

Automated Synthesis of Run-time Monitors to Enforce Authorization Policies in Business Processes

C. Bertolissi^{1,4}, **D.R. dos Santos**^{1,2,3}, S. Ranise¹

¹Fondazione Bruno Kessler, ²SAP Labs France,
³University of Trento, ⁴University of Marseille



SECENTIS

A European Industrial Doctorate on Security and Trust

Outline

- 1 Introduction
- 2 Automated Synthesis of Run-time Monitors
 - Off-line
 - On-line
- 3 Experiments
 - Real-world workflows
 - Synthetic Benchmarks
- 4 Conclusions

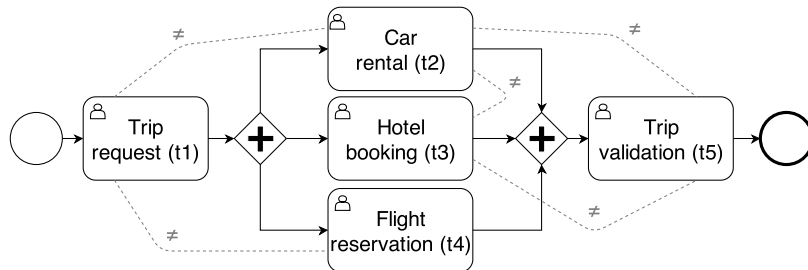
Outline

- 1 Introduction
- 2 Automated Synthesis of Run-time Monitors
 - Off-line
 - On-line
- 3 Experiments
 - Real-world workflows
 - Synthetic Benchmarks
- 4 Conclusions

Context

- A workflow specifies a collection of tasks and the causal relationships between them
- Authorization policies specify which users can execute which tasks
- Additional constraints, such as Separation/Binding of Duty, further restrict the execution of tasks by users

Example



task	roles
t1	r3
t2	r2
t3	r2
t4	r1
t5	r2

roles	users
r1	a
r2	a, b, c
r3	a, b

Problem

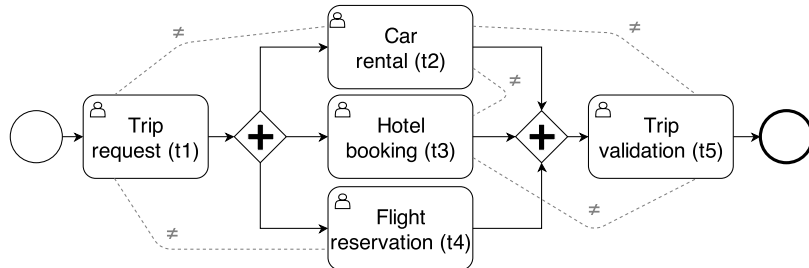
Workflow Satisfiability Problem (WSP)

Is there an assignment of users to tasks such that a workflow terminates while satisfying all authorization constraints?

Run-time WSP

Answering sequences of user requests at execution time ensuring termination with the satisfaction of authorization constraints

Run-time version allows us to divide the problem in two steps



task	roles
t1	r3
t2	r2
t3	r2
t4	r1
t5	r2

roles	users
r1	a
r2	a, b, c
r3	a, b

Satisfying trace: $t1(b)$, $t2(a)$, $t3(c)$, $t4(a)$, $t5(b)$

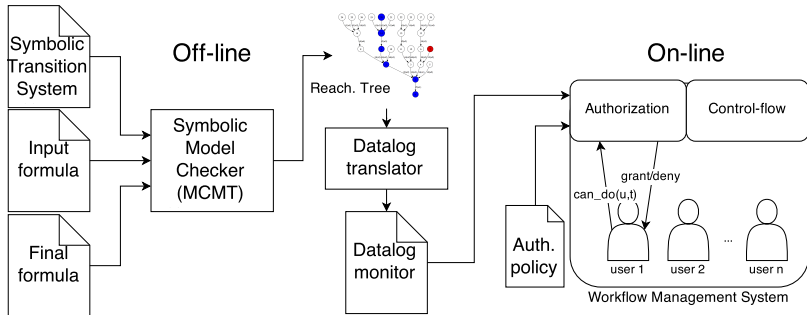
Contribution

- Automated technique to synthesize run-time monitors solving the run-time WSP
- Divided in off-line (workflow+constraints) and on-line (policy) phases
- Changes in the authorization policies can be accommodated without re-running from scratch the approach
- Main advantage: symbolic representation of users

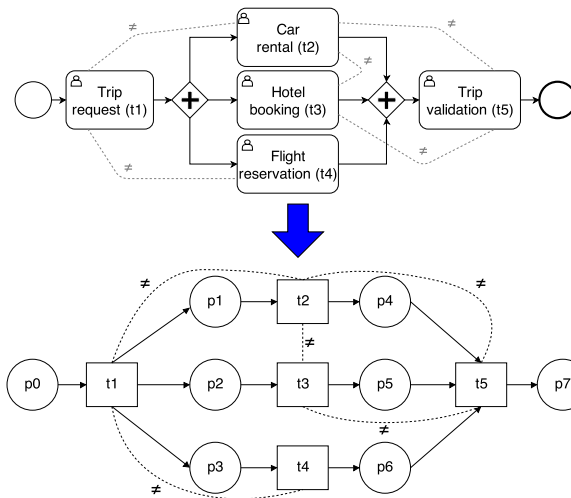
Outline

- 1 Introduction
- 2 Automated Synthesis of Run-time Monitors
 - Off-line
 - On-line
- 3 Experiments
 - Real-world workflows
 - Synthetic Benchmarks
- 4 Conclusions

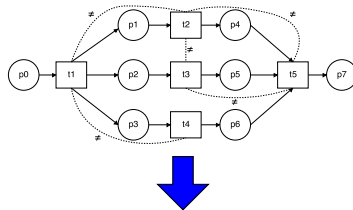
Overview



BPMN to Petri Net



Petri Net to Transition System



$$t(u): \text{enabled}_{CF} \wedge \text{enabled}_{Auth} \longrightarrow \text{act}_{CF} || \text{act}_{Auth}$$

$$t2(u): p1 \wedge \neg d_{t2} \wedge a_{t2}(u) \wedge \neg h_{t3}(u) \wedge \neg h_{t1}(u) \\ \longrightarrow p1, p4, d_{t2} := F, T, T || h_{t2}(u) := T$$

$$I: p0 \wedge \dots \wedge p7 \neg p_i \wedge \neg d_{t_i}, \dots, d_{t5} \wedge \bigwedge_{i=1, \dots, 5} \forall u. \neg h_{t_i}(u) \\ F: p7 \wedge \neg p0 \wedge \dots \neg p6 \wedge d_{t1} \dots \wedge d_{t5}$$

Computing the Reachability Graph

Require: $S = (V_{CF} \cup V_{Auth} \cup V_{User}, Ev_S)$ and F

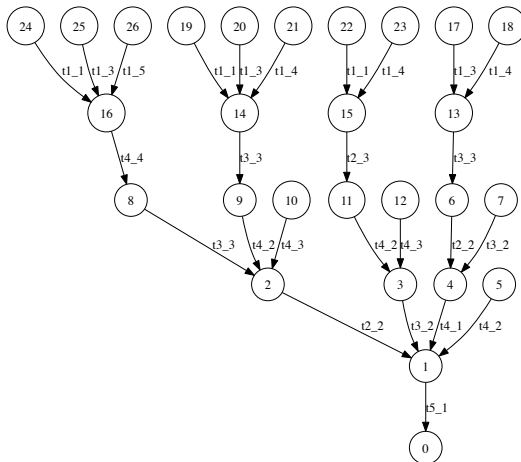
Ensure: $RG = (N, \lambda, E)$

```

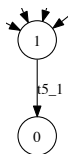
1: #start with a node for the final state
2:  $i \leftarrow \text{new}(); N \leftarrow \{i\}; E \leftarrow \emptyset; \lambda[i] \leftarrow F; TBV \leftarrow \{i\};$ 
3: while  $TBV \neq \emptyset$  do #until a fix-point is reached
4:   if  $\text{subsumed}(i, N, N')$  then
5:      $\text{connect}(N', i); TBV \leftarrow TBV - \{i\};$ 
6:   end if
7:   for all  $ev \in Ev_S$  do #then for every transition
8:      $P \leftarrow \text{wlp}(ev, \lambda[i]);$ 
9:     if  $P$  is satisfiable then #add a new node if applicable
10:       $j \leftarrow \text{new}(); N \leftarrow N \cup \{j\}; E \leftarrow E \cup \{(i, \overline{ev}, j)\};$ 
11:       $\lambda[j] \leftarrow P; TBV \leftarrow TBV \cup \{j\};$ 
12:    end if
13:  end for
14:   $i \leftarrow \text{pickOne}(TBV); TBV \leftarrow TBV - \{i\};$ 
15: end while
16: return  $(N, \lambda, E);$ 

```

Reachability Graph



Reachability Graph to Datalog



$$\neg p_0 \wedge \neg p_1 \wedge \neg p_2 \wedge \neg p_3 \wedge p_4 \wedge p_5 \wedge p_6 \wedge d_{t_1} \wedge d_{t_2} \wedge d_{t_3} \wedge d_{t_4} \wedge \neg d_{t_5} \wedge (a_{t_5}(u_1) \wedge \neg h_{t_2}(u_1) \wedge \neg h_{t_3}(u_1))$$


$$can_do(u_1, t_5) \leftarrow \neg p_0 \wedge \neg p_1 \wedge \neg p_2 \wedge \neg p_3 \wedge p_4 \wedge p_5 \wedge p_6 \wedge d_{t_1} \wedge d_{t_2} \wedge d_{t_3} \wedge d_{t_4} \wedge \neg d_{t_5} \wedge a_{t_5}(u_1) \wedge \neg h_{t_2}(u_1) \wedge \neg h_{t_3}(u_1)$$

Policy

RBAC Policy

$$U = \{a, b, c\} \quad R = \{r_1, r_2, r_3\}$$

$$UA = \{(a, r_1), (a, r_2), (a, r_3), (b, r_2), (b, r_3), (c, r_2)\}$$

$$TA = \{(r_3, t_1), (r_2, t_2), (r_2, t_3), (r_1, t_4), (r_2, t_5)\}$$

Policy in Datalog

$$ua(a, r_1) \quad ua(a, r_2) \quad ua(a, r_3) \quad ua(b, r_2) \quad ua(b, r_3) \quad ua(c, r_2)$$

$$pa(r_3, t_1) \quad pa(r_2, t_2) \quad pa(r_2, t_3) \quad pa(r_1, t_4) \quad pa(r_2, t_5)$$

$$a_t(u) \leftarrow ua(u, r) \wedge pa(r, t) \text{ for each } t \in \{t_1, \dots, t_5\}$$

Example trace

	CF	Auth					<i>can_do</i>	
#	Token in	h_{t1}	h_{t2}	h_{t3}	h_{t4}	h_{t5}	(u, t)	Resp.
0	$p0$	-	-	-	-	-	$(a, t1)$	deny
1	$p0$	-	-	-	-	-	$(b, t1)$	grant
2	$p1, p2, p3$	b	-	-	-	-	$(b, t2)$	deny
3	$p1, p2, p3$	b	-	-	-	-	$(a, t2)$	grant
4	$p4, p2, p3$	b	a	-	-	-	$(c, t3)$	grant
5	$p4, p5, p3$	b	a	c	-	-	$(a, t4)$	grant
6	$p4, p5, p6$	b	a	c	a	-	$(b, t5)$	grant
7	$p7$	b	a	c	a	b	-	-

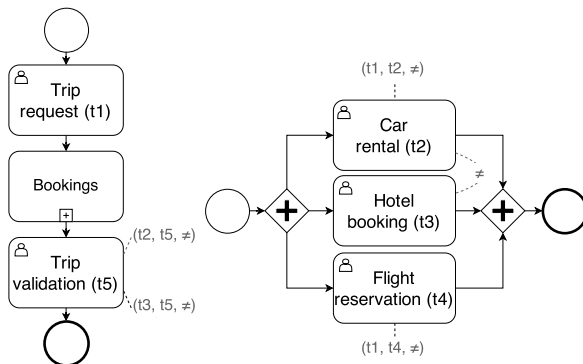
Outline

- 1 Introduction
- 2 Automated Synthesis of Run-time Monitors
 - Off-line
 - On-line
- 3 Experiments
 - Real-world workflows
 - Synthetic Benchmarks
- 4 Conclusions

Description

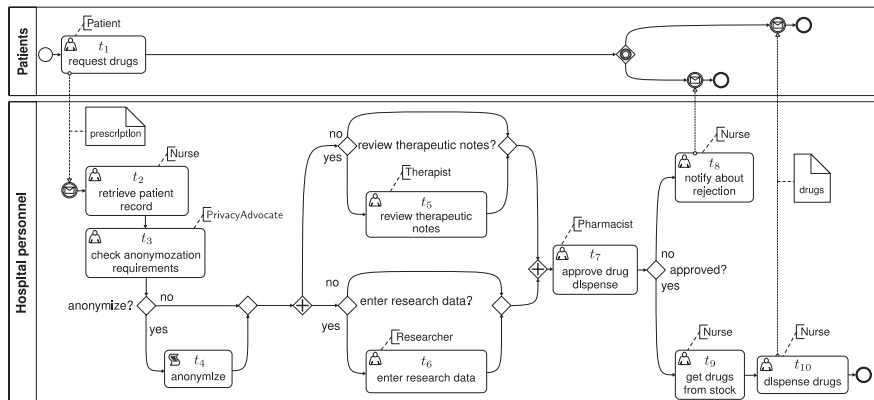
- Real-world for validity
 - Moderate number of tasks
 - Complex control-flow
- Synthetic for scalability
 - Huge number of tasks
 - Simple control-flow
- State space explosion → need for heuristics

Hierarchical descriptions



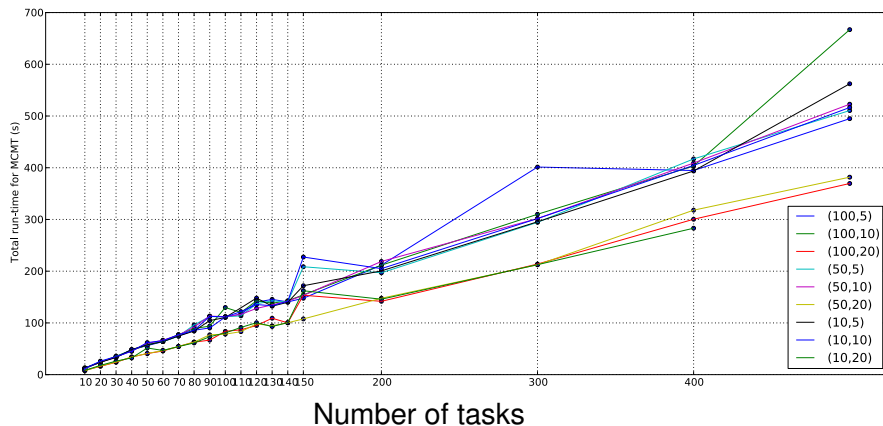
Real-world workflows

Drug dispensation process



Synthetic Benchmarks

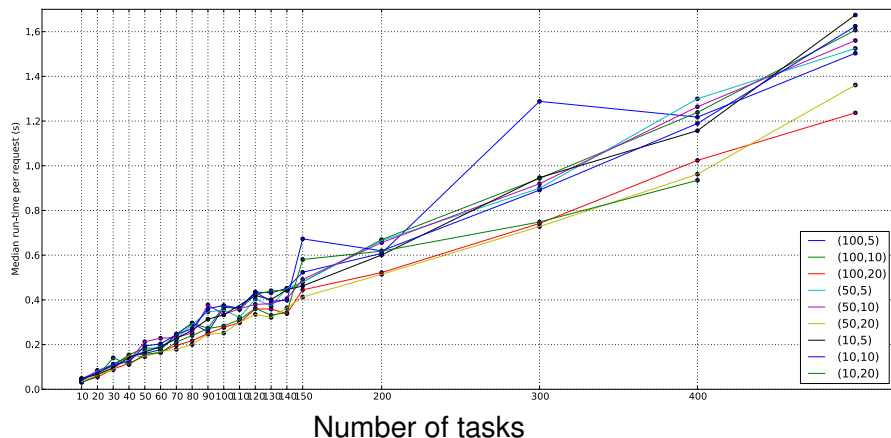
Off-line



(pa,pc) = (authorization density, constraint density)

Synthetic Benchmarks

On-line



(pa,pc) = (authorization density, constraint density)

Outline

- 1 Introduction
- 2 Automated Synthesis of Run-time Monitors
 - Off-line
 - On-line
- 3 Experiments
 - Real-world workflows
 - Synthetic Benchmarks
- 4 Conclusions

Conclusions

- Synthesize run-time monitors to ensure the termination of workflows with authorization constraints
 - Off-line: compute a symbolic representation of all behaviors
 - On-line: add the authorization policy and derive the monitor
- Parametric wrt the number of users
- Scalable with the use of hierarchical representations
- Future work: integrate the monitor in a real workflow engine to collect data about performance

Thank you!

www.secentis.eu

bertolissi@fbk.eu

dossantos@fbk.eu

ranise@fbk.eu



UNIVERSITY
OF TRENTO