Réunion d’équipe - 27 novembre 2014

Verifying Modal Workflow Specifications using Constraint Solving

Hadrien Bride  Olga Kouchnarenko  Fabien Peureux

Labex ACTION program (contract ANR-11-LABX-0001-01)
Outline

1. Introduction
2. Workflow Net / Modal Specification
3. Verification of Modal Specifications
4. Experimentation: Issue Tracking Systems
5. Conclusion
Nowadays workflows are extensively used by companies.

**Why ?**
- To improve organizational efficiency and productivity.

**How ?**
- By managing the tasks and steps of business processes.
Introduction

- **Modal specifications**
  - Refinement approaches for workflow development
  - Loose specifications with restrictions on activities
  - Activities are *necessary or admissible*

---

**Proposals**

- Verification of modal specifications of workflows
- Workflow Petri nets
- Constraint solving as a computational tool
Outline

1. Introduction
2. Workflow Net / Modal Specification
3. Verification of Modal Specifications
4. Experimentation : Issue Tracking Systems
5. Conclusion
Workflow Petri nets (WF-nets)

- Suited for modelling and analysing workflows exhibiting behaviours such as:
  - concurrency
  - conflict
  - and causal dependency between events

- Some advantages of WF-nets:
  - Represent finite or infinite-state processes in a readable graphical and/or a formal manner
  - Several important verification problems, like reachability or soundness, are decidable
Definition (Petri net)

A Petri net is a tuple \((P, T, F)\) where \(P\) is a finite set of places, \(T\) is a finite set of transitions \((P \cap T = \emptyset)\), and \(F \subseteq (P \times T) \cup (T \times P)\) is a set of arcs.
Petri nets

Definition (PreSet)

Let $g \in P \cup T$ and $G \subseteq P \cup T$:

- $\bullet g = \{g' | (g', g) \in F\}$ and $\bullet G = \bigcup_{g \in G} \bullet g$

Example

- $\bullet \{P6, P1\} =$
Petri nets

Definition (PreSet)

Let \( g \in P \cup T \) and \( G \subseteq P \cup T \):

- \( \cdot g = \{ g' | (g', g) \in F \} \) and \( \cdot G = \bigcup_{g \in G} \cdot g \)

Example

- \( \cdot \{ P6, P1 \} = \{ T0, T3, T4 \} \)
**Petri nets**

**Definition (PostSet)**

Let $g \in P \cup T$ and $G \subseteq P \cup T$:

- $g^\bullet = \{g'|(g, g') \in F\}$ and $G^\bullet = \bigcup_{g \in G} g^\bullet$

**Example**

- $\{P6, P1\}^\bullet =$
**Petri nets**

Definition (PostSet)

Let \( g \in P \cup T \) and \( G \subseteq P \cup T \):

\[
g^* = \{ g' | (g, g') \in F \} \quad \text{and} \quad G^* = \bigcup_{g \in G} g^*
\]

Example

\[
\{P6, P1\}^* = \{ T1, T5 \}
\]
Petri nets

- A marking of Petri net is a function $M : P \rightarrow \mathbb{N}$.
- A transition $t$ is enabled if and only if $\forall p \in \bullet t, M(p) \geq 1$.
- When an enabled transition $t$ is fired, it consumes one token from each place of $\bullet t$ and produces one token for each place of $t^\bullet$.
Petri nets

- A *marking* of Petri net is a function $M : P \to \mathbb{N}$.
- A transition $t$ is *enabled* if and only if $\forall p \in \cdot t, M(p) \geq 1$.
- When an *enabled* transition $t$ is *fired*, it *consumes* one token from each place of $\cdot t$ and *produces* one token for each place of $t^\ast$.
**Definition (Trap)**

Let $N \subseteq P$ such that $N \neq \emptyset$:

- $N$ is a trap if and only if $N^\bullet \subseteq \bullet N$.

**Example**

- $\{P_0, P_1\}^\bullet \subseteq \bullet \{P_0, P_1\} \iff \{T_0, T_1\} \subseteq \{T_0, T_1, T_5\}$
### Definition (Siphon)

Let $N \subseteq P$ such that $N \neq \emptyset$:

- $N$ is a siphon if and only if $\bullet N \subseteq N^\bullet$.

### Example

- $\bullet \{ P6 \} \subseteq \{ P6 \}^\bullet \iff \{ T3, T4 \} \subseteq \{ T3, T4, T5 \}$
**Definition (WF-net)**

A Petri net $PN = (P, T, F)$ is a WF-net (Workflow net) if and only if $PN$ have two special places $i$ and $o$, where $\bullet i = \emptyset$ and $o^\bullet = \emptyset$, and for each node $n \in (P \cup T)$ there exists a path from $i$ to $o$ passing through $n$. 
Modal Specifications

- Impose restrictions on transitions
- In the framework of WF-nets:
  - *may*-transition – transition fired by **at least one** execution
  - *must*-transition – transition fired by **all** executions

**Weakness**
- only individual transitions are concerned with
Example of an Industrial Business Workflow

- A proprietary issue tracking system used to manage bugs and issues requested by the customers of a tool provider company.
Example of an Industrial Business Workflow

Goal

- Verify some required behavioural properties
  - Specification or design stage of the development
  - Derived from textual requirements and business analyst expertise

Examples of properties:
- either the scenario SubA or the scenario SubB (and not both of them) must be executed
- when the scenario SubB is considered then the user must login
- a service request may be upgraded to a critical situation request
Extended Modal Specifications

- Express requirements on several transitions and on their causalities

**Definition (Well-formed Modal Formula)**

Let $S$ be the language of well-formed *modal* specification formulae:

- $\forall t \in T, t$ is a well-formed *modal* formula.
- Given $A_1, A_2 \in S$, $A_1 \land A_2$, $A_1 \lor A_2$, and $\neg A_1$ are well-formed *modal* formulae.

- A *modal* specification formula $m \in S$ can be interpreted as:
  - a *may*-formula – a behaviour that has to be ensured by *at least one* correct execution
  - a *must*-formula – a behaviour that has to be ensured by *all* correct executions
Example of an Industrial Business Workflow

- Examples of properties and their corresponding modal formulae:
  - either the scenario SubA or the scenario SubB (and not both of them) must be executed
    
    \[ PN \models_{\text{must}} (SubA \land \neg SubB) \lor (SubB \land \neg SubA) \]
  
  - when the scenario SubB is considered then the user must login
    
    \[ PN \models_{\text{must}} SubB \Rightarrow Login \]
  
  - a service request may be upgraded to a critical situation request
    
    \[ PN \models_{\text{may}} SR\_UpgradeToCRITSIT \]
Outline

1 Introduction

2 Workflow Net / Modal Specification

3 Verification of Modal Specifications

4 Experimentation : Issue Tracking Systems

5 Conclusion
Modelling Executions of WF-nets

Definition (Fundamental equation – Murata89)
Let $PN = (P, T, F)$ be a WF-net and $M_a, M_b$ two markings of $PN$, the minimum places potential constraint system $\varphi(PN, M_a, M_b)$ associated with it is:

$$\forall p \in P. \nu(p) = \sum_{t \in p^\bullet} \nu(t) + M_b(p) = \sum_{t \in \bullet p} \nu(t) + M_a(p) \quad (1)$$

where $\nu : P \times T \to \mathbb{N}$ is a valuation function.
Modelling Executions of WF-nets

**Definition (Fundamental equation – Murata89)**

Let \( PN = (P, T, F) \) be a WF-net and \( M_a, M_b \) two markings of \( PN \), the minimum places potential constraint system \( \varphi(PN, M_a, M_b) \) associated with it is:

\[
\forall p \in P. \nu(p) = \sum_{t \in p^*} \nu(t) + M_b(p) = \sum_{t \in \bullet p} \nu(t) + M_a(p) \quad (1)
\]

where \( \nu : P \times T \rightarrow \mathbb{N} \) is a valuation function.
Modelling Executions of WF-nets

Definition (Fundamental equation – Murata89)

Let $PN = (P, T, F)$ be a WF-net and $Ma, Mb$ two markings of $PN$, the minimum places potential constraint system $\varphi(PN, Ma, Mb)$ associated with it is:

$$\forall p \in P. \nu(p) = \sum_{t \in p'} \nu(t) + Mb(p) = \sum_{t \in p} \nu(t) + Ma(p) \quad (1)$$

where $\nu : P \times T \rightarrow \mathbb{N}$ is a valuation function.
Modelling Executions of WF-nets

Definition (Fundamental equation – Murata89)

Let $PN = (P, T, F)$ be a WF-net and $M_a, M_b$ two markings of $PN$, the minimum places potential constraint system $\varphi(PN, M_a, M_b)$ associated with it is:

$$\forall p \in P. \nu(p) = \sum_{t \in p^\bullet} \nu(t) + M_b(p) = \sum_{t \in \bullet p} \nu(t) + M_a(p) \quad (1)$$

where $\nu : P \times T \rightarrow \mathbb{N}$ is a valuation function.
Modelling Executions of WF-nets

Definition (Fundamental equation – Murata89)

Let $PN = (P, T, F)$ be a WF-net and $M_a, M_b$ two markings of $PN$, the minimum places potential constraint system $\varphi(PN, M_a, M_b)$ associated with it is:

$$\forall p \in P. \nu(p) = \sum_{t \in p^\bullet} \nu(t) + M_b(p) = \sum_{t \in \bullet p} \nu(t) + M_a(p) \quad (1)$$

where $\nu : P \times T \rightarrow \mathbb{N}$ is a valuation function.
Modelling Executions of WF-nets

Definition (Fundamental equation – Murata89)

Let $PN = (P, T, F)$ be a WF-net and $M_a, M_b$ two markings of $PN$, the minimum places potential constraint system $\varphi(PN, M_a, M_b)$ associated with it is:

$$\forall p \in P. \nu(p) = \sum_{t \in p^\bullet} \nu(t) + M_b(p) = \sum_{t \in \bullet p} \nu(t) + M_a(p) \quad (1)$$

where $\nu : P \times T \rightarrow \mathbb{N}$ is a valuation function.
The set of solutions of \( \varphi(PN, M_i, M_o) \) constitutes an over-approximation of the set of correct executions of \( PN \).
Existence of Siphon

Theorem

Let $\theta(PN)$ be the following constraint system associated with a WF-net $PN = (P, T, F)$:

$\forall p \in P, \forall t \in \bullet p. \sum_{p' \in \bullet t} \xi(p') \geq \xi(p)$, and $\sum_{p \in P} \xi(p) > 0$

where $\xi : P \rightarrow \{0, 1\}$ is a valuation function. $PN$ contains a siphon if and only if there is a valuation satisfying $\theta(PN)$. 
Structural Properties over Executions

**Theorem**

Let $PN = (P, T, F)$ a WF-net, $M_a, M_b$ two markings of $PN$, and $\nu : P \times T \rightarrow \mathbb{N}$ a valuation satisfying $\varphi(PN, M_a, M_b)$. We define the subnet $sPN(\nu) = (sP, sT, sF)$

- $sP = \{p \in P \setminus \{i, o\} \mid \nu(p) > 0\}$
- $sT = \{t \in T \mid \nu(t) > 0\}$
- $sF = \{(a, b) \in F \mid a \in (sP \cup sT) \land b \in (sP \cup sT)\}$

If $sPN(\nu)$ contains a trap (resp. siphon) $N$ then $N$ is also a siphon (resp. trap).
The set of solutions of $\text{SAT}(\varphi(\text{PN}, M_i, M_o), \nu) \land \text{UNSAT}(\theta(s\text{PN}(\nu)))$ also constitutes an over-approximation of the set of correct executions of $\text{PN}$.
Modelling Executions of WF-nets

- Model an execution that we call a segment

**Theorem**

Let $PN = (P, T, F)$ be a WF-net, and $M_a$, $M_b$ its two markings. If there is $\nu : P \times T \rightarrow \mathbb{N}$ such that $SAT(\varphi(PN, M_a, M_b), \nu) \land UNSAT(\theta(sPN(\nu))) \land \forall n \in P \times T. \nu(n) \leq 1$ then $M_a \xrightarrow{\sigma} M_b$ and $\forall t \in T. O_t(\sigma) = \nu(t)$. 
Modelling Executions of WF-nets

**Theorem**

Let $PN = (P, T, F)$ be a WF-net, and $M_a, M_b$ its two markings. $M_a \xrightarrow{\sigma} M_b$ if and only if there exists $k \in \mathbb{N}$ such that $M_1 \xrightarrow{\sigma_1} M_2 \cdots M_k \xrightarrow{\sigma_{(k)}} M_{k+1}$, where $M_1 = M_a$, $M_{k+1} = M_b$ and for every $i$, $0 < i \leq k$, there is $\nu_i$ s.t.

$$\text{SAT}(\varphi(PN, M_i, M_{i+1}, \nu_i)) \land \text{UNSAT}(\theta(sPN(\nu_i))) \land \forall n \in P \times T. \nu_i(n) \leq 1.$$ 

This constraint system (denoted $\phi(PN, M_a, M_b, k)$) can be used to model any execution of $PN$ composed of $k$ or less segments.
Two decision problems:

- The \textit{\textbf{K-bounded validity}} of a modal formula
  - considers executions formed by \( K \) segments, at most
- The \textit{\textbf{unbounded validity}} of a modal formula
  - considers executions formed by an arbitrary number of segments

Verifying a modal formula \( f \) relies on its expression by constraints (denoted \( C(f, \nu) \)):

- For every transition \( t \in T \), the corresponding terminal symbol of the formulae is replaced by \( \nu(t) > 0 \), where \( \nu \) is the valuation function of the constraint system.
The $K$-bounded validity of a modal formula can then be evaluated using constraint solving.

**Theorem**

- $m$ is a $K$-valid may-formula if and only if
  \[
  \text{SAT}(\phi(PN, M_i, M_o, K) \land C(m, \nu), \nu)
  \]

- $M$ is a $K$-valid must-formula if and only if
  \[
  \text{UNSAT}(\phi(PN, M_i, M_o, K) \land \neg C(m, \nu), \nu)
  \]
Verifying Modal Formulae

**Theorem**

Let $PN = (P, T, F)$ be a WF-net, $\bar{R}_{must}$ the set of all well-formed must-formulae not satisfied by $PN$, and $R_{may}$ the set of all well-formed may-formulae satisfied by $PN$. There exists $K_{max}$ such that:

- $\forall f \in \bar{R}_{must}, \exists \nu, k \leq K_{max}.
  \SAT(\phi(PN, M_i, M_o, k) \land \neg C(f, \nu), \nu),$

- $\forall f \in R_{may}, \exists \nu, k \leq K_{max}.
  \SAT(\phi(PN, M_i, M_o, k) \land C(f, \nu), \nu),$

- For $K \geq K_{max}$:
  - $K$-bounded validity $\iff$ unbounded validity
Implementation

- Fully automated
- CLP(FD) library of Sicstus Prolog
Example of an Industrial Business Workflow

Complete workflow: 91 places and 113 transitions
Results: Issue Tracking Systems

<table>
<thead>
<tr>
<th>Formula</th>
<th>$\phi$</th>
<th>$K$</th>
<th>$\phi(K)$</th>
<th>Result</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>either the scenario SubA or the scenario SubB (and not both of them) must be executed</td>
<td>$\exists$ $\text{SubA} \land \neg \text{SubB} \lor (\text{SubB} \land \neg \text{SubA})$</td>
<td>$\text{TRUE}$</td>
<td>-</td>
<td>-</td>
<td>$\leq 1s.$</td>
</tr>
<tr>
<td>when the scenario SubB is considered then the user must login</td>
<td>$\exists$ $\text{SubB} \Rightarrow \text{Login}$</td>
<td>$\text{TRUE}$</td>
<td>-</td>
<td>-</td>
<td>$\leq 1s.$</td>
</tr>
<tr>
<td>a service request can be upgraded to a critical situation request</td>
<td>$\exists$ $\text{SR}_\text{-UpgradeToCRITSIT}$</td>
<td>$\text{TRUE}$</td>
<td>1</td>
<td>FALSE</td>
<td>FALSE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>TRUE</td>
<td>TRUE</td>
</tr>
</tbody>
</table>

- The proposed method is feasible and efficient
Results: Issue Tracking Systems

- Particularly efficient when verifying *must*-formulae that are satisfied by the WF-net, or *may*-formulae that are not satisfied by the WF-net
- Values of $K_{\text{max}}$ are usually moderate (less than 10)
- Conclude about the (in)validity of the studied properties in a very short time (less than a second)
Outline

1. Introduction
2. Workflow Net / Modal Specification
3. Verification of Modal Specifications
4. Experimentation : Issue Tracking Systems
5. Conclusion
Conclusion

- An original and innovative formal framework based on constraint systems to model executions of WF-nets and their structural properties
  - verify their modal specifications
- Encouraging experimental results obtained using a proof-of-concept tool chain

Publication

- H. Bride and O. Kouchnarenko and F. Peureux
  Verifying Modal Workflow Specifications using Constraint Solving
Ongoing work

- Improving constraint solving resolution of segments
  - Mixed constraint solving

- Generalizing our approach by handling coloured Petri nets
  - Abstraction of infinite colours
  - Extend modal formulae expressibility
Thank you for your attention!

Questions and Comments?