An Introduction to Petri nets and how to analyse them...

G. Geeraerts

Groupe de Vérification - Département d’Informatique
Université Libre de Bruxelles
Introduction
Introduction

• **Concurrency:** property of a “system” in which many “entities” act at the same time and interact.

• Often found in many application:
  • Computer science (e.g.: parallel computing)
  • Workflow
  • Manufacturing systems
  • ....
Introduction

Concurrency
Introduction
Concurrency

Work in parallel
Introduction
Concurrency

Work in parallel

Must wait for the two other machines
Introduction
Concurrency

Can write or read on the DB

Can write or read on the DB
Introduction

Concurrency

Boss
Introduction
Concurrency

Boss
Introduction
Concurrency

Boss

Employees: work in parallel
Introduction

Concurrency

Boss

gives work

Employees: work in parallel
Introduction
Concurrency

Boss

gives work

Employees: work in parallel
Introduction
Concurrency

Boss

gives work

Employees: work in parallel
Introduction
Concurrency

Boss

gives work

receives credit for the results

Employees: work in parallel
Introduction

- Petri nets are a tool to model concurrent systems and reason about them.
- Invented in 1962 by C.A. Petri.
The aim of the talk

• **Introduce** you to Petri nets (and some of their extensions)
• **Explain** several analysis methods for PN
  • i.e., what can you ‘**ask**’ about a PN?
• **Give a rough idea of the** research in the verification group at ULB...
• **... and foster new collaborations**?
How I use Petri nets

```cpp
template <typename T> T Max(T a, T b)
{
    return a < b ? b : a;
}

#include <string>

int main() // fonction main
{
    int i = Max(3, 5);
    char c = Max('e', 'b');
    std::string s = Max(std::string("hello"), std::string("world"));
    float f = Max<float>(1, 2.2f);
}
```

Analysis method of PN
How you might use PN

Your favorite application

abstraction

Analysis method of PN

Figure 1: The Petri net ... to Finkel's algorithm. Nodes and edges in grey have been removed. Thick grey arrows represent the proofs.
Intuitions
**Ingredients**

A Petri net is made up of...

- **Places**
  - = some type of resource

- **Transitions**
  - consume and produce resources

- **Tokens**
  - = one unity of a certain resource

Tokens ‘live’ in the places
Transitions

Input places

Output places

2

3

13
Firing a transition

Transitions **consume** tokens from the **input** places and produce tokens in the **output** places
Firing a transition

Transitions **consume** tokens from the **input** places and produce tokens in the **output** places.

Now, the transition cannot be fired anymore.
Example 1

Can write or read on the DB

The two machines cannot write at the same time

Can write or read on the DB
The **token** tells us the **state** of the process.
Example 1

The **token** tells us the **state** of the process
The token tells us the state of the process
Example 1

The \textit{token} tells us the \textit{state} of the process
Example 1

The token tells us the state of the process
Example 1

Add a lock to ensure mutual exclusion
Example 1
Example 2

mutex M;

Process P {
    repeat {
        take M;
        critical;
        release M;
    }
}

![Diagram of SMPN Nµ](image-url)
mutex M;

Process P {
    repeat {
        take M;
        critical;
        release M;
    }
}

Here, we have applied a counting abstraction
Plan of the talk

- Preliminaries
- **Tools** for the analysis of PN
  - reachability tree and reachability graph
  - place invariants
  - Karp & Miller and the coverability set
- **The coverability problem**
- **More** on PN: extensions...
- Conclusion
Plan of the talk

- Preliminaries
- **Tools** for the analysis of PN
  - reachability tree and reachability graph
  - place invariants
  - Karp & Miller and the coverability set
- The coverability problem
- More on PN: extensions...
- Conclusion
Preliminaries
Formal definition

• A Petri net is a tuple $\langle P, T \rangle$ where:
  • $P$ is the (finite) set of places
  • $T$ is the (finite) set of transitions. Each transition $t$ is a tuple $\langle I, O \rangle$ where:
    • $I$: is a function s.t. $t$ consumes $I(p)$ tokens in each place $p$
    • $O$: is a function s.t. $t$ produces $O(p)$ tokens in each place $p$
Example

$I(p_1)=2 \quad I(p_2)=1 \quad I(p_3)=0 \quad I(p_4)=0 \quad I(p_5)=0$

$O(p_1)=0 \quad O(p_2)=0 \quad O(p_3)=1 \quad O(p_4)=3 \quad O(p_5)=1$
Markings

• The distribution of the tokens in the places is formalised by the notion of marking, which can be seen:

  • either as a function \( m \), s.t. \( m(p) \) is the number of tokens in place \( p \)
  
  • or as a vector \( m = \langle m_1, m_2, \ldots, m_n \rangle \) where \( m_i \) is the number of tokens in place \( p_i \)
Example

\[ m = \langle 1, 1, 1, 2, 0 \rangle \]

\[ m = \langle p_1, p_2, p_3, 2p_4 \rangle \]

\[ m(p_1) = 1, m(p_2) = 1, m(p_3) = 1, m(p_4) = 2, m(p_5) = 0 \]
Firing a transition

- A transition $t = \langle I, O \rangle$ can be fired from $m$ iff for any place $p$:
  $$m(p) \geq I(p)$$

- The firing transforms the marking $m$ into a marking $m'$ s.t. for any place $p$:
  $$m'(p) = m(p) - I(p) + O(p)$$

- Notation: $m \rightarrow m'$

- Notation: $\text{Post}(m) = \{ m' \mid m \rightarrow m' \}$
Example

$$\text{Post}(\langle 1, 1, 0 \rangle) = \{ \langle 2, 1, 0 \rangle, \langle 0, 0, 1 \rangle \}$$
Example

Post(⟨1, 1, 0⟩) =
{ ⟨2, 1, 0⟩, ⟨0, 0, 1⟩ }
Example

\[
\text{Post}(\langle 1, 1, 0 \rangle) = \{ \langle 2, 1, 0 \rangle, \langle 0, 0, 1 \rangle \}
\]
Example

\[ \text{Post}(\langle 1, 1, 0 \rangle) = \{ \langle 2, 1, 0 \rangle, \langle 0, 0, 1 \rangle \} \]
Example

\[ \text{Post}(\langle 1, 1, 0 \rangle) = \{ \langle 2, 1, 0 \rangle, \langle 0, 0, 1 \rangle \} \]
Initial marking
Reachable markings

• All PN are equipped with an initial marking $m_0$
• If two markings $m$ and $m'$ are s.t.:

$$m \rightarrow m_1 \rightarrow m_2 \rightarrow \cdots \rightarrow m'$$

Then $m'$ is reachable from $m$

• Let $N$ be a PN with initial marking $m_0$:

$$\text{Reach}(N) = \{m \text{ reachable from } m_0\}$$

is the set of reachable markings of $N$. 

Example

\[\text{Figure 2.1: The SMPN } N_\mu.\]

\[\text{Figure 2.2: The ... that an SMPN } N = \langle P, T, D^-, D^+, m_0 \rangle \text{naturally defines a transition system } S_N = \langle N_{kP}, m_0, \Rightarrow \rangle, \text{ where } \Rightarrow \text{ is such}\]
Example

\[
\operatorname{Reach}(N) = \left\{ \langle i, 1, 0 \rangle \mid i \in \mathbb{N} \right\} \cup \left\{ \langle i, 0, 1 \rangle \mid i \in \mathbb{N} \right\}
\]
Example

Reach(\(\mathcal{A}\)) =
\[
\{\langle i, 1, 0 \rangle \mid i \in \mathbb{N}\}
\cup
\{\langle i, 0, 1 \rangle \mid i \in \mathbb{N}\}
\]

This set allows us to prove that the mutual exclusion is indeed enforced.
Ordering on markings

• Markings can be compared thanks to $\preceq$:
  
  $m \preceq m'$ iff for any place $p$: $m(p) \leq m'(p)$

  $m \prec m'$ iff $m \preceq m'$ and $m \neq m'$

• Examples:
  
  • $\langle 1, 0, 0 \rangle \prec \langle 1, 1, 0 \rangle \preceq \langle 1, 1, 0 \rangle \preceq \langle 5, 7, 2 \rangle$

  • $\langle 1, 0, 0 \rangle$ is not comparable to $\langle 0, 1, 0 \rangle$
Meaningful questions about PN include:

- **Boundedness**: is the number of reachable markings bounded?
- **Place boundedness**: is there a bound on the maximal number of tokens that can be created in a given place?
- **Semi-liveness**: is there a reachable marking from which a given transition can fire?
- **Coverability**
Example

Bounded PN

All the places are bounded

All the transitions are semi-live
Example

- **Unbounded PN**
- $p_2$ and $p_3$ are **bounded**
- $p_1$ is **unbounded**
- All the transitions are **semi-live**
Some tools for the analysis of PN
Reachability tree and reachability graph
Reachability Tree

• **Idea:**
  
  • the **root** is labeled by $m_0$
  
  • for each node labeled by $m$, create one **child** for each marking of $\text{Post}(m)$
Reachability Tree
Reachability Tree

\[ \langle M, I_1, I_2 \rangle \]
Reachability Tree

\[
\langle M, I_1, I_2 \rangle \\
\langle I_1, W_2 \rangle \\
\langle W_1, I_2 \rangle \\
\langle M, R_1, I_2 \rangle \\
\langle M, I_1, R_2 \rangle
\]
Reachability Tree

\[ \langle M, I_1, I_2 \rangle \]
\[ \langle I_1, W_2 \rangle \]
\[ \langle W_1, I_2 \rangle \]
\[ \langle M, R_1, I_2 \rangle \]
\[ \langle R_1, W_2 \rangle \]
\[ \langle M, I_1, I_2 \rangle \]
Reachability Tree
Reachability Tree
Reachability Tree
Reachability Tree

Reachability trees can be infinite

\[ \langle M, I_1, I_2 \rangle \]

\[ \langle I_1, W_2 \rangle \]

\[ \langle M, I_1, R_2 \rangle \]

\[ \langle I_1, W_2 \rangle \langle W_1, I_2 \rangle \langle M, R_1, I_2 \rangle \]
Reachability graph

• Idea: build a node for each reachable marking and add an edge from m to m’ if some transition transforms m into m’

• remark: now, if we meet the same marking twice, we do not create a new node, but re-use the previously created node.
Reachability graph
Reachability graph

\[ \langle M, I_1, I_2 \rangle \]
Reachability graph

\[ \langle M, I_1, I_2 \rangle \quad \langle M, I_1, R_2 \rangle \quad \langle M, R_1, I_2 \rangle \]
Reachability graph
Reachability graph

\[
\begin{align*}
\langle M, \emptyset, \emptyset \rangle & \quad \langle M, i_1, \emptyset \rangle \\
\langle M, \emptyset, i_2 \rangle & \quad \langle M, i_1, i_2 \rangle \\
\langle M, R_1, \emptyset \rangle & \quad \langle M, R_1, i_2 \rangle \\
\langle M, \emptyset, R_2 \rangle & \quad \langle M, i_1, R_2 \rangle \\
\langle M, R_1, R_2 \rangle & \quad \langle M, i_1, R_2 \rangle 
\end{align*}
\]
Reachability graph

\[ \langle R_1, W_2 \rangle \quad \langle W_1, R_2 \rangle \quad \langle I_1, W_2 \rangle \quad \langle W_1, I_2 \rangle \quad \langle M, I_1, I_2 \rangle \quad \langle M, R_1, I_2 \rangle \quad \langle M, I_1, R_2 \rangle \quad \langle M, R_1, R_2 \rangle \]
Reachability graph
The reachability graph allows us to prove that the mutual exclusion is indeed enforced.
Reachability graph

• The reachability graph of a PN contains all the necessary information to decide:
  • boundedness
  • place boundedness
  • semi-liveness
  • ...

Reachability graph

• Unfortunately...

\( \langle p_2 \rangle \)
Reachability graph

- Unfortunately...

\[ \langle p_2 \rangle \]

\[ \langle p_1, p_2 \rangle \]
• Unfortunately...

\[
\langle p_2 \rangle\]

\[
\langle p_1, p_2 \rangle\]

\[
\langle 2p_1, p_2 \rangle\]

\[
\langle p_3 \rangle\]
Reachability graph

• Unfortunately...

\[
\begin{align*}
\langle p_2 \rangle & \quad \langle p_3 \rangle \\
\langle p_1, p_2 \rangle & \quad \langle 2p_1, p_2 \rangle \\
\langle 2p_1, p_2 \rangle & \quad \langle 3p_1, p_2 \rangle \\
\langle 3p_1, p_2 \rangle & \quad \langle p_1, p_3 \rangle
\end{align*}
\]

Figure 2.1: The SMPN \( N_{\mu} \).

Figure 2.2: The \( N = \langle P, T, D^{-}, D^{+}, m_0 \rangle \) naturally defines a transition system \( SN = \langle N_{kP}, m_0, \Rightarrow \rangle \), where \( \Rightarrow \) is such...
Reachability graph

- Unfortunately...

\[ \langle p_2 \rangle \]

\[ \langle p_1, p_2 \rangle \]

\[ \langle 2p_1, p_2 \rangle \rightarrow \langle p_3 \rangle \]

\[ \langle 3p_1, p_2 \rangle \rightarrow \langle p_1, p_3 \rangle \]

\[ \langle \cdot \rangle \]

Figure 2.1: The SMPN \( N \).

\[ \langle \cdot \rangle \]

Figure 2.2: The ... that an SMPN \( N = \langle P, T, D^{-}, D^{+}, m_0 \rangle \) naturally defines a transition system \( SN = \langle N^k, m_0, \Rightarrow \rangle \), where \( \Rightarrow \) is such...
Reachability graph

• Unfortunately...

Reachability graphs can be infinite
The hard stuff...

• The **main difficulty** in analysing Petri nets is due to the **possibly infinite** number of reachable markings.

• We have to find **techniques** to deal with this **infinity** set.
The hard stuff...

- **Remark**: finite doesn’t mean easy
- The set of reachable markings of a bounded net can be huge!
- Efficient techniques to deal with bounded nets have been developed.
- e.g.: net unfoldings
Place invariants
Place Invariants

\[ m(R_1) + m(R_2) + m(I_2) = 1 \]
Place Invariants

\[ m(R_1) + m(R_2) + m(l_2) = I \]
Place Invariants

\[ m(R_1) + m(R_2) + m(I_2) = 2 \]
Place Invariants

\[ m(R_1) + m(R_2) + m(I_2) = 0 \]
The total number of tokens in these places is not constant

\[ m(R_1) + m(R_2) + m(I_2) = 0 \]
Place Invariants

\[ m(R_1) + m(W_1) + m(l_1) = 1 \]
Place Invariants

\[ m(R_1) + m(W_1) + m(I_1) = 1 \]
Place Invariants

\[ m(R_1) + m(W_1) + m(I_1) = 1 \]
The total number of tokens in these places is constant.

\[ m(R_1) + m(W_1) + m(I_1) = 1 \]
The total number of tokens in these places is constant

This provides meaningful information about the system: a process is either idle, or reading or writing

\[ m(R_1) + m(W_1) + m(I_1) = 1 \]
Place Invariants

\[ m(p_1) + m(p_2) + m(p_3) + m(p_4) = 1 \]
Place Invariants

\[ m(p_1) + m(p_2) + m(p_3) + m(p_4) = 3 \]
Place Invariants

\[ m(p_1) + m(p_2) + m(p_3) + m(p_4) = 2 \]
Place Invariants

\[ m(p_1) + m(p_2) + m(p_3) + m(p_4) = 1 \]
Place Invariants

The total number of tokens in these places is not constant

\[ m(p_1) + m(p_2) + m(p_3) + m(p_4) = 1 \]
The total number of tokens in these places is **not constant**

In some sense, tokens in $p_1$ are **heavier** than those in $p_2$
Place Invariants

Let’s add **weights** to the places!

The total number of tokens in these places is **not constant**

In some sense, tokens in $p_1$ are **heavier** than those in $p_2$
Place Invariants

\[ 3 \, m(p_1) + m(p_2) + m(p_3) + 2 \, m(p_4) = 3 \]
Place Invariants

3 m(p₁) + m(p₂) + m(p₃) + 2 m(p₄) = 3
Place Invariants

3 \( m(p_1) + m(p_2) + m(p_3) + 2 \ m(p_4) = 3 \)
Place invariant: Definition

- **Definition**: a place-invariant (or p-semiflow) is a vector $i$ of natural numbers s.t. for any reachable marking $m$:

\[
\sum_{p \in P} i(p) \times m(p) = \sum_{p \in P} i(p) \times m_0(p)
\]

**Remark**: there exists a trivial invariant $i = \langle 0, 0, \ldots, 0 \rangle$
Example: other invariants

\[ m(p_1) + m(p_3) = 1 \]
\[ 2 \cdot m(p_1) + m(p_2) + 2 \cdot m(p_4) = 2 \]
Invariants as over-approximations

- A place-invariant expresses a \textbf{constraint} on the \textbf{reachable markings}.

- If $m$ is reachable and $i$ is an \textbf{invariant}, then:

$$\sum_{p \in P} i(p) \times m(p) = \sum_{p \in P} i(p) \times m_0(p)$$

- The \textbf{reverse is not true}!
Example

\[ m(p_1) + m(p_3) = 1 \]

is an invariant

but \( \langle 1, 25, 0, 234 \rangle \) is not reachable
Theorem: For any Petri net $N$:

$$\text{Reach}(N) \subseteq \{m \mid m \text{ respects some invariant of } N\}$$
Invariants as over-approximations

- **Theorem**: For any Petri net $N$:

  \[
  \text{Reach}(N) \subseteq \{ m \mid m \text{ respects some invariant of } N \}
  \]

This set overapproximates the reachable markings.
Invariants as over-approximations

- **Theorem**: For any Petri net $N$:
  \[
  \text{Reach}(N) \subseteq \{m \mid m \text{ respects some invariant of } N\}
  \]

This set overapproximates the reachable markings

Place invariants are thus useful to finitely approximate the set of reachable markings
Place invariant and boundedness

- **Theorem:** If there exists a place invariant $i$ and a place $p$ s.t. $i(p) > 0$ then $p$ is bounded.

- **Remark:** the reverse is not true.

- One can find a **bounded** net that doesn’t have a place invariant $i$ with $i(p) > 0$ for each place.
Place invariant

- **Question**: how do we compute them?
Matrix characterisation

• The **negative effect** (consumption) of all the transitions on all the places can be **summarised** in one matrix:

\[
W^- = \begin{pmatrix}
I_1(p_1) & I_2(p_1) & \cdots & I_k(p_1) \\
I_1(p_2) & I_2(p_2) & \cdots & I_k(p_2) \\
\vdots & \vdots & \ddots & \vdots \\
I_1(p_n) & I_2(p_n) & \cdots & I_k(p_n)
\end{pmatrix}
\]

where, for any \( i \): \( t_i = \langle I_i, O_i \rangle \)
Matrix characterisation

- The same can be done with the positive effects:

\[
W^+ = \begin{pmatrix}
O_1(p_1) & O_2(p_1) & \cdots & O_k(p_1) \\
O_1(p_2) & O_2(p_2) & \cdots & O_k(p_2) \\
\vdots & \vdots & \ddots & \vdots \\
O_1(p_n) & O_2(p_n) & \cdots & O_k(p_n)
\end{pmatrix}
\]

pos. eff. on \( p_1 \)
pos. eff. on \( p_2 \)

where, for any \( i \): \( t_i = \langle l_i, O_i \rangle \)
**Incidence Matrix**

- The **global effect** of every transition can be summarised as a single matrix:

\[ W = W^+ - W^- \]

\(W\) is called the **incidence matrix** of the net.
Example

\[ W^+ = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \quad W^- = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \]

\[ W = \begin{pmatrix} 1 & -1 & 1 \\ 0 & -1 & 1 \\ 0 & 1 & -1 \end{pmatrix} \]
Example

\[ W^+ = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \quad W^- = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \]

\[ W = \begin{pmatrix} 1 & -1 & 1 \\ 0 & -1 & 1 \\ 0 & 1 & -1 \end{pmatrix} \]

![Diagram](image)

Figure 2.1: The SMPN \( N_{\mu} \).

Figure 2.2: The ... that an SMPN \( N = \langle P, T, D^-, D^+, m_0 \rangle \) naturally defi nes a transition system \( SN = \langle N, P, m_0, \Rightarrow \rangle \), where \( \Rightarrow \) is such
Example

\[ W^+ = \begin{pmatrix}
1 & 0 & 1 \\
0 & 0 & 1 \\
0 & 1 & 0
\end{pmatrix} \quad W^- = \begin{pmatrix}
0 & 1 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix} \]

\[ W = \begin{pmatrix}
1 & -1 & 1 & 1 \\
0 & -1 & 1 & 1 \\
0 & 1 & -1 & 1
\end{pmatrix} \]
Computing place invariants

- Intuitively, if $i$ is a place invariant it should assign weights to the places such that the positive and negative effects of every transition are balanced.

- Thus, for any transition $t = \langle I, O \rangle$ we should have:

$$\sum_{p \in P} I(p) \times i(p) = \sum_{p \in P} O(p) \times i(p)$$
Computing place invariants

- **Intuitively**, if \( i \) is a place invariant it should assign **weights** to the places such that the **positive** and **negative** effects of every transition are balanced.

- Thus, for any transition \( t = \langle I, O \rangle \) we should have:

\[
\sum_{p \in P} I(p) \times i(p) = \sum_{p \in P} O(p) \times i(p)
\]
Computing place invariants

- Intuitively, if \( i \) is a place invariant it should assign weights to the places such that the positive and negative effects of every transition are balanced.

- Thus, for any transition \( t = \langle I, O \rangle \) we should have:

\[
\sum_{p \in P} I(p) \times i(p) = \sum_{p \in P} O(p) \times i(p)
\]
Computing place invariants

• Intuitively, if \( i \) is a place invariant it should assign weights to the places such that the positive and negative effects of every transition are balanced.

• Thus, for any transition \( t = \langle I, O \rangle \) we should have:

\[
\sum_{p \in P} I(p) \times i(p) = \sum_{p \in P} O(p) \times i(p)
\]
Computing place invariants

- Intuitively, if \( i \) is a place invariant it should assign \textbf{weights} to the places such that the \textbf{positive} and \textbf{negative} effects of every transition are \textbf{balanced}

- Thus, for any transition \( t = \langle I, O \rangle \) we should have:

\[
\sum_{p \in P} I(p) \times i(p) = \sum_{p \in P} O(p) \times i(p)
\]
Computing place invariants

\[ \sum_{p \in P} I(p) \times i(p) = \sum_{p \in P} O(p) \times i(p) \]

means

\[ \sum_{p \in P} \left( O(p) - I(p) \right) \times i(p) = 0 \]
Computing place invariants

\[ \sum_{p \in P} I(p) \times i(p) = \sum_{p \in P} O(p) \times i(p) \]

means

\[ \sum_{p \in P} \left( O(p) - I(p) \right) \times i(p) = 0 \]

\[ t = \langle I, O \rangle \]
Computing place invariants

\[ \sum_{p \in P} I(p) \times i(p) = \sum_{p \in P} O(p) \times i(p) \]

means

\[ \sum_{p \in P} (O(p) - I(p)) \times i(p) = 0 \]

\[ t = \langle I, O \rangle \quad \text{and} \quad W = \begin{pmatrix} \cdots & O(p_1) - I(p_1) & \cdots \\ \cdots & O(p_2) - I(p_2) & \cdots \\ \vdots & \ddots & \vdots \\ \cdots & O(p_n) - I(p_n) & \cdots \end{pmatrix} \]
Computing place invariants

\[
\sum_{p \in P} I(p) \times i(p) = \sum_{p \in P} O(p) \times i(p)
\]

means

\[
\sum_{p \in P} (O(p) - I(p)) \times i(p) = 0
\]

\[t = \langle I, O \rangle\]

\[W = \begin{pmatrix}
\cdots & O(p_1) - I(p_1) & \cdots \\
\cdots & O(p_2) - I(p_2) & \cdots \\
\vdots & \vdots & \vdots \\
\cdots & O(p_n) - I(p_n) & \cdots 
\end{pmatrix}\]
Computing place invariants

\[ \sum_{p \in P} (O(p) - I(p)) \times i(p) = 0 \]

is thus the scalar product of \( i \) and the column of \( W \) that corresponds to transition \( \tau \).
Computing place invariants

\[ \sum_{p \in P} \left( O(p) - I(p) \right) \times i(p) = 0 \]

is thus the scalar product of \( i \) and the column of \( W \) that corresponds to transition \( t \)

Since this must hold for any \( t \), we obtain:
Computing place invariants

\[ \sum_{p \in P} \left( O(p) - I(p) \right) \times i(p) = 0 \]

is thus the scalar product of \( i \) and the column of \( W \) that corresponds to transition \( t \).

Since this must hold for any \( t \), we obtain:

**Theorem:** any solution \( i \) to the following system of equations is a place-invariant:

\[ \sum_{p \in P} \left( O(p) - I(p) \right) \times i(p) = 0 \]
Computing place invariants

\[ \sum_{p \in P} (O(p) - I(p)) \times i(p) = 0 \]

is thus the scalar product of \( i \) and the column of \( W \) that corresponds to transition \( t \).

Since this must hold for any \( t \), we obtain:

**Theorem:** any solution \( i \) to the following system of equations is a place-invariant:

\[ i \times W = 0 \]
Example

\[ W = \begin{pmatrix}
1 & -1 & 1 \\
0 & -1 & 1 \\
0 & 1 & -1
\end{pmatrix} \]
Example

\[ \langle i_1, i_2, i_3 \rangle \times W = 0 \]

\[ W = \begin{pmatrix} 1 & -1 & 1 \\ 0 & -1 & 1 \\ 0 & 1 & -1 \end{pmatrix} \]
Example

\[ W = \begin{pmatrix} 1 & -1 & 1 \\ 0 & -1 & 1 \\ 0 & 1 & -1 \end{pmatrix} \]

\[ \langle i_1, i_2, i_3 \rangle \times W = 0 \]

\[
\begin{cases} 
  i_1 & = 0 \\
  -i_1 - i_2 + i_3 & = 0 \\
  i_1 + i_2 - i_3 & = 0 
\end{cases}
\]
Example

\begin{equation}
W = \begin{pmatrix}
1 & -1 & 1 \\
0 & -1 & 1 \\
0 & 1 & -1 \\
\end{pmatrix}
\end{equation}

\langle i_1, i_2, i_3 \rangle \times W = 0

\begin{align*}
i_1 & = 0 \\
-i_1 - i_2 + i_3 & = 0 \\
i_1 + i_2 - i_3 & = 0
\end{align*}

\begin{align*}
i_1 & = 0 \\
-i_2 + i_3 & = 0 \\
+i_2 - i_3 & = 0
\end{align*}
Example

Any vector of the form \( \langle 0, i, i \rangle \) is a place invariant

\[
\begin{align*}
\langle i_1, i_2, i_3 \rangle \times W & = 0 \\
i_1 & = 0 \\
-i_1 - i_2 + i_3 & = 0 \\
i_1 + i_2 - i_3 & = 0
\end{align*}
\]

\[
W = \begin{pmatrix}
1 & -1 & 1 \\
0 & -1 & 1 \\
1 & 0 & -1
\end{pmatrix}
\]
Proving properties

Let us choose $\langle 0, 1, 1 \rangle$ as place-invariant
Let us choose $\langle 0, 1, 1 \rangle$ as place-invariant.

This means that $p_2$ and $p_3$ are bounded!
Proving properties

Let us choose \( \langle 0, 1, 1 \rangle \) as place-invariant.

This means that \( p_2 \) and \( p_3 \) are bounded!

For any reachable marking \( m \):

\[
0 \ m(p_1) + 1 \ m(p_2) + 1 \ m(p_3) = 0 \ m_0(p_1) + 1 \ m_0(p_2) + 1 \ m_0(p_3)
\]

\[
m(p_2) + m(p_3) = 1
\]
Proving properties

Let us choose $\langle 0, 1, 1 \rangle$ as place-invariant.

This means that $p_2$ and $p_3$ are bounded! 

For any reachable marking $m$:

$$0 \cdot m(p_1) + 1 \cdot m(p_2) + 1 \cdot m(p_3) = 0 \cdot m_0(p_1) + 1 \cdot m_0(p_2) + 1 \cdot m_0(p_3)$$

$$m(p_2) + m(p_3) = 1$$

Hence, mutual exclusion is enforced!
Proving properties

\[ i(M) = i(W_1) = i(W_2) = 1 \text{ and } i(p) = 0 \text{ otherwise} \]

is a place invariant
Proving properties

\[
i(M) = i(W_1) = i(W_2) = 1 \text{ and } i(p) = 0 \text{ otherwise}
\]

is a place invariant

Hence, mutual exclusion is enforced!
Karp & Miller
and
the coverability set
The reachability tree revisited

- Reminder: reachability trees can be infinite

\[ \langle 0p1, p2 \rangle \]
\[ \langle 1p1, p2 \rangle \]
\[ \langle 2p1, p2 \rangle \]
\[ \langle 3p1, p2 \rangle \]
\[ \langle p1, p3 \rangle \]

Figure 2.1: The SMPN $N_{\mu}$.

Figure 2.2: The ... that an SMPN $N = \langle P, T, D^{-}, D^{+}, m_0 \rangle$ naturally defines a transition system $S_N = \langle N_{kP}, m_0, \Rightarrow \rangle$, where $\Rightarrow$ is such
The reachability tree revisited

• **Reminder:** reachability trees can be infinite
The reachability tree revisited

- Reminder: reachability trees can be infinite

Increasing sequences of markings appear on unbounded places
The reachability tree revisited

• Let us summarise this infinite sequence

\[ \langle 0p_1,p_2 \rangle \]
\[ \downarrow \]
\[ \langle 1p_1,p_2 \rangle \]
\[ \downarrow \]
\[ \langle 2p_1,p_2 \rangle \]
\[ \downarrow \]
\[ \langle 3p_1,p_2 \rangle \]
\[ \vdots \]
\[ \downarrow \]
The reachability tree revisited

- Let us summarise this infinite sequence

\[ \langle 0p_1,p_2 \rangle \]
\[ \langle 1p_1,p_2 \rangle \]
\[ \langle 2p_1,p_2 \rangle \]
\[ \langle 3p_1,p_2 \rangle \]
\[ \vdots \]
\[ \text{limit} \]
The reachability tree revisited

- Let us *summarise* this infinite sequence

\[
\langle 0p_1, p_2 \rangle \\
\downarrow \\
\langle 1p_1, p_2 \rangle \\
\downarrow \\
\langle 2p_1, p_2 \rangle \\
\downarrow \\
\langle 3p_1, p_2 \rangle \\
\vdots
\]

\[
\langle \omega p_1, p_2 \rangle
\]
The reachability tree revisited

• Let us *summarise* this infinite sequence

\[ \langle 0p_1,p_2 \rangle \]
\[ \downarrow \]
\[ \langle 1p_1,p_2 \rangle \]
\[ \downarrow \]
\[ \langle 2p_1,p_2 \rangle \]
\[ \downarrow \]
\[ \langle 3p_1,p_2 \rangle \]
\[ \vdots \]
\[ \langle \omega p_1,p_2 \rangle \]

\( \omega \) must be regarded as: “any number of tokens”
The reachability tree revisited

Let us summarise this infinite sequence

\[ \langle 0 \rangle \]
\[ \langle 1 \rangle \]
\[ \langle 2 \rangle \]
\[ \langle 3 \rangle \]
\[ \ldots \]
\[ \langle \omega \rangle \]

\( \omega \) must be regarded as: "any number of tokens"

Main idea of the Karp and Miller algorithm
Karp & Miller

• Propose in 1969 a solution to detect unbounded places of a Petri net
Monotonicity

• Petri nets induce (strongly) monotonic transition systems:

\[ m_3 \succ m_1 \rightarrow^t m_2 \]

• In particular:

if
Monotonicity

• Petri nets induce (strongly) monotonic transition systems:

\[
\begin{align*}
  m_3 & \prec m_4 \\
  m_1 & \xrightarrow{t} m_2
\end{align*}
\]

• In particular:

if
Monotonicity

• Petri nets induce (strongly) monotonic transition systems:

\[
\begin{align*}
  m_3 &\xrightarrow{t} m_4 \\
  m_1 &\xrightarrow{t} m_2
\end{align*}
\]

• In particular:

if
Monotonicity

• Petri nets induce (strongly) monotonic transition systems:

\[
\begin{align*}
  m_3 & \xrightarrow{t} m_4 \\
  m_1 & \xrightarrow{t} m_2
\end{align*}
\]

• In particular:

if
Monotonicity

• Petri nets induce (strongly) monotonic transition systems:

\[
\begin{align*}
    m_3 & \xrightarrow{t} m_4 \\
    m_1 & \xrightarrow{t} m_2
\end{align*}
\]

• In particular:

\[
\langle i_1, i_2, i_3 \rangle
\]
Monotonicity

- Petri nets induce (strongly) monotonic transition systems:

\[
\begin{align*}
    m_3 & \xrightarrow{t} m_4 \\
    m_1 & \xrightarrow{t} m_2
\end{align*}
\]

- In particular:

\[
\langle i_1, i_2, i_3 \rangle \quad \langle i'_1, i'_2, i'_3 \rangle
\]

if

\[
\langle i_1, i_2, i_3 \rangle \quad \langle i'_1, i'_2, i'_3 \rangle
\]
Monotonicity

• Petri nets induce (strongly) monotonic transition systems:

\[ m_3 \xrightarrow{t} m_4 \]

\[ m_1 \xrightarrow{t} m_2 \]

• In particular:

\[ \langle i_1, i_2, i_3 \rangle \rightarrow \langle i'_1, i'_2, i'_3 \rangle \] if
Monotonicity

• Petri nets induce (strongly) monotonic transition systems:

\[
\begin{align*}
m_3 & \xrightarrow{t} m_4 \\
m_1 & \xrightarrow{t} m_2
\end{align*}
\]

• In particular:

\[
\langle i_1, i_2, i_3 \rangle \xrightarrow{\triangleleft} \langle i'_1, i'_2, i'_3 \rangle
\]

if

\[
\langle i_1, i_2, i_3 \rangle \xrightarrow{t} \langle i'_1, i'_2, i'_3 \rangle
\]
Monotonicity

• Petri nets induce (strongly) monotonic transition systems:

• In particular:

\[ \langle i_1, i_2, i_3 \rangle \xrightarrow{m_3} \langle i'_1, i'_2, i'_3 \rangle \]
Monotonicity

• Petri nets induce (strongly) monotonic transition systems:

• In particular:

\[ \langle i_1, i_2, i_3 \rangle \preceq \langle i'_1, i'_2, i'_3 \rangle \quad \text{if} \quad \langle m_1, t, m_2 \rangle \preceq \langle m_3, t, m_4 \rangle \]

then \( p_2 \) is unbounded
Example

\[ \langle 1, 0, 0, 0 \rangle \]
Example

\[ \langle 1, 0, 0, 0 \rangle \]

\[ \langle 0, 0, 1, 0 \rangle \]
Example

\[ \langle 1, 0, 0, 0 \rangle \]

\[ \langle 0, 0, 1, 0 \rangle \]

\[ \langle 0, 0, 0, 1 \rangle \]
Example

\[ \langle 1, 0, 0, 0 \rangle \]

\[ \langle 0, 0, 1, 0 \rangle \]

\[ \langle 0, 0, 0, 1 \rangle \]

\[ \langle 0, 0, 0, 1 \rangle \rightarrow \langle 1, 0, 1, 1 \rangle \]
Example

\[ \langle 1, 0, 0, 0 \rangle \]

\[ \Rightarrow \langle 0, 0, 1, 0 \rangle \]

\[ \langle 0, 0, 0, 1 \rangle \]

\[ \Rightarrow \langle 0, 0, 1, 0 \rangle \]

\[ \Rightarrow \langle 1, 0, 1, 1 \rangle \]
Example

\[
\langle 1, 0, 0, 0 \rangle
\]

\[
\langle 0, 0, 1, 0 \rangle
\]

\[
\langle 0, 0, 0, 1 \rangle
\]

\[
\langle 0, 0, 1 \rangle
\]

\[
\langle 1, 0, 1, 1 \rangle
\]
Example
Example

$\langle 1, 0, 0, 0 \rangle$

$\langle 0, 0, 1, 0 \rangle$

$\langle 0, 0, 0, 1 \rangle$

$\langle 0, 0, 0, 1 \rangle \rightarrow \langle 1, 0, 1, 1 \rangle$

$p_1, p_3$ and $p_4$ are unbounded!
Example

\[ \langle 0, 0, 0, 1 \rangle \rightarrow \langle 1, 0, 1, 1 \rangle \rightarrow \langle 0, 0, 1, 0 \rangle \rightarrow \langle 1, 0, 0, 0 \rangle \rightarrow \langle \omega, 0, \omega, \omega \rangle \]

\[ p_1, p_3 \text{ and } p_4 \text{ are unbounded!} \]
p₁, p₃ and p₄ are unbounded!

Example

ω must be regarded as: “any number of tokens”
This is how we compute the successors of a node $n$:

```
foreach Successor $m'$ of $m$ do
    $m_\omega \leftarrow m'$;
    foreach ancestor $n_i$ s.t. $m_i < m'$ do
        foreach place $p$ s.t. $m_i(p) < m'(p)$ do
            $m_\omega(p) \leftarrow \omega$;
    Add $m_\omega$ as child of $n$;
```
This is how we compute the successors of a node $n$:

```plaintext
foreach Successor $m'$ of $m$ do
    $m_\omega \leftarrow m'$;
    foreach ancestor $n_i$ s.t. $m_i < m'$ do
        foreach place $p$ s.t. $m_i(p) < m'(p)$ do
            $m_\omega(p) \leftarrow \omega$;
    Add $m_\omega$ as child of $n$;
```
This is how we compute the successors of a node $n$:

```plaintext
foreach Successor $m'$ of $m$ do
  $m_\omega \leftarrow m'$;
  foreach ancestor $n_i$ s.t. $m_i < m'$ do
    foreach place $p$ s.t. $m_i(p) < m'(p)$ do
      $m_\omega(p) \leftarrow \omega$;
  Add $m_\omega$ as child of $n$;
```
This is how we compute the successors of a node \( n \):

\[
\text{foreach } \text{Successor } m' \text{ of } m \text{ do }
\]
\[
\quad m_\omega \leftarrow m';
\]
\[
\text{foreach } \text{ancestor } n_i \text{ s.t. } m_i \prec m' \text{ do }
\]
\[
\quad \text{foreach } \text{place } p \text{ s.t. } m_i(p) \prec m'(p) \text{ do }
\]
\[
\quad \quad m_\omega(p) \leftarrow \omega;
\]
\[
\text{Add } m_\omega \text{ as child of } n;
\]
This is how we compute the successors of a node $n$:

```
forall successor $m'$ of $m$ do
  $m_\omega \leftarrow m'$;
  forall ancestor $n_i$ s.t. $m_i < m'$ do
    forall place $p$ s.t. $m_i(p) < m'(p)$ do
      $m_\omega(p) \leftarrow \omega$;
  Add $m_\omega$ as child of $n$;
```
This is how we compute the successors of a node \( n \):

\[
\text{foreach } \text{Successor } m' \text{ of } m \text{ do}
\]
\[
\quad m_\omega \leftarrow m';
\]
\[
\quad \text{foreach ancestor } n_i \text{ s.t. } m_i < m' \text{ do}
\]
\[
\quad \quad \text{foreach place } p \text{ s.t. } m_i(p) < m'(p) \text{ do}
\]
\[
\quad \quad \quad m_\omega(p) \leftarrow \omega;
\]
\[
\quad \text{Add } m_\omega \text{ as child of } n;
\]
This is how we compute the successors of a node $n$:

```latex
deforeach$m'$ of $m$ do
    \begin{align*}
    m_\omega &\leftarrow m'; \\
    \text{foreach ancestor } n_i \text{ s.t. } m_i < m' \text{ do} \\
    \quad \text{foreach place } p \text{ s.t. } m_i(p) < m'(p) \text{ do} \\
    \quad \quad m_\omega(p) &\leftarrow \omega;
    \end{align*}
\end{verbatim}
Add $m_\omega$ as child of $n$;
```
This is how we compute the successors of a node $n$:

```
foreach Successor $m'$ of $m$ do
   $m_\omega \leftarrow m'$;
   foreach ancestor $n_i$ s.t. $m_i < m'$ do
      foreach place $p$ s.t. $m_i(p) < m'(p)$ do
         $m_\omega(p) \leftarrow \omega$;
   Add $m_\omega$ as child of $n$;
```
This is how we compute the successors of a node $n$:

```plaintext
foreach Successor $m'$ of $m$ do
  $m_\omega \leftarrow m'$;
  foreach ancestor $n_i$ s.t. $m_i < m'$ do
    foreach place $p$ s.t. $m_i(p) < m'(p)$ do
      $m_\omega(p) \leftarrow \omega$;
  Add $m_\omega$ as child of $n$;
```
This is how we compute the successors of a node $n$:

```latex
\begin{verbatim}
foreach Successor $m'$ of $m$ do
    $m_\omega \leftarrow m'$;
    foreach ancestor $n_i$ s.t. $m_i < m'$ do
        foreach place $p$ s.t. $m_i(p) < m'(p)$ do
            $m_\omega(p) \leftarrow \omega$;
    Add $m_\omega$ as child of $n$;
\end{verbatim}
```
This is how we compute the successors of a node \( n \):

\[
\text{foreach Successor } m' \text{ of } m \text{ do} \\
\quad m_\omega \leftarrow m'; \\
\text{foreach ancestor } n_i \text{ s.t. } m_i < m' \text{ do} \\
\quad \text{foreach place } p \text{ s.t. } m_i(p) < m'(p) \text{ do} \\
\quad \quad m_\omega(p) \leftarrow \omega; \\
\text{Add } m_\omega \text{ as child of } n;
\]

Karp & Miller
Acceleration
This is how we compute the successors of a node $n$:

\[
\text{foreach } \text{Successor } m' \text{ of } m \text{ do } \\
m_\omega \leftarrow m' ; \\
\text{foreach ancestor } n_i \text{ s.t. } m_i < m' \text{ do } \\
\text{foreach place } p \text{ s.t. } m_i(p) < m'(p) \text{ do } \\
m_\omega(p) \leftarrow \omega ; \\
\text{Add } m_\omega \text{ as child of } n ;
\]
Karp & Miller
Stopping a branch

This node doesn’t have to be developed
Example of K&M tree

\[\langle 0, 1, 0 \rangle\]
Example of K&M tree

\[ \langle 0, 1, 0 \rangle \]

(0, 1, 0) \xrightarrow{t_1} (1, 1, 0) \nRightarrow (0, 1, 0)
Example of K&M tree

\[ \langle 0, 1, 0 \rangle \]

\[ \langle \omega, 1, 0 \rangle \]

\[ (0,1,0) \xrightarrow{t_1} (1,1,0) \succ (0,1,0) \]
Example of K&M tree

\[ \langle 0, 1, 0 \rangle \]

\[ \langle \omega, 1, 0 \rangle \]

\[ \langle \omega, 1, 0 \rangle \]

\[ (0,1,0) \xrightarrow{t_1} (1,1,0) \xrightarrow{t_3} (0,1,0) \]
Example of K&M tree

\[
\langle 0, 1, 0 \rangle
\]

\[
\langle \omega, 1, 0 \rangle
\]

\[
\langle \omega, 0, 1 \rangle
\]

\[
(0, 1, 0) \xrightarrow{t_1} (1, 1, 0) \xrightarrow{t_1} (0, 1, 0)
\]

Figure 2.1: The SMPN \( N_\mu \).

Figure 2.2: The ... that an SMPN \( N = \langle P, T, D^-, D^+, m_0 \rangle \) naturally defi nes a transition system \( SN = \langle \mathcal{N}_P, m_0, \Rightarrow \rangle \), where \( \Rightarrow \) is such
Example of K&M tree

\[
\langle 0, 1, 0 \rangle \\
\langle \omega, 1, 0 \rangle \\
\langle \omega, 0, 1 \rangle
\]

\[
\langle 0, 1, 0 \rangle \xrightarrow{t_1} (1,1,0) \xrightarrow{t_1} (0,1,0)
\]
Example of K&M tree

\[ \langle 0, 1, 0 \rangle \]
\[ \langle \omega, 1, 0 \rangle \]
\[ \langle \omega, 0, 1 \rangle \]
\[ \langle \omega, 0, 1 \rangle \]
\[ \langle \omega, 1, 0 \rangle \]

\( (0, 1, 0) \xrightarrow{t_1} (1, 1, 0) \Rightarrow (0, 1, 0) \)
Properties

• **Theorem**: the K&M tree is *always finite*.

• **Idea of the proof**:
  
  • if the net is not bounded, it is because of some *infinite increasing sequence* of markings.
  
  • such sequences are detected in a *finite amount of time* by adding $\omega$ in the unbounded places.
Properties

• **Theorem**: a net is **bounded** iff there is no node containing an $\omega$ in its K&M tree.

• **Theorem**: place $p$ is **unbounded** iff there exists a node labeled by $m$ in the K&M tree s.t. $m(p) = \omega$.

• **Theorem**: transition $t$ is **semi-live** iff there exists a node labeled by $m$ in the K&M tree s.t. $t$ can fire in $m$. 
Example

\[
\begin{align*}
\langle 0, 1, 0 \rangle \\
\langle \omega, 1, 0 \rangle &\xrightarrow{t_1} \langle \omega, 1, 0 \rangle \\
\langle \omega, 1, 0 \rangle &\xrightarrow{t_2} \langle \omega, 0, 1 \rangle \\
\langle \omega, 0, 1 \rangle &\xrightarrow{t_1} \langle \omega, 0, 1 \rangle \\
\langle \omega, 0, 1 \rangle &\xrightarrow{t_3} \langle \omega, 1, 0 \rangle \\
\end{align*}
\]
Example

\[
\begin{align*}
&\langle 0, 1, 0 \rangle \\
&\langle \omega, 1, 0 \rangle \\
&\langle \omega, 0, 1 \rangle \\
&\langle \omega, 0, 0 \rangle \\
&\langle \omega, 1, 0 \rangle
\end{align*}
\]

\[t_1\]

\[t_2\]

\[t_3\]

\[t_2\] is semi-live
Example

\[
\begin{align*}
\langle 0, 1, 0 \rangle \\
\langle \omega, 1, 0 \rangle \\
\langle \omega, 0, 1 \rangle \\
\langle \omega, 0, 1 \rangle \\
\langle \omega, 1, 0 \rangle \\
\langle \omega, 1, 0 \rangle
\end{align*}
\]

\( t_2 \) is semi-live

\( p_2 \) and \( p_3 \) are bounded
Example

\[ \langle 0, 1, 0 \rangle \]
\[ \langle \omega, 1, 0 \rangle \]
\[ \langle \omega, 0, 1 \rangle \]
\[ \langle \omega, 0, 1 \rangle \]
\[ \langle \omega, 1, 0 \rangle \]

\[ t_1 \]
\[ t_2 \]
\[ t_3 \]

\( t_2 \) is semi-live

\( p_1 \) is unbounded

\( p_2 \) and \( p_3 \) are bounded
Example

\[
\begin{align*}
&\langle 0, 1, 0 \rangle \\
&\langle \omega, 1, 0 \rangle \\
&\langle \omega, 0, 1 \rangle \\
&\langle \omega, 0, 1 \rangle \\
&\langle \omega, 1, 0 \rangle
\end{align*}
\]

\(t_2\) is semi-live

\(p_2\) and \(p_3\) are bounded

\(p_1\) is unbounded

The net is unbounded
Coverability set

- **Question**: what is the relationship between:
  - the set of reachable markings and
  - the set of labels of the nodes of the K&M tree?
Coverability set

- **Question:** what is the relationship between:
  - the set of *reachable markings* and
  - the set of *labels* of the nodes of the K&M tree?

might be infinite
Question: what is the relationship between:

- the set of reachable markings and
- the set of labels of the nodes of the K&M tree?
Example
Example
Example
Example
Example

• Set of reachable markings:

\[ \{ \langle 1, 0, 3, i \rangle, \langle 0, 1, 3, i \rangle \mid i \geq 0 \} \]

• Set of nodes of the K&M tree:

\[ \{ \langle 1, 0, 0 \rangle, \langle 1, 0, \omega \rangle, \langle 0, 1, \omega \rangle \} \]

• This set “represents”:

\[ \{ \langle 1, 0, i \rangle, \langle 0, 1, i \rangle \mid i \geq 0 \} \]
Example

• Set of reachable markings:

\{ \langle 1, 0, 3.i \rangle, \langle 0, 1, 3.i \rangle \mid i \geq 0 \} 

• Set of nodes of the K&M tree:

\{ \langle 1, 0, 0 \rangle, \langle 1, 0, \omega \rangle, \langle 0, 1, \omega \rangle \} 

• This set “represents”:

\{ \langle 1, 0, i \rangle, \langle 0, 1, i \rangle \mid i \geq 0 \} 

Clearly: \quad \neq 

Clearly, the K&M set contains more markings than the set of reachable markings:

$\{ \langle 1, 0, 3.1 \rangle, \langle 0, 1, 3.1 \rangle \mid i \geq 0 \}$ vs $\{ \langle 1, 0, i \rangle, \langle 0, 1, i \rangle \mid i \geq 0 \}$

- However, for every marking $m$ in the K&M set, there exists a reachable marking $m'$ s.t.:

  $m' \succeq m$
Clearly, the K&M set contains more markings than the set of reachable markings:

\[
\{ \langle 1, 0, 3.i \rangle, \langle 0, 1, 3.i \rangle \mid i \geq 0 \} \subseteq \{ \langle 1, 0, i \rangle, \langle 0, 1, i \rangle \mid i \geq 0 \}
\]

However, for every marking \( m \) in the K&M set, there exists a reachable marking \( m' \) s.t.:

\[
m' \succ m
\]

\[
\text{Reach} \ K&\text{M} = \{ m \mid \text{there is } m' \text{ in set with } m' \succ m \} + \{ m' \}
\]

\[
\text{Reach } KS = \{ m \mid \text{there is } m' \text{ in set with } m' \succ m \}
\]
Downward-closure

• Let us assume that any natural number \( i \) is s.t. \( i < \omega \).

• Let \( m \) be a marking (possibly with \( \omega \)), then its \textit{downward-closure} is the set:

\[
\downarrow m = \{ m' | m' \preceq m \}
\]

• Let \( S = \{ m_1, m_2, \ldots, m_k \} \) be a \textit{set of markings}, then:

\[
\downarrow S = \downarrow m_1 \cup \downarrow m_2 \cup \ldots \cup \downarrow m_k
\]
Examples in 2 dim.

\{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle 3, 1 \rangle \}

\{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle \omega, 1 \rangle \}
Examples in 2 dim.

\[ \{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle 3, 1 \rangle \} \]

\[ \{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle \omega, 1 \rangle \} \]
Examples in 2 dim.

\[ \{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle 3, 1 \rangle \} \]

\[ \{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle \omega, 1 \rangle \} \]
Examples in 2 dim.

\{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle 3, 1 \rangle \} 

\{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle \omega, 1 \rangle \}
Examples in 2 dim.

\[ \{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle 3, 1 \rangle \} \]

\[ \{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle \omega, 1 \rangle \} \]
Examples in 2 dim.

\{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle 3, 1 \rangle \}
Examples in 2 dim.

\[ \{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle 3, 1 \rangle \} \]

\[ \{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle \omega, 1 \rangle \} \]
Examples in 2 dim.

\{ \langle 1, 2 \rangle , \langle 2, 4 \rangle , \langle 3, 1 \rangle \}
Examples in 2 dim.

\[
\{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle 3, 1 \rangle \}
\]

\[
\{ \langle 1, 2 \rangle, \langle 2, 4 \rangle, \langle \omega, 1 \rangle \}
\]
Properties of the K&M tree

• The set of all the markings that appear in a K&M tree is called a coverability set of the net.

• Notation: $\text{Cover}(N)$

• Theorem: $\downarrow \text{Cover}(N) = \downarrow \text{Reach}(N)$

• Theorem: $\text{Reach}(N) \subseteq \downarrow \text{Cover}(N)$

• Hence, $\downarrow \text{Cover}(N)$ is a finite over-approximation of $\text{Reach}(N)$
Example

Reach(N) = \{ \langle i, 1, 0 \rangle, \langle i, 0, 1 \rangle \mid i \geq 0 \}

Cover(N) = \downarrow \{ \langle \omega, 1, 0 \rangle, \langle \omega, 0, 1 \rangle \} = Reach(N) \cup \{ \langle 0, 0, 0 \rangle \}
Recently, we have defined a new algorithm to compute the coverability set of a Petri net.

It is several order of magnitudes more efficient than K&M
The coverability problem
Reachability: a natural question

- The reachability problem: given a marking $m$ is it reachable from $m_0$?
Reachability: a natural question

- The reachability problem: given a marking $m$ is it reachable from $m_0$?
Reachability: a natural question

- The reachability problem: given a marking $m$ is it reachable from $m_0$?
Reachability: a natural question

- The reachability problem: given a marking $m$ is it reachable from $m_0$?
Reachability: a natural question

- The reachability problem: given a marking $m$, is it reachable from $m_0$?
Reachability: a natural question

- The reachability problem: given a marking $m$ is it reachable from $m_0$?
Reachability: a natural question ??

• In the case of Petri nets, asking whether a given marking is reachable does not always make sense...

• ... because Petri nets are monotonic
Example
Example

p1

p2

p3

nasty

2
Example

Question is $\langle 0, 0, 2, 0 \rangle$ reachable?
Example

Better question is a marking with at least 2 tokens in p₃ reachable?

Question is x₀, 0, 2, 0 y reachable?
Example

Better question is a marking \( m \cong \langle 0, 0, 2, 0 \rangle \) reachable?

nasty

KA-BOOM
The coverability problem

Does there exist a reachable marking which is larger than some marking $b$?
The coverability problem

Does there exist a reachable marking which is larger than some marking \( b \)?
The coverability problem

Does there exist a reachable marking which is larger than some marking $b$?
The coverability problem

Does there exist a reachable marking which is larger than some marking $b$?
The coverability problem

Does there exist a reachable marking which is larger than some marking $b$?
The coverability problem

Does there exist a reachable marking which is larger than some marking $b$?
The coverability problem

Does there exist a reachable marking which is larger than some marking \( b \)?
The coverability problem
The **coverability** problem

$m_0$
The **coverability problem**

\[ m_0 \quad b \]
The **coverability problem**
The coverability problem
The coverability problem

Reach(N)
The coverability problem

Reach(\(N\))

\(m_0\)
The coverability problem

Reach(ℕ)
The coverability problem

$\text{Reach}(\mathbb{N})$

$m_0$

$\{m \mid m \geq b\}$
The coverability problem

- Two alternative definitions:
  - Is there a reachable marking \( m \) s.t. \( m \succcurlyeq b \)?
  - Does \( \text{Reach}(N) \cap \{m \mid m \succcurlyeq b\} \neq \emptyset \)?
Coverability: a natural question (indeed)

• **Coverability** might be regarded as the *most natural reachability question* in the framework of Petri nets

• Besides, coverability is *much more easily solved* than *reachability*
Safety Properties
Safety Properties

A marking $m$ is unsafe when $m \succeq \langle 0, 0, 2, 0 \rangle$
Safety Properties

No more than one token at a time in this place!!

A marking $m$ is unsafe when $m \succeq \langle 0, 0, 2, 0 \rangle$
First idea

- Use the **coverability set**!
- **Remember**: the coverability set over-approximates the reachable states:

\[
\text{Reach}(N) \subseteq \downarrow \text{Cover}(N)
\]
First idea

- Use the **coverability set**!
- **Remember**: the coverability set **over-approximates** the reachable states:

\[ \text{Reach}(N) \subseteq \downarrow \text{Cover}(N) \]
First idea

- Use the coverability set!
- **Remember**: the coverability set over-approximates the reachable states:

\[
\text{Reach}(N) \subseteq \downarrow \text{Cover}(N)
\]
First idea

• Use the **coverability set**!

• **Remember**: the coverability set **over-approximates** the reachable states:

\[
\text{Reach}(N) \subseteq \downarrow \text{Cover}(N)
\]
First idea
First idea

Reach(N)
First idea

Reach(\(N\)) \downarrow \text{Cover}(\(N\))
First idea

Reach(N) \Downarrow \text{Cover}(N) \quad U
First idea

\[
\downarrow \text{Cover}(N) \cap U = \emptyset
\]

implies

\[
\text{Reach}(N) \cap U = \emptyset
\]
What if?

- There is $m$ in $\downarrow \text{Cover}(N) \cap U$
- Hence, there is $m' \succeq m$ which is in $\text{Reach}(N)$
- However, any $m' \succeq m$ is also in $U$
- Thus, there is $m'$ both in $\text{Reach}(N)$ and $U$
• There is $m$ in $\downarrow \text{Cover}(N) \cap U$

• Hence, there is $m' \supseteq m$ which is in $\text{Reach}(N)$

• However, any $m' \supseteq m$ is also in $U$

• Thus, there is $m'$ both in $\text{Reach}(N)$ and $U$
What if?

- There is $m$ in $\downarrow \text{Cover}(N) \cap U$
- Hence, there is $m' \sqsupseteq m$ which is in Reach($N$)
- However, any $m' \sqsupseteq m$ is also in $U$
- Thus, there is $m'$ both in Reach($N$) and $U$
What if?

- There is $m$ in $\downarrow \text{Cover}(N) \cap \text{U}$
- Hence, there is $m' \succ m$ which is in $\text{Reach}(N)$
- However, any $m' \succ m$ is also in $\text{U}$
- Thus, there is $m'$ both in $\text{Reach}(N)$ and $\text{U}$
What if?

\[ \downarrow \text{Cover}(N) \]

\[ \text{Reach}(N) \]

\[ \cup \]
What if?

\[ \text{Reach}(N) \cap U = \emptyset \]

implies

\[ \downarrow \text{Cover}(N) \cap U = \emptyset \]
Coverability set and coverability problem
Coverability set and coverability problem

• Theorem:

\[ \text{Reach}(N) \cap U = \emptyset \text{ iff } \downarrow \text{Cover}(N) \cap U = \emptyset \]
Coverability set and coverability problem

- **Theorem:**
  \[ \text{Reach}(N) \cap U = \emptyset \iff \downarrow \text{Cover}(N) \cap U = \emptyset \]

- **Nice,**...
Coverability set and coverability problem

- Theorem:

  \[ \text{Reach}(N) \cap U = \emptyset \iff \downarrow \text{Cover}(N) \cap U = \emptyset \]

- Nice,...

- ...but \(U\) and \(\downarrow \text{Cover}(N)\) might both be infinite!
Coverability set and coverability problem

• Theorem:
  \[ \text{Reach}(N) \cap U = \emptyset \quad \text{iff} \quad \downarrow \text{Cover}(N) \cap U = \emptyset \]

• Nice,...

• ...but \( U \) and \( \downarrow \text{Cover}(N) \) might both be infinite!

• How do we test that \( \downarrow \text{Cover}(N) \cap U = \emptyset \)?
Coverability set and coverability problem
Coverability set and coverability problem
Coverability set and coverability problem

\[ \text{Coverability set and } \text{coverability problem} \]
Coverability set and coverability problem

c \supseteq b

\downarrow \text{Cover}(N)

p_1

p_2

U
Coverability set and coverability problem

All we need to remember is the (finite) set of minimal elements $\text{Min}(U)$.
Coverability set and coverability problem

All we need to remember is the (finite) set of minimal elements $\text{Min}(U)$.
Coverability set and coverability problem

\[ \downarrow \text{Cover}(N) \cap U \neq \emptyset \]
iff
there is \( c \) in \( \text{Cover}(N) \) and \( b \) in \( \text{Min}(U) \) s.t.
\( c \succ b \)

All we need to remember is the (finite) set of minimal elements \( \text{Min}(U) \)
Backward approach

\[ U = \{ m | m \succ b \} \]
Backward approach

\[ U = \{ m \mid m \geq b \} \]
Backward approach

All the markings that can reach $U$ in one step

$U = \{m|m \geq b\}$
Backward approach

\[ U = \{m|m \geq b\} \]
Backward approach

\[ \mathcal{U} = \{ m \mid m \succ b \} \]
Backward approach

\[ U = \{ m \mid m \geq b \} \]
Backward approach

In the end, we want to obtain all the markings that can reach $U$ in any number of steps.

$$U = \{m|m \geq b\}$$
Backward approach

In the end, we want to obtain all the markings that can reach $U$ in any number of steps.

$U = \{m|m \leq b\}$
Backward approach

In the end, we want to obtain all the markings that can reach $U$ in any number of steps.

$U = \{m|m \geq b\}$

$\text{Pre}^*(U)$
Backward approach

In the end, we want to obtain all the markings that can reach $U$ in any number of steps.

$U = \{m | m \succeq b\}$

$\text{Pre}^*(U)$

$m_0$
Backward Approach

• Clearly:

\[ m_0 \text{ is in } \text{Pre}^*(U) \iff \text{Reach}(N) \cap U \neq \emptyset \]

• **Question**: can we **compute** \( \text{Pre}^*(U) \)?

• **Yes**!
Predecessor operator

- Symmetrically to the Post, we define the predecessor operator:

  \[ \text{Pre}(m) = \{ m' \mid m \text{ is in Post}(m') \} \]

- Let us consider the sequence

  \[ U, \text{Pre}(U), \text{Pre}(\text{Pre}(U)), \text{Pre}(\text{Pre}(\text{Pre}(U))), ... \]

- **Theorem:** After a finite amount of steps, the sequence stabilises, and we obtain \( \text{Pre}^*(U) \)
• **Efficient data structures** to implement this algorithm have been defined by researchers of the verification group at ULB.
More on Petri nets
Marking dependent effects
The effect of a transition is not constant anymore, but depends on the current marking.
Marking-dependent effect

- The effect of a transition is not constant anymore, but depends on the current marking.

Mathematical expression:

\[ m(p_1) + m(p_2) \]

Diagram:

- Transition labeled with 2
- Places labeled with \( p_1, p_2, p_3, p_4 \)
- Arrows indicating the flow of tokens
- Tokens distributed among places
Marking-dependent effect - resets

- In particular, we can define resets.

\[ \text{reset of } p_2 \]
Marking-dependent effect - resets

- In particular, we can define resets.

\[ \text{reset of } p_2 \]
Marking-dependent effect - resets

In particular, we can define resets.

reset of $p_2$
Reset nets

- When we have only classical PN transitions + resets:
  - Coverability is **decidable**
  - Boundedness is **decidable**
  - Place boundedness is **undecidable**
  - The coverability set is **not computable**
Marking-dependent effect - transfers

- In particular, we can define transfers.

\[ p_1 \rightarrow p_2 \quad m(p_2) \rightarrow p_3 \]

transfer from \( p_2 \) to \( p_3 \)
Marking-dependent effect - transfers

- In particular, we can define transfers.

\[ \text{transfer from } p_2 \text{ to } p_3 \]
Usefulness of transfers

- Modelisation of broadcasts:
  - A single message is sent to every process
  - Each process that receives the message moves to another state.
Transfer nets

• When we have only classical PN transitions + transfers:
  • Coverability is decidable
  • Boundedness is decidable
  • Place boundedness is undecidable
  • The coverability set is not computable
Marking-dependent effect - zero-test

- In particular, we can define test for zero.

enabled only if $p_2$ is empty
Marking-dependent effect - zero-test

- In particular, we can define test for zero.

enabled only if $p_2$ is empty
Marking-dependent effect - zero-test

- In particular, we can define test for zero.

enabled only if $p_2$ is empty
Test for zero

• Once we have test-for-zero everything becomes undecidable.
Coloured Petri nets
Coloured Petri nets

- **Popular extension** of the basic model.
- Introduced by the team of Kurt Jensen, in the ‘80s
- used in many applications
Coloured Petri nets

• Idea: add colours to the tokens
• Allow to distinguish between different types of tokens
• The colours can model data carried by the processes
• Transitions are aware of the colours
Phone example

• We have a set of customers:
  • Each customer is represented by a token.
  • Color of the token = Phone number.
  • A customer is either inactive or connected.
Phone example

- A **pair** of **inactive** customers can **establish** a connection.
- We want to **distinguish** between sender and receiver.

The transition consumes a **sender** $x$ and a **receiver** $y$

Connections are recorded here as tokens whose color is a **pair** $(\text{snd, rcv})$
Phone example

- A **pair of inactive** customers can **establish** a connection.

- We want to **distinguish** between sender and receiver.

The transition consumes a **sender** $x$ and a **receiver** $y$.

Connections are recorded here as tokens whose color is a **pair** $(snd, rcv)$. 
Phone example

- The connection can be **closed** either by the **sender** or by the **receiver**.
Phone example

- The connection can be **closed** either by the **sender** or by the **receiver**.
Phone example
Coloured Petri nets

• Several analysis methods have been developed for this model (finite number of colours)

• e.g.: invariants

• Some results can be achieved when the colors have good properties
Practical Tools: Pep
Practical Tools: Pep

• = language to describe PN + a suite of tools to analyse them:
  • simulation
  • verification (SPIN, SMV)
  • translation from/to different formalisms
  • ...

• Everything can be accessed through a single graphical interface (Tcl/Tk)

http://theoretica.informatik.uni-oldenburg.de/~pep/
Practical Tools: CPNTools
Practical Tools: CPNTools

• Specialised in Coloured Petri nets

• **Features** similar to Pep:
  • modelisation
  • simulation
  • state space analysis
  • ...

http://wiki.daimi.au.dk/cpntools/cpntools.wiki
Conclusion
To conclude

• Petri nets (and their extensions) are a nice tool to reason about concurrent systems:
  • very popular
  • non-trivial decision problems are decidable
  • appealing graphical representation
  • tool supported
To conclude

- There is still a lot to explore:
  - other *extensions*:
    - Time Petri nets
    - Timed Petri nets
    - Stochastic Petri nets,...
To conclude

- There is still a lot to explore:
  - **Subclasses of Petri nets:**
    - 1-safe
    - marked graphs
    - free-choice
    - conflict free
    - ...
  - Some problems are *easier* to decide on these subclasses.
To conclude

• There is still a lot to explore:
  • other problems:
    • liveness
    • deadlock freedom
    • semi-linearity
    • non-termination
    • ...

...
To conclude

• Very active field of research!
• Several conference and journals entirely dedicated to Petri nets
• ... just hop in and join us!

http://www.informatik.uni-hamburg.de/TGI/PetriNets/
Some references

- **On Petri nets:**

- **On Petri nets with marking dependent effects:**
Some references

- **On the coverability problem:**
Some references

- **On Coloured Petri nets:**
Some references

- **On other extensions of Petri nets:**

- **On net unfoldings:**

- **More at:**
  - [http://www.informatik.uni-hamburg.de/TGI/PetriNets/introductions/](http://www.informatik.uni-hamburg.de/TGI/PetriNets/introductions/)
Questions ?