# Accelerating the Computation of Dead and Concurrent Places using Reductions

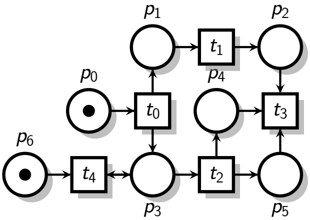
*This is a model checking paper where no transitions are fired!*

**Nicolas Amat**, **Silvano Dal Zilio**, **Didier Le Botlan**

LAAS-CNRS, Université de Toulouse, CNRS, INSA, Toulouse, France

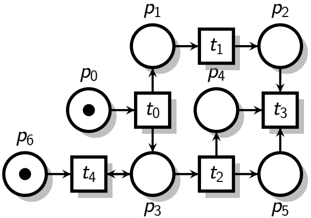
SPIN 2021, July 12 2021

A pair of places $(p, q)$ are concurrent, denoted $p \| q$, if there is $m$ in $R(N, m_0)$ such that both $m(p) > 0$ and $m(q) > 0$.




A *safe* Petri net

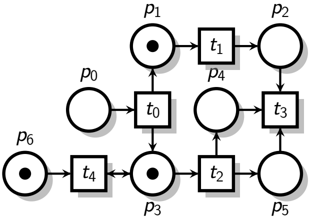
A pair of places $(p, q)$ are concurrent, denoted $p \| q$, if there is $m$ in $R(N, m_0)$ such that both $m(p) > 0$ and $m(q) > 0$.




A *safe* Petri net

$p_0 \| p_6$

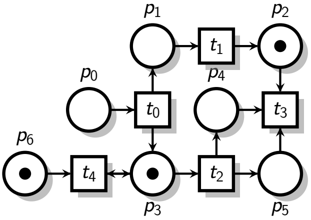
A pair of places $(p, q)$ are concurrent, denoted $p \| q$, if there is $m$ in $R(N, m_0)$ such that both $m(p) > 0$ and $m(q) > 0$.




A *safe* Petri net

$p_0 \| p_6$
$p_1 \| p_3$
$p_1 \| p_6$
$p_3 \| p_6$

A pair of places $(p, q)$ are concurrent, denoted $p \| q$, if there is $m$ in $R(N, m_0)$ such that both $m(p) > 0$ and $m(q) > 0$.



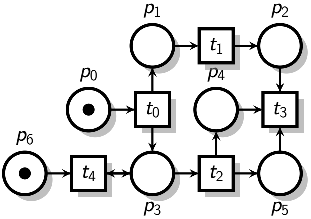

A *safe* Petri net

$p_0 \| p_6$
$p_1 \| p_3$
$p_1 \| p_6$
$p_3 \| p_6$
$p_2 \| p_3$
$p_2 \| p_6$

# Concurrent Places Problem

## Introduction

A pair of places $(p, q)$ are concurrent, denoted $p \| q$, if there is $m$ in $R(N, m_0)$ such that both $m(p) > 0$ and $m(q) > 0$.



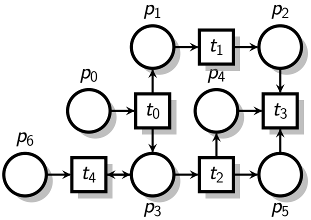

A *safe* Petri net

$$C = \begin{array}{cc} & \begin{array}{ccccccc} p_0 & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \end{array} \\ \begin{array}{c} p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{array} & \begin{pmatrix} 1 & & & & & & \\ 0 & 1 & & & & & \\ 0 & 0 & 1 & & & & \\ 0 & 1 & 1 & 1 & & & \\ 0 & 1 & 1 & 0 & 1 & & \\ 0 & 1 & 1 & 0 & 1 & 1 & \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

# Concurrent Places Problem



A pair of places $(p, q)$ are concurrent, denoted $p \| q$, if there is $m$ in $R(N, m_0)$ such that both $m(p) > 0$ and $m(q) > 0$.




A *safe* Petri net

$$C = \begin{array}{c} \\ p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{array} \begin{array}{c} \begin{array}{ccccccc} p_0 & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \end{array} \\ \begin{pmatrix} 1 & & & & & & \\ 0 & 1 & & & & & \\ 0 & 0 & 1 & & & & \\ 0 & 1 & 1 & 1 & & & \\ 0 & 1 & 1 & 0 & 1 & & \\ 0 & 1 & 1 & 0 & 1 & 1 & \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

- Used for the decomposition into networks of automata

- Many results based on **linear algebra** and linear programming techniques [Murata, 1989] [Silva et al., 1996]
  - **Potentially reachable markings**
  - **Place invariants**
  - ...

- Many results based on **linear algebra** and linear programming techniques [Murata, 1989] [Silva et al., 1996]
  - **Potentially reachable markings**
  - **Place invariants**
  - ...
- **Structural reductions** [Berthelot, 1987]

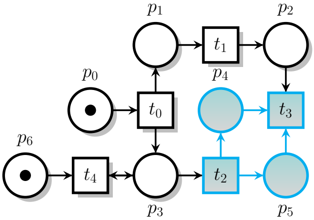
- Many results based on **linear algebra** and linear programming techniques [Murata, 1989] [Silva et al., 1996]
  - **Potentially reachable markings**
  - **Place invariants**
  - ...
- **Structural reductions** [Berthelot, 1987]
- And 30 years after... [Berthomieu et al., 2018]
  **Structural reductions** with **linear equations**

- Many results based on **linear algebra** and linear programming techniques [Murata, 1989] [Silva et al., 1996]
  - **Potentially reachable markings**
  - **Place invariants**
  - ...
- **Structural reductions** [Berthelot, 1987]
- And 30 years after... [Berthomieu et al., 2018]
  **Structural reductions** with **linear equations**
- **Combination with SMT** [Amat et al., 2021]

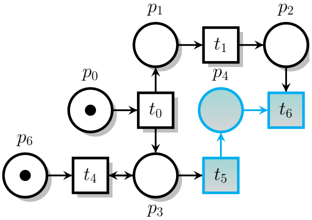
- Many results based on **linear algebra** and linear programming techniques [Murata, 1989] [Silva et al., 1996]
  - **Potentially reachable markings**
  - **Place invariants**
  - ...
- **Structural reductions** [Berthelot, 1987]
- And 30 years after... [Berthomieu et al., 2018]
  **Structural reductions** with **linear equations**
- **Combination with SMT** [Amat et al., 2021]
- **Concurrent Places**: Good testbed for our new approach!

# Petri Nets and Polyhedral Abstraction

**Rule:** RED

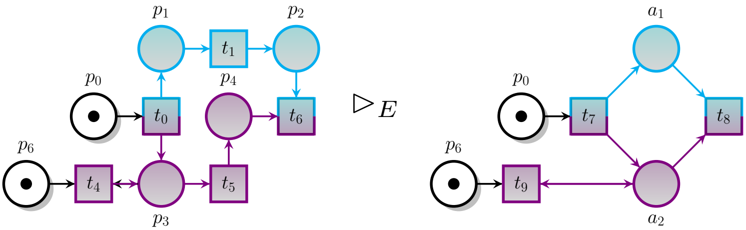



$\triangleright_E$



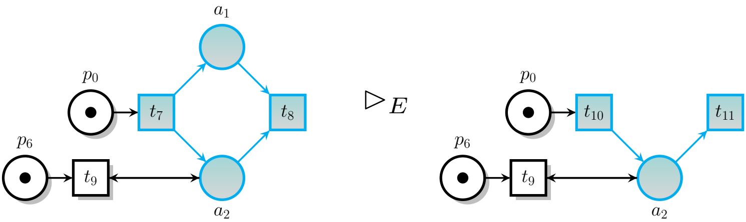

$$E \triangleq \{\ p_5 = p_4$$

**Rule:** CONCAT

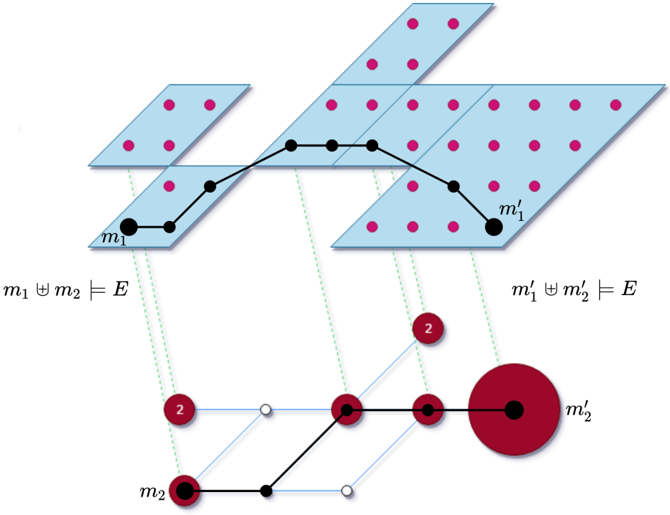



$$E \triangleq \begin{cases} p_5 = p_4 \\ a_1 = p_1 + p_2 \\ a_2 = p_3 + p_4 \end{cases}$$

**Rule:** RED




$$E \triangleq \begin{cases} p_5 = p_4 \\ a_1 = p_1 + p_2 \\ a_2 = p_3 + p_4 \\ a_1 = a_2 \end{cases}$$



$m_1$
$m'_1$
$m_1 \uplus m_2 \models E$
$m'_1 \uplus m'_2 \models E$
2
2
$m'_2$
$m_2$

*On the Combination of Polyhedral Abstraction and SMT-based Model Checking for Petri Nets*, N. Amat et al., Petri Nets 2021

## Definition ($E$-abstraction)

$(N_1, m_1) \sqsupseteq_E (N_2, m_2)$ iff

(A1) initial markings are compatible with $E$, meaning $m_1 \uplus m_2 \models E$

(A2) for all observation sequences $\sigma \in \Sigma^\star$ such that $(N_1, m_1) \overset{\sigma}{\Rrightarrow} (N_1, m_1')$

- there is at least one marking $m_2' \in R(N_2, m_2)$ such that $m_1' \uplus m_2' \models E$
- for all markings $m_2'$ we have that $m_1' \uplus m_2' \models E$ implies $(N_2, m_2) \overset{\sigma}{\Rrightarrow} (N_2, m_2')$

*On the Combination of Polyhedral Abstraction and SMT-based Model Checking for Petri Nets*, N. Amat et al., Petri Nets 2021

**Definition ($E$-abstraction)**

$(N_1, m_1) \sqsupseteq_E (N_2, m_2)$ iff

(A1) initial markings are compatible with $E$, meaning $m_1 \uplus m_2 \models E$

(A2) for all observation sequences $\sigma \in \Sigma^\star$ such that $(N_1, m_1) \overset{\sigma}{\Rightarrow} (N_1, m_1')$

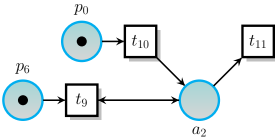
- there is at least one marking $m_2' \in R(N_2, m_2)$ such that $m_1' \uplus m_2' \models E$
- for all markings $m_2'$ we have that $m_1' \uplus m_2' \models E$ implies $(N_2, m_2) \overset{\sigma}{\Rightarrow} (N_2, m_2')$

**$E$-abstraction equivalence**

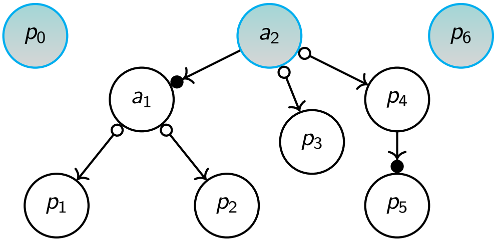
$(N_1, m_1) \rhd_E (N_2, m_2)$ iff $(N_1, m_1) \sqsupseteq_E (N_2, m_2)$ and $(N_2, m_2) \sqsupseteq_E (N_1, m_1)$

# Token Flow Graphs

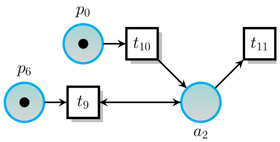
$$E \triangleq \begin{cases} p_5 = p_4 \\ a_1 = p_1 + p_2 \\ a_2 = p_3 + p_4 \\ a_1 = a_2 \end{cases}$$



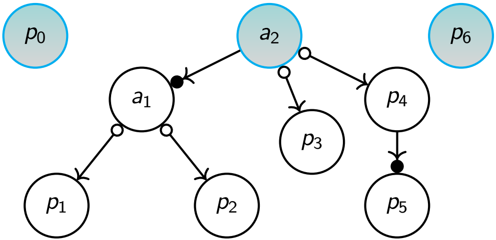

$(N_2, m_2)$



$$E \triangleq \begin{cases} p_5 = p_4 \\ a_1 = p_1 + p_2 \\ a_2 = p_3 + p_4 \\ a_1 = a_2 \end{cases}$$

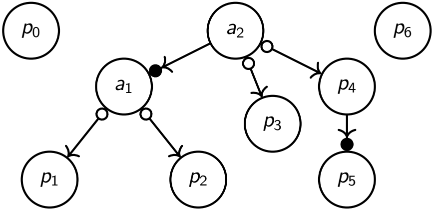



$(N_2, m_2)$

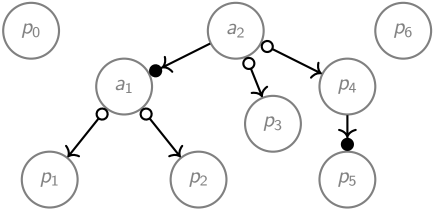



- *Remark*: Roots are places of the reduced net $(N_2, m_2)$

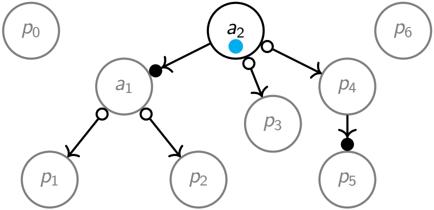
# Configuration of a TFG
## Token Flow Graphs



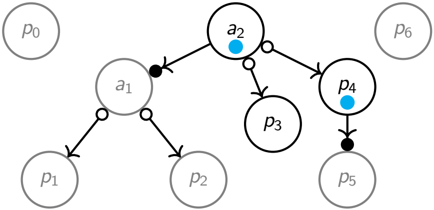

- Configuration $c$: partial function from $V$ to $\mathbb{N}$




- Configuration $c$: partial function from $V$ to $\mathbb{N}$
  - $c(v) = \perp$ when $c$ not defined on $v$
  - *Total* if for all $v$ in $V$ we have $c(v) \neq \perp$



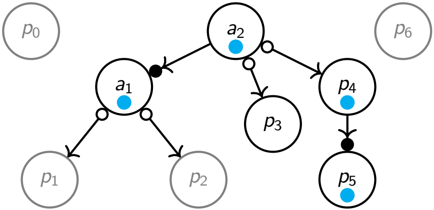

- Configuration $c$: partial function from $V$ to $\mathbb{N}$
  - $c(v) = \perp$ when $c$ not defined on $v$
  - *Total* if for all $v$ in $V$ we have $c(v) \neq \perp$
- **Well-defined** configuration




- Configuration $c$: partial function from $V$ to $\mathbb{N}$
  - $c(v) = \bot$ when $c$ not defined on $v$
  - *Total* if for all $v$ in $V$ we have $c(v) \neq \bot$
- **Well-defined** configuration
  - *Agglomeration*: if $c(v) \neq \bot$ then $c(v) = \sum_{w \mid v \circ\!\!\rightarrow w} c(w)$

# Configuration of a TFG

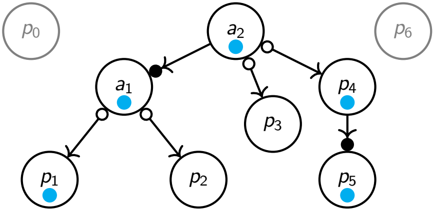
## Token Flow Graphs




- Configuration $c$: partial function from $V$ to $\mathbb{N}$
  - $c(v) = \perp$ when $c$ not defined on $v$
  - *Total* if for all $v$ in $V$ we have $c(v) \neq \perp$
- **Well-defined** configuration
  - *Agglomeration*: if $c(v) \neq \perp$ then $c(v) = \sum_{w | v \circ\!\!\rightarrow w} c(w)$
  - *Redundancy*: if $c(w) \neq \perp$ then $c(w) = \sum_{v | v \rightarrow\!\!\bullet w} c(v)$

# Configuration of a TFG

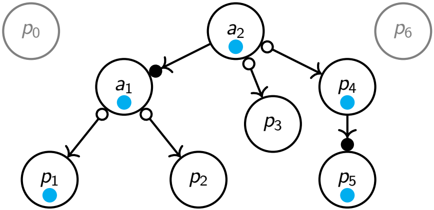
## Token Flow Graphs




- Configuration $c$: partial function from $V$ to $\mathbb{N}$
  - $c(v) = \perp$ when $c$ not defined on $v$
  - *Total* if for all $v$ in $V$ we have $c(v) \neq \perp$
- **Well-defined** configuration
  - *Agglomeration*: if $c(v) \neq \perp$ then $c(v) = \sum_{w \mid v \circ\!\!\rightarrow w} c(w)$
  - *Redundancy*: if $c(w) \neq \perp$ then $c(w) = \sum_{v \mid v \rightarrow\!\!\bullet w} c(v)$
  - *Bottom*: if $v \rightarrow v'$ then $c(v) = \perp$ if and only if $c(v') = \perp$

# Configuration of a TFG






- Configuration $c$: partial function from $V$ to $\mathbb{N}$
  - $c(v) = \bot$ when $c$ not defined on $v$
  - *Total* if for all $v$ in $V$ we have $c(v) \neq \bot$

- **Well-defined** configuration
  - *Agglomeration*: if $c(v) \neq \bot$ then $c(v) = \sum_{w \mid v \circ\!\!\rightarrow w} c(w)$
  - *Redundancy*: if $c(w) \neq \bot$ then $c(w) = \sum_{v \mid v \rightarrow\!\!\bullet w} c(v)$
  - *Bottom*: if $v \to v'$ then $c(v) = \bot$ if and only if $c(v') = \bot$

- If $c$ total then $c_{|N_1}$ (resp. $c_{|N_2}$) is marking of $N_1$ (resp. $N_2$)

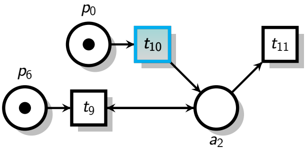
$[\![E]\!]$ is a well-formed TFG for the equivalence $(N_1, m_1) \rhd_E (N_2, m_2)$

**Theorem (Reachable marking extension)**

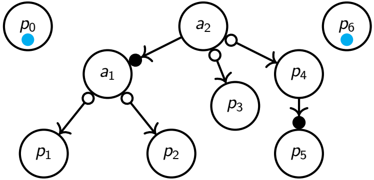
*If $m$ is a marking in $R(N_1, m_1)$ (resp. $R(N_2, m_2)$) then there exists a total, well-defined configuration $c$ of $[\![E]\!]$ such that $c_{|N_1} = m$ (resp. $c_{|N_2} = m$).*

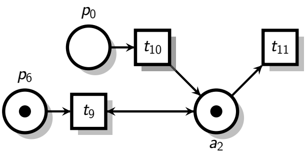
**Theorem (Reachability equivalence)**

*Given a total, well-defined configuration $c$ of $[\![E]\!]$, if marking $c_{|N_1}$ is reachable in $(N_1, m_1)$ then $c_{|N_2}$ is reachable in $(N_2, m_2)$.*

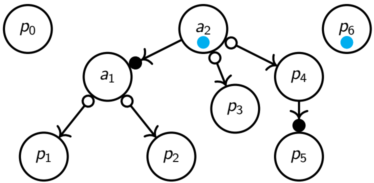


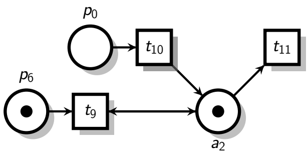

$(N_2, m_2)$



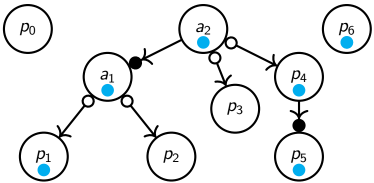



$(N_2, m_2')$

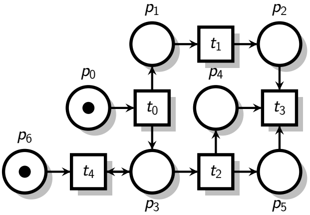




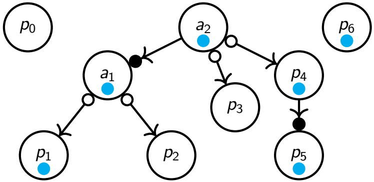

$(N_2, m'_2)$



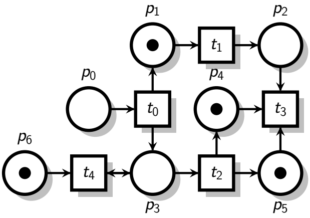

- Total and well-defined configuration $c$ such that $c_{|N_2} = m'_2$



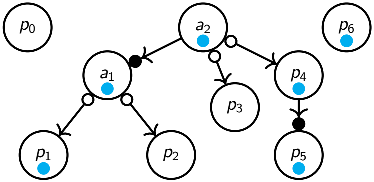

$(N_1, m_1)$




- Total and well-defined configuration $c$ such that $c_{|N_2} = m'_2$




$(N_1, c_{|N_1})$

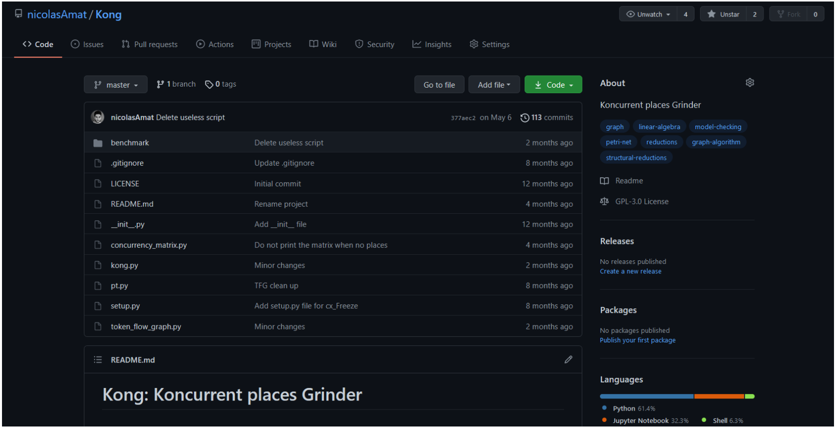


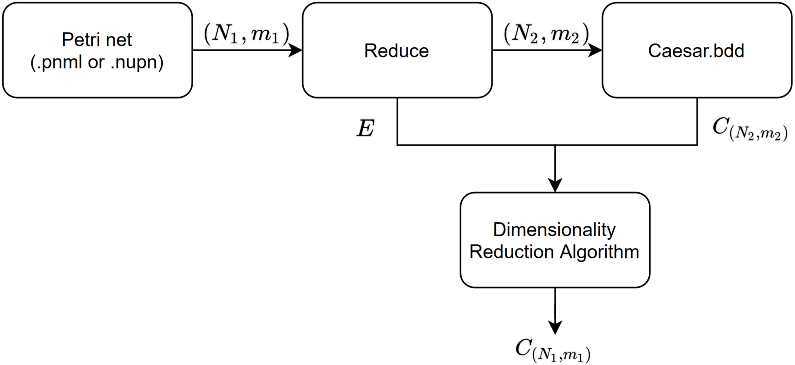

- Total and well-defined configuration $c$ such that $c_{|N_2} = m'_2$
- $c_{|N_2}$ reachable in $(N_2, m_2)$ implies $c_{|N_1}$ reachable in $(N_1, m_1)$

# Kong: Koncurrent Places Grinder

# Tool Overview

## Kong

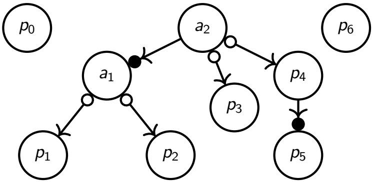



Petri net
(.pnml or .nupn)
$(N_1, m_1)$
Reduce
$(N_2, m_2)$
Caesar.bdd
$E$
$C_{(N_2,m_2)}$
Dimensionality
Reduction Algorithm
$C_{(N_1,m_1)}$

# Dimensionality Reduction Algorithm

## Kong

$$C_{(N_2,m_2)} = \begin{array}{c} \\ a_2 \\ p_0 \\ p_6 \end{array} \begin{array}{c} \begin{array}{ccc} a_2 & p_0 & p_6 \end{array} \\ \begin{pmatrix} 1 & & \\ 0 & 1 & \\ 1 & 1 & 1 \end{pmatrix} \end{array}$$

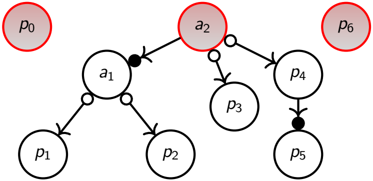



$$C_{(N_1,m_1)} = \begin{array}{c} \\ p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{array} \begin{array}{c} \begin{array}{ccccccc} p_0 & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \end{array} \\ \begin{pmatrix} ? & & & & & & \\ ? & ? & & & & & \\ ? & ? & ? & & & & \\ ? & ? & ? & ? & & & \\ ? & ? & ? & ? & ? & & \\ ? & ? & ? & ? & ? & ? & \\ ? & ? & ? & ? & ? & ? & ? \end{pmatrix} \end{array}$$

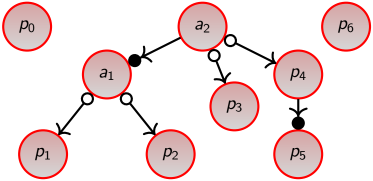
# Dimensionality Reduction Algorithm

## Kong

$$C_{(N_2, m_2)} = \begin{array}{c c} & \begin{array}{ccc} a_2 & p_0 & p_6 \end{array} \\ \begin{array}{c} a_2 \\ p_0 \\ p_6 \end{array} & \begin{pmatrix} \color{red}{1} & & \\ 0 & \color{red}{1} & \\ 1 & 1 & \color{red}{1} \end{pmatrix} \end{array}$$



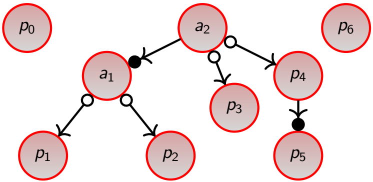
$$C_{(N_2,m_2)} = \begin{array}{c} \\ a_2 \\ p_0 \\ p_6 \end{array} \begin{array}{c} \begin{array}{ccc} a_2 & p_0 & p_6 \end{array} \\ \begin{pmatrix} 1 & & \\ 0 & 1 & \\ 1 & 1 & 1 \end{pmatrix} \end{array}$$




**Lemma (Propagation of Live Nodes)**

*If $v \| v$ and $v \rightarrow^* w$ then $w \| w$*

$$C_{(N_2,m_2)} = \begin{array}{c c} & \begin{array}{ccc} a_2 & p_0 & p_6 \end{array} \\ \begin{array}{c} a_2 \\ p_0 \\ p_6 \end{array} & \begin{pmatrix} 1 & & \\ 0 & 1 & \\ 1 & 1 & 1 \end{pmatrix} \end{array}$$

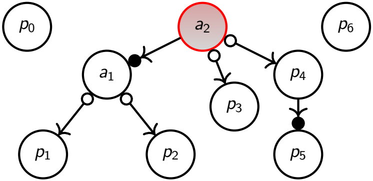



$$C_{(N_1,m_1)} = \begin{array}{c c} & \begin{array}{ccccccc} p_0 & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \end{array} \\ \begin{array}{c} p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{array} & \begin{pmatrix} 1 & & & & & & \\ ? & 1 & & & & & \\ ? & ? & 1 & & & & \\ ? & ? & ? & 1 & & & \\ ? & ? & ? & ? & 1 & & \\ ? & ? & ? & ? & ? & 1 & \\ ? & ? & ? & ? & ? & ? & 1 \end{pmatrix} \end{array}$$

# Dimensionality Reduction Algorithm

## Kong

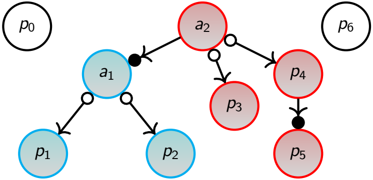
$$C_{(N_2, m_2)} = \begin{array}{c c} & \begin{array}{ccc} a_2 & p_0 & p_6 \end{array} \\ \begin{array}{c} a_2 \\ p_0 \\ p_6 \end{array} & \begin{pmatrix} {\color{red}1} & & \\ 0 & 1 & \\ 1 & 1 & 1 \end{pmatrix} \end{array}$$



# Dimensionality Reduction Algorithm

## Kong

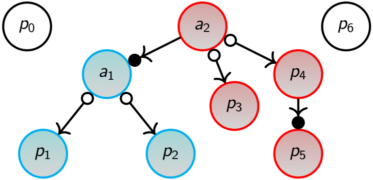
$$C_{(N_2, m_2)} = \begin{array}{c c} & \begin{array}{ccc} a_2 & p_0 & p_6 \end{array} \\ \begin{array}{c} a_2 \\ p_0 \\ p_6 \end{array} & \begin{pmatrix} \color{red}{1} & & \\ 0 & 1 & \\ 1 & 1 & 1 \end{pmatrix} \end{array}$$




### Lemma

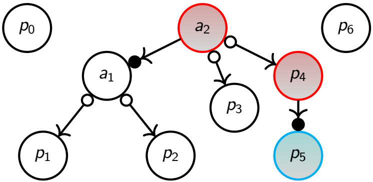
*If $v \| v$ and $v \to\!\!\bullet\; w$ then $v' \| w'$ for every pair of nodes $(v', w')$ such that $v' \in (\downarrow(v) \setminus \downarrow w)$ and $w' \in \downarrow w$.*

$$C_{(N_2,m_2)} = \begin{array}{c} \\ a_2 \\ p_0 \\ p_6 \end{array} \begin{array}{c} \begin{array}{ccc} a_2 & p_0 & p_6 \end{array} \\ \begin{pmatrix} \color{red}{1} & & \\ 0 & 1 & \\ 1 & 1 & 1 \end{pmatrix} \end{array}$$



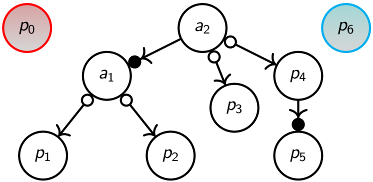

$$C_{(N_1,m_1)} = \begin{array}{c|ccccccc} & p_0 & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \\ \hline p_0 & 1 & & & & & & \\ p_1 & ? & 1 & & & & & \\ p_2 & ? & ? & 1 & & & & \\ p_3 & ? & \color{red}{1} & \color{red}{1} & 1 & & & \\ p_4 & ? & \color{red}{1} & \color{red}{1} & ? & 1 & & \\ p_5 & ? & \color{red}{1} & \color{red}{1} & ? & ? & 1 & \\ p_6 & ? & ? & ? & ? & ? & ? & 1 \end{array}$$

$$C_{(N_2,m_2)} = \begin{array}{c} \\ a_2 \\ p_0 \\ p_6 \end{array} \begin{array}{c} \begin{array}{ccc} a_2 & p_0 & p_6 \end{array} \\ \begin{pmatrix} 1 & & \\ 0 & 1 & \\ 1 & 1 & 1 \end{pmatrix} \end{array}$$




$$C_{(N_1,m_1)} = \begin{array}{c} \\ p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{array} \begin{array}{c} \begin{array}{ccccccc} p_0 & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \end{array} \\ \begin{pmatrix} 1 & & & & & & \\ ? & 1 & & & & & \\ ? & ? & 1 & & & & \\ ? & 1 & 1 & 1 & & & \\ ? & 1 & 1 & ? & 1 & & \\ ? & 1 & 1 & ? & 1 & 1 & \\ ? & ? & ? & ? & ? & ? & 1 \end{pmatrix} \end{array}$$

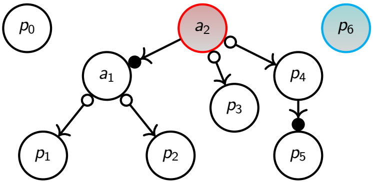
$$C_{(N_2,m_2)} = \begin{array}{c} \\ a_2 \\ p_0 \\ p_6 \end{array} \begin{array}{c} \begin{array}{ccc} a_2 & p_0 & p_6 \end{array} \\ \begin{pmatrix} 1 & & \\ 0 & 1 & \\ 1 & \color{red}{1} & 1 \end{pmatrix} \end{array}$$




$$C_{(N_1,m_1)} = \begin{array}{c|ccccccc} & p_0 & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \\ \hline p_0 & 1 & & & & & & \\ p_1 & ? & 1 & & & & & \\ p_2 & ? & ? & 1 & & & & \\ p_3 & ? & 1 & 1 & 1 & & & \\ p_4 & ? & 1 & 1 & ? & 1 & & \\ p_5 & ? & 1 & 1 & ? & 1 & 1 & \\ p_6 & \color{red}{1} & ? & ? & ? & ? & ? & 1 \end{array}$$

# Dimensionality Reduction Algorithm
## Kong

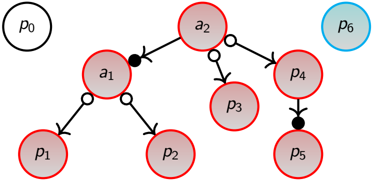
$$C_{(N_2,m_2)} = \begin{array}{c} \\ a_2 \\ p_0 \\ p_6 \end{array} \begin{array}{c} \begin{array}{ccc} a_2 & p_0 & p_6 \end{array} \\ \begin{pmatrix} 1 & & \\ 0 & 1 & \\ \color{red}{1} & 1 & 1 \end{pmatrix} \end{array}$$



# Dimensionality Reduction Algorithm

## Kong

$$C_{(N_2,m_2)} = \begin{array}{c c} & \begin{array}{ccc} a_2 & p_0 & p_6 \end{array} \\ \begin{array}{c} a_2 \\ p_0 \\ p_6 \end{array} & \begin{pmatrix} 1 & & \\ 0 & 1 & \\ \color{red}{1} & 1 & 1 \end{pmatrix} \end{array}$$



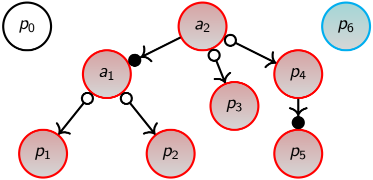

**Lemma**

*Assume $v, w$ are two nodes in $[\![E]\!]$ such that $v \notin \downarrow w$ and $w \notin \downarrow v$. If $v \| w$ then $v' \| w'$ for all pairs of nodes $(v', w') \in \downarrow v \times \downarrow w$.*

# Dimensionality Reduction Algorithm

## Kong

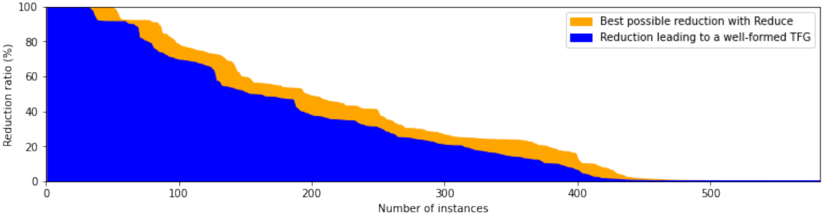
$$C_{(N_2,m_2)} = \begin{array}{c} \\ a_2 \\ p_0 \\ p_6 \end{array} \begin{array}{c} \begin{array}{ccc} a_2 & p_0 & p_6 \end{array} \\ \begin{pmatrix} 1 & & \\ 0 & 1 & \\ 1 & 1 & 1 \end{pmatrix} \end{array}$$




$$C_{(N_1,m_1)} = \begin{array}{c} \\ p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{array} \begin{array}{c} \begin{array}{ccccccc} p_0 & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \end{array} \\ \begin{pmatrix} 1 & & & & & & \\ ? & 1 & & & & & \\ ? & ? & 1 & & & & \\ ? & 1 & 1 & 1 & & & \\ ? & 1 & 1 & ? & 1 & & \\ ? & 1 & 1 & ? & 1 & 1 & \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

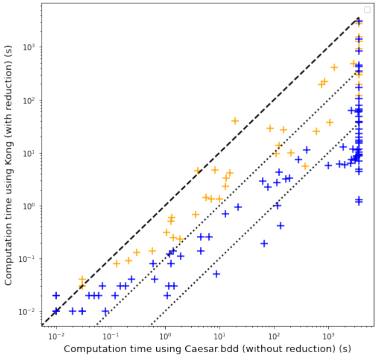
$$C_{(N_1,m_1)} = \begin{array}{c} \\ p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{array} \begin{array}{c} \begin{array}{ccccccc} p_0 & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \end{array} \\ \begin{pmatrix} 1 & & & & & & \\ \color{red}{0} & 1 & & & & & \\ \color{red}{0} & \color{red}{0} & 1 & & & & \\ \color{red}{0} & 1 & 1 & 1 & & & \\ \color{red}{0} & 1 & 1 & \color{red}{0} & 1 & & \\ \color{red}{0} & 1 & 1 & \color{red}{0} & 1 & 1 & \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

# Experimental Results



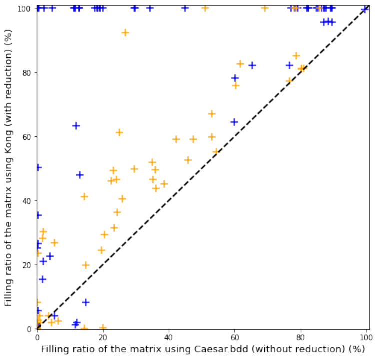
100
80
60
40
20
0
Reduction ratio (%)
0
100
200
300
400
500
Number of instances
Best possible reduction with Reduce
Reduction leading to a well-formed TFG

Kong ($y$-axis) versus CÆSAR.BDD ($x$-axis)




**Reduction ratio**: $[0.25, 0.5[$ (orange) and $[0.5, 1]$ (blue)

Kong ($y$-axis) versus CÆSAR.BDD ($x$-axis)




**Reduction ratio**: $[0.25, 0.5[$ (orange) and $[0.5, 1]$ (blue)

# Conclusion and Perspectives

# Conclusion

- We propose a new method for accelerating the computation of concurrent places $\Rightarrow$ by transposing the problem from an initial, high-dimensionality domain, into a smaller one
- Algorithm: complexity is linear in the size of the output (but with a rich formalization and a complex proof)
- Token Flow Graphs: a new data-structure + gives a better insight on the structure of reduction equations

# Perspectives

- New applications:
  - Model Counting
  - Max-marking
  - Generalized Mutual Exclusion Constraints
- $k$-concurrent relation
- Max-concurrent

Thank you for your attention!

Any questions?

# Results for a Selected Set of Instances

## Experimental Results

| Model Name | Reduc. Ratio ($r$) | Size | Computed Values (Kong Improvement) | Only 0s | Exec Time | |
|---|---|---|---|---|---|---|
| | | | | | Kong | Caesar |
| BART-030 | 100% | 8 696 535 | $\times$300.04 | $\times$291.39 | 8.73 s | 87.14 s |
| SmartHome-19 | 82% | 274 911 | $\times$1.06 | $\times$1.13 | 71.17 s | 1405.61 s |
| Peterson-6 | 67% | 885 115 | $\times$22.41 | $\times$38.22 | 212.66 s | 389.15 s |
| Database-20 | 37% | 5 315 430 | $\times$11.08 | $\times$49.27 | 62.39 s | 60.21 s |