Coverage Techniques for Checking Temporal-Observation Subsumption

A. Ducoli, G. Lamperti, E. Piantoni, M. Zanella

Dipartimento di Elettronica per l'Automazione
Università degli Studi di Brescia
Brescia, Italy
Diagnosis of discrete-event systems (DESs) faced both by artificial intelligence and control theory approaches.

DES modeled as a network of reactive components.

Each component is a communicating automaton.

Diagnosis task generates a variety of model-based structures in the abductive process.

Throw-away model-based reasoning may not be ecological.

Need for some sort of *reuse* of model-based reasoning.
Diagnosis with Reuse

- *Similarity-based diagnosis* aims to pursue reuse of model-based reasoning ⇒ *diagnosis with reuse*.

- Knowledge Base to store the structures generated for solving each diagnostic problem.

- When solving a new problem, the KB is first browsed in order to find a compatible (already solved) problem.

- If found, the new problem can be solved efficiently by reusing the structures in the KB.
Observation Subsumption

- Diagnosis with reuse requires observation subsumption (among other constraints).

- Temporal observation represented by a DAG.

- Observation replaced by the index space in the diagnosis task.

- Index space: automaton derived from the observation.

- Subsumption-checking involves the comparison of the index spaces of the two observations (regular-language containment): costly.

- Need for an alternative approach for subsumption checking of temporal observations.
Temporal Observation

- Reaction of a system is a sequence of component transitions:
  \[ \langle T_1, T_2, \ldots, T_n \rangle \]

- Each visible transition leaves a clue of its occurrence, namely an observable label.

- A reaction is perceived as a sequence of observable labels, its signature:
  \[ S = \langle \ell_1, \ell_2, \ldots, \ell_m \rangle \]

- However, what is actually observed, is a relaxation of \( S \), the temporal observation, \( O = (\mathcal{N}, \mathcal{L}, \mathcal{A}) \).
Figure: Observations $\mathcal{O}_1$ (left) and $\mathcal{O}_2$ (right)
For the diagnostic process, better reasoning on a surrogate of the observation, its *index space*.

*\text{Isp}(\mathcal{O})* is a deterministic automaton with the property that its language is the set of *candidate signatures* of \( \mathcal{O} \):

\[
\text{Lang}(\text{Isp}(\mathcal{O})) = \parallel \mathcal{O} \parallel
\]

Generation of *\text{Isp}(\mathcal{O})* in two steps:

- Yielding the *prefix space* of \( \mathcal{O} \), the nondeterministic automaton where each node identifies the set of consumed nodes in \( \mathcal{O} \) up to now;
- Generating the deterministic automaton equivalent to the prefix space.
Figure: Index spaces $lsp(\Theta_1)$ (left) and $lsp(\Theta_2)$ (right)
Definition of Observation Subsumption

- In similarity-based diagnosis, essential to understand if the solution of a new problem $\mathcal{O}'$ can be supported by the knowledge stored in the KB for solving a previous problem $\mathcal{O}$.

- In particular, need for checking subsumption $\mathcal{O} \supseteq \mathcal{O}'$.

- Observation subsumption defined as language containment of the index spaces:

$$\mathcal{O} \supseteq \mathcal{O}' \iff \text{Lang}(lsp(\mathcal{O})) \supseteq \text{Lang}(lsp(\mathcal{O}'))$$

- Observation subsumption supports reuse because, if $\mathcal{O} \supseteq \mathcal{O}'$ then the structures based on $\mathcal{O}$ contain (in fact, subsumes) the structures based on $\mathcal{O}'$. 
We have $\mathcal{O}_1 \supseteq \mathcal{O}_2$, as $\text{Lang}(lsp(\mathcal{O}_1)) \supseteq \text{Lang}(lsp(\mathcal{O}_2))$.

Figure: Index spaces $lsp(\mathcal{O}_1)$ (left) and $lsp(\mathcal{O}_2)$ (right)
The *checking problem* is the problem of testing $\mathcal{O} \supseteq \mathcal{O}'$.

Systematic checking based on formal definition: prohibitive in real applications.

Better finding necessary ($N_c$) and sufficient ($S_c$) conditions to subsumption that can be checked using a reasonable computational effort:

- If $N_c$ is violated then the answer will be *no*;
- If $S_c$ holds then the answer will be *yes*;
- If either $N_c$ holds or $S_c$ is violated then the checking problem will remain *unanswered* (and another observation in the KB will be tried).

Necessary conditions and sufficient conditions for the checking problem are given.
Theorem
Let \( O = (N, L, A) \) and \( O' = (N', L', A') \) be two temporal observations. Let \( n \) and \( n' \) be the number of nodes in \( N \) and \( N' \), respectively. Let \( n_\epsilon \) and \( n'_\epsilon \) be the number of nodes that include the empty label \( \epsilon \) in \( N \) and \( N' \), respectively. Let \( M \) and \( M' \) be the multisets of observable labels occurring in \( O \) and \( O' \), respectively. Then, \( O \) subsumes \( O' \) only if the following conditions hold:

\[
\begin{align*}
    n &\geq n' \\
    n_\epsilon - n'_\epsilon &\geq n - n' \\
    M &\supseteq M'.
\end{align*}
\]
Necessary Conditions for Subsumption: Example

Figure: Observations $\mathcal{O}_1$ (left) and $\mathcal{O}_2$ (right)

where:

- $n_1 = 5$, $n_2 = 4$, $n_1\epsilon = 3$, $n_2\epsilon = 2$

- $\mathcal{M}_1 = [a, a, a, b, b, b, c, d, f, \epsilon, \epsilon, \epsilon]$, $\mathcal{M}_2 = [a, a, b, c, d, \epsilon, \epsilon]$

Since $\mathcal{O}_1 \supseteq \mathcal{O}_2$, we expect necessary conditions to hold, in fact:

\[
\begin{align*}
n_1 &\geq n_2 \\
n_1\epsilon - n_2\epsilon &\geq n_1 - n_2 \\
\mathcal{M}_1 &\supseteq \mathcal{M}_2.
\end{align*}
\]
Coverage

Definition

Let $\mathcal{O} = (\mathcal{N}, \mathcal{L}, \mathcal{A})$ and $\mathcal{O'} = (\mathcal{N}', \mathcal{L}', \mathcal{A}')$ be two temporal observations, where $\mathcal{N} = \{N_1, \ldots, N_n\}$ and $\mathcal{N}' = \{N'_1, \ldots, N'_{n'}\}$. We say that $\mathcal{O}$ covers $\mathcal{O}'$, written $\mathcal{O} \supseteq \mathcal{O}'$, when there exists a subset $\tilde{\mathcal{N}}$ of $\mathcal{N}$, with $\tilde{\mathcal{N}} = \{\tilde{N}_1, \ldots, \tilde{N}_{n'}\}$ isomorphic to $\mathcal{N}'$, such that, denoting $\mathcal{N}^\epsilon = (\mathcal{N} - \tilde{\mathcal{N}})$, we have:

1. **(node coverage):** $\forall N \in \mathcal{N}^\epsilon (\epsilon \in \|N\|)$;

2. **(logical coverage):** $\forall i \in [1..n'] (\|\tilde{N}_i\| \geq \|N'_i\|)$;

3. **(temporal coverage):** For each path $\tilde{N}_i \leadsto \tilde{N}_j$ in $\mathcal{O}$, the corresponding nodes in $\mathcal{O}'$ are such that $N'_i < N'_j$. 

A. Ducoli, G. Lamperti, E. Piantoni, M. Zanella

Coverage Techniques for Checking Temporal-Observation Subsumption
Choosing $\mathcal{N}_1 = \{N_2, N_1, N_4, N_5\}$, hence, $\mathcal{N}_1^e = \{N_3\}$, we have:

1. (node coverage): $\epsilon \in \|N_3\|$;

2. (logical coverage): $\|N_2\| \supseteq \|N'_1\|$, $\|N_1\| \supseteq \|N'_2\|$, $\|N_4\| \supseteq \|N'_3\|$, and $\|N_5\| \supseteq \|N'_4\|$;

3. (temporal coverage): for instance, for $N_1 \leadsto N_5$ in $\mathcal{O}_1$, we have $N'_2 < N'_4$ in $\mathcal{O}_2$. 

Figure: Observations $\mathcal{O}_1$ (left) and $\mathcal{O}_2$ (right)
Theorem

Coverage entails subsumption:

\[ \emptyset \triangleright \emptyset' \iff \emptyset \supseteq \emptyset' \]
Coverage is *not* a necessary condition for subsumption:

Coverage is *not equivalent to subsumption*:

\[ \mathcal{O} \ni \mathcal{O}' \iff \mathcal{O} \supseteq \mathcal{O}'. \]

**Example:**

\[ \begin{array}{c}
\mathcal{N}_1 \xrightarrow{a} \mathcal{N}_2 \\
\mathcal{N}_1' \xrightarrow{a} \mathcal{N}_2'
\end{array} \]

- \( \mathcal{O} \ni \mathcal{O}' \), as \( \| \mathcal{O} \| = \| \mathcal{O}' \| = \{aa\} \).
- \( \mathcal{N} = \{\mathcal{N}_1, \mathcal{N}_2\} \), thus \( \mathcal{N}_\varepsilon = \emptyset \).
- Temporal coverage is missing, as for \( \mathcal{N}_1 \mapsto \mathcal{N}_2 \) in \( \mathcal{O} \), we have \( \mathcal{N}_1' \not\mapsto \mathcal{N}_2' \) in \( \mathcal{O}' \).
- The same negative result comes by choosing \( \mathcal{N} = \{\mathcal{N}_2, \mathcal{N}_1\} \).
- Thus, \( \mathcal{O} \not\ni \mathcal{O}' \) even if \( \mathcal{O} \ni \mathcal{O}' \).
Algorithm $\text{Covers}(\mathcal{O}, \mathcal{O}')$: Boolean

$\mathcal{O} = (\mathcal{N}, \mathcal{L}, \mathcal{A})$, $\mathcal{O}' = (\mathcal{N}', \mathcal{L}', \mathcal{A}')$;

begin

$n := |\mathcal{N}|$; $n_\epsilon := \{N \mid N \in \mathcal{N}, \epsilon \in \|N\|\}$;

$n' := |\mathcal{N}'|$, $n'_\epsilon := \{N' \mid N' \in \mathcal{N}', \epsilon \in \|N'\|\}$;

if $n < n'$ or $n_\epsilon - n'_\epsilon < n - n'$ or $\mathcal{L} \not\supseteq \mathcal{L}'$ then return \textbf{false};

Create the multisets $\mathcal{M}$ and $\mathcal{M}'$ of instances of labels in $\mathcal{O}$, $\mathcal{O}'$;

d := n - n';

Remove $d$ instances of label $\epsilon$ from $\mathcal{M}$;

if $\mathcal{M} \not\supseteq \mathcal{M}'$ then return \textbf{false};

return $\text{CovStep}(\mathcal{O}, \mathcal{O}', \emptyset, \emptyset, \mathcal{M}, \mathcal{M}', d, \emptyset)$

end.
Function CovStep(\(\mathcal{O}, \mathcal{O}', \mathcal{C}, \mathcal{C}', \mathcal{M}, \mathcal{M}', d, \mathcal{R}\)): Boolean
begin
if \(|\mathcal{R}| = n'| then return true;
Pick up a node \(N' \in (\mathcal{N}' - \mathcal{C}')\) where all parents of \(N'\) are in \(\mathcal{C}'\);
\(\mathcal{F} := \) the set of \(N \in (\mathcal{N} - \mathcal{C})\) where all parents of \(N\) are in \(\mathcal{C}\);
for each \(N \in \mathcal{F}\) do
if \(\|N\| \geq \|N'\| \) and \((\mathcal{M} - \|N\|) \supseteq (\mathcal{M}' - \|N'\|)\) then
\(\mathcal{N}_a := \) the set of nearest ancestors of \(N\) which are in \(\mathcal{R}(N')\);
if \(\forall N_a \in \mathcal{N}_a, (N_a, N'_a) \in \mathcal{R} (N'_a < N')\) then
if CovStep(\(\mathcal{O}, \mathcal{O}', \mathcal{C} \cup \{N\}, \mathcal{C}' \cup \{N'\}, \mathcal{M} - \|N\|, \mathcal{M}' - \|N'\|, d, \mathcal{R} \cup \{(N, N')\}\)) then return true end-if
end-if
end-if;
if \(\epsilon \in \|N\| \) and \(d > 0 \) and \((\mathcal{M} - \|N\|) \supseteq \mathcal{M}'\) then
if CovStep(\(\mathcal{O}, \mathcal{O}', \mathcal{C} \cup \{N\}, \mathcal{C}', \mathcal{M} - \|N\|, \mathcal{M}', d - 1, \mathcal{R}\)) then return true end-if
end-if
end-for;
return false
end.
Testing Coverage – Example: Tracing of Covers($\mathcal{O}_1, \mathcal{O}_2$)

Figure: CovStep call tree

Figure: $\mathcal{O}_1$ and $\mathcal{O}_2$

<table>
<thead>
<tr>
<th>Call</th>
<th>$\mathcal{C}$</th>
<th>$\mathcal{C}'$</th>
<th>$\mathcal{M}$</th>
<th>$\mathcal{M}'$</th>
<th>$d$</th>
<th>$\mathcal{R}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$a^3 b^4 cdf \epsilon^2$</td>
<td>$a^2 bcd \epsilon^2$</td>
<td>1</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>$\emptyset$</td>
<td>$a^2 b^3 cdf \epsilon^2$</td>
<td>$a^2 bcd \epsilon^2$</td>
<td>0</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>1$'$</td>
<td>$ab^2 cdf \epsilon^2$</td>
<td>$abcd \epsilon^2$</td>
<td>0</td>
<td>$2, 1'$</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>1$'$</td>
<td>$ab^3 cdf \epsilon^2$</td>
<td>$abcd \epsilon^2$</td>
<td>1</td>
<td>$2, 1'$</td>
</tr>
<tr>
<td>5</td>
<td>12</td>
<td>1$'2'$</td>
<td>$ab^2 cdf \epsilon$</td>
<td>$acd \epsilon$</td>
<td>1</td>
<td>$2, 1')(1, 2')$</td>
</tr>
<tr>
<td>6</td>
<td>123</td>
<td>1$'2'$</td>
<td>$ab^2 cde$</td>
<td>$acd \epsilon$</td>
<td>0</td>
<td>$2, 1')(1, 2')$</td>
</tr>
<tr>
<td>7</td>
<td>1234</td>
<td>1$'2'3'$</td>
<td>$ab \epsilon$</td>
<td>$a \epsilon$</td>
<td>0</td>
<td>$2, 1')(1, 2')(4, 3') $</td>
</tr>
<tr>
<td>8</td>
<td>12345</td>
<td>1$'2'3'4'$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>0</td>
<td>$(2,1')(1,2')(4,3')(5,4')$</td>
</tr>
</tbody>
</table>
Experimental Results

- Rapid prototyping in *Haskell* functional language.
- Subsumption checking via *Subsumes* and *Covers*.
- Different classes of observations.
- Test cases where subsumption holds.
- No significant difference in space allocation.
- Considerable difference in response time.
Experimental Results: Space Allocation

**Figure:** Checking subsumption: space allocation

A. Ducoli, G. Lamperti, E. Piantoni, M. Zanella

Coverage Techniques for Checking Temporal-Observation Subsumption
Experimental Results: Response Time

Figure: Checking subsumption: response time
Proposal of a technique for checking observation-subsumption in diagnosis of DESs.

Need for subsumption-checking in similarity-based diagnosis, in order to pursue reuse of model-based reasoning.

Subsumption-checking based on its definition requires comparison of index spaces.

Need for avoiding index-space manipulation during the search in the KB.

Better checking either necessary or sufficient conditions for subsumption within a unified algorithm.

Technique implemented and tested.

Considerable advantages in response time.