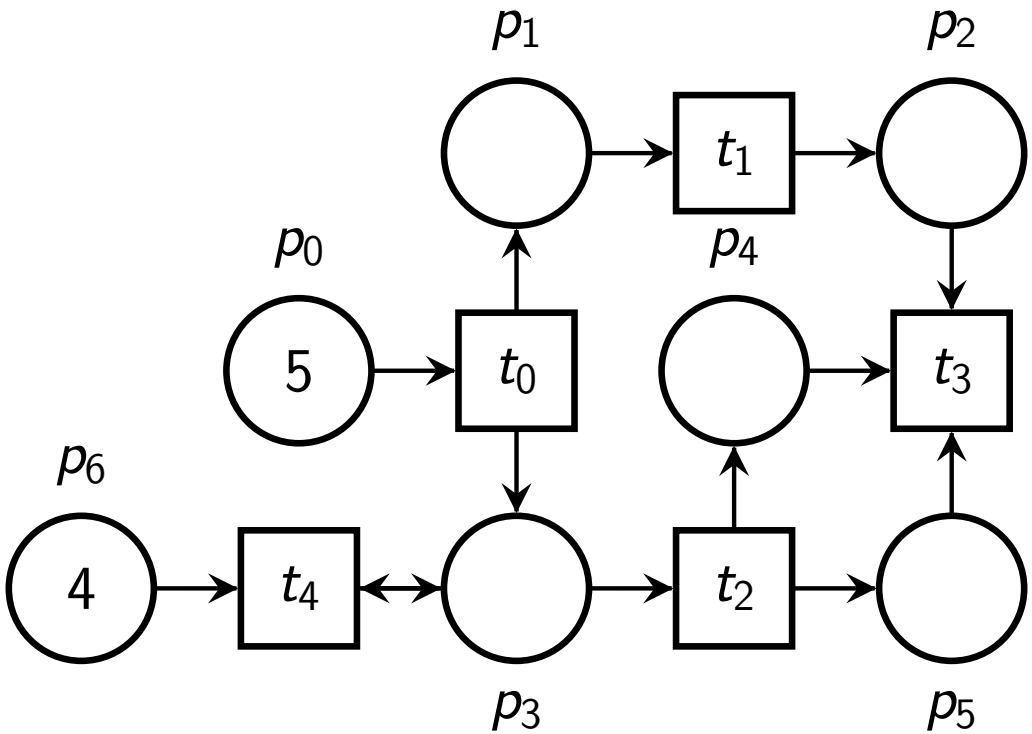
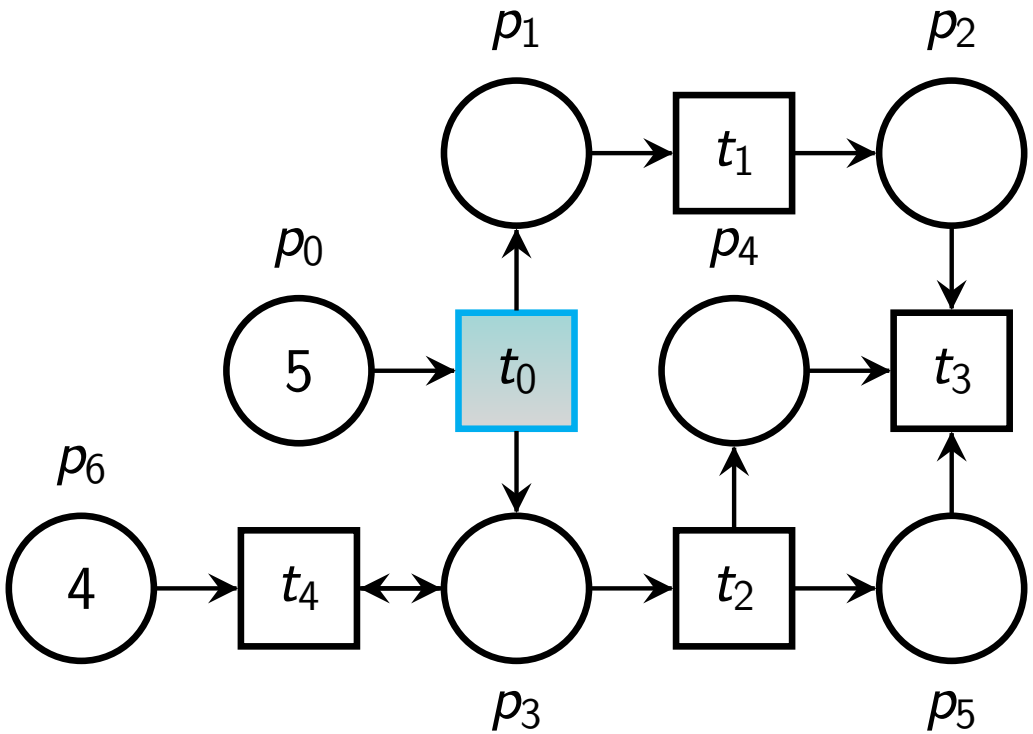
SMPT: A Testbed for Reachability Methods in Generalized Petri Nets

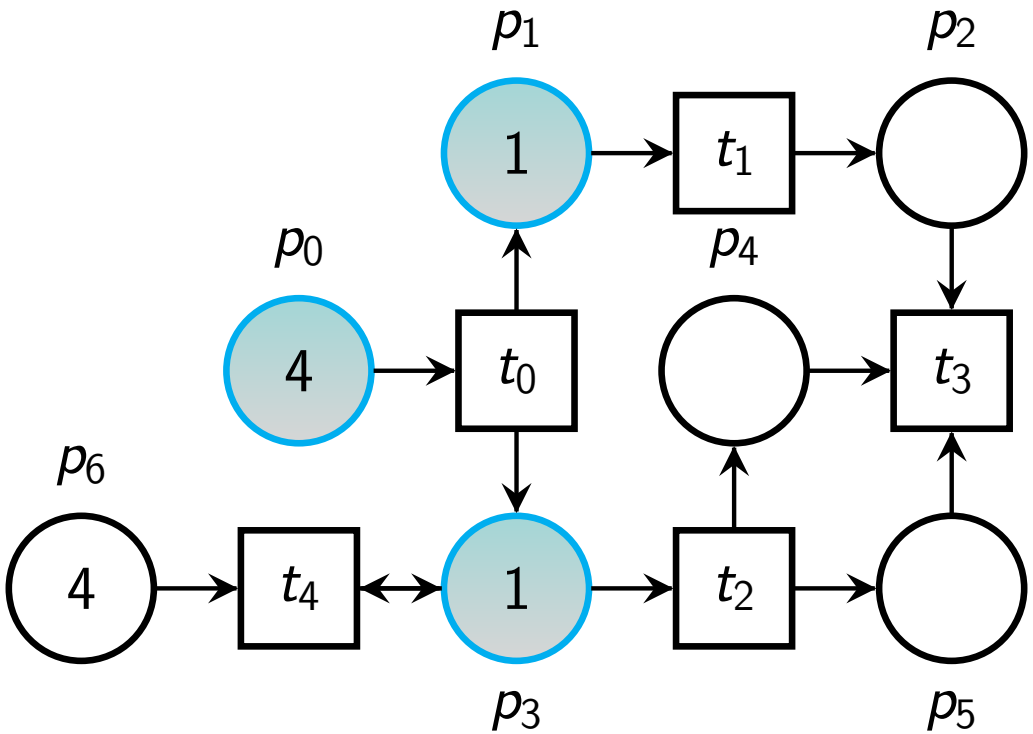
Nicolas Amat, **Silvano Dal Zilio**

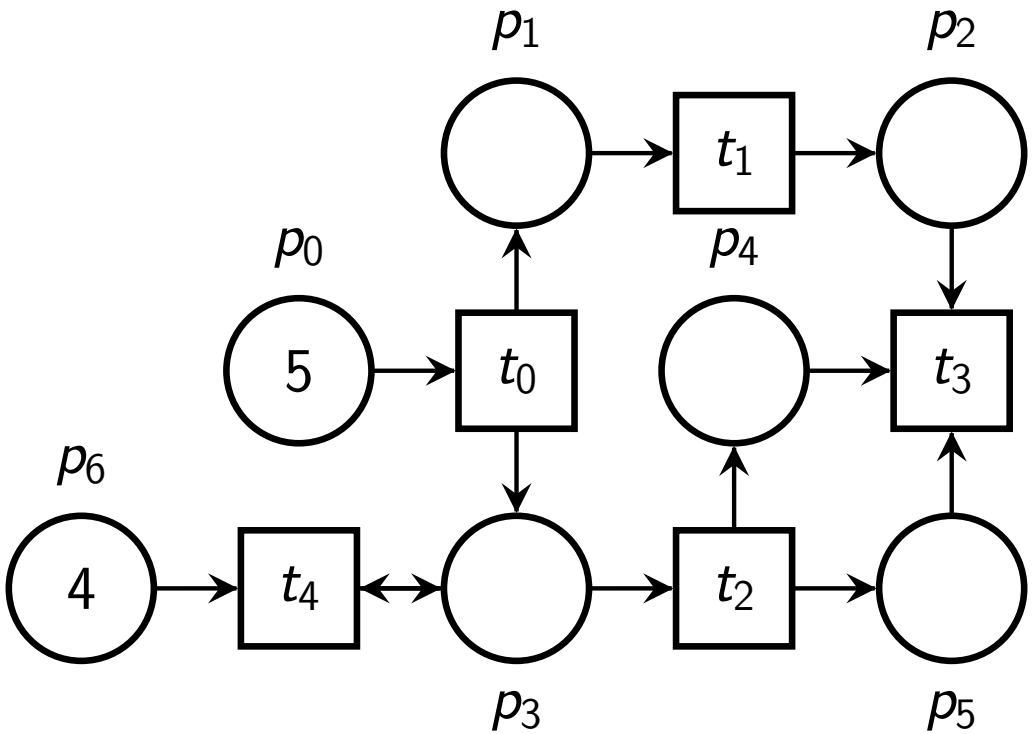
LAAS-CNRS

FM, March 9 2023

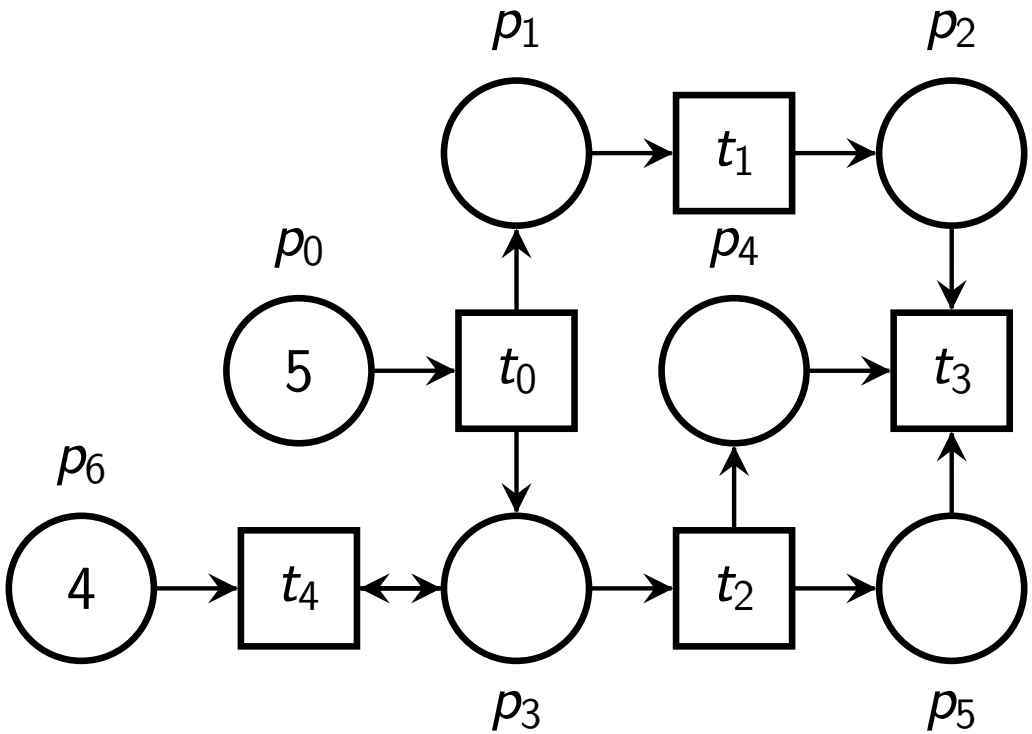

p_1
p_2
t_1
p_0
p_4
5
t_0
t_3
p_6
4
t_4
t_2
p_3
p_5


p_1
p_2
t_1
p_0
p_4
5
t_0
t_3
p_6
4
t_4
t_2
p_3
p_5


p_1
p_2
1
t_1
p_0
p_4
4
t_0
t_3
p_6
4
t_4
1
t_2
p_3
p_5

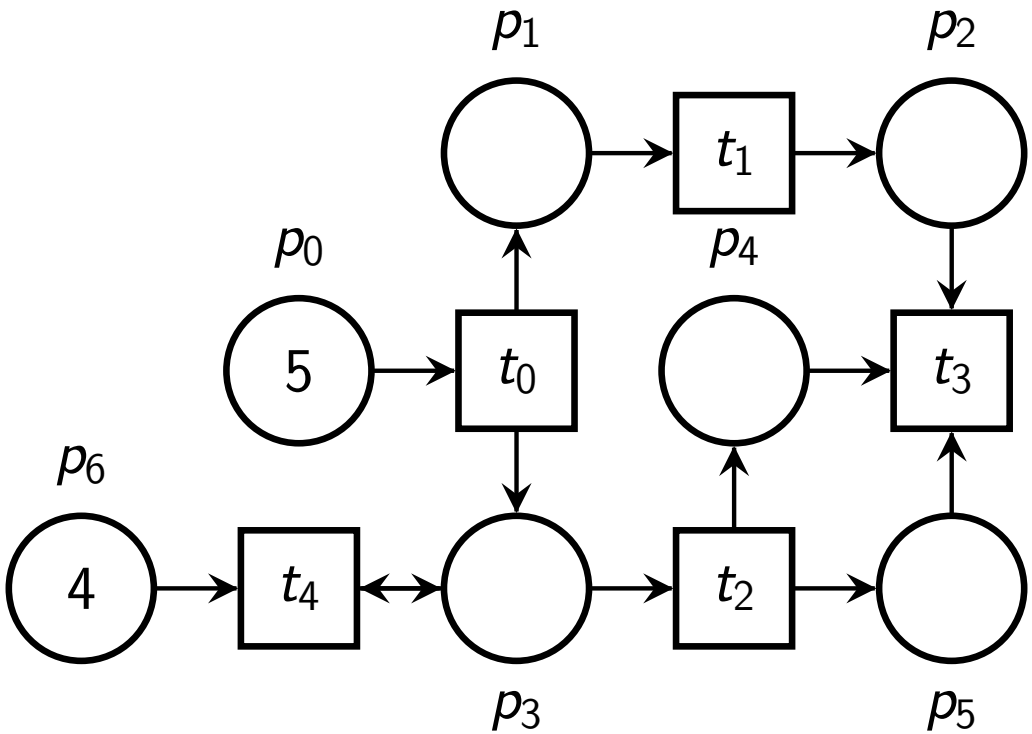


Is $(p_1 \geqslant 1) \wedge (p_6 \leqslant 2)$ **reachable** from the initial marking?



Is $(p_1 \geqslant 1) \wedge (p_6 \leqslant 2)$ **reachable** from the initial marking?

Is $(p_1 + p_2 \leqslant 5) \wedge (p_4 = p_5)$ an **invariant** for all the reachable markings?



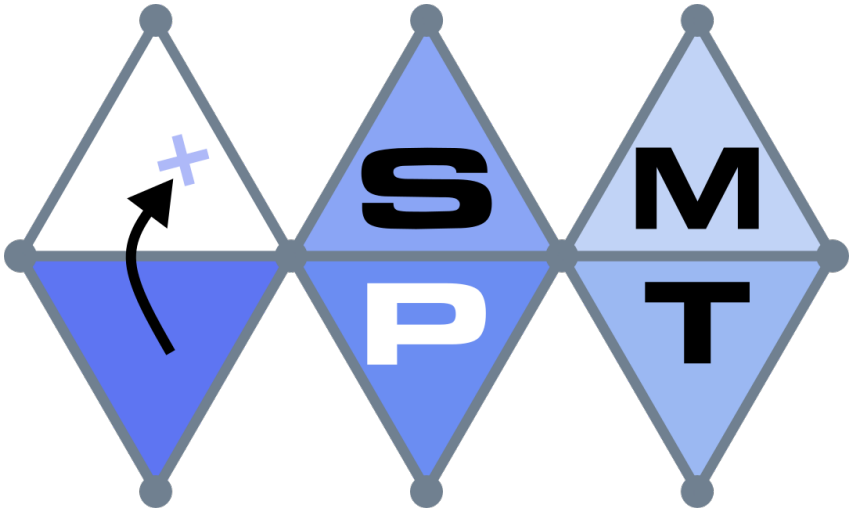
Is $(p_1 \geqslant 1) \wedge (p_6 \leqslant 2)$ **reachable** from the initial marking?

Is $(p_1 + p_2 \leqslant 5) \wedge (p_4 = p_5)$ an **invariant** for all the reachable markings?

Good fit with Quantifier-Free Linear Integer Arithmetic!
(and more generally Presburger arithmetic)

What is SMPT?

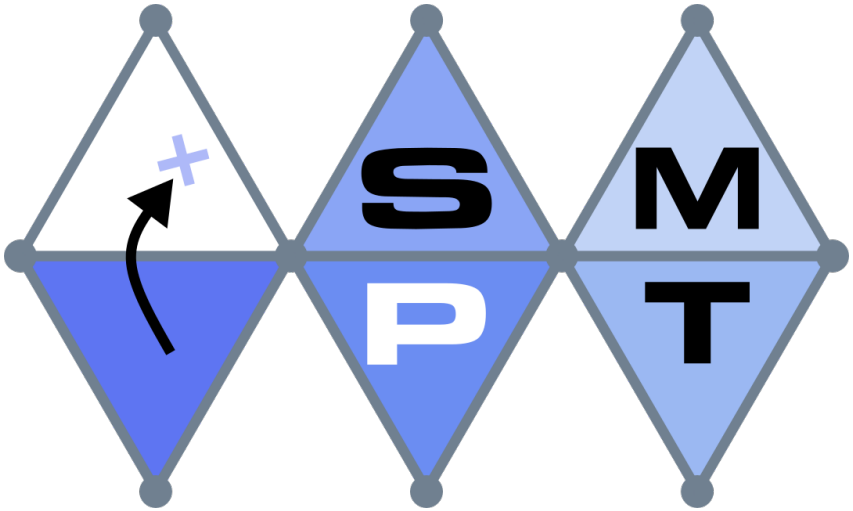
Introduction



- A Petri nets model-checker for **reachability problems**

What is SMPT?

Introduction



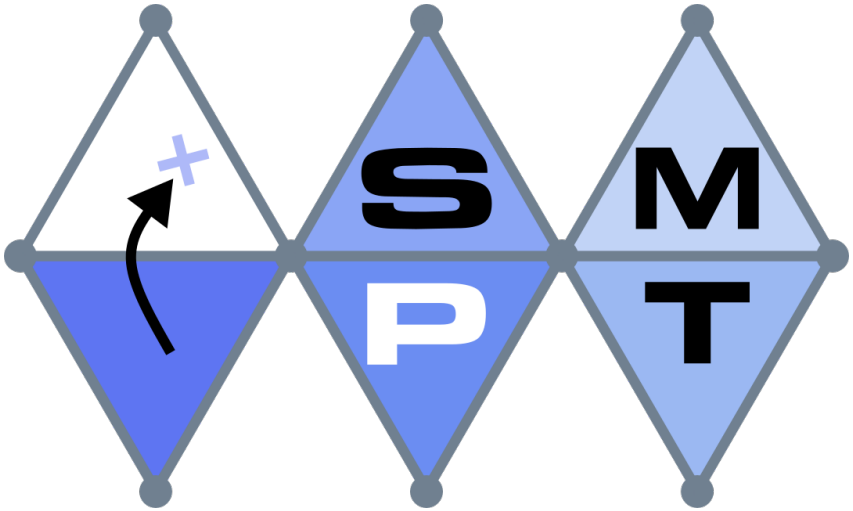
- A Petri nets model-checker for **reachability problems**

  No restriction on the marking of places or the weight of arcs

  Can handle **unbounded nets**!

What is SMPT?

Introduction



- A Petri nets model-checker for **reachability problems**

  No restriction on the marking of places or the weight of arcs

  Can handle **unbounded nets**!

- Support a generalized notion of **reachability properties** (boolean combination of linear constraints between place)

  Includes deadlock detection, quasi-liveness, reachability, etc.

Why the tool can interest you?

- Lots of use-cases ("assembly" of concurrent systems):

  Verification of **concurrent** systems: biological, business processes, ..

  Verification of **software** systems

  Analysis of **infinite state** systems

Why the tool can interest you?

Introduction

- Lots of use-cases ("assembly" of concurrent systems):

  Verification of **concurrent** systems: biological, business processes, ..

  Verification of **software** systems

  Analysis of **infinite state** systems

- **Decidable** problem [Mayr'1981] [Kosaraju'1982]

Why the tool can interest you?

Introduction

- Lots of use-cases ("assembly" of concurrent systems):

  Verification of **concurrent** systems: biological, business processes, ..

  Verification of **software** systems

  Analysis of **infinite state** systems

- **Decidable** problem [Mayr'1981] [Kosaraju'1982]

- Timely subject

  Recent **tools** [Dixon et al.'2020] [Blondin et al.'2021]
  Ackermann-complete [Czerwiński et al.'2021] [Leroux'2021]

Why the tool can interest you?

Introduction

- Lots of use-cases ("assembly" of concurrent systems):

  Verification of **concurrent** systems: biological, business processes, ..

  Verification of **software** systems

  Analysis of **infinite state** systems

- **Decidable** problem [Mayr'1981] [Kosaraju'1982]

- Timely subject

  Recent **tools** [Dixon et al.'2020] [Blondin et al.'2021]
  Ackermann-complete [Czerwiński et al.'2021] [Leroux'2021]

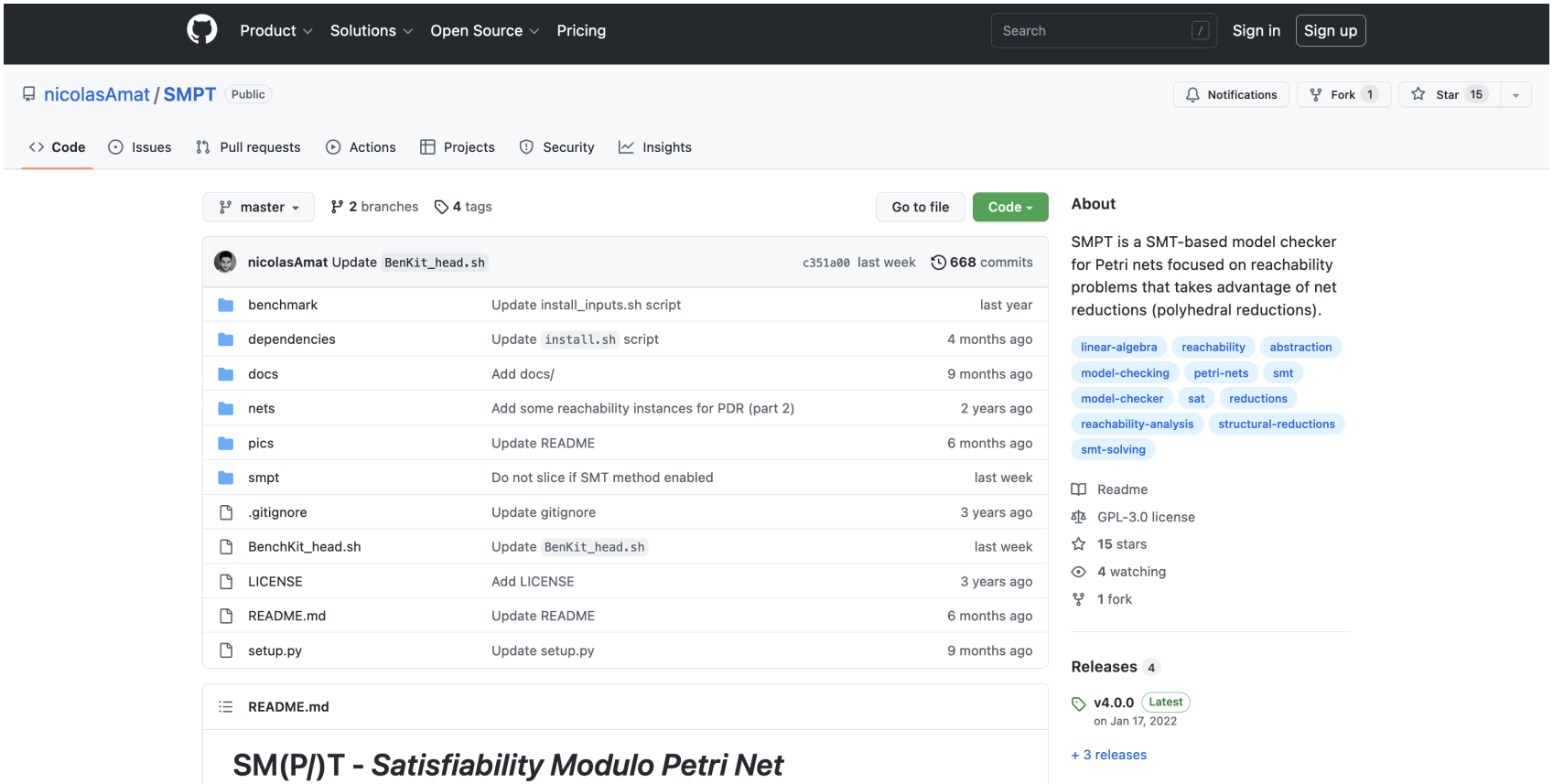
- **Portfolio** tool participating in the Model Checking Contest

Where can I find the tool?

Introduction

Freely available under the GPLv3 license

`github.com/nicolasAmat/SMPT`



Outline

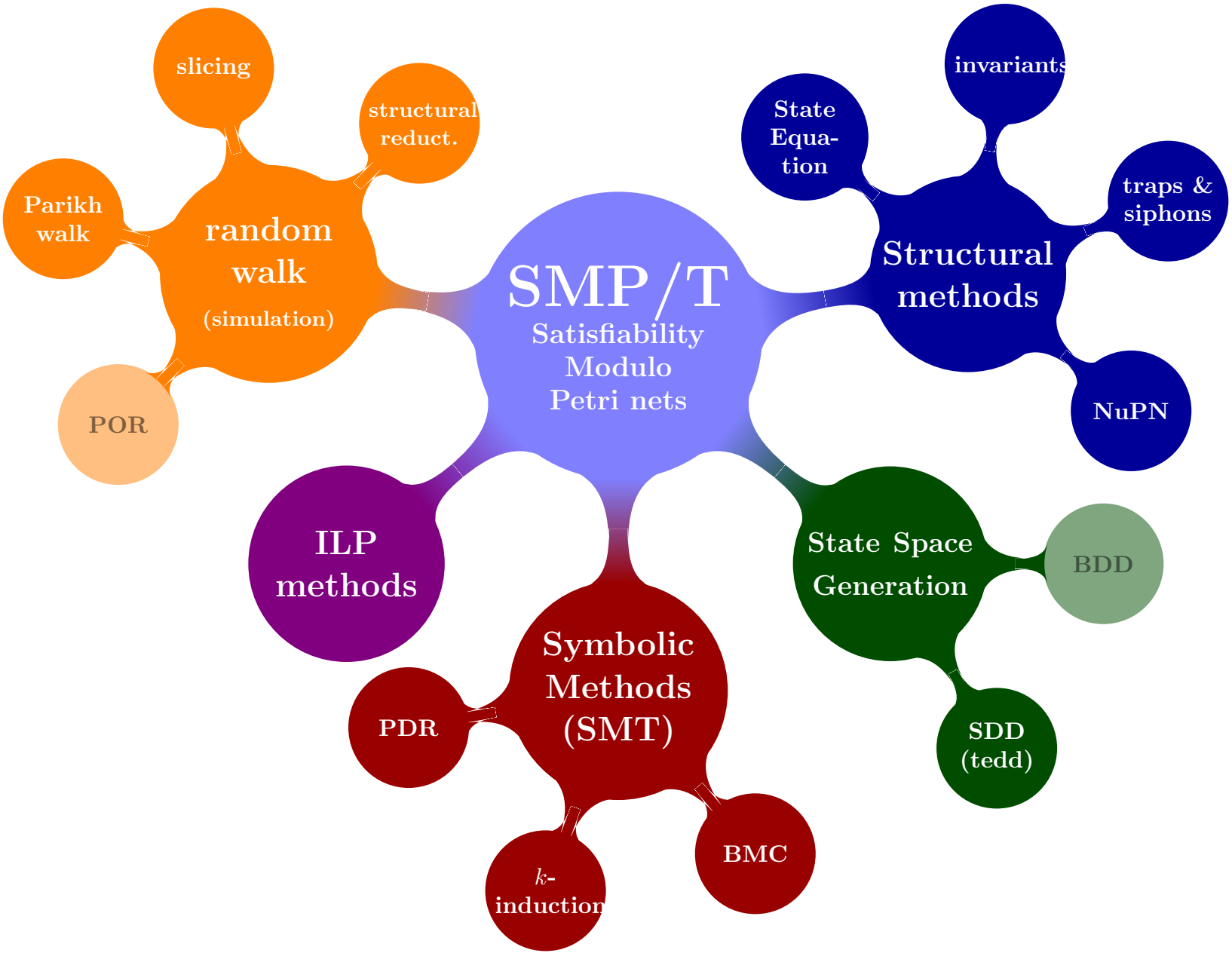
1. Overview
2. Distinctive features
3. Live demonstration
4. Model Checking Contest retrospective
5. What's next?

Overview

- **Python** project, **fully typed** using `mypy`
- Mainly based on **SMT methods** (SMT-LIB format)
- Many **tracing** and **debugging** options
- Packaged into libraries, and provide abstract classes to help with future **extensions**

Portfolio of methods

Overview



Input and output formats

Overview

Nets:

- Petri Net Markup Language (**PNML**) for nets
- Support **colored** Petri nets (high-level nets)

Input and output formats

Overview

Nets:

- Petri Net Markup Language (**PNML**) for nets
- Support **colored** Petri nets (high-level nets)

Formulas:

- **MCC** property language (XML format encoding)
- Textual format
- Some specific options (deadlock, quasi-liveness, etc.)

Input and output formats

Overview

Nets:

- Petri Net Markup Language (**PNML**) for nets
- Support **colored** Petri nets (high-level nets)

Formulas:

- **MCC** property language (XML format encoding)
- Textual format
- Some specific options (deadlock, quasi-liveness, etc.)

Output is MCC compliant.

SMPT can be interchanged with other participating tools `ITS-Tools`, `Tapaal`, `LoLA`, `GreatSPN`, ...

Outline

1. Overview
2. Distinctive features
3. Live demonstration
4. Model Checking Contest retrospective
5. What's next?

"There exist checkable certificates of non-reachability in the Presburger arithmetic" [Leroux, 2009]

Definition (Certificate of Invariance)

A predicate R is a Certificate of Invariance (CI) for F if:

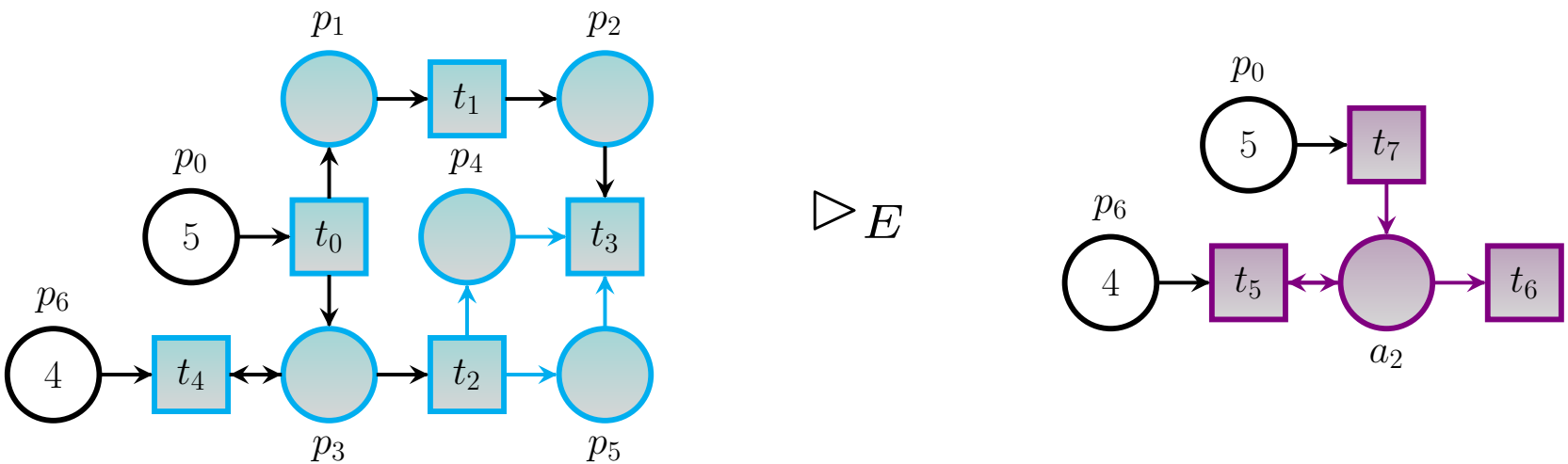
- R inductive
- R entails F: $R(\boldsymbol{p}) \wedge \neg F(\boldsymbol{p})$ unsatisfiable

$$\underbrace{(N, m_0)}_{\text{initial net}} \quad \underbrace{\rhd_E}_{\text{linear system}} \quad \underbrace{(N', m'_0)}_{\text{reduced net}}$$

Correspondence between the set of reachable markings "modulo" the linear equations E

$$\underbrace{(N, m_0)}_{\text{initial net}} \quad \underbrace{\rhd_E}_{\text{linear system}} \quad \underbrace{(N', m'_0)}_{\text{reduced net}}$$

Correspondence between the set of reachable markings "modulo" the linear equations E



$$E = (p_5 = p_4) \wedge (a_1 = p_2 + p_1) \wedge (a_2 = p_4 + p_3) \wedge (a_1 = a_2)$$

Outline

1. Overview
2. Distinctive features
3. Live demonstration
4. Model Checking Contest retrospective
5. What's next?

Outline

1. Overview
2. Distinctive features
3. Live demonstration
4. Model Checking Contest retrospective
5. What's next?

Model Checking Contest retrospective

We are grateful to the Model Checking Contest (MCC):

- Annual **competition** for model-checking tools
- Huge **benchmark** (1 618 models and 51 744 queries)
- Compare approaches and improve **reliability**!

Model Checking Contest retrospective

We are grateful to the Model Checking Contest (MCC):

- Annual **competition** for model-checking tools
- Huge **benchmark** (1 618 models and 51 744 queries)
- Compare approaches and improve **reliability**!

Bronze medal and 100% confidence award! (high reliability)

Outline

1. Overview
2. Distinctive features
3. Live demonstration
4. Model Checking Contest retrospective
5. What's next?

What's next?

Continue to explore the relation of Petri nets with **Presburger arithmetic**:

- **Quantifier elimination** in a specific fragment of Presburger (make better use of polyhedral reductions)

- Extract part of nets that are **Presburger-definable** (*flattable*)

  Automated procedure to prove polyhedral reductions

  Improve our use of the "state equation" method (identifying new classes of Petri nets for which all **potentially reachable markings** are indeed **reachable**)

Thank you for your attention!

Have a look at the GitHub repository :)

Any questions?