

# SMPT: A Testbed for Reachability Methods in Generalized Petri Nets

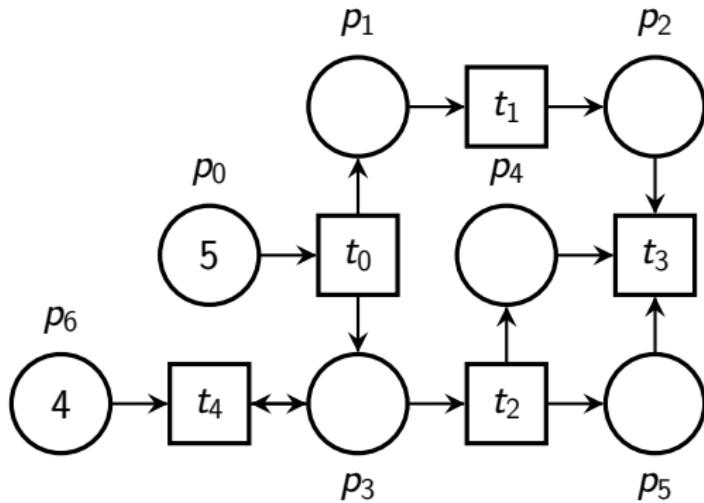
**Nicolas Amat, Silvano Dal Zilio**

LAAS-CNRS

FM, March 9 2023

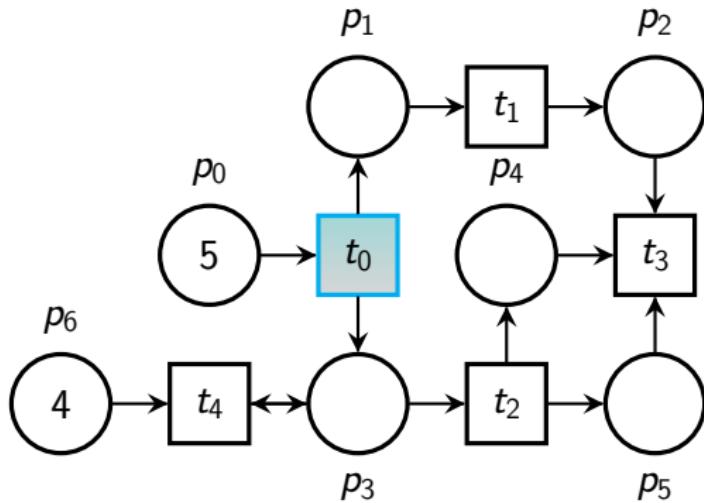
# Petri nets & Reachability problems

## Introduction



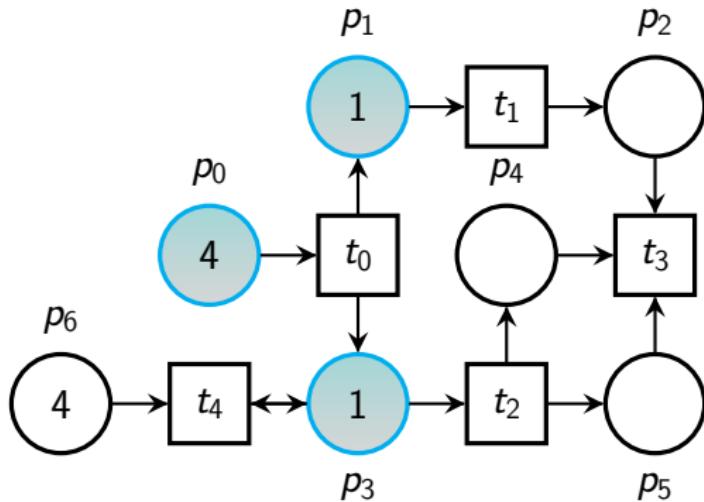
# Petri nets & Reachability problems

## Introduction



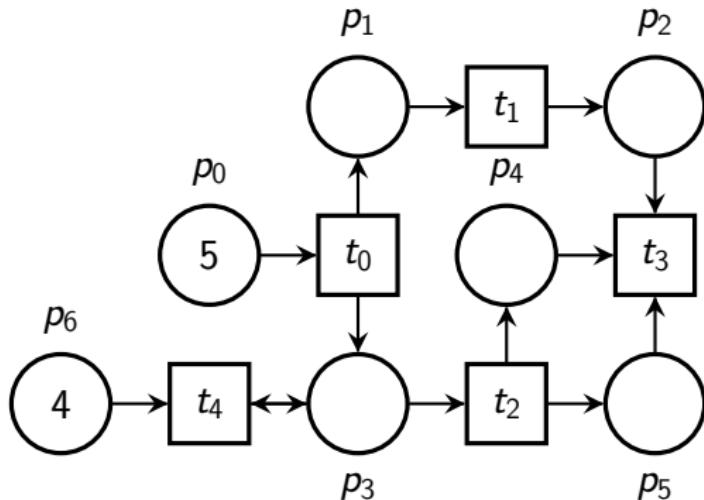
# Petri nets & Reachability problems

## Introduction



# Petri nets & Reachability problems

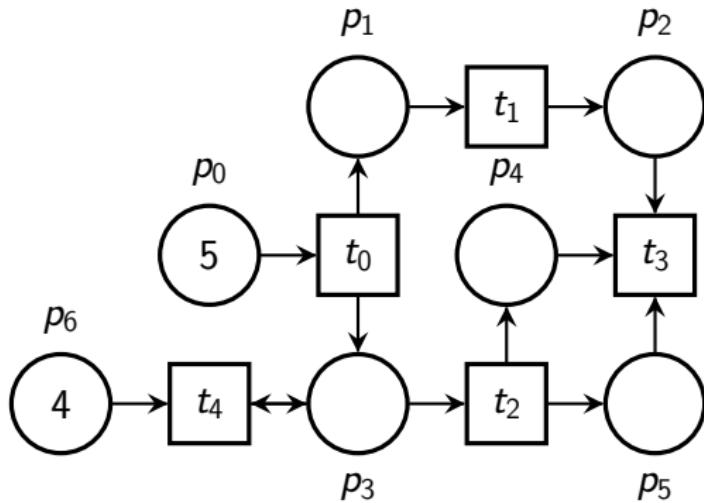
## Introduction



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

# Petri nets & Reachability problems

## Introduction

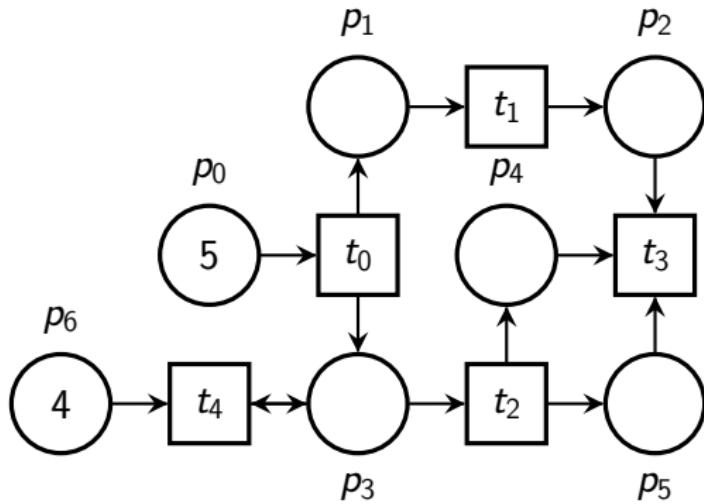


Is  $(p_1 \geq 1) \wedge (p_6 \leq 2)$  **reachable** from the initial marking?

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

# Petri nets & Reachability problems

## Introduction



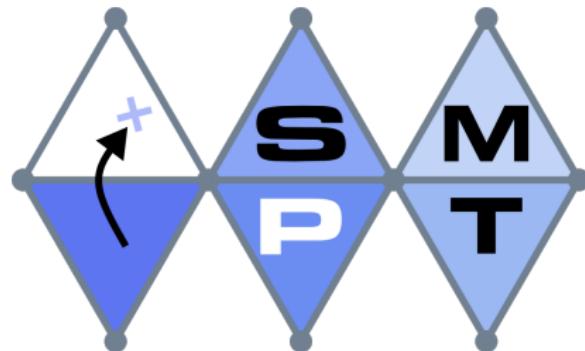
Is  $(p_1 \geq 1) \wedge (p_6 \leq 2)$  **reachable** from the initial marking?

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

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

# What is SMPT?

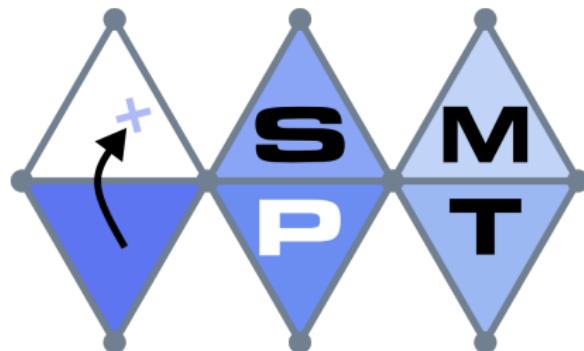
## Introduction



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

# What is SMPT?

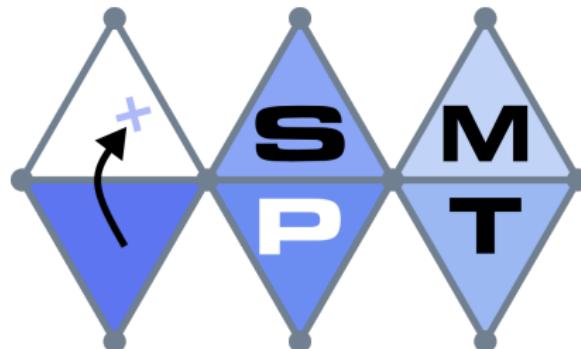
## Introduction



- 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?

## 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

# 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`

The screenshot shows the GitHub repository page for `nicolasAmat/SMPT`. The top navigation bar includes links for Product, Solutions, Open Source, Pricing, Search, Sign in, and Sign up. Below the header, there are buttons for Code, Issues, Pull requests, Actions, Projects, Security, and Insights. The main content area displays a list of recent commits from the `master` branch. Each commit includes the author, message, date, and number of commits. To the right of the commit list is a detailed description of the project, its tags, and its releases.

**Commits**

- nicolasAmat Update BenKit\_head.sh (c351a08 last week) 668 commits
- benchmark Update install\_inputs.sh script (last year)
- dependencies Update install.sh script (4 months ago)
- docs Add docs/ (9 months ago)
- nets Add some reachability instances for PDR (part 2) (2 years ago)
- pics Update README (6 months ago)
- smpt Do not slice if SMT method enabled (last week)
- .gitignore Update gitignore (3 years ago)
- BenchKit\_head.sh Update BenKit\_head.sh (last week)
- LICENSE Add LICENSE (3 years ago)
- README.md Update README (6 months ago)
- setup.py Update setup.py (9 months ago)

**About**

SMPT is a SMT-based model checker for Petri nets focused on reachability problems that takes advantage of net reductions (polyhedral reductions).

Tags: linear-algebra, reachability, abstraction, model-checking, petri-nets, smt, model-checker, sat, reductions, reachability-analysis, structural-reductions, smt-solving

Readme: Readme

License: GPL-3.0 license

Statistics: 15 stars, 4 watching, 1 fork

**Releases** 4

- v4.0.0 (Latest) on Jan 17, 2022

+ 3 releases

**SM(P)T - Satisfiability Modulo Petri Net**

## 1 Overview

## 2 Distinctive features

## 3 Live demonstration

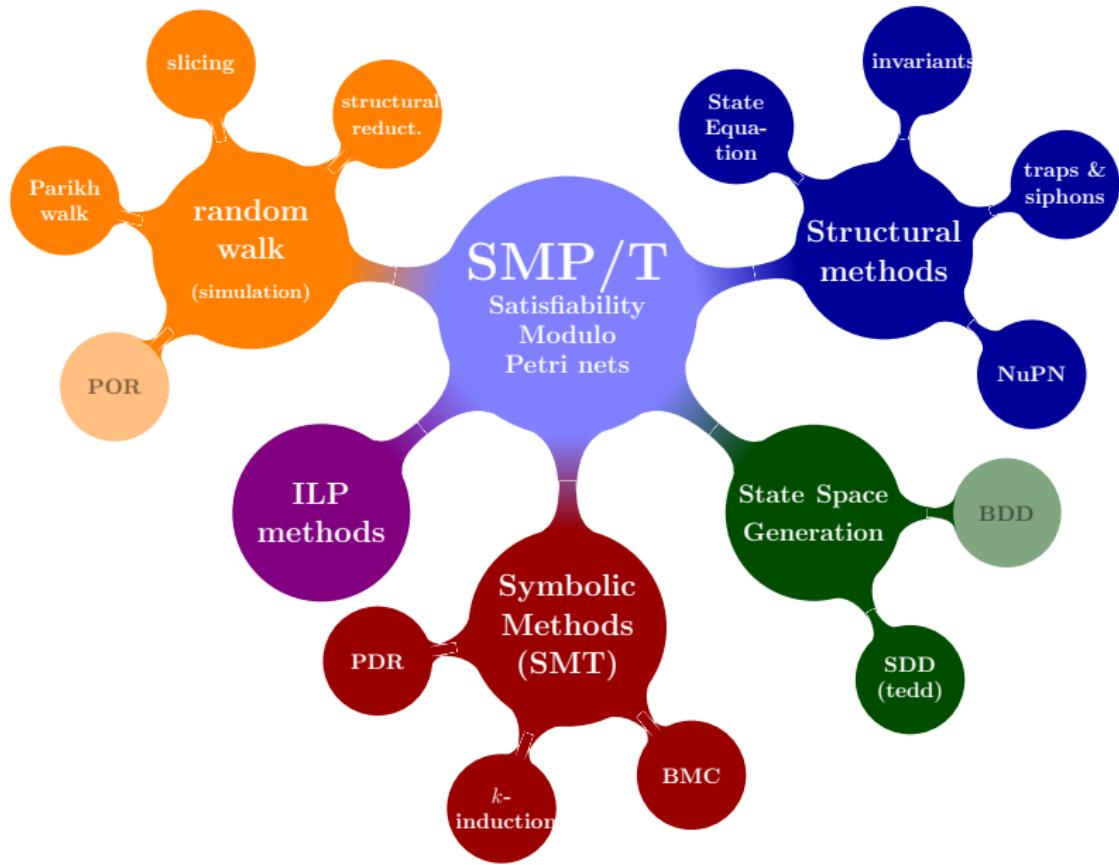
## 4 Model Checking Contest retrospective

## 5 What's next?

- 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?

## Distinctive features

*“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  $F$ :  $R(\mathbf{p}) \wedge \neg F(\mathbf{p})$  unsatisfiable

## Distinctive features

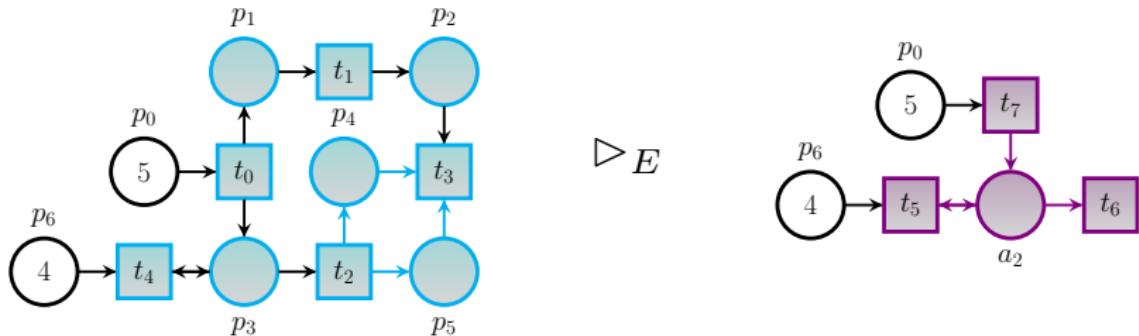
$$\underbrace{(N, m_0)}_{\text{initial net}} \quad \underbrace{\triangleright_E}_{\text{linear system}} \quad \underbrace{(N', m'_0)}_{\text{reduced net}}$$

Correspondence between the set of reachable markings  
“modulo” the linear equations  $E$

## Distinctive features

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

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?