Linear algebra + Petri nets

Piotr Hofman
University of Warsaw
Petri Nets.

- Places.
- Transitions.
Petri Nets.

- Places.
- Transitions.
- Tokens, a Marking.
Petri Nets.

- Places.
- Transitions.
- Tokens, a Marking.
- Firing a transition.
Petri Nets.

- **Places.**
- **Transitions.**
- **Tokens, a Marking.**
- **Firing a transition.**
Petri Nets.

- Places.
- Transitions.
- Tokens, a Marking.
- Firing a transition.
Questions and tools.

We focus on analysis of systems modelled with Petri nets.

Most important questions:

1. Place coverability,
2. Reachability,
3. Liveness,
4. Death of a transition,
5. Deadlock-freeness.

Most important tools:

1. Coverability: ExpSpace complete,
2. Boundedness: ExpSpace complete,
Two solutions:

Do not try to be precise (approximations).
1. Place invariant.
2. State equation.
3. Continuous reachability.
4. Traps and siphons.

Do not try to be general (sub-classes).
1. Free-choice Petri Nets.
2. Conflict free Petri nets.
3. One counter systems.
4. 2-dimensional VASS.
5. Flat systems.
Linear algebra

Integer programming.

**Input:** An integer matrix $M$ and a vector $\vec{y}$.

**Question:** If there is a vector $\vec{x} \in \mathbb{N}^d$ such that

$$M \cdot \vec{x} = \vec{y}$$

**Theorem**

The integer programming problem is NP-complete.
**Linear algebra.**

**Linear programming.**

**Input:** An integer matrix $M$ and a vector $\vec{y}$.

**Question:** If there is a vector $\vec{x} \in \mathbb{Q}^d_{\geq 0}$ such that

$$M \cdot \vec{x} = \vec{y}?$$

**Theorem**

The linear programming problem is P-complete.
Description of the net, three matrices.

\[ \vec{0}[i] = 0 \text{ for all } i \]

\[ P_1 \xrightarrow{T_1} P_2 \]

\[ P_3 \xrightarrow{T_2} P_4 \]

\[ \begin{align*}
\text{Pre}(\mathcal{N}) &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \\
\text{Post}(\mathcal{N}) &= \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix} \\
\Delta &= \text{Post}(\mathcal{N}) - \text{Pre}(\mathcal{N}) \\
&= \begin{bmatrix} -1 & 1 \\ 1 & -1 \\ 1 & -1 \end{bmatrix}
\end{align*} \]
Description of the net, three matrices.

\[ P_1 \xrightarrow{T_1} P_2 \xrightarrow{T_2} P_3 \xrightarrow{T_2} P_4 \]

\[ \vec{0}[i] = 0 \text{ for all } i \]

\[ \vec{1}_p[i] = \begin{cases} 1 & \text{if } p = i \\ 0 & \text{otherwise} \end{cases} \]

\[
Pre(N) = \begin{bmatrix}
1 & 0 \\
0 & 1 \\
0 & 0
\end{bmatrix}
\]

\[
Post(N) = \begin{bmatrix}
0 & 1 \\
1 & 0 \\
1 & 0 \\
0 & 1
\end{bmatrix}
\]

\[ \Delta = Post(N) - Pre(N) = \begin{bmatrix}
-1 & 1 \\
1 & -1 \\
1 & -1 \\
0 & 1
\end{bmatrix} \]
Let $\text{Reach}(\mathcal{N}, i)$ be a set of configurations reachable from $i$ in $\mathcal{N}$.
Let $Reach(N, i)$ be a set of configurations reachable from $i$ in $N$. Let $L_{NR}(N, i) = \{ \vec{y} : \exists \vec{x} \in N^d \ M \cdot \vec{x} = \vec{y} - i \}$. Easier to describe (PTime).

Lemma $Reach(N, i) \subseteq L_{NR}(N, i) \subseteq L_{NZ}(N, i)$. Hard to describe.
State equation.

Let $\text{Reach}(\mathcal{N}, i)$ be a set of configurations reachable from $i$ in $\mathcal{N}$.

Hard to describe.

Let $L_{\mathbb{N}}RS(\mathcal{N}, i) = \{ \vec{y} : \exists \vec{x} \in \mathbb{N}^d \ M \cdot \vec{x} = \vec{y} - i \}$.

Easier to describe (NP-complete).
Let $\text{Reach}(\mathcal{N}, i)$ be a set of configurations reachable from $i$ in $\mathcal{N}$.

Hard to describe.

Let $L_{\mathbb{N}}RS(\mathcal{N}, i) = \{\vec{y} : \exists \vec{x} \in \mathbb{N}^d \ M \cdot \vec{x} = \vec{y} - i\}$.

Easier to describe (NP-complete).

Let $L_{\mathbb{Z}}RS(\mathcal{N}, i) = \{\vec{y} : \exists \vec{x} \in \mathbb{Z}^d \ M \cdot \vec{x} = \vec{y} - i\}$.
State equation.

Let $\text{Reach}(\mathcal{N}, i)$ be a set of configurations reachable from $i$ in $\mathcal{N}$.

Hard to describe.

Let $L_{\mathbb{N}}RS(\mathcal{N}, i) = \{ \vec{y} : \exists \vec{x} \in \mathbb{N}^d \ M \cdot \vec{x} = \vec{y} - i \}$. 

Easier to describe (NP-complete).

Let $L_{\mathbb{Z}}RS(\mathcal{N}, i) = \{ \vec{y} : \exists \vec{x} \in \mathbb{Z}^d \ M \cdot \vec{x} = \vec{y} - i \}$. 

Easy to describe (PTime).
State equation.

Let \( \text{Reach}(\mathcal{N}, i) \) be a set of configurations reachable from \( i \) in \( \mathcal{N} \).

Hard to describe.

Let \( L_{\mathbb{N}} RS(\mathcal{N}, i) = \{ \vec{y} : \exists \vec{x} \in \mathbb{N}^d \ M \cdot \vec{x} = \vec{y} - \vec{i} \} \).

Easier to describe (NP-complete).

Let \( L_{\mathbb{Z}} RS(\mathcal{N}, i) = \{ \vec{y} : \exists \vec{x} \in \mathbb{Z}^d \ M \cdot \vec{x} = \vec{y} - \vec{i} \} \).

Easy to describe (PTime).

Lemma

\( \text{Reach}(\mathcal{N}, i) \subseteq L_{\mathbb{N}} RS(\mathcal{N}, i) \subseteq L_{\mathbb{Z}} RS(\mathcal{N}, i) \).
An application.

Algorithm 1 for reachability.

Start from the initial configuration $i$ and exhaustively build a graph of reachable configurations adding nodes one by one.

- if you find $f$ then return 1;
- if you can not visit any new configuration then return 0;
- if you run out of memory then return "I don’t know".

Algorithm 2 for reachability.

Start from the initial configuration $i$ and exhaustively build a graph of reachable configurations adding nodes one by one; but whenever you want to add a new node $\vec{x}$ to the graph you check if $f \in L_{\text{SN}}(N, \vec{x})$. You add the node if and only if the answer is yes.

- if you find $f$ then return 1;
- if you can not add any new node then return 0;
- if you run out of memory then return "I don’t know".
An application.

Algorithm 1 for reachability.

Start from the initial configuration \( i \) and exhaustively build a graph of reachable configurations adding nodes one by one.

- if you find \( f \) then return 1;
- if you can not visit any new configuration then return 0;
- if you run out of memory then return "I don’t know".

Algorithm 2 for reachability.

Start from the initial configuration \( i \) and exhaustively build a graph of reachable configurations adding nodes one by one; but whenever you want to add a new node \( \vec{x} \) to the graph you check if \( f \in L_N SR(N, \vec{x}) \). You add the node if and only if the answer is yes.

- if you find \( f \) then return 1;
- if you can not add any new node then return 0;
- if you run out of memory then return "I don’t know".
\( \vec{y} \) is called a P-flow iff \( \vec{y} \cdot M = 0 \).

If \( \vec{y} \geq 0 \) then we call it P-semiflow.
P-flows

\( \vec{y} \) is called a P-flow iff \( \vec{y} \cdot M = 0 \).

If \( \vec{y} \geq 0 \) then we call it a P-semiflow.

Lemma

If \( f \) is reachable from \( i \) then \( \vec{y} \cdot f = \vec{y} \cdot i \).
\( \vec{y} \) is called a P-flow iff \( \vec{y} \cdot M = 0 \).
If \( \vec{y} \geq 0 \) then we call it P-semiflow.

**Lemma**
If \( f \) is reachable from \( i \) then \( \vec{y} \cdot f = \vec{y} \cdot i \).

**Question**
How do we test a boundedness of a place using P-semiflows?
P-flows

\( \vec{y} \) is called a P-flow iff \( \vec{y} \cdot M = 0 \).

If \( \vec{y} \geq 0 \) then we call it a P-semiflow.

**Lemma**

If \( \vec{f} \) is reachable from \( i \) then \( \vec{y} \cdot \vec{f} = \vec{y} \cdot i \).

**Question**

How do we test a boundedness of a place using P-semiflows?

**Lemma**

Let \( \vec{y} \) be a P-semiflow of the net \( \mathcal{N} \), then the number of tokens is bounded for all \( 1 \leq i \leq d \) such that \( \vec{y}[i] > 0 \).
A place $p$ in a net $\mathcal{N}$ is structurally bounded if for every initial marking $i$ the
\[
\max \{ \mathbf{1}_p^T \cdot \mathbf{m} : \mathbf{m} \in RS(\mathcal{N}, i) \} \text{ is finite.}
\]
A place $p$ in a net $\mathcal{N}$ is structurally bounded if for every initial marking $i$ the
\[
\max \{ \mathbf{1}_p^T \cdot \mathbf{m} : \mathbf{m} \in RS(\mathcal{N}, i) \} \text{ is finite.}
\]

Theorem
A following conditions are equivalent:
1. a place $p$ in the net $\mathcal{N}$ is structurally bounded,
2. there exists $\mathbf{y} \geq \mathbf{1}_p$ such that $\mathbf{y} \cdot \Delta \leq \mathbf{0}$,
3. there does not exist $\mathbf{x} \geq \mathbf{0}$ such that $\Delta \cdot \mathbf{x} \geq \mathbf{1}_p$. 
proof

Theorem

A following conditions are equivalent:

1. a place $p$ in the net $\mathcal{N}$ is structurally bounded,
2. there exists $\vec{y} \geq 1_p$ such that $\vec{y} \cdot \Delta \leq \vec{0}$,
3. there does not exist $\vec{x} \geq \vec{0}$ such that $\Delta \cdot \vec{x} \geq 1_p$.

1 $\Rightarrow$ 3 by $\neg 3 \Rightarrow \neg 1$

2 $\Rightarrow$ 1 Direct.
Proof

Theorem

A following conditions are equivalent:

1. a place $p$ in the net $\mathcal{N}$ is structurally bounded,
2. there exists $\vec{y} \geq 1_p$ such that $\vec{y} \cdot \Delta \leq 0$,
3. there does not exist $\vec{x} \geq 0$ such that $\Delta \cdot \vec{x} \geq 1_p$.

1. $1 \implies 3$ by $\neg 3 \implies \neg 1$
Proof

Theorem

A following conditions are equivalent:

1. a place \( p \) in the net \( \mathcal{N} \) is structurally bounded,
2. there exists \( \bar{y} \succeq 1_p \) such that \( \bar{y} \cdot \Delta \leq 0 \),
3. there does not exist \( \bar{x} \succeq 0 \) such that \( \Delta \cdot \bar{x} \geq 1_p \).

1 \( \implies \) 3 by \( \neg 3 \implies \neg 1 \)
2 \( \implies \) 2 by a theorem related to dual programs theorem called alternative theorem.

Theorem

Exactly one of the following systems of equations has a solution:

\[
\begin{align*}
A\bar{x} & \succeq \bar{b}.
\end{align*}
\]

\[
\begin{align*}
\bar{y} & \succeq 0 \\
\bar{y}^T \cdot A & = 0 \\
\bar{y}^T \cdot \bar{b} & > 0.
\end{align*}
\]
Proof

Theorem

A following conditions are equivalent:
1. a place $p$ in the net $\mathcal{N}$ is structurally bounded,
2. there exists $\vec{y} \geq \vec{1}_p$ such that $\vec{y} \cdot \Delta \leq \vec{0}$,
3. there does not exist $\vec{x} \geq \vec{0}$ such that $\Delta \cdot \vec{x} \geq \vec{1}_p$.

1. $1 \implies 3$ by $\neg 3 \implies \neg 1$
2. $3 \implies 2$ by a theorem related to dual programs theorem called alternative theorem.

Theorem

Exactly one of the following systems of equations has a solution:

- $A\vec{x} \geq \vec{b}$.
- $\vec{y} \geq \vec{0}$
  - $\vec{y}^T \cdot A = \vec{0}$
  - $\vec{y}^T \cdot \vec{b} > 0$.

3. $2 \implies 1$ Direct.
Continuous reachability.
Linear programming + If formula.

Input: A $r \times c$- integer matrix $M$ and a vector $\vec{y} \in \mathbb{Z}^r$ and a set of predicates of a form $\vec{x}[i] > 0 \implies \vec{x}[j] > 0$.

Question: If there is a vector $\vec{x} \in \mathbb{Q}^c_{\geq 0}$ such that $M \cdot \vec{x} = \vec{y}$ and all predicates are satisfied?

Theorem

The Linear programming + If formula problem is in PTime.
Linear programming + If formula.

**Input:** A $r \times c$-integer matrix $M$ and a vector $\vec{y} \in \mathbb{Z}^r$ and a set of predicates of a form $\vec{x}[i] > 0 \implies \vec{x}[j] > 0$.

**Question:** If there is a vector $\vec{x} \in \mathbb{Q}^c_{\geq 0}$ such that $M \cdot \vec{x} = \vec{y}$ and all predicates are satisfied?

**Theorem**

The Linear programming + If formula problem is in PTime.

**Proof**

1. The set of solutions is convex.
2. If for every $i$ there is a solution such that $\vec{x}[i] > 0$ then there is a solution such that $\vec{x}[j] > 0$ for all $j$. 

solve( Matrix $\Delta$, Vector $\vec{y}$, set_of_implications $\mathcal{S}$, set_of_zeros $\mathbb{X}$)
{
    If there is no solution $\Delta \cdot \vec{x} = \vec{y}$ in $\mathbb{Q}_{\geq 0}^d$,
    where $x_i = 0$ for all $x_i \in \mathbb{X}$ then return false;
    If there is a solution $\Delta \cdot \vec{x} = \vec{y}$ in $\mathbb{Q}_{\geq 0}^d$,
    where $x_i = 0$ iff $x_i \in \mathbb{X}$ then return true;
    Find a new coordinate $x_i$
        which has to be equal 0 in every solution;
    Add $x_i$ to $\mathbb{X}$;
    Add to $\mathbb{X}$ all $x_j$ that has to be added due to implications;
    return solve($M$, $\vec{y}$, $\mathcal{S}$, $\mathbb{X}$);
}
Continuous Petri Nets.

- Marking: $\mathcal{M}: \mathbb{P} \rightarrow \mathbb{Q}$
- Transitions: $\mathbb{T}$
- Firing a transition $t \in \mathbb{T}$ with a coefficient $a \in \mathbb{Q}$. 

![Diagram of Continuous Petri Nets]
Continuous Petri Nets.

- **Marking**: $\mathcal{M} : \mathbb{P} \rightarrow \mathbb{Q}$
- **Transitions**: $\mathbb{T}$
- **Firing a transition $t \in \mathbb{T}$ with a coefficient $a \in \mathbb{Q}$.**
Continuous Petri Nets.

- **Marking**: $\mathcal{M} : P \rightarrow Q$
- **Transitions**: $T$
- **Firing a transition**: $t \in T$ with a coefficient $a \in \mathbb{Q}$. 
Continuous Petri Nets.

- **Marking**: \( M : P \rightarrow Q \)
- **Transitions**: \( T \)
- **Firing a transition** \( t \in T \) **with a coefficient** \( a \in \mathbb{Q} \).
Continuous Petri Nets.

- **Marking**: $\mathcal{M} : P \rightarrow Q$
- **Transitions**: $T$
- **Firing a transition** $t \in T$ with a coefficient $a \in Q$. 

![Diagram of Continuous Petri Nets]
Continuous Petri Nets Reachability.

**Input:** Two configurations \( i \) and \( f \)

**Question:** If there is a run from \( i \) to \( f \) under continuous semantics.

**A simpler variant of the problem.**

Suppose, that

\[ \forall_i \ (i[i] > 0 \text{ and } f[i] > 0). \]

\( f \) is reachable from \( i \) iff

\[ f - i = \Delta \cdot \bar{x} \text{ where } \bar{x} \in \mathbb{Q}^d_{\geq 0}. \]
Lemma

\( f \) is reachable from \( i \) if

1.

\[ f - i = \Delta \cdot \tilde{x} \text{ where } \tilde{x} \in \mathbb{Q}^{d}_{\geq 0} \]

2.

\( \tilde{x}[i] > 0 \text{ and } Pre[j, i] > 0 \implies i[j] > 0, \)

3.

\( \tilde{x}[i] > 0 \text{ and } Post[j, i] > 0 \implies f[j] > 0. \)
Continuous Petri Nets Reachability.

**Lemma**

$f$ is reachable from $i$ if

1. $f - i = \Delta \cdot \vec{x}$ where $\vec{x} \in \mathbb{Q}^d_{\geq 0}$
2. $\vec{x}[i] > 0$ and $Pre[j, i] > 0 \implies i[j] > 0$,
3. $\vec{x}[i] > 0$ and $Post[j, i] > 0 \implies f[j] > 0$.

**Theorem**

$f$ is reachable from $i$ iff there are two configurations $i'$ and $f'$ such that

1. there is a run form $i$ to $i'$ that is using at most $d$ steps.
2. there is a run form $f'$ to $f$ that is using at most $d$ steps.
3. There is a run form $i'$ to $f'$ due to Lemma.
Translation to a formula (linear + If).

**Lemma**

For a given Petri net $\mathcal{N}$ and two configurations $i$ and $f$ in PTime one can compute a formula (linear programming + if) such that it is satisfiable if and only if $f$ is continuously reachable from $i$ in the net $\mathcal{N}$.

We use:

**Theorem**

$f$ is reachable from $i$ iff there are two configurations $i'$ and $f'$ such that

1. there is a run form $i$ to $i'$ that is using at most $d$ steps.
2. there is a run form $f'$ to $f$ that is using at most $d$ steps.
3. There is a run form $i'$ to $f'$ due to Lemma.
IDEA: Take a backward coverability algorithm, and speed it up.
IDEA: Take a backward coverability algorithm, and speed it up.

What is the main obstacle?
Q-cover 2015.

**IDEA:** Take a backward coverability algorithm, and speed it up.

**CHALLENGE:** Size of the representation of the representation of the upward-closed set may get too big.
Q-cover 2015.

**IDEA:** Take a backward coverability algorithm, and speed it up.

**CHALLENGE:** Size of the representation of the representation of the upward-closed set may get too big.

How to cut the upward-closed set?
IDEA: Take a backward coverability algorithm, and speed it up.

CHALLENGE: Size of the representation of the representation of the upward-closed set may get too big.

IDEA: Let \( \vec{x} \in M \uparrow \), if there is no \( \vec{y} \geq \vec{x} \) such that \( \vec{y} \in RS(\mathcal{N}, i) \) then we can throw \( \vec{x} \) away.
IDEA: Take a backward coverability algorithm, and speed it up.

CHALLENGE: Size of the representation of the representation of the upward-closed set may get too big.

IDEA: Let $\vec{x} \in M \uparrow$, if there is no $\vec{y} \geq \vec{x}$ such that $\vec{y} \in RS(N, i)$ then we can throw $\vec{x}$ away.

M. Blondin, A. Finkel, Ch. Haase, S. Haddad, 2015

SOLUTION: Let $\vec{x} \in M \uparrow$, if there is no $\vec{y} \geq \vec{x}$ such that $\vec{y} \in CRS(N, i)$ then we can throw $\vec{x}$ away.
**IDEA**: Take a backward coverability algorithm, and speed it up.

**CHALLENGE**: Size of the representation of the representation of the upward-closed set may get too big.

**IDEA**: Let $\vec{x} \in \mathcal{M} \uparrow$, if there is no $\vec{y} \geq \vec{x}$ such that $\vec{y} \in RS(\mathcal{N}, i)$ then we can throw $\vec{x}$ away.

**SOLUTION**: Let $\vec{x} \in \mathcal{M} \uparrow$, if there is no $\vec{y} \geq \vec{x}$ such that $\vec{y} \in CRS(\mathcal{N}, i)$ then we can throw $\vec{x}$ away.

**Thomas Geffroy, Jérôme Leroux, Grégoire Sutre, 2017**

Actually, any over-approximation will work: $LRS$ instead of $CRS$. 
Advertisement.
Internships at the University of Warsaw.

Possibilities:

Prof. Mikołaj Bojanczyk
Logic, Automata, Formal Languages.
Email: bojan@mimuw.edu.pl

Prof. Piotr Sankowski
Algorithms.
Email: sank@mimuw.edu.pl

Prof. Stefan Dziembowski
Cryptography.
Email: S.Dziembowski@crypto.edu.pl