MCC

A tool to remove colors from your High-Level Petri nets !

Silvano DAL ZILIO

LAAS-CNRS, Vertics team

presentation for our paper: *MCC: a Tool for Unfolding Colored Petri Nets in PNML Format* for the Petri Nets Conference—June 2020

TINA

MCC is not an acronym !

`mcc` is a tool designed for a very specific task:

> transform models of High-Level Petri nets
> (symmetric nets in PNML), into equivalent P/T nets

`mcc` has been developed and made available for the last 3 years and designed with the goal to be open, easily extensible, and good enough for the Model-Checking Contest

Why mcc ?

“There are only two hard things in Computer Science: cache invalidation and naming things.”
-- Phil Karlton

- short answer ≡ solve a problem we faced when entering the Model-Checking Contest three years ago.

- the initial goal ≡ develop a collection of helper apps to deal with colored models in PNML
  - unfolding
  - computing invariants on colored models
  - computing symmetries

why not compute directly on colored models ?

MCC

A tool to remove colors from your High-Level Petri nets !

quick tool demo: *mcc in 50”*

see also our 10’ tool demonstration vidéo for the Petri Nets Conference—June 2020

Installing MCC or building it from source

- MCC is a classic CLI tool

  Just install the right binary file in your PATH.

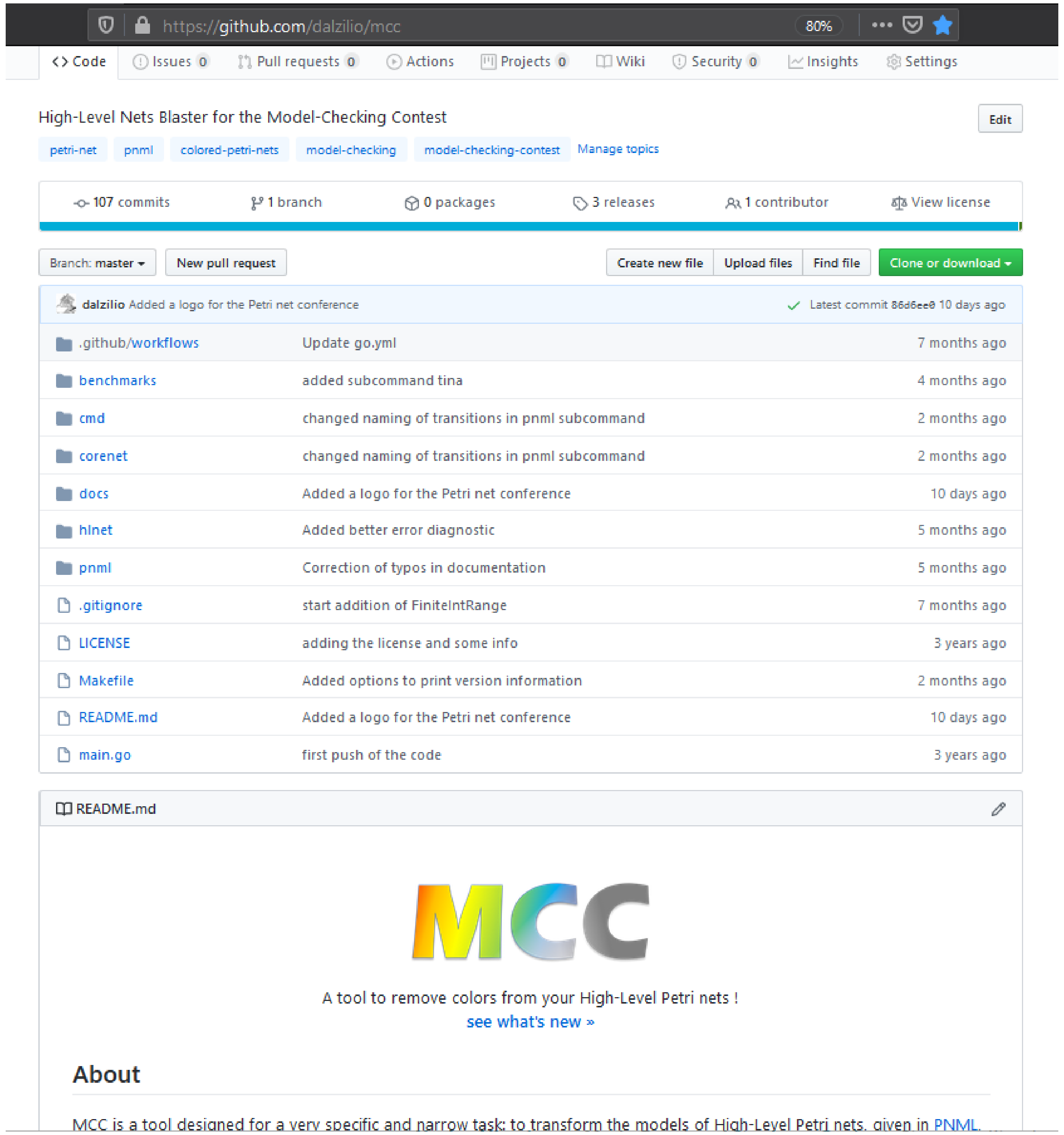
  Binary files for Windows, Linux and MacOS.

  See the latest releases on GitHub

- You have the option to install the tool from source, using a recent Go distribution. Just:

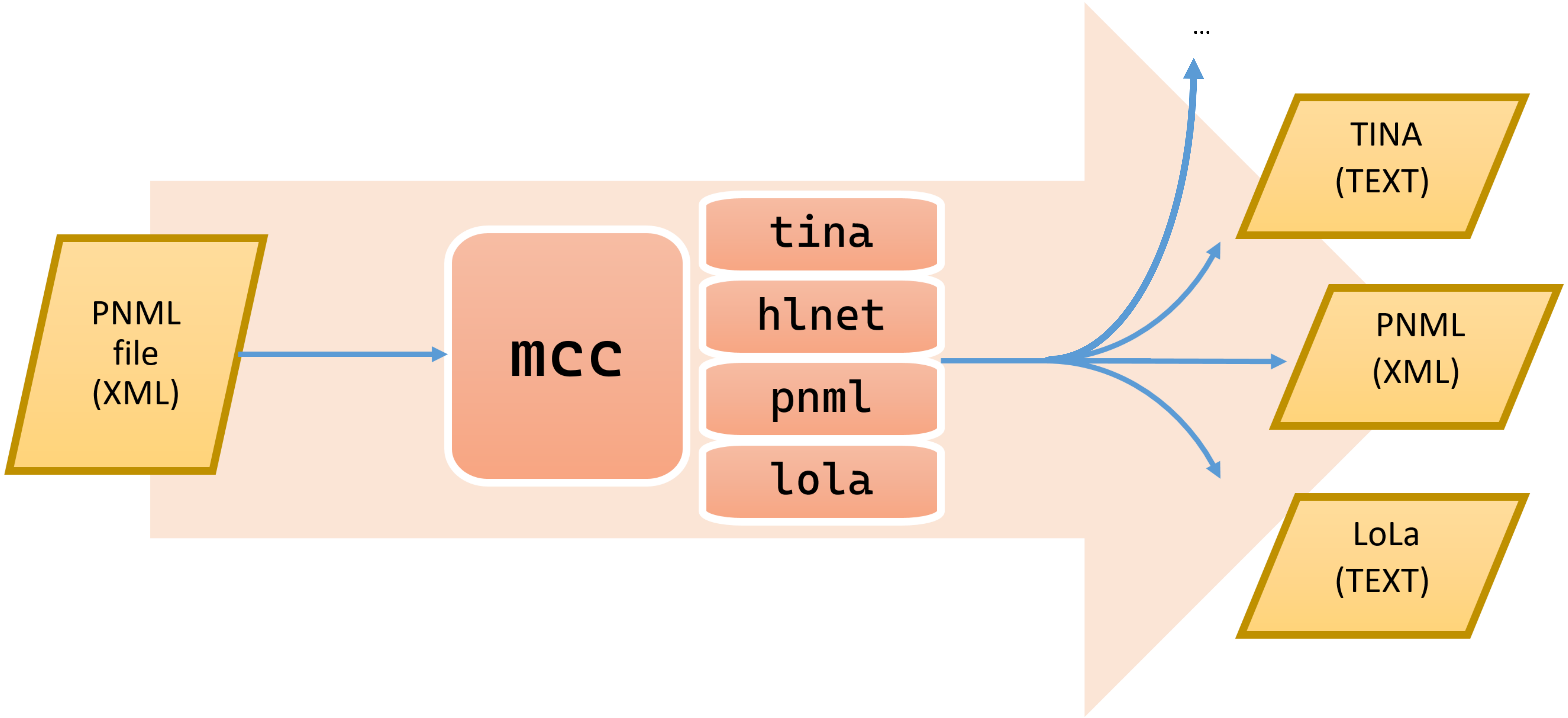
```
$> go get github.com/dalzilio/mcc
```

github.com/dalzilio/mcc

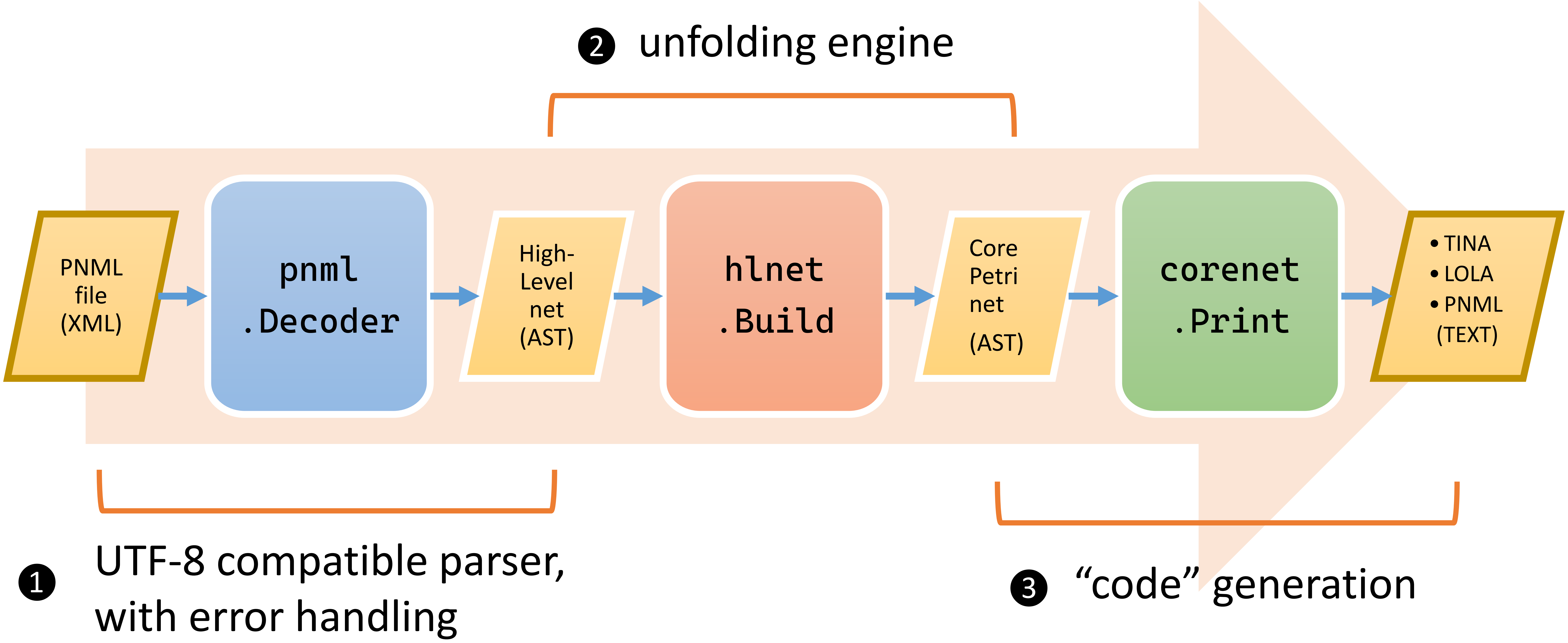


Architecture of MCC

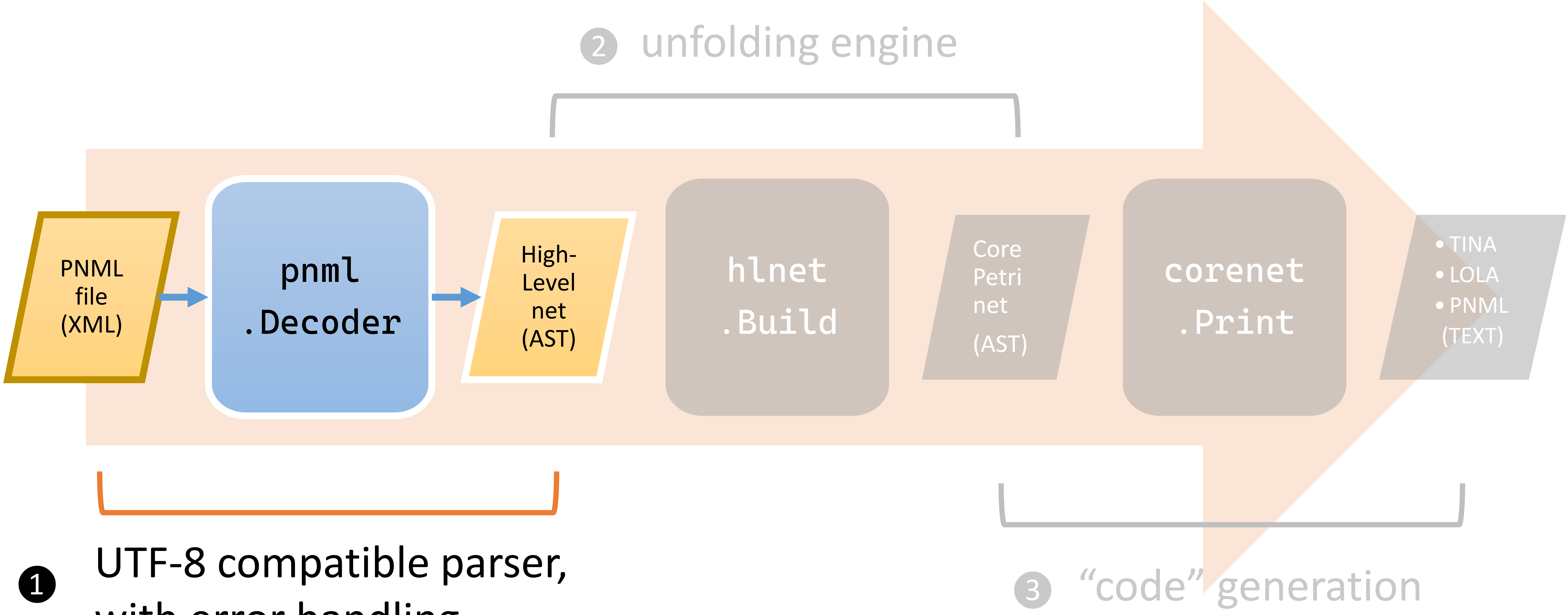
```
$> mcc tina –i lamport.pnml --name
```



Architecture of MCC



Parsing PNML files



❶ UTF-8 compatible parser, with error handling

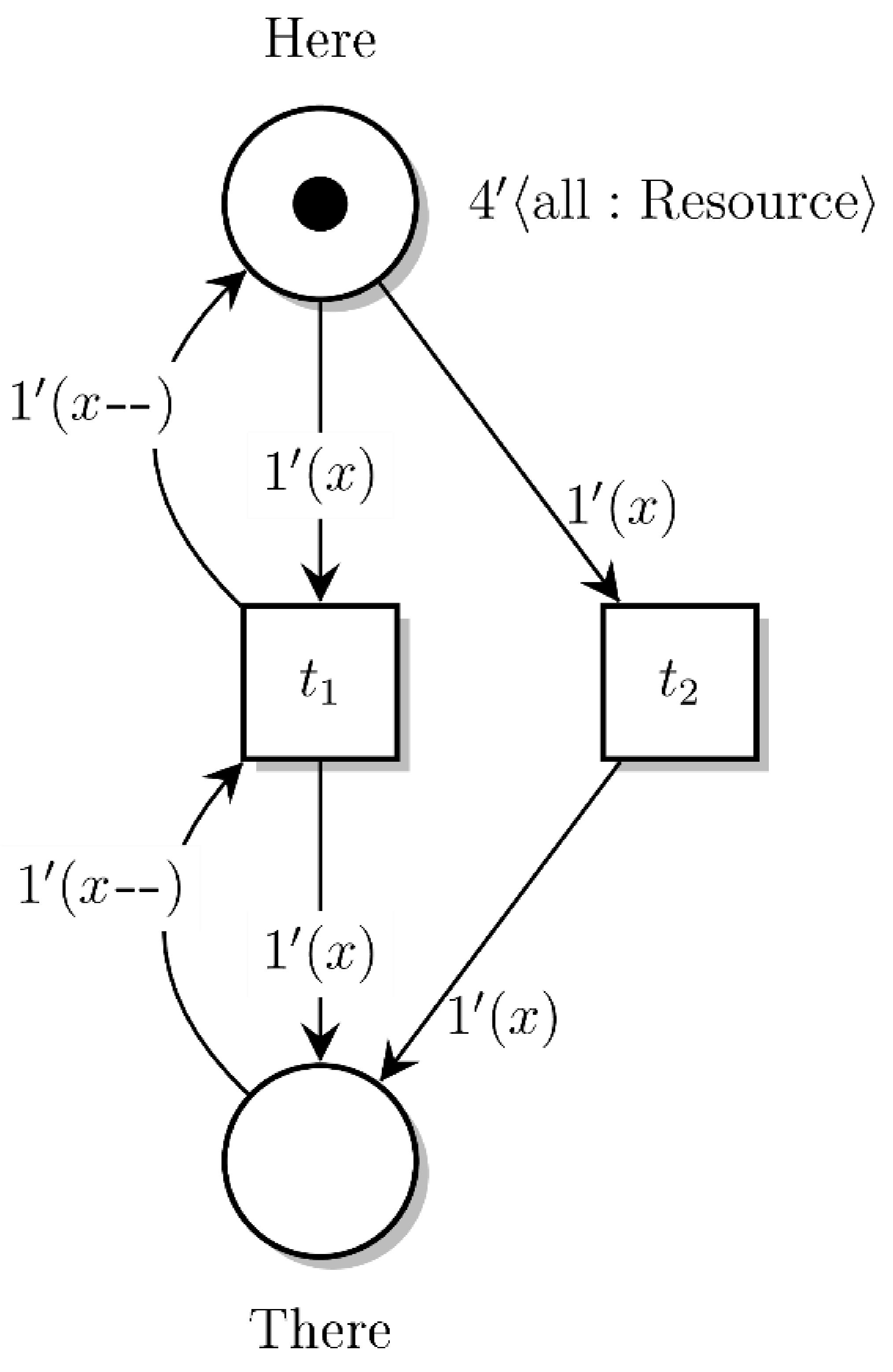
What is in a PNML model ?

types ≡ what is the color of a place

constants ≡ token (from a given color)

expressions ≡ operations over
multiset of constants

conditions ≡ used in guards



Here, There : RESOURCE
RESOURCE : 1..10

Supported PNML elements

types ::= dot
| cyclicenumeration
| finiteenumeration
| finiteintrange
| productsort
| partition
| partitionelement

constants ::= dotconstant (●)
| feconstant
| finiteintrangeconstant

expressions ::= variable (x)
| successor (x++) | predecessor (x--)
| tuple
| all | add | subtract

conditions ::= or | and | equality
| inequality
| lessthan | greaterthan
| greaterthanorequal
| lessthanorequal

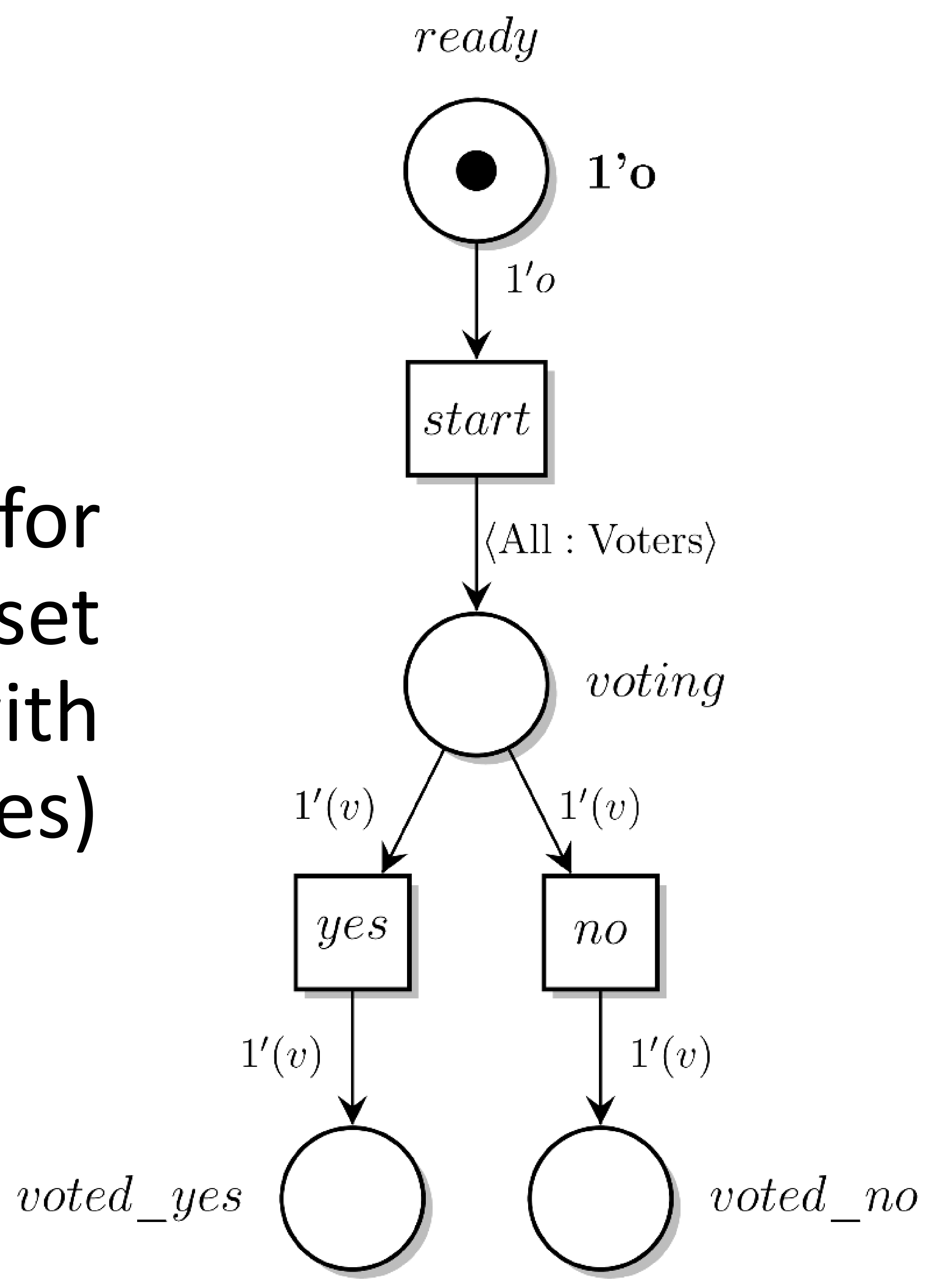
PNML 101: symmetric net

```xml
<?xml version="1.0" ?>
<pnml xmlns="http://www.pnml.org/version-2009/grammar/pnml">
    <net id="Referendum-COL-0010" type="http://www.pnml.org/version-2009/grammar/symmetricnet">
        <page id="DocumentDefaultPage">
            <place id="voted_no">
                <name><text>voted_no</text></name>
                <type><text>Voters</text></type>
            </place>

            <transition id="no"><name><text>no</text></name></transition>

            <arc id="arc12" source="no" target="voted_no">...</arc>
        </page>
        <name>
            <text>Referendum-COL-010</text>
        </name>
        <declaration>
            <structure>
                <declarations>
                    <namedsort id="Voters" name="Voters">
                        <cyclicenumeration>
                            <feconstant id="Voters1" name="1" />
                            <feconstant id="Voters2" name="2" />
                            ...
                        </cyclicenumeration>
                    </namedsort>
                </declarations>
            </structure>
        </declaration>
    </net>
</pnml>
```

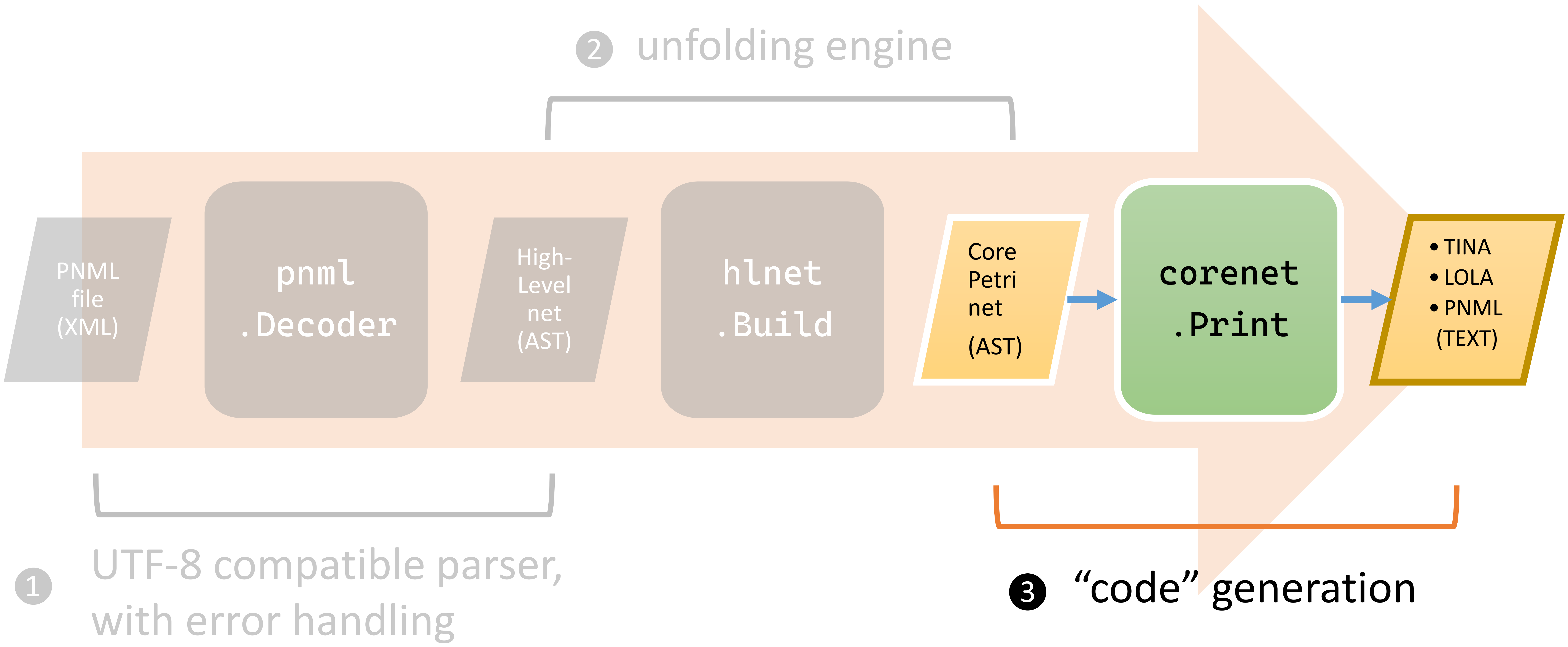
XML language for describing a subset of colored nets (with values of finite types)



URI: http://www.pnml.org/version-2009/grammar/symmetricnet

image generated with mcc + nd with tikz export

Output format



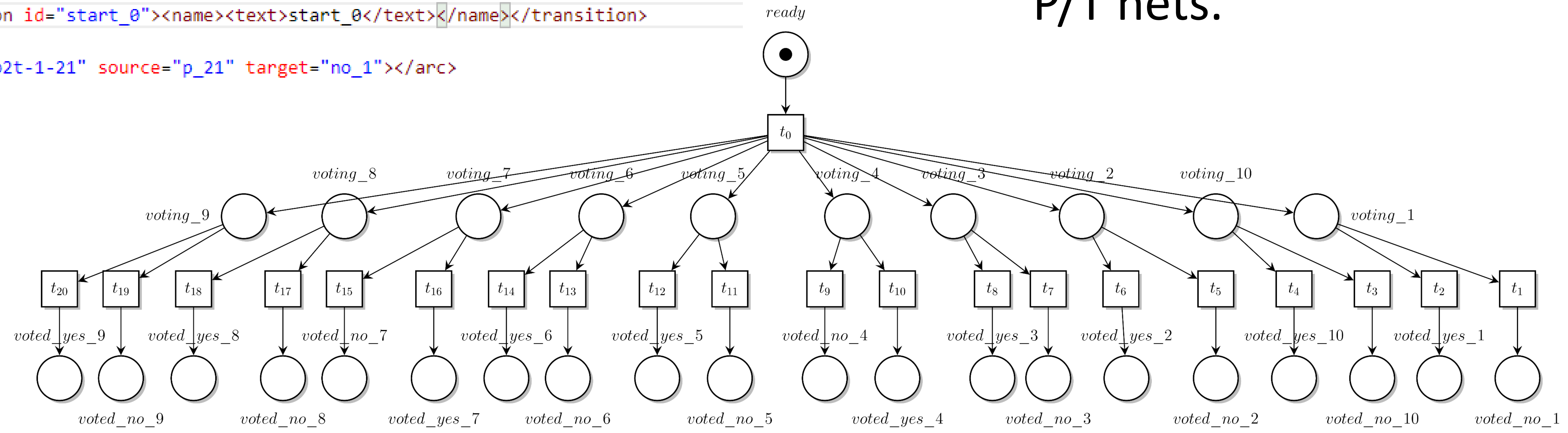
PNML 101: Place/Transition nets

```xml
<?xml version="1.0" encoding="UTF-8"?>
<pnml xmlns="http://www.pnml.org/version-2009/grammar/pnml">
  <net type="http://www.pnml.org/version-2009/grammar/ptnet" id="Referendum-COL-010">
    <name><text>MCC-PT-Referendum-COL-010</text></name>
    <page id="page">
      <place id="p_0"><name><text>p_0</text></name></place>

      <transition id="start_0"><name><text>start_0</text></name></transition>

      <arc id="p2t-1-21" source="p_21" target="no_1"></arc>
    </page>
  </net>
</pnml>
```

XML language for describing "core" P/T nets.



URI: http://www.pnml.org/version-2009/grammar/ptnet

net generated with mcc; image from nd with tikz export

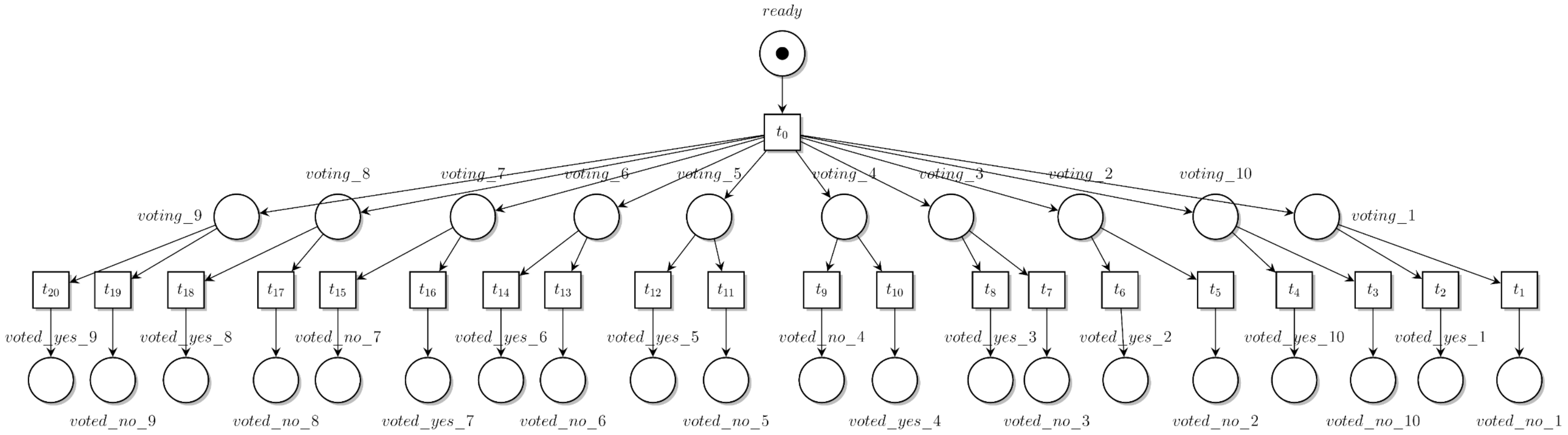
The .net format from Tina

```
$>  mcc tina -i referendum.pnml -o -
# net Referendum-COL-002 has 7 places
# and 5 transitions
net {Referendum-COL-002}
pl ready (1)
tr t0  ready -> voting_1 voting_2
tr t1  voting_1 -> voted_no_1
tr t2  voting_1 -> voted_yes_1
tr t3  voting_2 -> voted_no_2
tr t4  voting_2 -> voted_yes_2
```

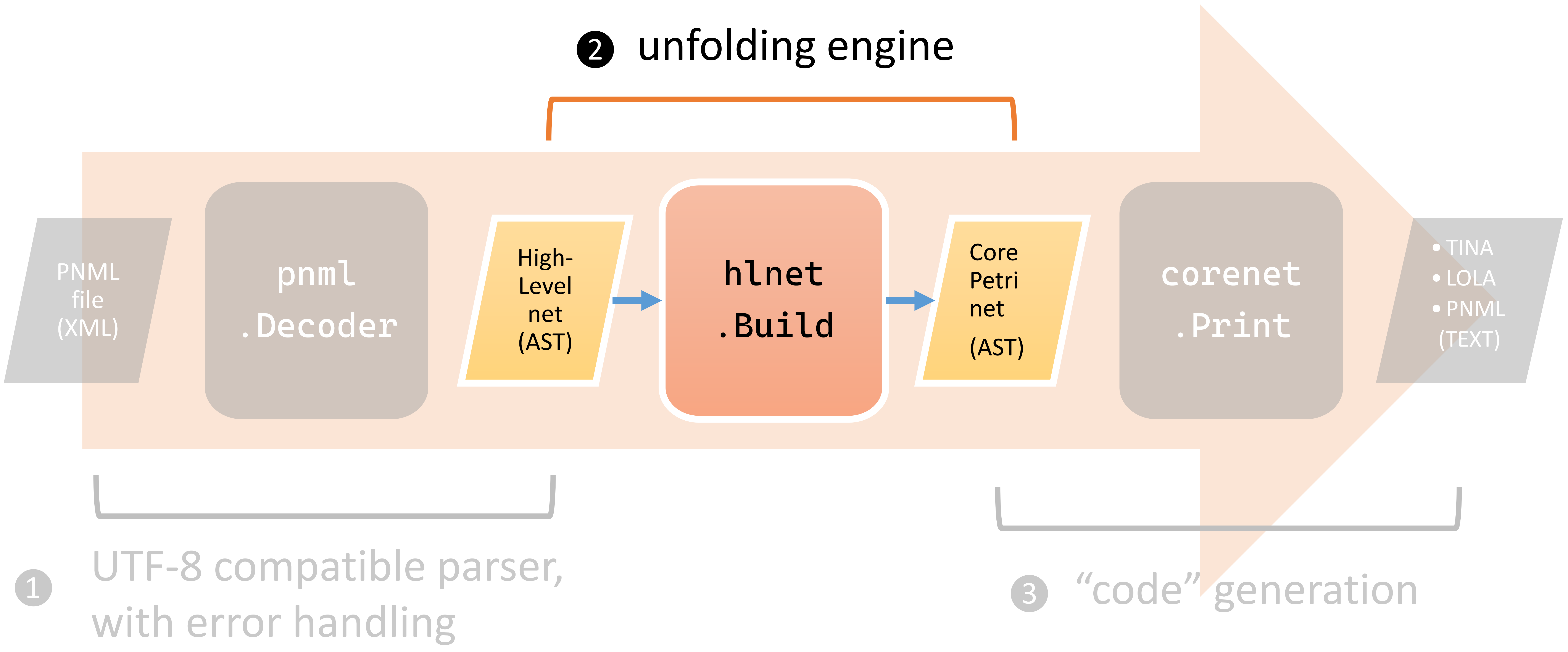
Same information than in the PNML model for P/T nets

⇒ this is just an oriented graph

Most tools in Tina actually supports the PNML ptnet format

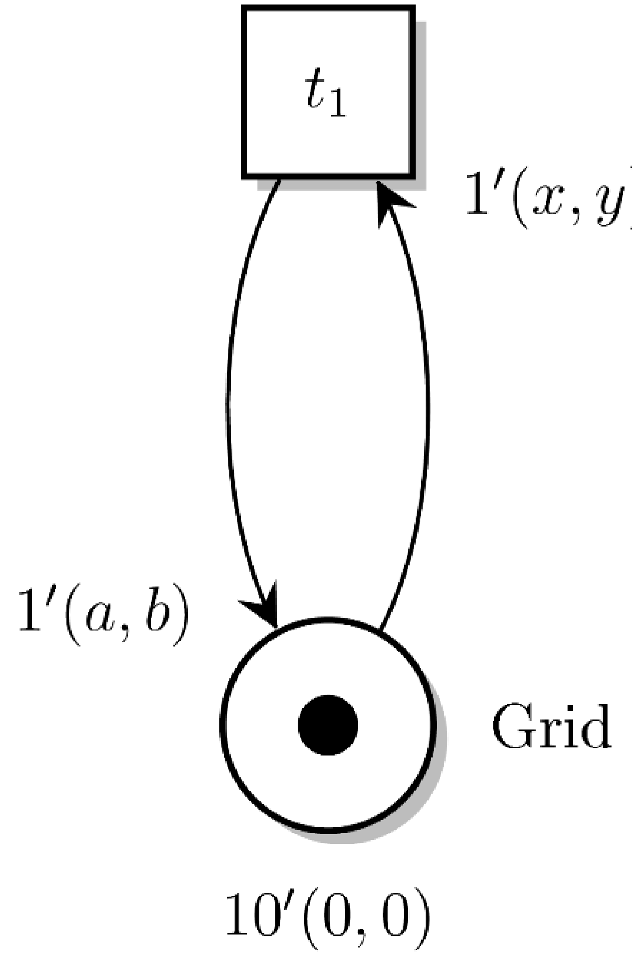


Unfolding engine

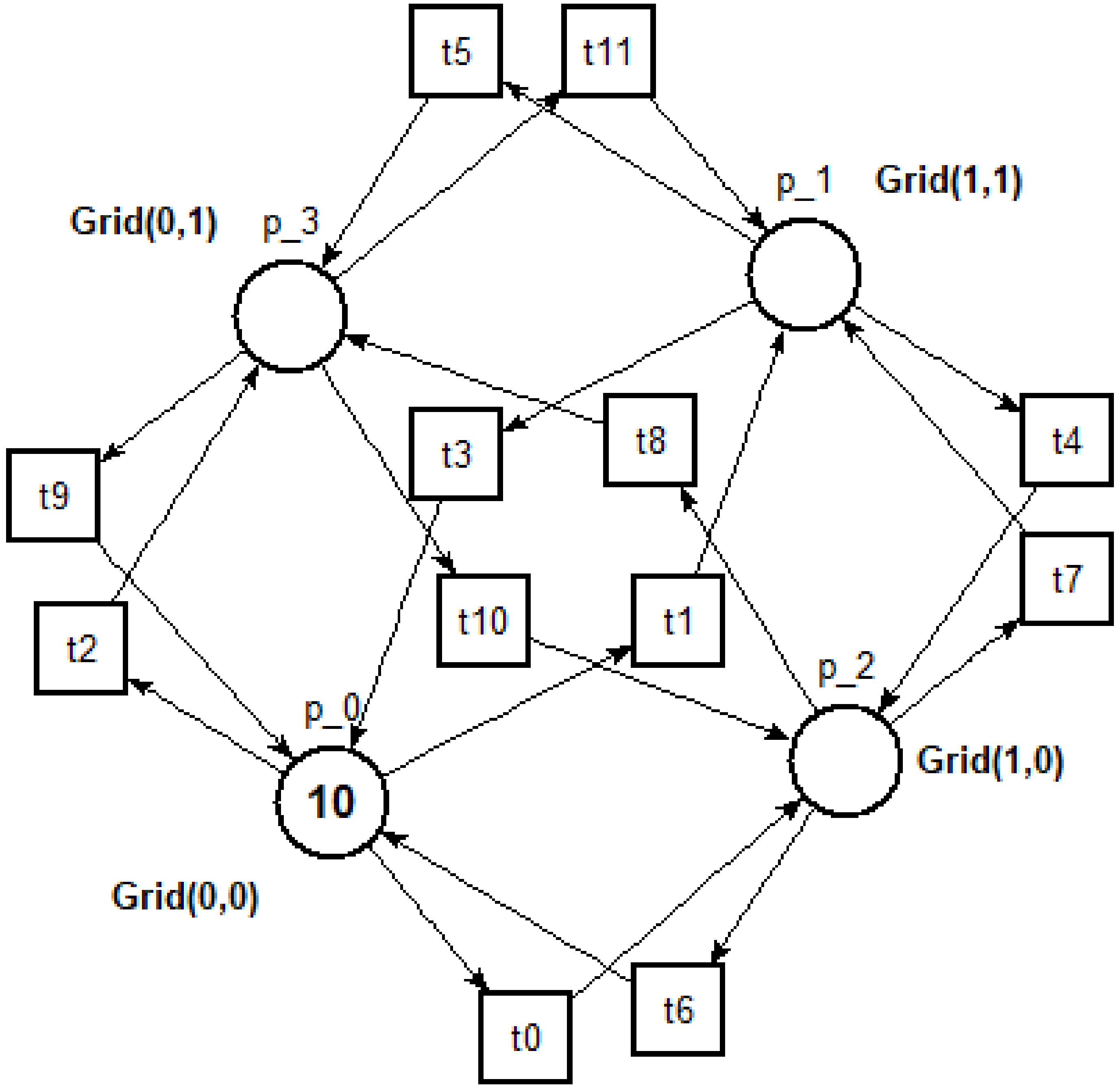


Unfolding: an example

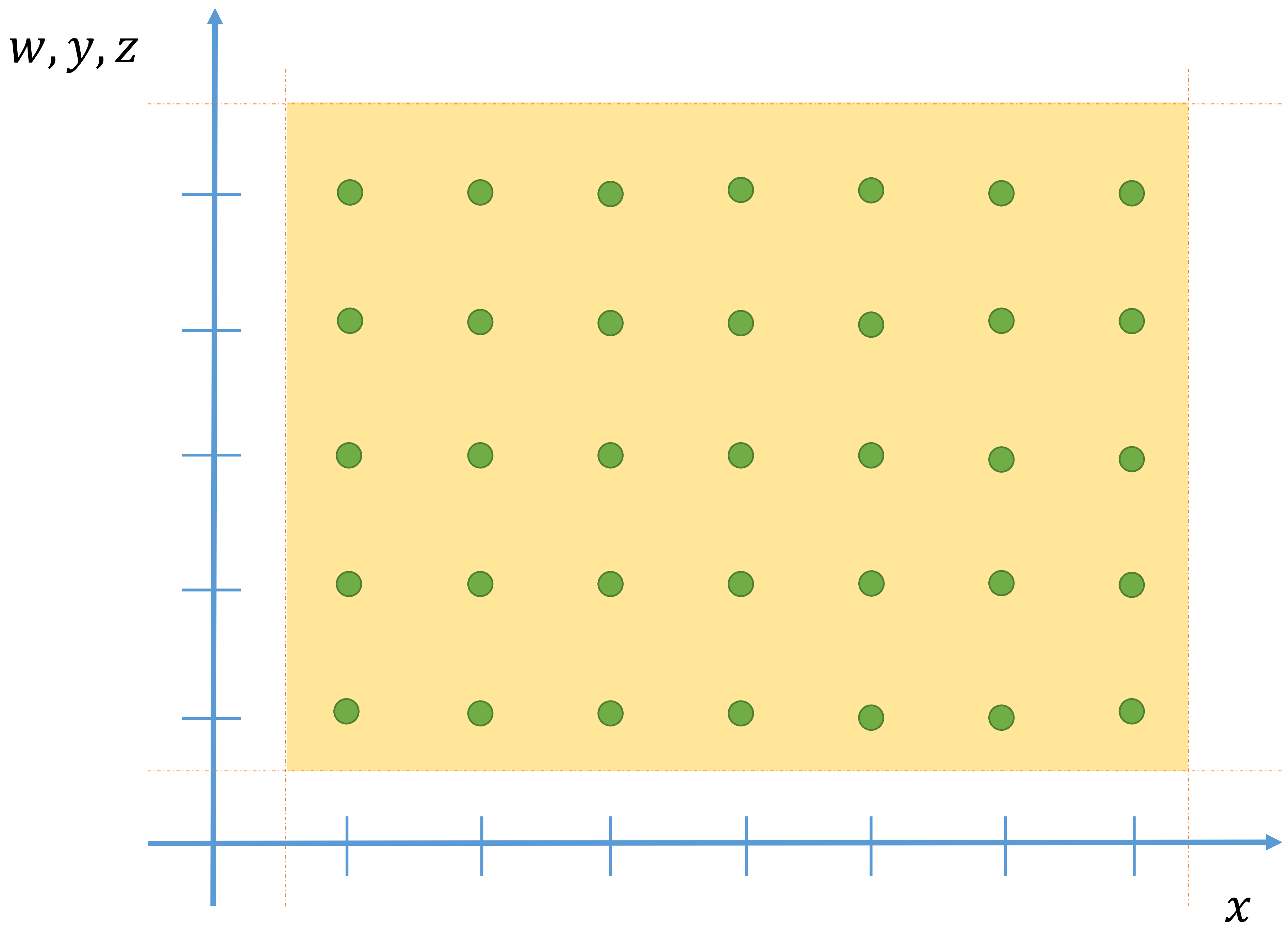
$$(a = x \lor a = x{+}{+} \lor a = x{-}{-}) \land (b = y \lor b = y{+}{+} \lor b = y{-}{-}) \land (a \neq x \lor b \neq y)$$

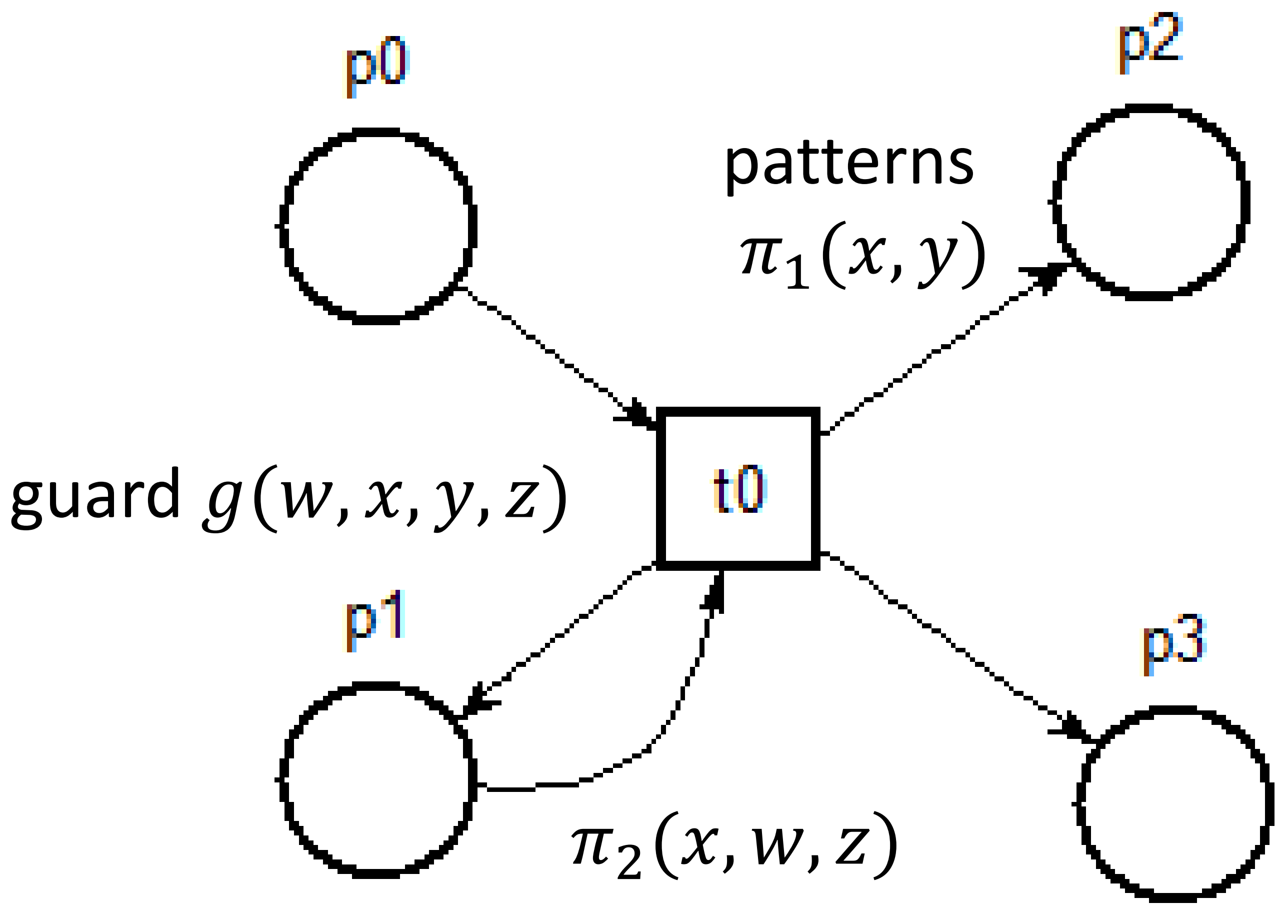


Grid : CD $\times$ CD
CD : $0\,..\,1$



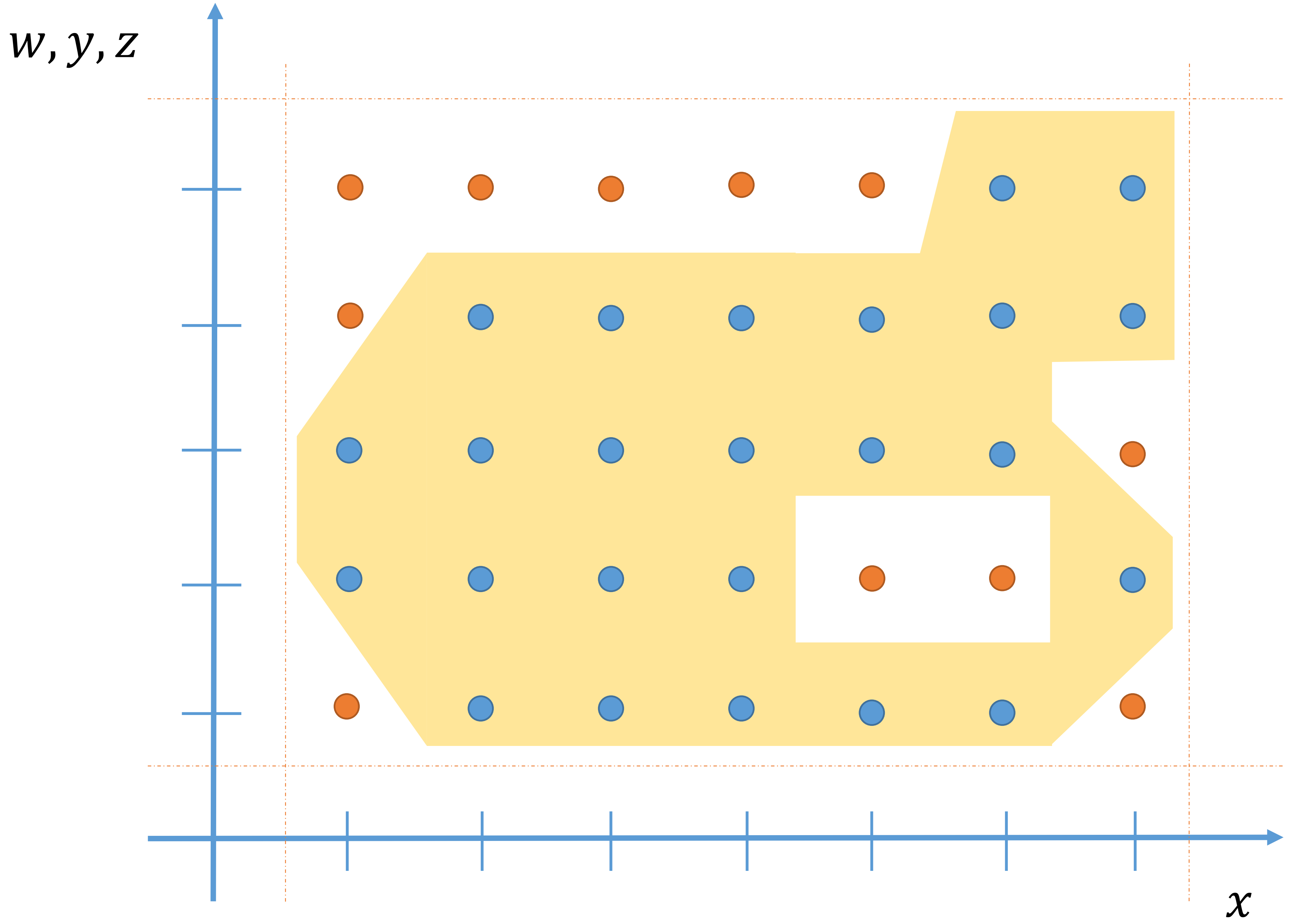
Unfolding: types and environment

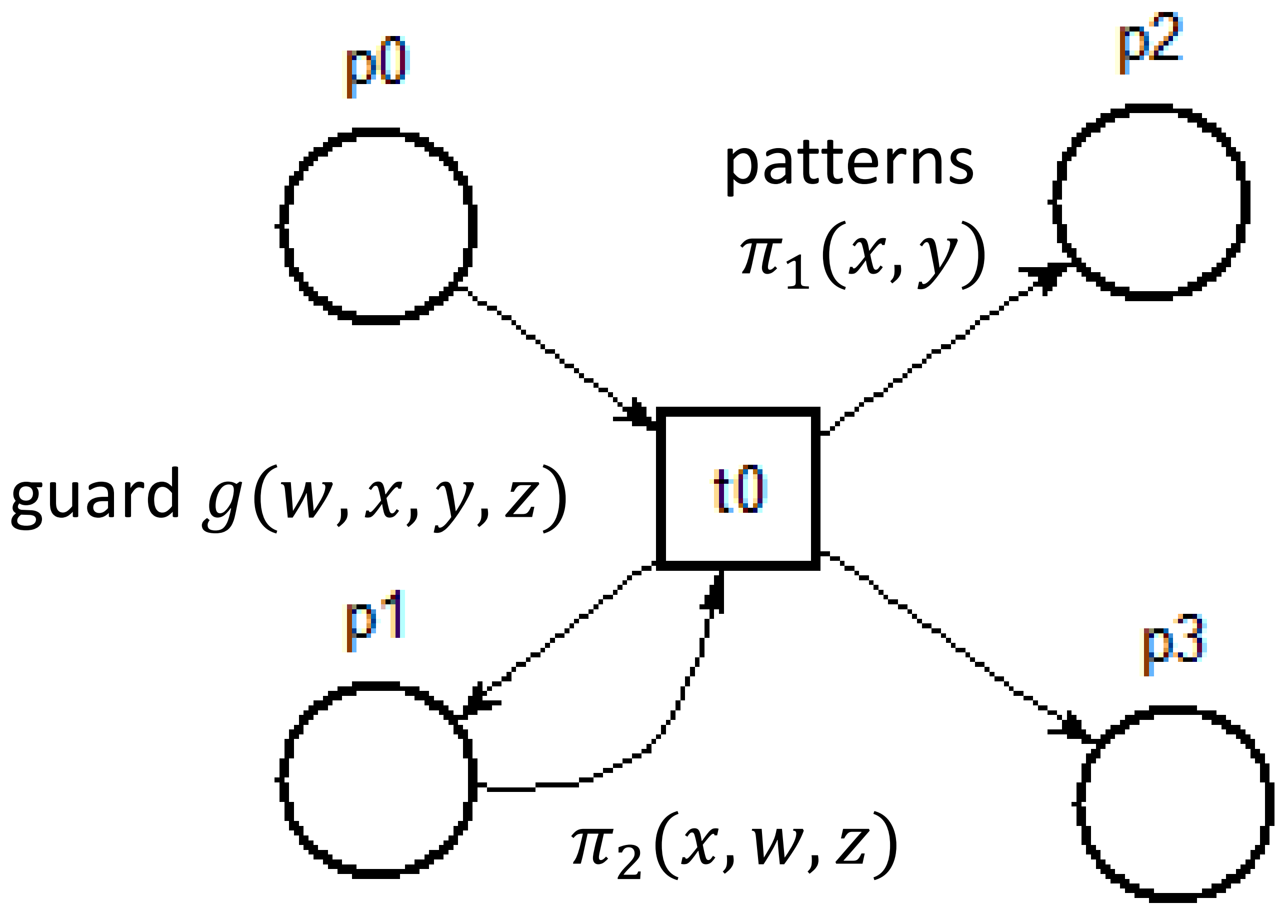




Environment $\equiv x : T_x, \ldots, z : T_z$

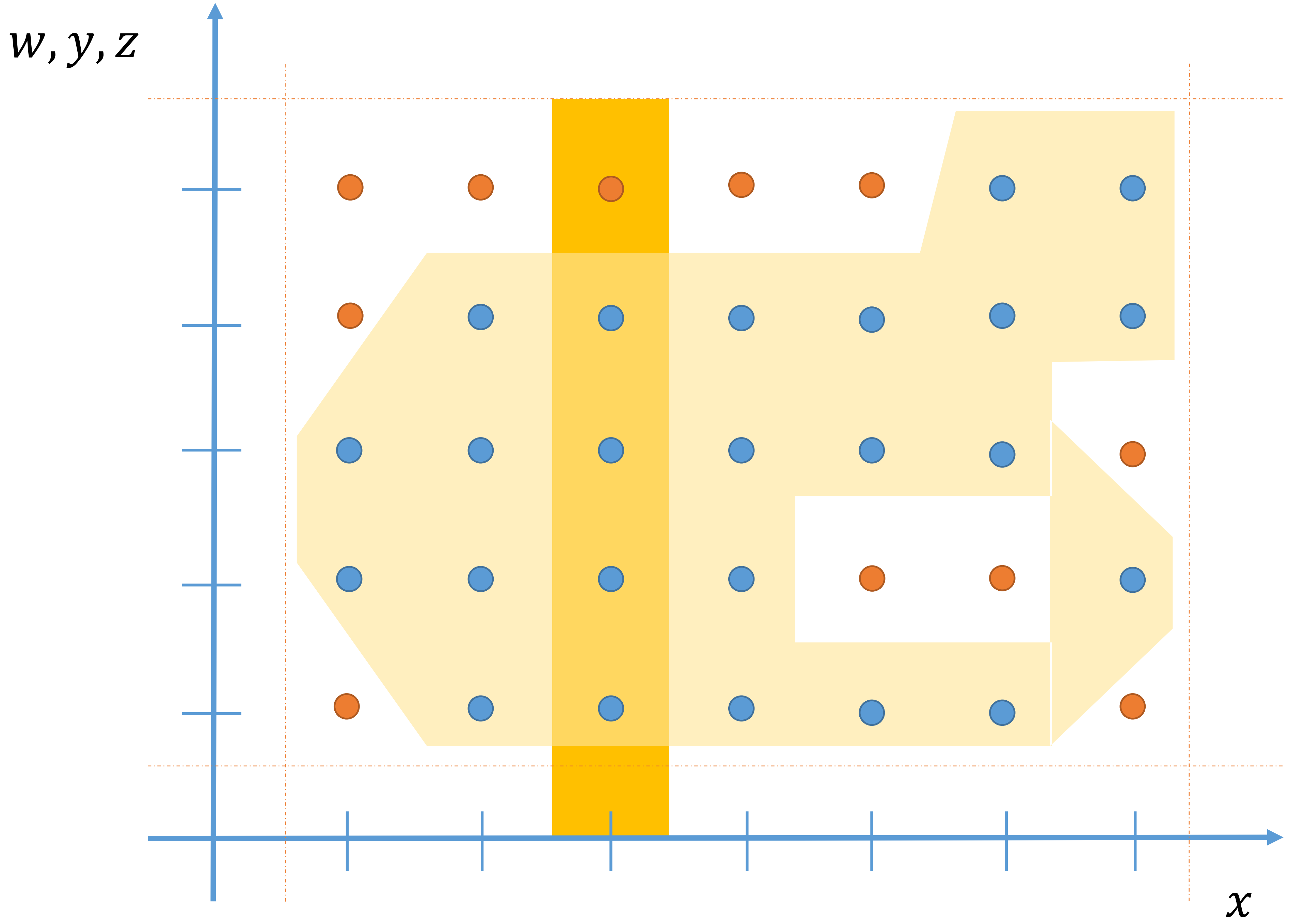
Unfolding: types and environment

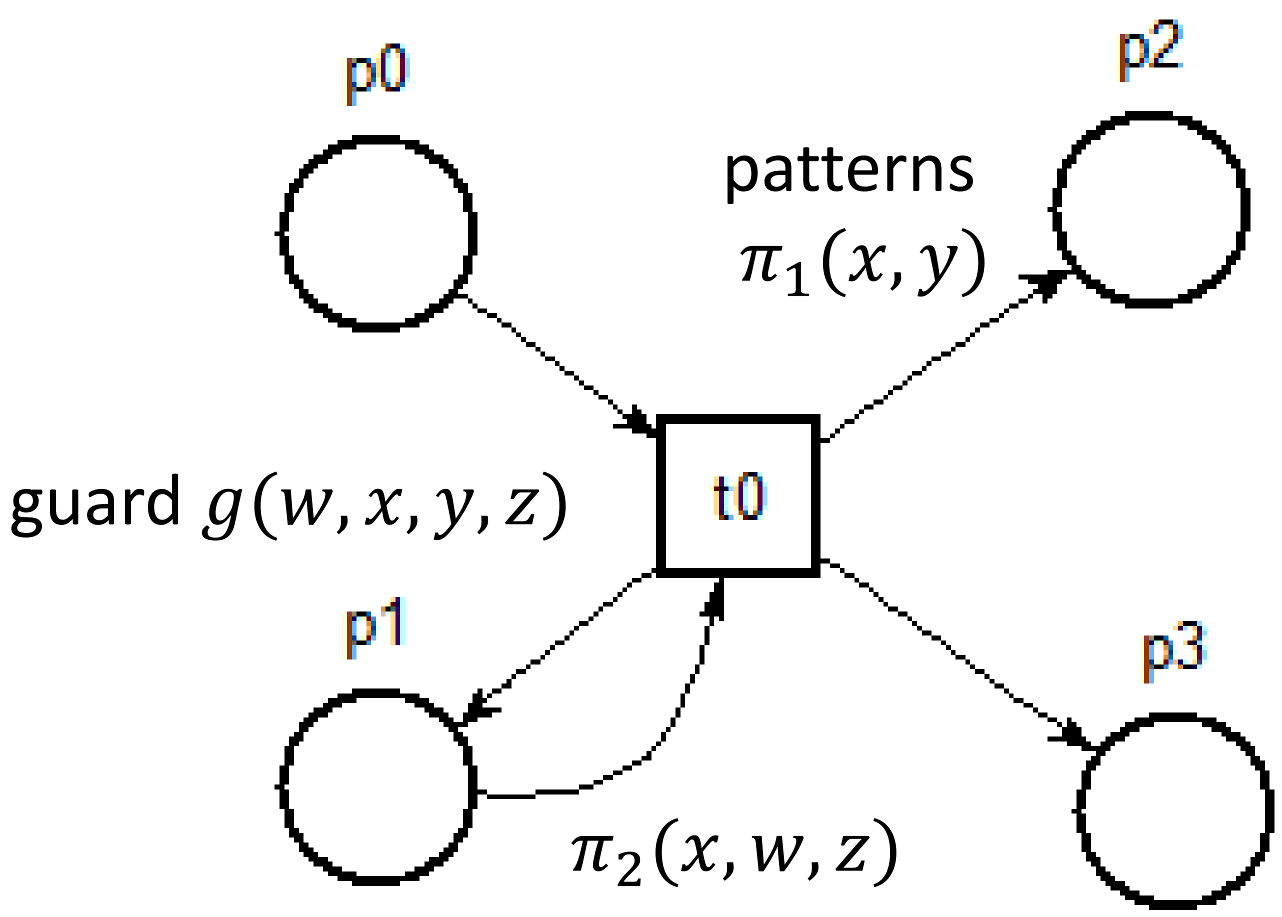




Environment $\equiv x : T_x, \ldots, z: T_z$

Unfolding: a constraint solving approach





Environment $\equiv x : T_x, \ldots, z : T_z$

What else can you do with mcc ?

— structured naming of places

— debugging + prettifying of colored models

How good is it ?

— mcc finishes on all the feasible models in the MCC

How does it look like under the hood ?

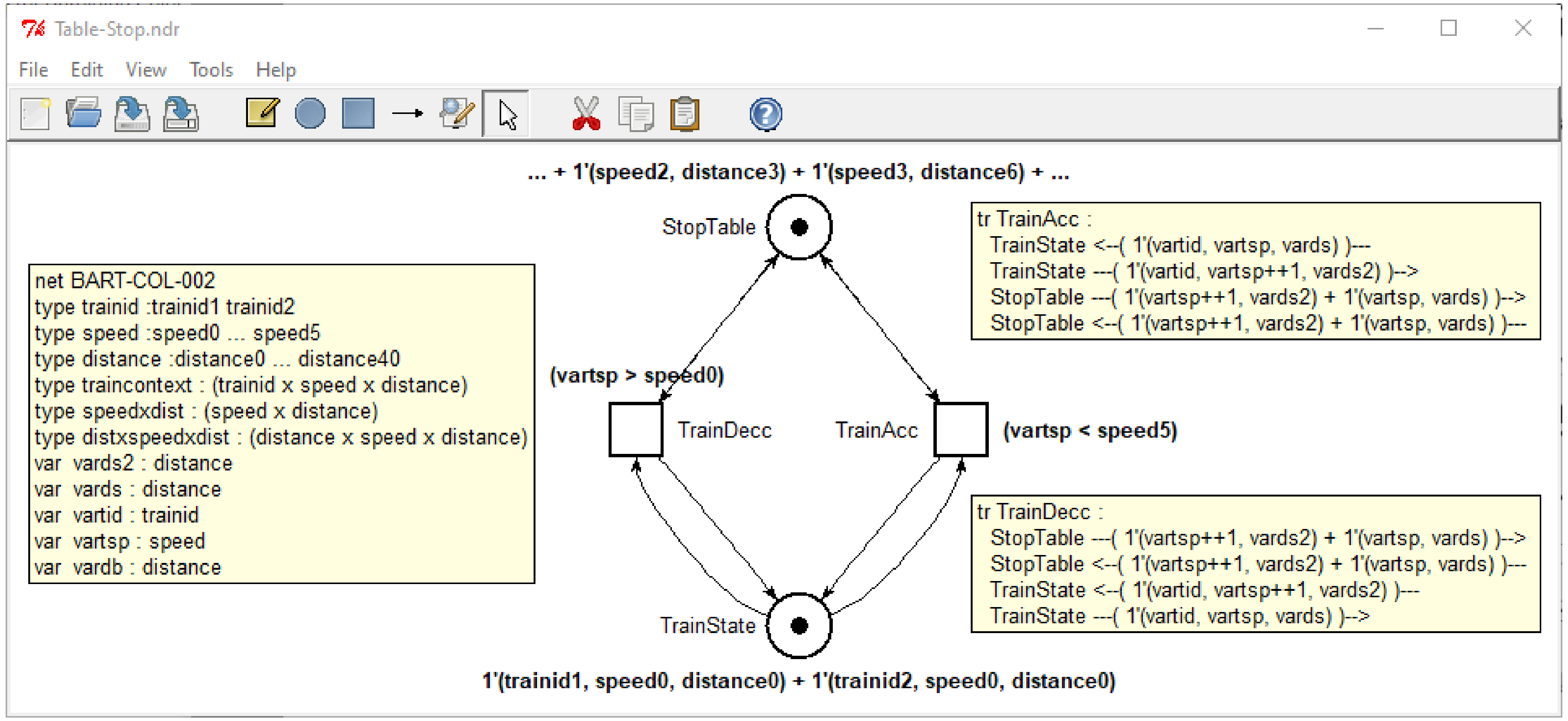
— we follow a constraint-solving approach

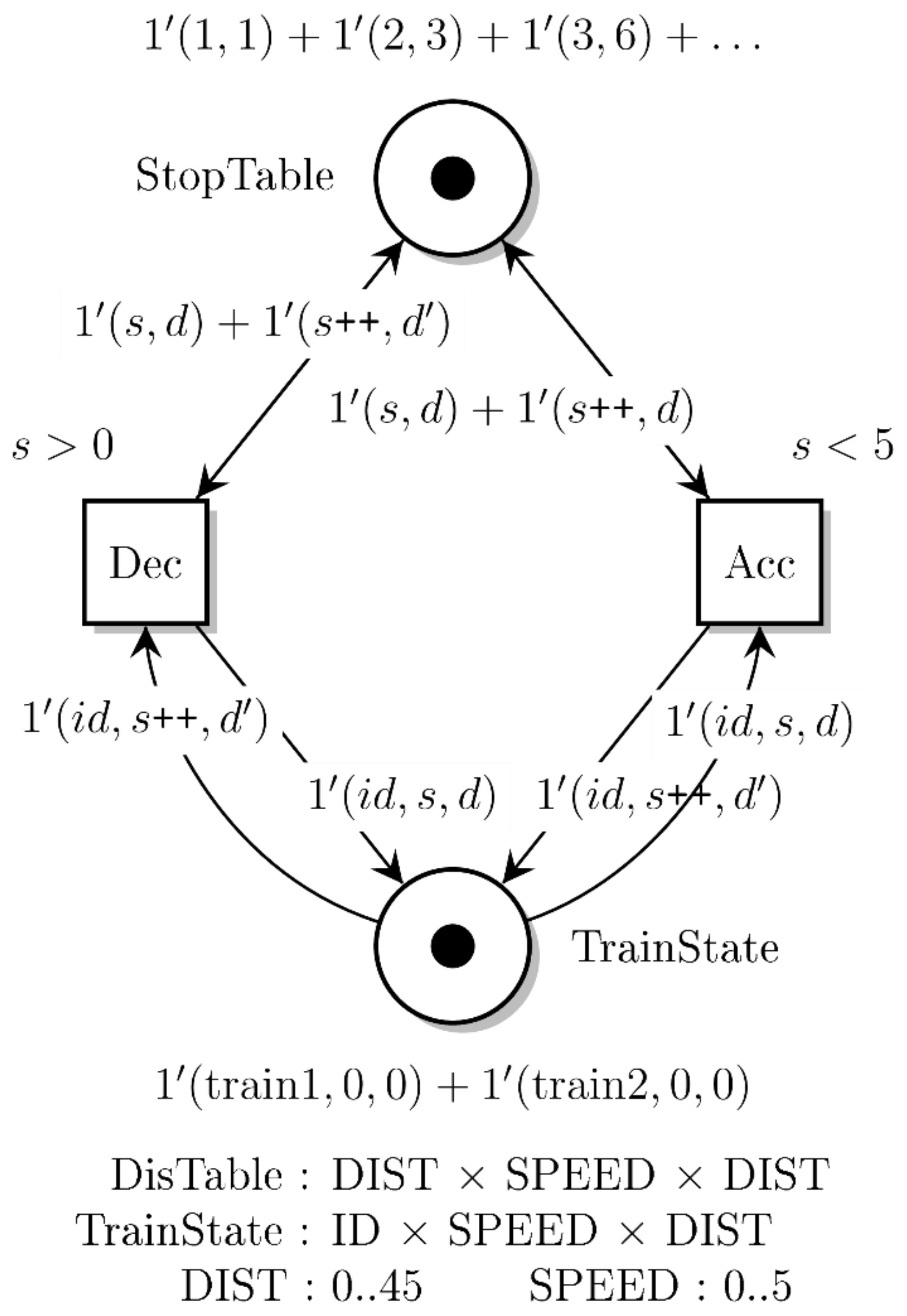
— use of colored invariant

— use of a Petri scripting language

Prettifying PNML models

```
$> mcc hlnet -i table.pnml --debug
```





Comparison with other Tools

Colored net (PNML) unfolding in the literature

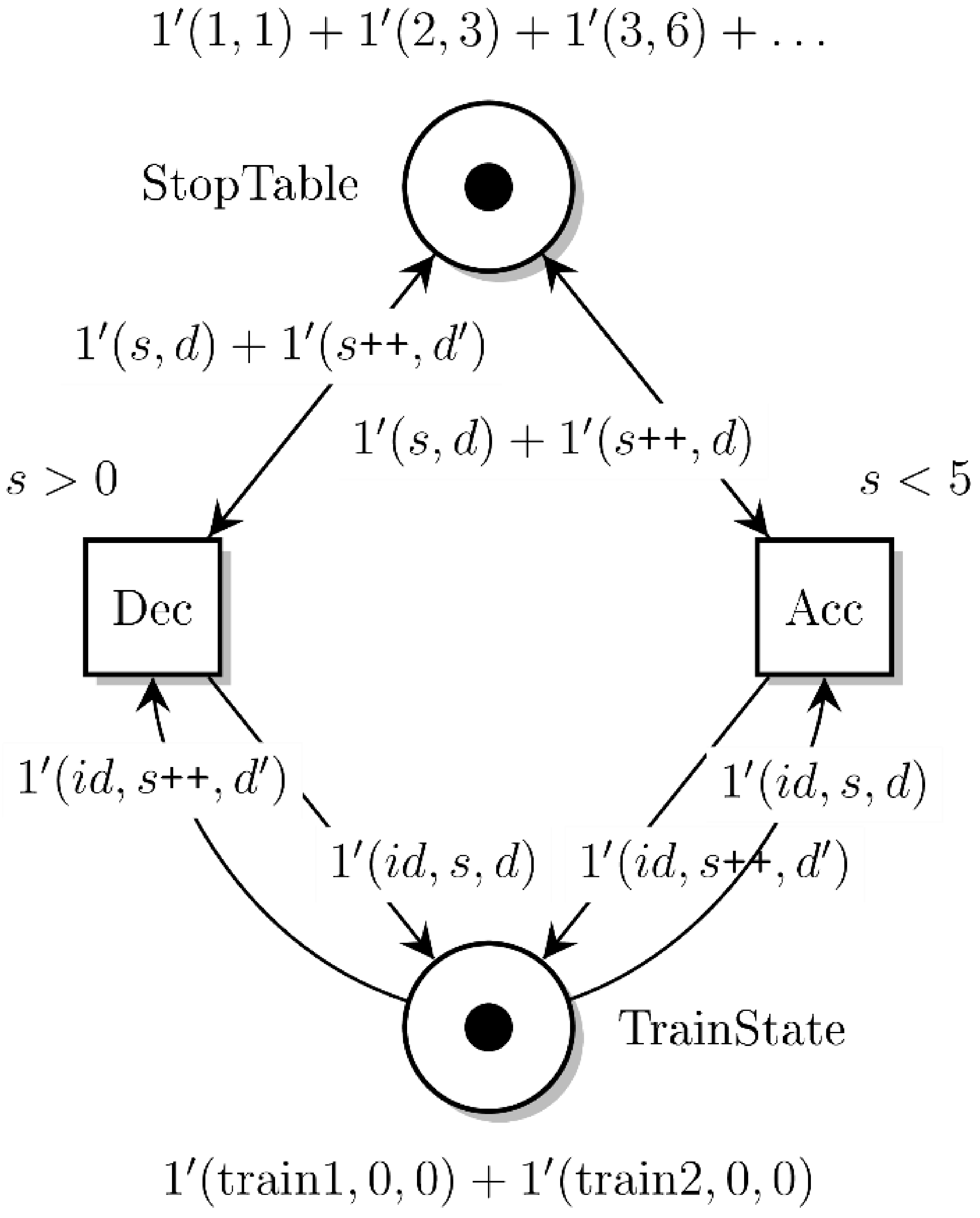
MARIA [Mäkelä, 2001] ; CPN-AMI [Kordon 2006] ; MARCIE [Heiner, 2011 **and 2020**] ; …

Unfolding tools in the MCC

- verifypn (part of Tapaal)
- andl_converter (part of Marcie)
- GreatSPN editor

Model	Places	Trans.	MCC	Tapaal	Marcie	GSPN
GlobalResAllocation-07	133	291 067	**1.7**	3	14.4	22.3
GlobalResAllocation-11	297	2.10^6	**15.1**	29.3	144.6	—
DrinkVendingMachine-16	192	10^6	15.5	10.7	52.8	108.1
DrinkVendingMachine-24	288	8.10^6	97.1	95.9	—	—
PhilosophersDyn-50	2 850	255 150	**1**	2.1	11.1	15.7
PhilosophersDyn-80	6 960	10^6	**4.1**	9.9	55.9	61.0
Diffusion-D050	2 500	8 109	14.5	**0.6**	4.1	—
Diffusion-D100	10 000	31 209	243.3	**8.6**	31.3	—
TokenRing-100	10 201	10^6	**4**	8.2	33.5	49.3
TokenRing-200	40 401	8.10^6	**67.4**	166.1	—	—
SafeBus-50	5 606	140 251	14.2	**1.4**	6.2	25.1
SafeBus-80	13 766	550 801	89.5	**7**	20.6	133.1
TrainTable-Dist	722	602	**1.4**	12.6	59.5	69.4
TrainTable-Stop+Dist	728	602	**2.1**	—	—	—
BART-002	764	646	**3.1**	—	—	—
BART-060	15 032	19 380	**3.2**	—	—	—
SharedMemory-000200	40 801	80 400	**0.3**	1.7	2.6	5.1
SharedMemory-001000	10^6	2.10^6	**8.9**	—	60.3	160.2
SharedMemory-002000	4.10^6	8.10^6	**55.3**	—	—	—
FamilyReunion-L800	2.10^6	2.10^6	**5.5**	—	84.8	143.0
FamilyReunion-L3000	28.10^6	27.10^6	**89.5**	—	—	—

Implementation: colored invariants



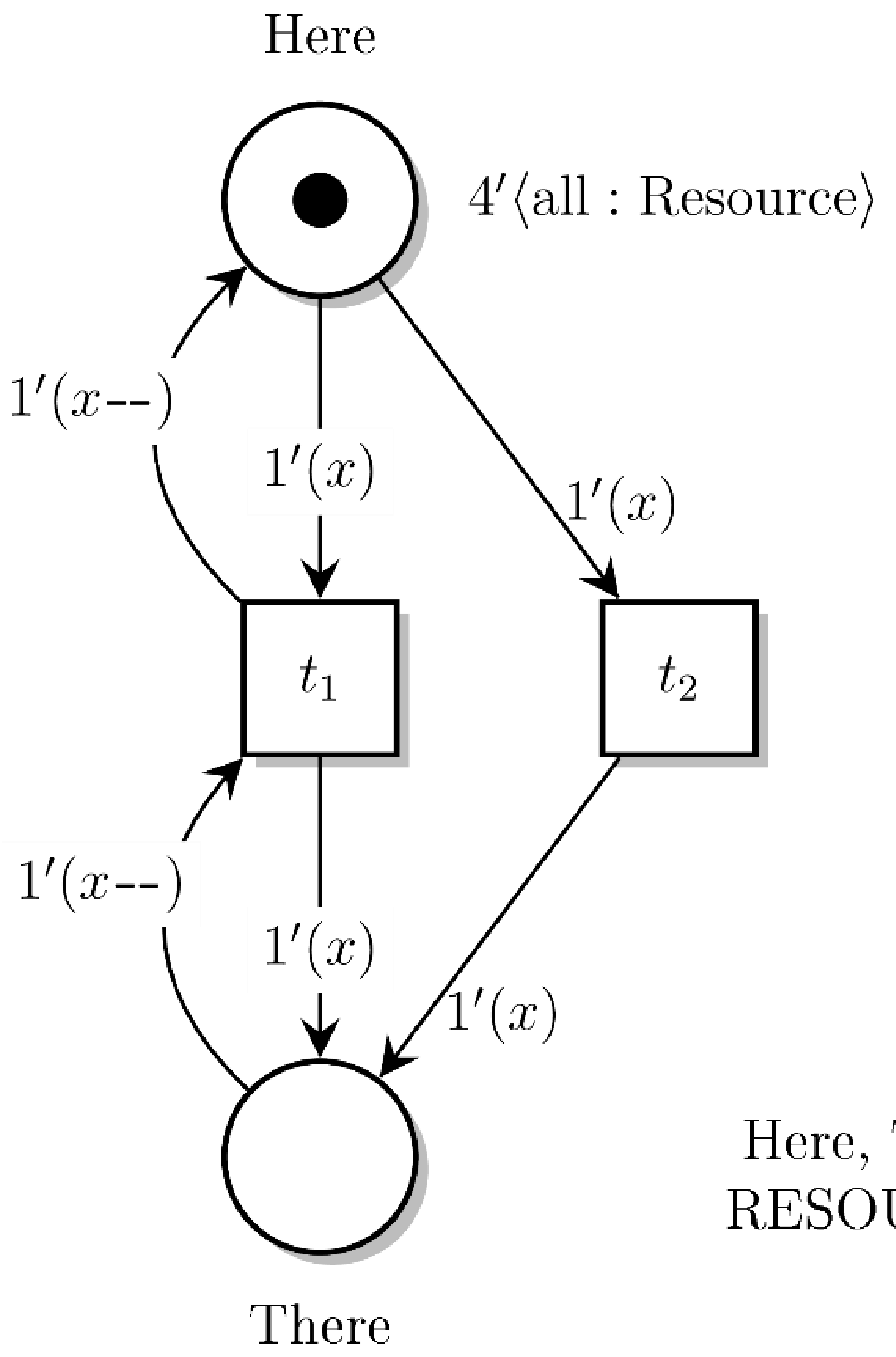
Place $StopTable$ is stable.

Its type has $46 \times 6 = 276$ possible values, its initial marking only 6

Therefore there are $276^2 = 76\,176$ potential combinations to test for transitions Dec and Acc; instead of 36

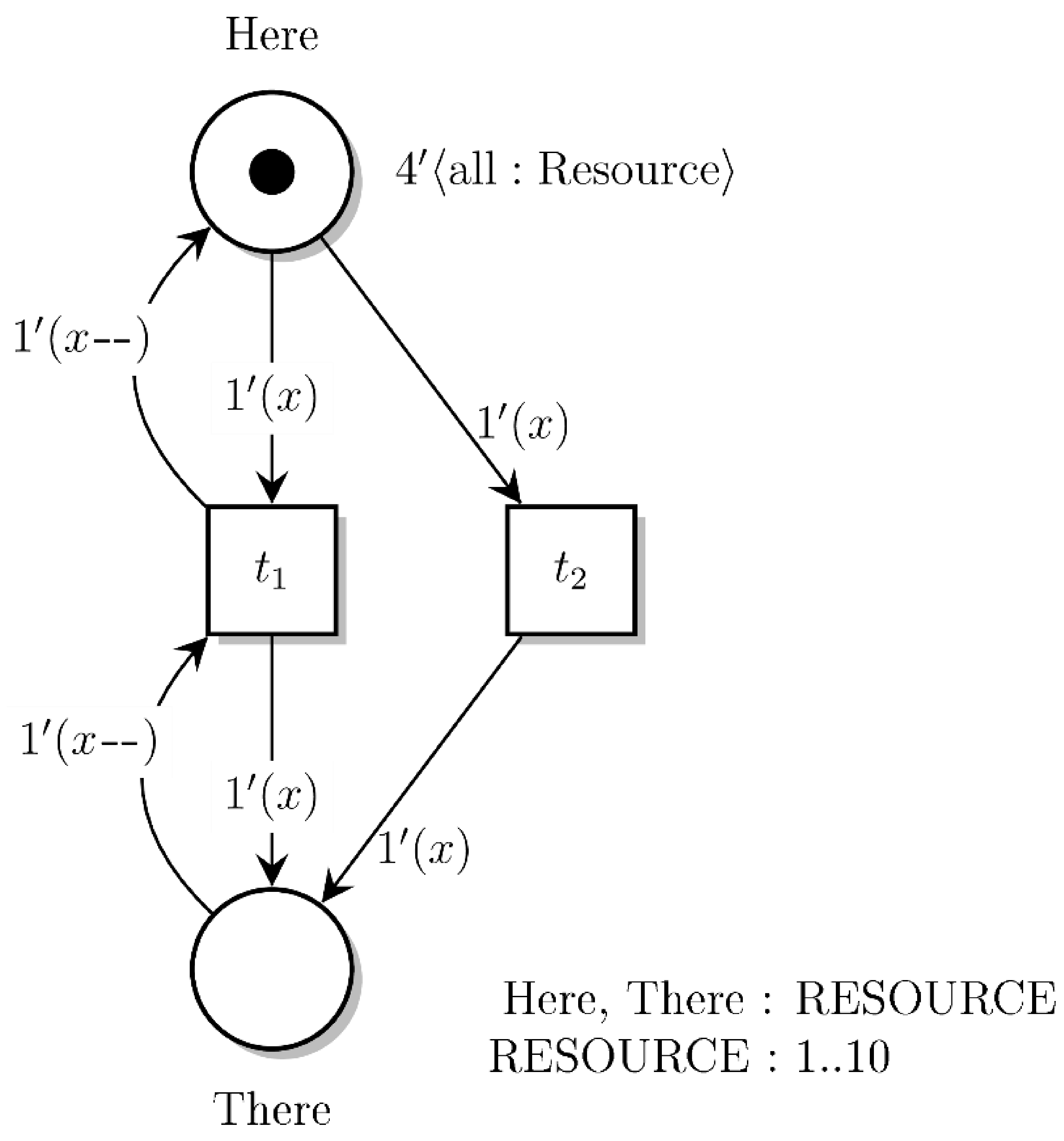
$$\begin{aligned} \mathrm{DisTable} &: \mathrm{DIST} \times \mathrm{SPEED} \times \mathrm{DIST} \\ \mathrm{TrainState} &: \mathrm{ID} \times \mathrm{SPEED} \times \mathrm{DIST} \\ \mathrm{DIST} &: 0..45 \qquad \mathrm{SPEED} : 0..5 \end{aligned}$$

Implementation: Petri scripting language



Here, There : RESOURCE
RESOURCE : 1..10

Implementation: Petri scripting language




```
net {Swap-COL-P10-N4}

pl p0 : {Here} (4)
pl p1 : {There}
tr t0 : {l-t1} p1 -> p0
tr t1 : {r-t1} p0 -> p1
tr t2 : {h-t2} p0 -> p1

RING sync 10 /{l-t1},{r-t1} {h-t2}
```

MCC

A tool to remove colors from your High-Level Petri nets !

https://github.com/dalzilio/mcc

tina-users @ laas . fr

thank you to:

- paxtonhare/demo-magic: for repeatable shell script demos
- Audacity: for the audio editing
- Captura: for the screen capture
- Shotcut: for my first experience editing videos