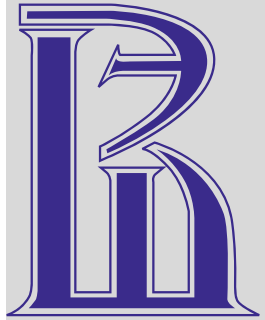
Whats the Plan?

- **Prerequisites of NP-nets**
- Ostensive definition of NP-nets: Simple model of P2P protocol
- Formal definition of NP-nets
- Need for analysis methods
- Compositionality of NP-net properties
- **Technical achievements**
- Conclusions and further directions



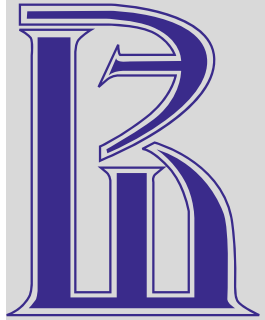
Technical achievements: NPN2CPN

- Idea: we don't have any tools at all, so why don't translate NP-nets to models with tool support



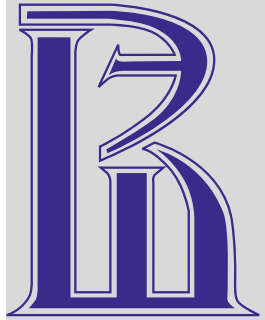
Technical achievements: NPN2CPN

- Idea: we don't have any tools at all, so why don't translate NP-nets to models with tool support
- What tool?



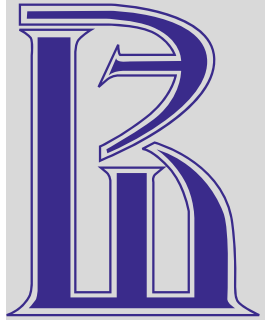
Technical achievements: NPN2CPN

- Idea: we don't have any tools at all, so why don't translate NP-nets to models with tool support
- What tool? - CPNtools



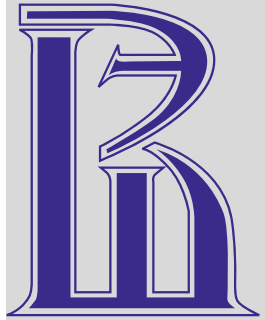
Technical achievements: NPN2CPN

- Idea: we don't have any tools at all, so why don't translate NP-nets to models with tool support
- What tool? - CPNtools
- Why?

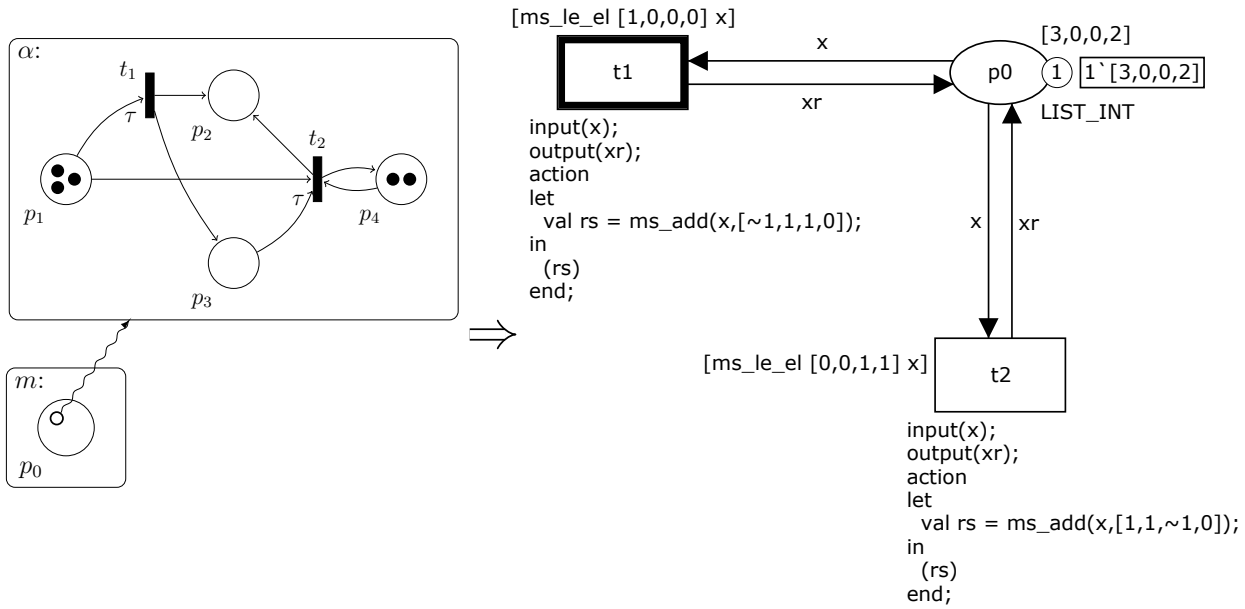


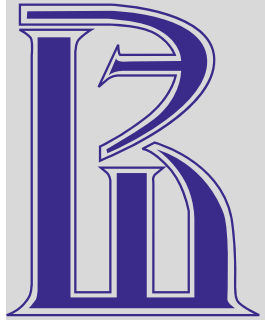
Technical achievements: NPN2CPN

- Idea: we don't have any tools at all, so why don't translate NP-nets to models with tool support
- What tool? - CPNtools
- Why? - Simulation, Performance analysis, Reachability graph, CTL model checking et cetera.
- Implementation of the translation: Hattu Pavel, Sysoev German.

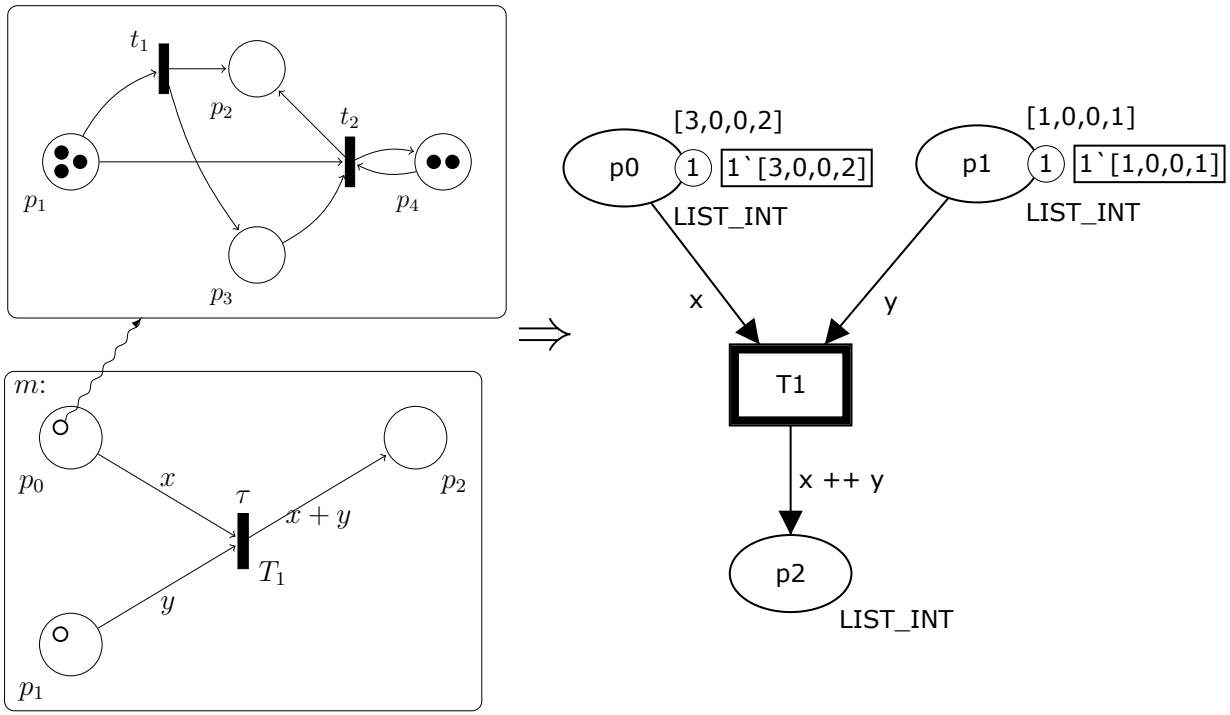


Technical achievements: NPN2CPN

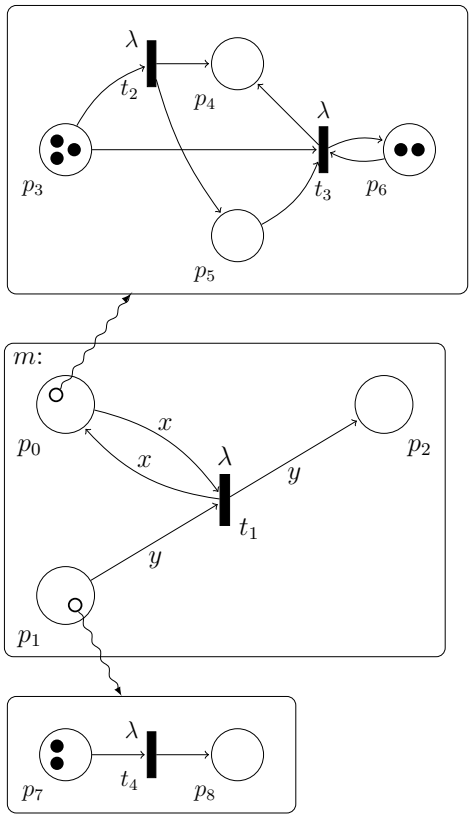


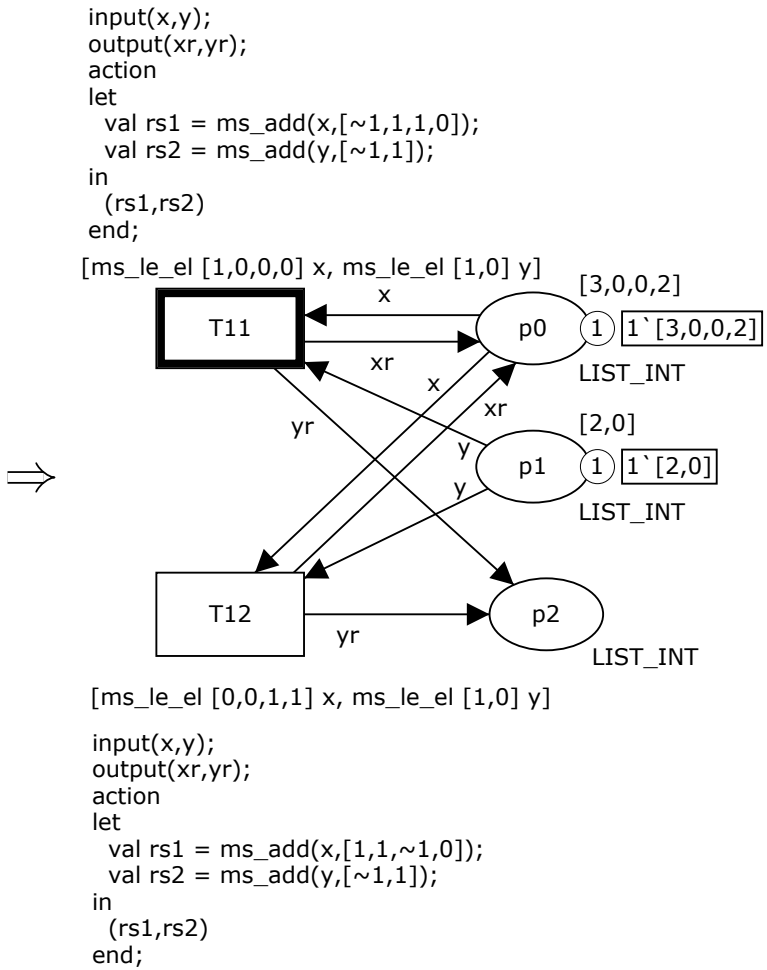


Technical achievements: NPN2CPN



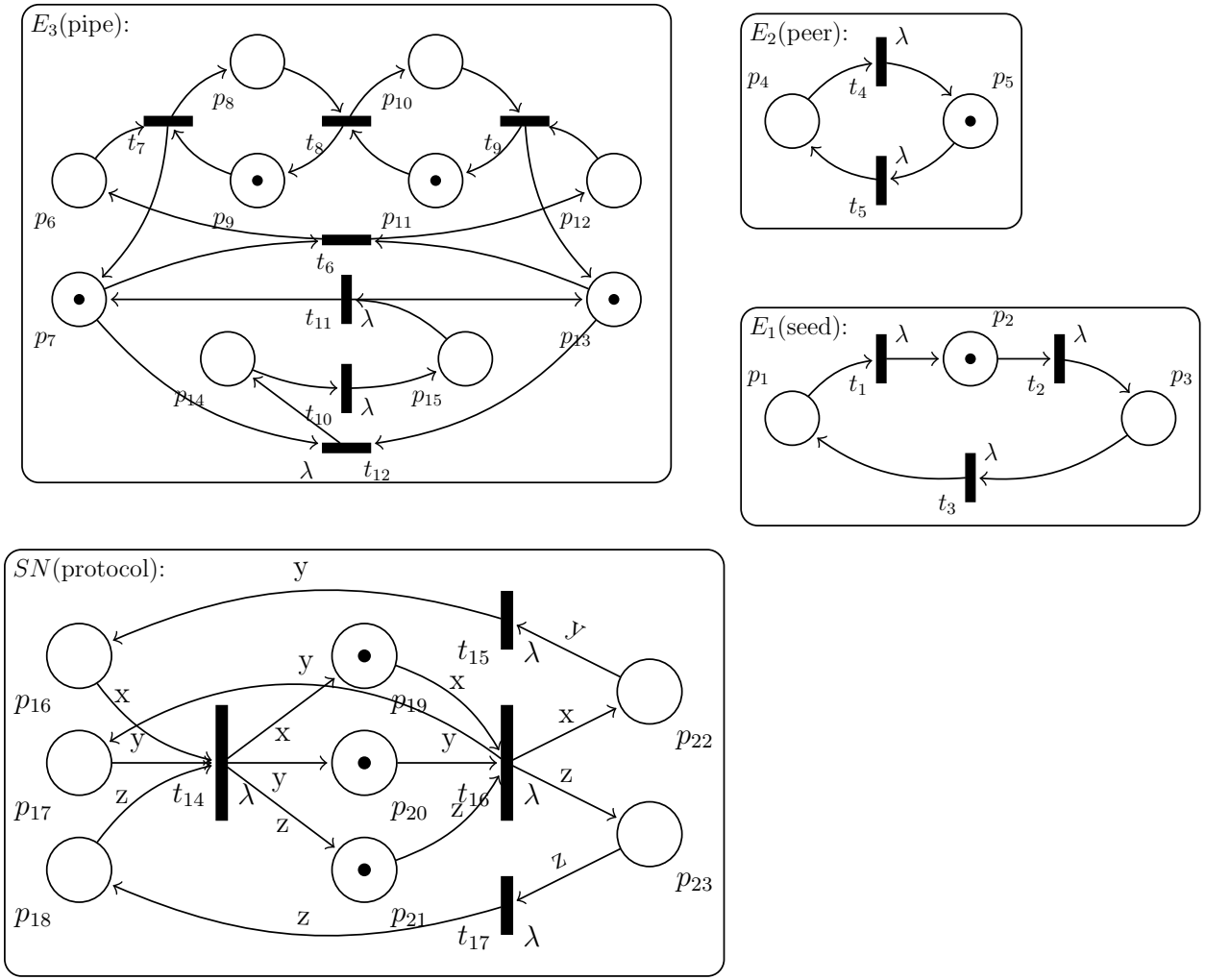
Technical achievements: NPN2CPN

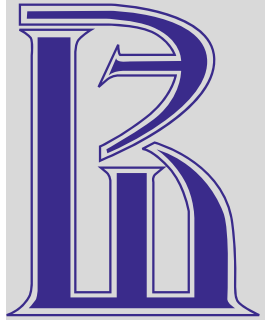




Technical achievements: NPN2CPN

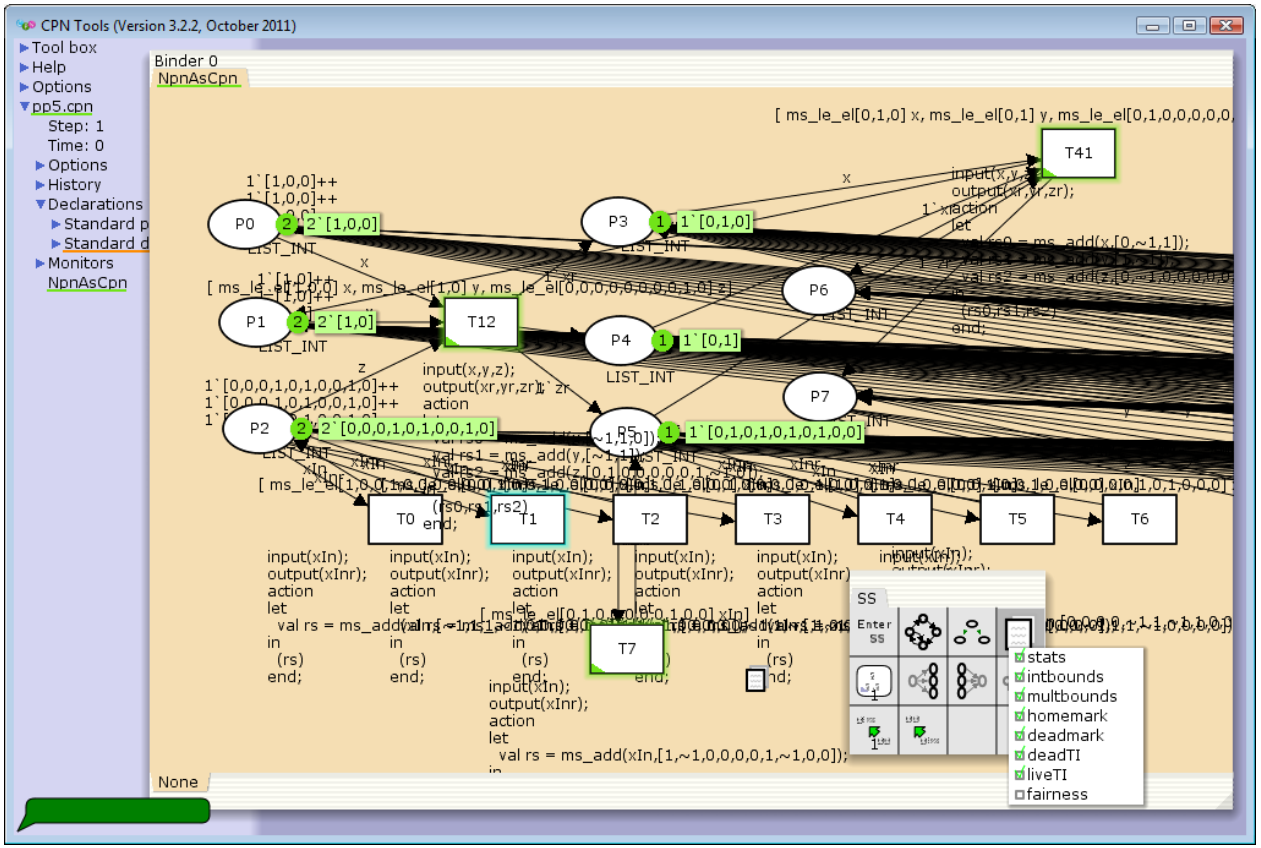
Source NP-net:

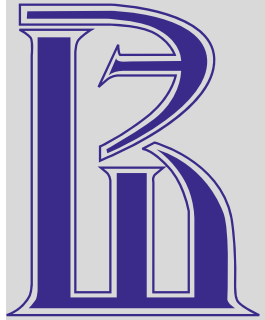




Technical achievements: NPN2CPN

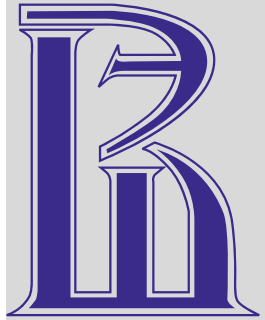
Resultant CP-net:





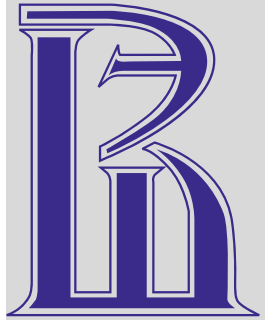
Technical achievements: EMF model

- It would be better to have our own tool;

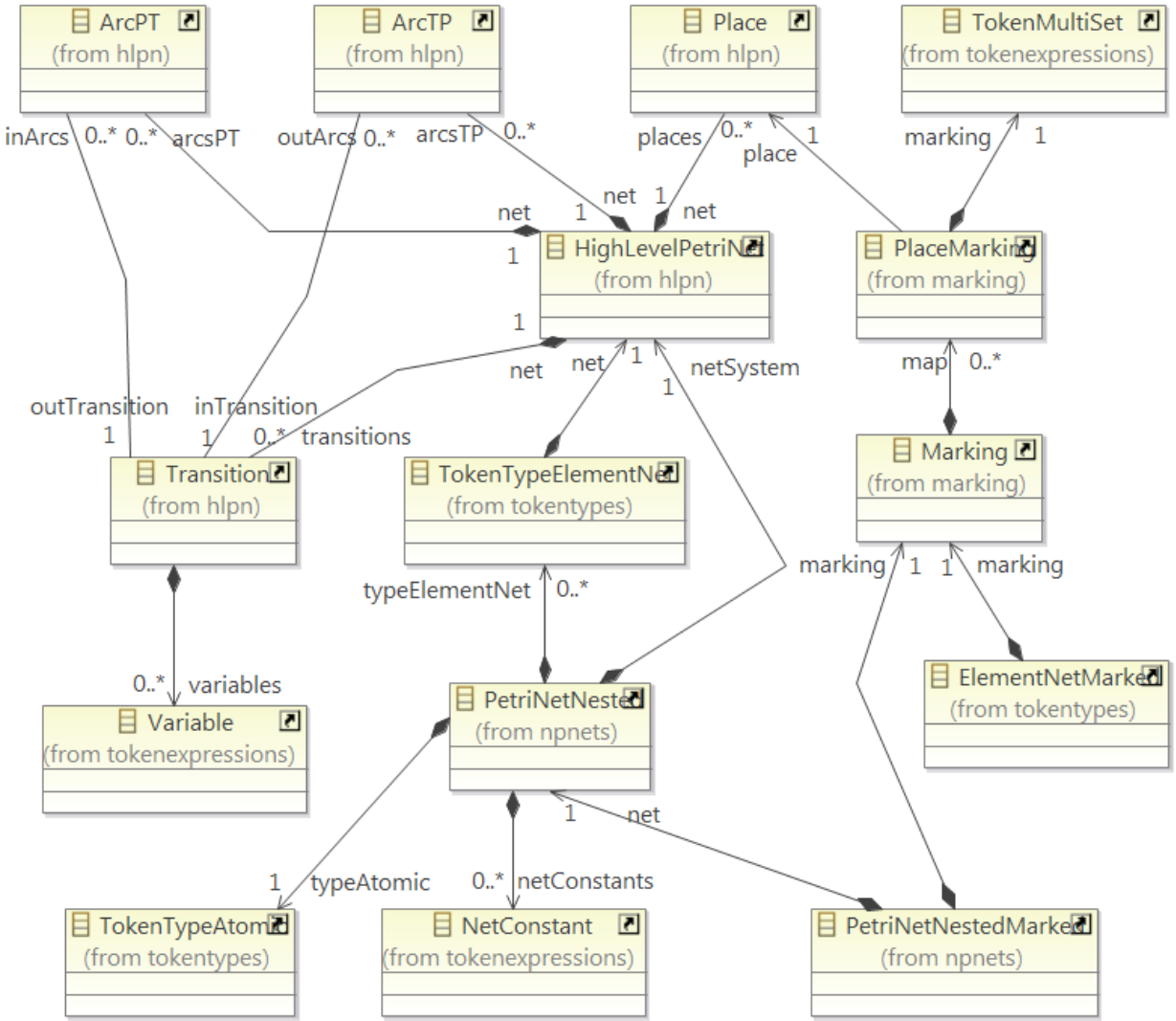


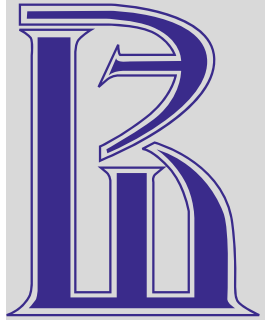
Technical achievements: EMF model

- It would be better to have our own tool;
- We have to start from a data model;



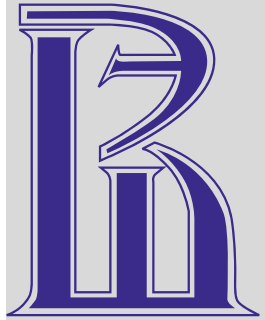
Technical achievements: EMF model





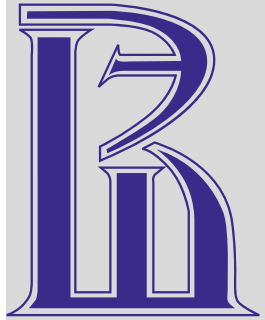
Technical achievements

- It would be better to have our own tool;
- We have to start from a data model;
- Haskell backend (by Daniil Frumin);
  1. Model checking of separate components;
  2. Compositional checking of NP-nets liveness (alpha-trail nets, m-bisimulation algorithm).

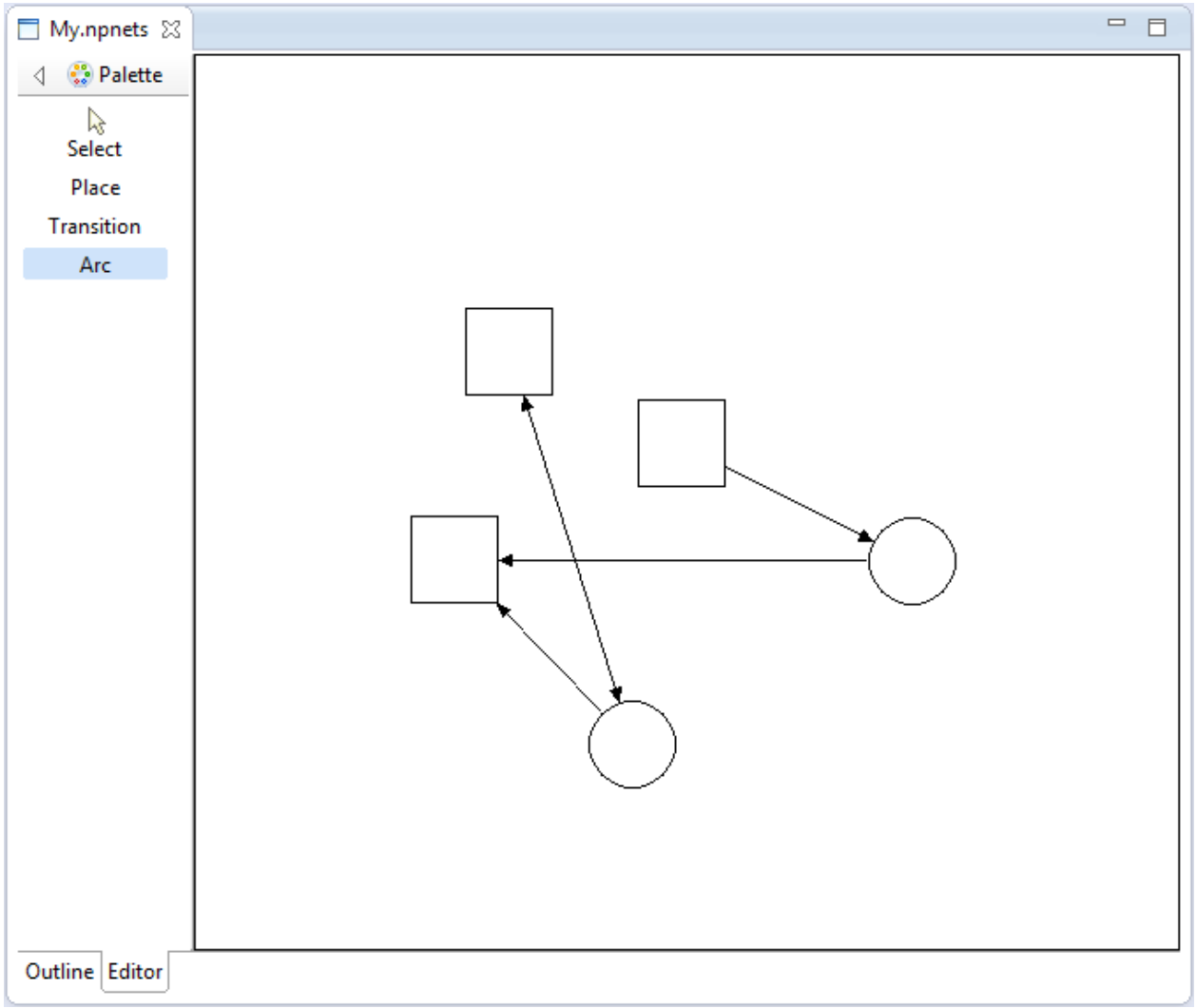


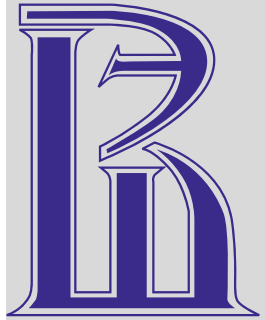
Technical achievements

- It would be better to have our own tool;
- We have to start from a data model;
- Haskell backend (by Daniil Frumin);
- Graphical editor (by Ilya Zubarev);



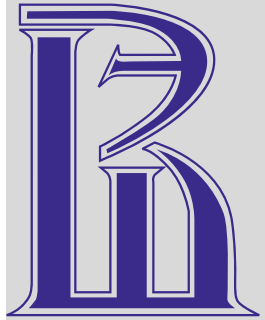
Technical achievements: Eclipse editor



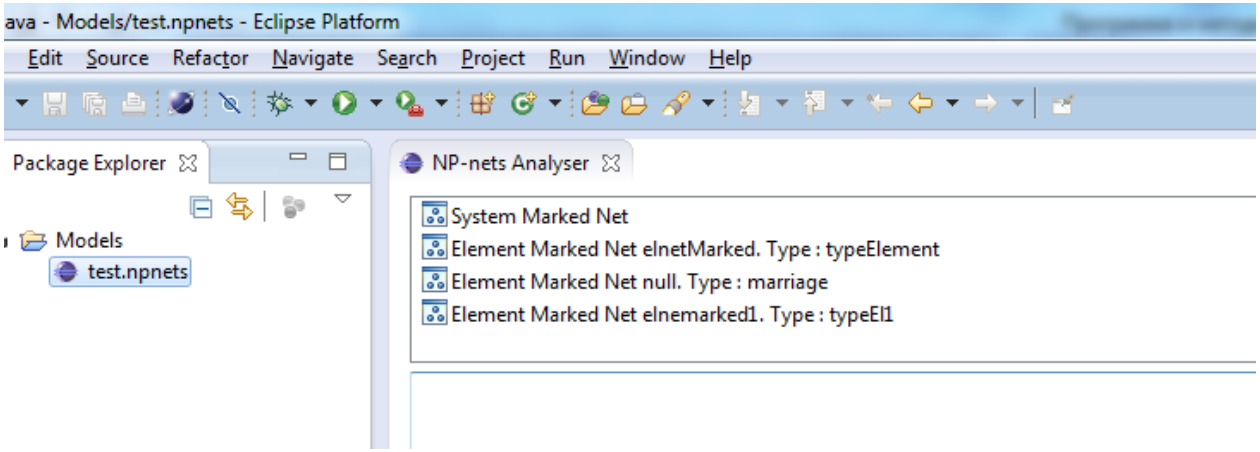


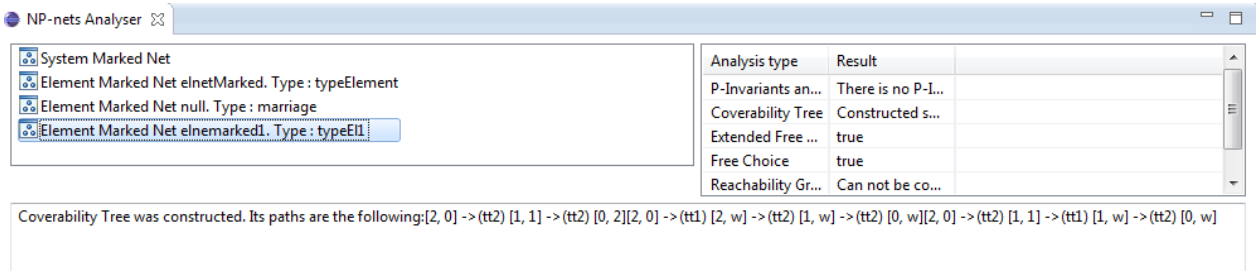
Technical achievements

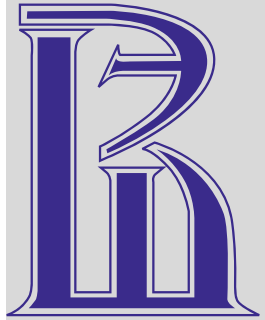
- It would be better to have our own tool;
- We have to start from a data model;
- Haskell backend (by Daniil Frumin);
- Graphical editor (by Ilya Zubarev);
- Analysis of components (by Anton Fillipov);
  - reachability graph, reachability tree, coverability tree, P-invariants, covering P-invariants, (extended) free-choiceness checking.



Technical achievements: EMF model

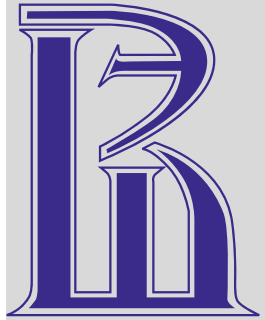






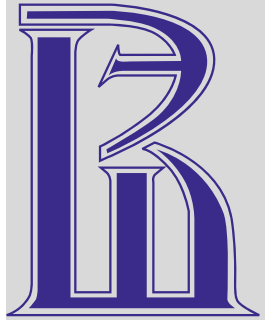
Technical achievements

- It would be better to have our own tool;
- We have to start from a data model;
- Haskell backend (by Daniil Frumin);
- Graphical editor (by Ilya Zubarev);
- Analysis of components (by Anton Fillipov);
- Code generation:
  - From NP-nets to Java apps over TCP/IP (by Dmitry Kuznecov)
  - From NP-nets to EJB system (by Nikolenko Artem)



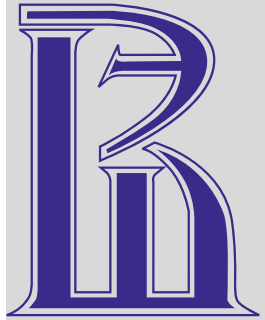
Whats the Plan?

- **Prerequisites of NP-nets**
- Ostensive definition of NP-nets: Simple model of P2P protocol
- Formal definition of NP-nets
- Need for analysis methods
- Compositionality of NP-net properties
- Technical achievements
- Ongoing research
- **Conclusions and further directions**



Ongoing research

1. Compositionality of P-invariants for NP-nets;
2. Compositionality of T-invariants for NP-nets;
3. Time NP-nets;
4. Compositional bounding of unbounded NP-nets by timing;



Conclusions and further directions

1. Nested Petri Nets with well-formed components can be analysed for boundedness and liveness in polynomial time;
2. Boundedness and liveness holds compositionality for NP-nets (under some constraints)
3. Compositionality of:
   - P- and T- invariants;
   - Fairness, impairness, justice.