Symbolic Execution of Acyclic Business Process Models
Cédric Favre and Hagen Völzer
Control-flow analysis of business process models

- Check the absence of control-flow errors in business process models:
  - As early as possible: during the modeling activity

- We want an efficient control-flow analysis technique that provides diagnostic information adequate for a non-verification expert
Our work

- Symbolic execution of acyclic business process models, which may contain IOR-joins

- Quadratic time and space complexity control-flow analysis for acyclic processes

- New type of diagnostic information

- Approach to dismiss false positives that are due to data abstraction
Agenda

- Overview
- Techniques
- Dismissing false positives
An example
Soundness: a notion of correctness

- A workflow graph is *sound* if it is free from deadlock and lack of synchronization.

**Deadlock**
- A token blocked in the graph.

**Lack of Synchronization**
- Two tokens on the same edge.
State of the art in control-flow analysis (1/2)

- Trade off between efficiency and consumability:
  
a. Polynomial time complexity approaches that have to limited diagnostic information (Rank theorem and reduction approaches)
  b. State space exploration based approaches that return an error trace as diagnostic information
State of the art in control-flow analysis (2/2)

- Trade off between efficiency and consumability:
  a. Polynomial time complexity approaches that have to limited diagnostic information (Rank theorem and reduction approaches)
  b. State space exploration based approaches that return an error trace as diagnostic information
State of the art in control-flow analysis (2/2)

- Trade off between efficiency and consumability:
  
  a. Polynomial time complexity approaches that have to limited diagnostic information (Rank theorem and reduction approaches)
  
  b. State space exploration based approaches that return an error trace as diagnostic information

Lack of synchronization on this edge
Symbolic execution (1/3)

- Show error graphically as reduced error trace
- Edges get labeled with conditions under which they carry a token
  - Allow us to reason about the condition under which the error can occur
- User can decide whether an error can occur in practice: dismiss or repair
- Quadratic time and space complexity
Symbolic execution (2/3)

- Show error graphically as *reduced error trace*

- Edges get labeled with *conditions* under which they carry a token
  - Allow us to reason about the condition under which the error can occur

- User can decide whether an error can occur in practice: *dismiss* or *repair*

- Quadratic time and space complexity
Symbolic execution (3/3)

- Show error graphically as reduced error trace
- Edges get labeled with conditions under which they carry a token
  - Allow us to reason about the condition under which the error can occur
- User can decide whether an error can occur in practice: dismiss or repair
- Quadratic time and space complexity
Agenda

- Overview

- Techniques
  - Lack of synchronization detection and trace display
  - Conditions and deadlock detection

- Dismissing false positives
Handles

- Two disjoint paths between an IOR-split or an AND-split and an XOR-join [EsparzaSilva, Aalst]

In a deadlock free acyclic business process model, there is a lack of synchronization if and only if there is a handle.
Handles: An adequate diagnostic information.

- Executing a handle results in a lack of synchronization

- In acyclic processes, handles can be detected in quadratic time using a modified graph theoretical approach.
Agenda

- Overview

- Techniques
  - Lack of synchronization detection and trace display
  - Conditions and deadlock detection

- Dismissing false positives
Characterization of an execution

- In an acyclic process, an execution is characterized by the conditions that are evaluated to true

- Evaluated to true: New customer, Is not gold customer, and Not eligible
Characterization of an execution

- In an acyclic process, an execution is characterized by the conditions that are evaluated to true

- Evaluated to true: New customer, Is not gold customer, and Not eligible
Symbolic Execution

- Characterizes the set of executions that lead an edge to carry a token in terms of conditions

```
Identify customer

Existing customer

New customer

Create customer record

Retrieve Customer record

Update customer records

Check customer status

Check eligibility to become gold customer

Add free gift

Add taxes

Is gold customer OR
New customer

Is not gold customer

Deduct 2% of bill

Is gold customer

Eligible

Not eligible
```
Symbolic execution - Example

Propagation rules:

\[
\begin{align*}
S_1 &\rightarrow \{a\} \\
S_1 &\rightarrow \{a\} \\
S_1 &\rightarrow S_1 \\
S_1 \cup S_2 &\rightarrow S_1 \\
S_1 \cup S_2 &\rightarrow S_1 \\
S_1 &\rightarrow S_1 \\
S_1 &\rightarrow S_1 \\
\text{iff } S_1 \equiv S_2 &\rightarrow S_1 \\
\end{align*}
\]
Symbolic Execution – Detecting deadlocks

- A symbol maps to a set of executions

- Multiples symbols are equivalent, i.e., map to the same set of executions

- To check the absence of deadlock at an AND-join, we check that the incoming edges carry equivalent symbols

- To check edge equivalence, check symbol equivalence by computing the maximal symbol of each symbol
  - Largest symbol such that it is equivalent to the original symbol
  - Normal form of the equivalent symbols
Symbolic execution - Continued

Maximal symbol of $S$:

- If $S \supseteq S_1$ then $S \leftarrow S \cup \{x_1, x_2\}$
- If $S \supseteq \{x_1, x_2\}$ then $S \leftarrow S \cup S_1$

Propagation rules:

- $S_1 \leftarrow S_1$ if $S_1 \equiv S_2$
- $S_1 \leftarrow S_1 \cup S_2$ if $S_1 \equiv S_2$
- $S_1 \leftarrow S_1 \cup S_2$ if $S_1 \equiv S_2$
- $S_1 \leftarrow S_1 \cup S_2$ if $S_1 \equiv S_2$
- $S_1 \leftarrow S_1 \cup S_2$ if $S_1 \equiv S_2$
- $S_1 \leftarrow S_1 \cup S_2$ if $S_1 \equiv S_2$
- $S_1 \leftarrow S_1 \cup S_2$ if $S_1 \equiv S_2$
- $S_1 \leftarrow S_1 \cup S_2$ if $S_1 \equiv S_2$
- $S_1 \leftarrow S_1 \cup S_2$ if $S_1 \equiv S_2$
Symbolic execution - Continued

Maximal symbol of $S$:

If $S \supseteq S_1$ then $S \leftarrow S \cup \{x_1, x_2\}$
If $S \supseteq \{x_1, x_2\}$ then $S \leftarrow S \cup S_1$

Propagation rules:
Symbolic execution - Continued

Maximal symbol of $S$:

If $S \supseteq S_1$ then $S \leftarrow S \cup \{x_1, x_2\}$

If $S \supseteq \{x_1, x_2\}$ then $S \leftarrow S \cup S_1$

Propagation rules:
Symbolic execution - Continued

Maximal symbol of $S$:

If $S \supseteq S_1$ then $S \leftarrow S \cup \{x_1, x_2\}$

If $S \supseteq \{x_1, x_2\}$ then $S \leftarrow S \cup S_1$

Propagation rules:
Symbolic execution - Continued

Maximal symbol of $S$:

- If $S \supseteq S_1$ then $S \leftarrow S \cup \{x_1, x_2\}$
- If $S \supseteq \{x_1, x_2\}$ then $S \leftarrow S \cup S_1$

Propagation rules:

- $S_1 \rightarrow \{a\}$
- $S \rightarrow \{b\}$
- $S_1 \rightarrow S_1$
- $S_1 \rightarrow S_1 \cup S_2$
- $S_1 \rightarrow S_1 \cup S_2$
- $S_1 \rightarrow S_1 \cup S_2$

+ $S_1 \rightarrow S_1$

iff $S_1 \equiv S_2$
Symbolic execution - Continued

Maximal symbol of $S$:

- If $S \supseteq S_1$ then $S \leftarrow S \cup \{x_1, x_2\}$
- If $S \supseteq \{x_1, x_2\}$ then $S \leftarrow S \cup S_1$

Propagation rules:
Symbolic execution - Continued

Maximal symbol of $S$:

- If $S \supseteq S_1$ then $S \leftarrow S \cup \{x_1, x_2\}$
- If $S \supseteq \{x_1, x_2\}$ then $S \leftarrow S \cup S_1$

Propagation rules:
Symbolic execution - Continued

Maximal symbol of $S$:

If $S \supseteq S_1$ then $S \leftarrow S \cup \{x_1, x_2\}$

If $S \supseteq \{x_1, x_2\}$ then $S \leftarrow S \cup S_1$

Propagation rules:
Symbolic execution - Continued

Maximal symbol of $S$:

- If $S \supseteq S_1$ then $S \leftarrow S \cup \{x_1, x_2\}$
- If $S \supseteq \{x_1, x_2\}$ then $S \leftarrow S \cup S_1$

Propagation rules:

- If $S_1$ then $S \leftarrow S_1$
- If $S_2$ then $S \leftarrow S_2$
- If $S_1 \cup S_2$ then $S \leftarrow S_1 \cup S_2$
- If $S_1$ then $S \leftarrow S_1$
- If $S_2$ then $S \leftarrow S_2$
- If $S_1 \equiv S_2$ then $S \leftarrow S_1$
Symbolic execution - Continued

**Maximal symbol of S:**

- If $S \supseteq S_1$ then $S \leftarrow S \cup \{x_1, x_2\}$
- If $S \supseteq \{x_1, x_2\}$ then $S \leftarrow S \cup S_1$

**Propagation rules:**

The propagation rules are illustrated with diagrams showing how symbols are updated based on the conditions specified in the maximal symbol of the set $S$. Each rule corresponds to a specific operation on the sets, such as union ($\cup$) or set addition ($+$), and examples of these operations are shown with corresponding symbols and set updates.
Symbolic execution - Continued

Maximal symbol of \( S \):

If \( S \supseteq S_1 \) then \( S \leftarrow S \cup \{x_1, x_2\} \)

If \( S \supseteq \{x_1, x_2\} \) then \( S \leftarrow S \cup S_1 \)

Propagation rules:

\[
\begin{align*}
S_1 & \rightarrow \{a\} & S_1 & \rightarrow \{a\} & S_1 & \rightarrow S_1 \\
\quad \{b\} & \quad \{b\} & \quad + & \quad S_1 & \iff S_1 \equiv S_2 \\
\{s\} & \rightarrow \{a\} & s_2 & \rightarrow S_1 & s_1 & \rightarrow S_1 \\
\quad \{s\} & \quad \{b\} & s_2 & \rightarrow S_1 & s_1 & \rightarrow S_1 \\
\quad \{s\} & \quad \{c\} & s_1 & \rightarrow S_1 & s_1 & \rightarrow S_1 \\
\quad \{s\} & \quad \{d\} & \quad + & \quad S_1 & \quad + \\
\{s, a, b, c, d\} & \quad \{s, a, b, c, d\} & \quad \{s, a, b, c, d\} & \quad \{s, a, b, c, d\} & \neq \{d\}
\end{align*}
\]
Symbolic execution - Continued

**Maximal symbol of S:**

If \( S \supseteq S_1 \) then \( S \leftarrow S \cup \{x_1, x_2\} \)

If \( S \supseteq \{x_1, x_2\} \) then \( S \leftarrow S \cup S_1 \)

**Propagation rules:**

\[ \begin{align*}
S_1 &\Rightarrow \{a\} \\
S_1 &\Rightarrow \{b\} \\
S_1 &\Rightarrow + \\
S_1 &\Rightarrow S_1 \\
S_1 \cup S_2 &\Rightarrow S_1 \\
S_1 \cup S_2 &\Rightarrow + \\
S_1 &\Rightarrow S_1 \\
\end{align*} \]

\( \text{iff } S_1 \equiv S_2 \)
Symbolic execution - Continued

Maximal symbol of \( S \):

- If \( S \supseteq S_1 \) then \( S \leftarrow S \cup \{x_1, x_2\} \)
- If \( S \supseteq \{x_1, x_2\} \) then \( S \leftarrow S \cup S_1 \)

Propagation rules:

- Computation of the maximal symbol in linear time
- Symbolic execution in quadratic time
Agenda

- Overview
- Techniques
- Dismissing false positives
False positives

- Some information not contained in the business process model
- Conditions are abstracted during the analysis
- Lead to detect errors that cannot happen in an ‘actual’ execution
Dismissing false positives via user interaction

- **Deadlock**: dismiss when symbols are ‘equivalent’
  - For each execution where an $x_i$ is evaluated to true a $y_j$ is evaluated to true and vice versa

- **Lack of synchronization**: dismiss when symbols are ‘mutually exclusive’
  - For each $x_i$ there exist no $y_j$ that can be evaluated to true during the same execution

- **Dismissed error**: use the IOR-join propagation rule

![Diagram](image-url)
Not every error should be dismissed

- Every edge should be marked during a sound execution (relaxed soundness [DehnertRittgen])
- Heuristics for errors that should not be dismissed
  - Deadlock without handle when replacing AND-join by an XOR-join
  - Lack of synchronization where a condition occurs in multiple maximal symbols
Conclusion

- Symbolic execution
  - of acyclic business process models that may contain IOR-joins which allows to characterize the execution that mark an edge
  - Leads to a control-flow analysis with a new type of diagnostic information in term of conditions and which has quadratic time complexity
  - Approach to dismiss false positives that are due to data abstraction
  - Provide control-flow relationships that are useful beyond control-flow analysis (e.g. data-flow analysis)

- Future work:
  - Cyclic business process models
  - Composition of business process models
  - Data-flow analysis