Methods for the specification and verification of business processes

MPB (6 cfu, 295AA)

Roberto Bruni

http://www.di.unipi.it/~bruni

14 - Analysis of WF nets
We study suitable soundness properties of Workflow nets
WF nets as business processes

Transition realised by another workflow net
Structural analysis

No distinguished entry / exit point

**no entry**: when should the case start?

**no exit**: when should the case end?

**ruled out by definition of workflow nets**
Structural analysis

Multiple entry / exit points

**multiple entries**: when should the case start?
**multiple exit**: when should the case end?
**ruled out by definition of workflow nets**
Structural analysis

Tasks \( t \) without incoming and/or outgoing arcs

**no input**: when should \( t \) be carried out?

**no output**: \( t \) does not contribute to case completion

**ruled out by definition of workflow nets**
Structural analysis

Wrong decorations of transitions

split with only one outgoing arc

join with only one incoming arc

left to designer responsibility
Activity analysis

Dead tasks

Tasks that can never be carried out
Each transitions lies on a path from i to o: not sufficient

can arise in workflow nets
Token analysis

Some tokens left in the net after case completion

(when a token is in the final place the case should end)

**can arise in workflow nets**
Activity analysis

If tokens are left in the net after case completion then activities may still take place after case completion it is a (worse) consequence of the previous flaw can arise in workflow nets
Token analysis

More than one token reach the end place

it can be a consequence of the previous flaws

can arise in workflow nets
Net analysis

Deadlock

a case blocks without coming to an end

can arise in workflow nets
Question time

Do you see any problem in the workflow net below?
Question time

Do you see any problem in the net below?
Question time

Do you see any problem in the workflow net below?
Question time

Do you see any problem in the workflow net below?
Exercise

Livelock (divergence without producing output)

a case is trapped in a cycle with no opportunity to end

**can arise in workflow nets**

Draw a workflow net that suffers from livelock
Remark

All the previous flaws are typical errors that can be detected without any knowledge about the actual content of the Business Process
Verification and validation

**Validation** is concerned with the relation between the model and the reality
How does a model fit log files?
Which model does fit better?

**Verification** aims to answer qualitative questions
Is there a deadlock possible?
Is it possible to successfully handle a specific case?
Will all cases terminate eventually?
Is it possible to execute a certain task?
Simulation techniques

Test analysis
Try and see if certain firing sequences are allowed by the workflow net

Using WoPeD:
Play (forward and backward) with net tokens
Record certain runs (to replay or explain)
Randomly select alternatives

Problem: how to make sure that all possible runs have been examined?
Reachability analysis

Verification by inspection
All possible runs of a workflow net are represented in its Reachability Graph (when it is finite)

Using WoPeD:
all reachable states are shown
(a single run does not necessarily visit all nodes)
End states are evident (no outgoing arc)

Useful to check if dangerous or undesired states can arise
(e.g. the green-green state in the two-traffic-lights)

Problem: state explosion
Boundedness (for Nets)

**Proposition:**
The reachability graph of a net is finite if and only if the net is bounded.
Boundedness (for Nets)

Proposition:
A net is unbounded if and only if
its reachability graph is not finite
Coverability graph

A **coverability graph** is a finite over-approximation of the reachability graph.

It allows for markings with infinitely many tokens in one place (called extended bags).

\[ B : P \rightarrow \mathbb{N} \cup \{\infty\} \]
Discover unbounded places

Suppose

\[ M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} M_2 \ldots \xrightarrow{t_i} M_i \ldots \xrightarrow{t_j} M_j \]

with \( M_i \subset M_j \)

Let \( M = M_i \) and \( M' = M_j \) and \( L = M' - M \)

By the monotonicity Lemma we have, for any \( n \in \mathbb{N} \):

\[ M \rightarrow^* M + L \rightarrow^* M + 2L \rightarrow^* \ldots \rightarrow^* M + nL \]

Hence all places \( p \) marked by \( L \) (i.e. if \( L(p) > 0 \)) are unbounded
Cover unbounded places

Idea:
When computing the RG, if $M'$ is found s.t.

$$M_0 \rightarrow^* M \rightarrow^* M' \text{ with } M \subset M'$$

Add the extended bag $B$ (instead of $M'$) to the graph

where

$$B(p) = \begin{cases} M'(p) & \text{if } M'(p) - M(p) = 0 \\ \infty & \text{otherwise} \end{cases}$$
A few remarks

**Idea:** mark unbounded places by $\infty$

**Remind:** $M \subset M'$ means that $M \subseteq M' \land M \neq M'$, i.e.,
1. for any $p \in P$, $M'(p) \geq M(p)$
2. there exists at least one place $q \in P$ such that $M'(q) > M(q)$

**Remark:**
Requiring $M_0 \rightarrow^* M \rightarrow^* M'$ is different than requiring $M, M' \in [M_0]$
Operations on extended bags

**Inclusion:** Let $B, B' : P \rightarrow \mathbb{N} \cup \{\infty\}$
We write $B \subseteq B'$ if for any $p$ we have
$B'(p) = \infty$ or $B(p), B'(p) \in \mathbb{N} \land B(p) \leq B'(p)$

**Sum:** Let $B, B' : P \rightarrow \mathbb{N} \cup \{\infty\}$

$$(B + B')(p) = \begin{cases} \infty & \text{if } B(p) = \infty \text{ or } B'(p) = \infty \\ B(p) + B'(p) & \text{if } B(p), B'(p) \in \mathbb{N} \end{cases}$$

**Difference:** Let $B : P \rightarrow \mathbb{N} \cup \{\infty\}$ and $M : P \rightarrow \mathbb{N}$ with $M \subseteq B$

$$(B - M)(p) = \begin{cases} \infty & \text{if } B(p) = \infty \\ B(p) - M(p) & \text{if } B(p) \in \mathbb{N} \end{cases}$$
Compute a reachability graph

1. Initially $N = \{ M_0 \}$ and $A = \emptyset$
2. Take a bag $B \in N$ and a transition $t \in T$ such that
   1. $B$ enables $t$ and there is no arc labelled $t$ leaving from $B$
3. Let $B' = B - \cdot t + t^*$
4. Add $B'$ to $N$ and $(B,t,B')$ to $A$
5. Repeat steps 2,3,4 until no new arc can be added

(all bags are finite in this case)
Compute a **coverability** graph

1. Initially \( N = \{ M_0 \} \) and \( A = \emptyset \)
2. Take a bag \( B \in N \) and a transition \( t \in T \) such that
   1. \( B \) enables \( t \) and there is no arc labelled \( t \) leaving from \( B \)
3. Let \( B' = B - \cdot t + t^* \)
4. Let \( B_c' \) such that for any \( p \in P \)
   1. \( B_c'(p) = \infty \) if there is a node \( B'' \in N \) such that
      1. there is a directed path from \( B'' \) to \( B \) in the graph \((N,A)\)
      2. \( B'' \subset B' \),
      3. \( B''(p) < B'(p) \)
   2. \( B_c'(p) = B'(p) \) otherwise
5. Add \( B_c' \) to \( N \) and \((B,t,B_c')\) to \( A \)
6. Repeat steps 2,3,4,5 until no new arc can be added
Example

\[ p_3 \rightarrow p_2 \rightarrow (0 \ 1 \ 0 \ 0) \rightarrow t_3 \rightarrow o \]
Example
Example
Example
Example

Graph with state transitions and annotations:

- Initial state: p1
- Transition t1: p2
- Transition t2: p3
- Transition t3: o

Annotations:
- p3 (0 1 0 0)
- p2 (0 0 1 0)
- p3 (1 0 0 0)
- p2 (0 0 1 1)
Example
Example
Properties of coverability graphs

A coverability graph is always finite, but in general it is not uniquely defined (it depends on which B and t are selected at step 2)

Every firing sequence has a corresponding path in the CG (the converse is not necessarily true)

Any path in a CG that visits only finite markings corresponds to a firing sequence

If the RG is finite, then it coincides with the CG
Reachability analysis by coverability

All possible behaviours of a workflow net are represented exactly in the Reachability Graph (if finite)

We use Coverability Graph when necessary (RG not finite)

WoPeD computes a Coverability Graph
Exercise

Do you see any problem in the workflow net below?
Exercise

Which problem(s) in the workflow net below? How would you redesign the business process?
Soundness
Soundness of Business Processes

A process is called sound if

1. it contains no unnecessary tasks
2. every case is always completed in full
3. no pending items are left after case completion
Soundness of Business Processes
Soundness of Workflow nets

A workflow net is called **sound** if

1. for each transition $t$, there is a marking $M$ (reachable from $i$) that enables $t$

2. for each token put in place $i$, one token eventually appears in the place $o$

3. when a token is in place $o$, all other places are empty
Fairness assumption

**Remark:**
Condition 2 does not mean that iteration must be forbidden or bound

It says that from any reachable marking $M$
there must be possible to reach $o$ in some steps

**Fairness assumption:**
A task cannot be postponed indefinitely
Soundness, Formally

A workflow net is called **sound** if

no dead task  no transition is dead
\[ \forall t \in T. \exists M \in [i]. M \xrightarrow{t} \]

option to complete  place \( o \) is eventually marked
\[ \forall M \in [i]. \exists M' \in [M]. M'(o) \geq 1 \]

proper completion  when \( o \) is marked, no other token is left
\[ \forall M \in [i]. M(o) \geq 1 \Rightarrow M = o \]
Dead, live or non-live

A remark about terminology:

- t is **dead** = its firing is always ruled out
- t is **live** = its firing can never be ruled out
- t is **non-live** = its firing is possibly ruled out
1: no dead tasks

Reachable marking that enables the transition
1: no dead tasks

The check must be repeated for each task
2: option to complete

Able to produce one token in o
2: option to complete

The check must be repeated for each reachable marking
We should show that it is not a reachable marking.
3: proper completion

The check must be repeated for each marking $M$ such that $M(o) \geq 1$
Brute-force analysis

First, check if the Petri net is a workflow net
   easy "syntactic" check

Second, check if it is sound (more difficult):
   build the Reachability Graph

**to check 1**: for each transition \( t \) there must be an arc in the
   RG that is labelled with \( t \)

**to check 2&3**: the RG must have only one final state (sink),
   that consists of one token in \( o \)
   and is reachable from any other state,
   and no other marking has a token in \( o \)
Some Pragmatic Considerations

All checks can better be done automatically (computer aided)

but nevertheless RG construction...
1. can be computationally expensive for large nets (because of state explosion)
2. provides little support in repairing unsound processes
3. can be infinite (CG can be used, but it is not exact)
Translate soundness to other well-known properties that can be checked more efficiently:

boundedness and liveness
Play once

Business Process
Play twice

Business Process

reset

i

o
Play any number of times
From N to N*
Let us denote by $N : i \rightarrow o$ a workflow net with entry place $i$ and exit place $o$.

Let $N^*$ be the net obtained by adding the “reset” transition to $N$

$\text{reset} : o \rightarrow i$.

**Proposition:**

$N^*$ is strongly connected.

If $x, y \neq \text{reset}$, then

- $y$ lies on a path $i \rightarrow^* y \rightarrow^* o$, because $N$ is a workflow net,
- $x$ lies on a path $i \rightarrow^* x \rightarrow^* o$, because $N$ is a workflow net,

we combine the paths $y \rightarrow^* o \rightarrow \text{reset} \rightarrow i \rightarrow^* x$.
Strong connectedness of $N^*$

Let us denote by $N : i \rightarrow o$ a workflow net with entry place $i$ and exit place $o$.

Let $N^*$ be the net obtained by adding the “reset” transition to $N$.

$reset : o \rightarrow i$.

**Proposition:**

$N^*$ is strongly connected. If $x = o$, $y = reset$, then take any path $i \rightarrow^* o$, we build the path $reset \rightarrow i \rightarrow^* o$.

Take two nodes of $(x, y) \in F_{N^*}$, we want to build a path from $y$ to $x$. 
Strong connectedness
of N*

Let us denote by $N : i \rightarrow o$ a workflow net with entry place $i$ and exit place $o$.

Let $N^*$ be the net obtained by adding the “reset” transition to $N$:

$reset : o \rightarrow i$.

Proposition:

$N^*$ is strongly connected.

Take two nodes of $(x, y) \in F_{N^*}$, we want to build a path from $y$ to $x$.

If $x = reset$, $y = i$, then we build the path $i \rightarrow^* o \rightarrow reset$. If $x \neq reset$, we take any path $i \rightarrow^* o$. Then, we build the path $i \rightarrow^* o \rightarrow reset$. 

Take two nodes of $(x, y) \in F_{N^*}$, we want to build a path from $y$ to $x$.
Let us denote by $N : i \rightarrow o$ a workflow net with entry place $i$ and exit place $o$

Let $N^*$ be the net obtained by adding the "reset" transition to $N$
reset : $o \rightarrow i$

**Theorem:**
$N$ is sound iff $N^*$ is live and bounded
Proof of MAIN THEOREM (1)

\(N^*\) live and bounded implies \(N\) sound:

Since \(N^*\) is live: for each \(t \in T\) there is \(M \in [i]. M \xrightarrow{t}\)

Take any \(M \in [i]\) enabling \(\text{reset}: o \rightarrow i\), hence \(M \supseteq o\)

Let \(M \xrightarrow{\text{reset}} M'\). Then \(M' \in [i]\) and \(M' \supseteq i\)

Since \(N^*\) is bound, it must be \(M' = i\) (and \(M = o\))
Otherwise all places marked by \(M' - i = M - o\) would be unbounded

Hence \(N^*\) just allows multiple runs of \(N\):
"option to complete" and "proper completion" hold (see above)
"no dead task" holds because \(N^*\) is live
A technical lemma

Lemma:
If $N$ is sound, $M$ is reachable in $N$ iff $M$ is reachable in $N^*$

$\implies$ straightforward

$\iff$ Let $i \xrightarrow{\sigma} M$ in $N^*$ for $\sigma = t_1 t_2 \ldots t_n$
We proceed by induction on the number $r$ of instances of reset in $\sigma$
If $r = 0$, then reset does not occur in $\sigma$ and $M$ is reachable in $N$
If $r > 0$, let $k$ be the least index such that $t_k = \text{reset}$
Let $\sigma = \sigma' t_k \sigma''$ with $\sigma' = t_1 t_2 \ldots t_{k-1}$ fireable in $N$
Since $N$ is sound: $i \xrightarrow{\sigma'} o$ and $i \xrightarrow{\sigma''} M$
Since $\sigma''$ contains $r - 1$ instances of reset:
by inductive hypothesis $M$ is reachable in $N$
Proof of MAIN THEOREM (2)

$N$ sound implies $N^*$ bounded:

We proceed by contradiction, assuming $N^*$ is unbounded.

Since $N^*$ is unbounded:

$\exists M, M'$ such that $i \rightarrow^* M \rightarrow^* M'$ with $M \subseteq M'$

Let $L = M' - M \neq \emptyset$

Since $N$ is sound:

$\exists \sigma \in T^*$ such that $M \xrightarrow{\sigma} o$

By the monotonicity Lemma: $M' \xrightarrow{\sigma} o + L$ and thus $o + L \in [i]$

Which is absurd, because $N$ is sound
Proof of MAIN THEOREM (3)

\(N\) sound implies \(N^*\) live:
Take any transition \(t\) and let \(M\) be a marking reachable in \(N^*\)
By the technical lemma, \(M\) is reachable in \(N\)

Since \(N\) is sound: \(\exists \sigma \in T^*\) with \(M \xrightarrow{\sigma} o\)
Since \(N\) is sound: \(\exists \sigma' \in T^*\) with \(i \xrightarrow{\sigma'} M'\) and \(M' \xrightarrow{t}\)

Let \(\sigma'' = \sigma\) reset \(\sigma'\), then:
\(M \xrightarrow{\sigma''} M'\) in \(N^*\) and \(M' \xrightarrow{t}\)
WoPeD (3.6.0)
Workflow Petri Net Designer
Download WoPeD at sourceforge!
Exercise

Use some tools to check if the net below is a sound workflow net or not
Exercise

Use some tools to check if the net below is a sound workflow net or not
Exercise

Analyse the following net