Conflicts Detection and Diagnosis in Configuration

Alexander Felfernig*, Stefan Reiterer†, Florian Reinfrank*, Gerald Ninaus*, and Michael Jeran*

*Graz University of Technology, Graz, Austria
†SelectionArts, Graz, Austria
Contents

• Anomalies

• Conflict Detection
  • Simple Conflict Detection
  • QuickXPlain

• Diagnosis
  • HSDAG
  • Duality of Conflicts and Diagnoses
  • FastDiag
Anomalies

• Parts of a knowledge base that conform to a defined pattern of unintended structures [Chandola et al., 2009].

• Anomaly detection:
  • Automated testing and debugging (minimal subsets)
  • Redundancy detection (maximal subsets)
Working Example: Knowledge Base

\[ C_{KB} = \{
\]
\[ c_\alpha : \forall X : MB(X) \rightarrow \exists Y : cpu-of-mb(Y, X).
\]
\[ c_\beta : \forall X : CPU(X) \rightarrow \exists Y : mb-of-cpu(Y, X).
\]
\[ c_\gamma : \forall X : MB(X) \leftrightarrow MBSilver(X) \lor MBDiamond(X).
\]
\[ c_\delta : \forall X : \neg MBSilver(X) \lor \neg MBDiamond(X).
\]
\[ c_\epsilon : \forall X : CPU(X) \leftrightarrow CPUD(X) \lor CPUS(X).
\]
\[ c_\phi : \forall X, Y : cpu-of-mb(X,Y) \leftrightarrow mb-of-cpu(Y,X).
\]
\[ c_\iota : \forall X : \neg CPUD(X) \lor \neg CPUS(X).
\]
\[ \]
\[ c_1 : \forall X, Y : cpu-of-mb(Y, X) \land CPUS(Y) \rightarrow MBDiamond(X).
\]
\[ c_2 : \forall X, Y : cpu-of-mb(Y, X) \land CPUS(Y) \rightarrow MBSilver(X). /*should be: replacement of c1*/
\]
\[ c_3 : \forall X, Y : cpu-of-mb(Y, X) \land CPUD(Y) \land MBSilver(X) \rightarrow false.
\]
\[ c_4 : \forall X, Y : cpu-of-mb(Y, X) \land CPUS(Y) \land MBDiamond(X) \rightarrow false.
\]
\[ c_5 : \forall X : CPUD(X) \rightarrow false. /*should be disabled, but still active*/
\]
\[ \}
\]
Minimal Conflicts

**Definition (Minimal Conflict Set).** A conflict set $CS = \{ca, cb, \ldots, cz\}$ is a subset of $C$ such that inconsistent $(B \cup CS)$. $AC = B \cup C$ represents the set of all constraints in the knowledge base ($AC = \{c_1, c_2, \ldots, c_n\}$), $B$ represents the background knowledge (no conflict elements are assumed to be included in $B$), and $C$ represents the set of constraints subject of conflict search. A conflict set $CS$ is *minimal* if there does not exist a $CS \subset CS$ that has the conflict property.
Minimal Conflicts

**FIGURE 7.1**

A conflict set CS is a subset of C ($AC = C \cup B$), which is inconsistent with B. CS is minimal if no subset of CS fulfills the conflict set property. In this context, B is the background knowledge that includes all constraints considered correct. An example conflict set is $CS_1 = \{c_1, c_4, c_5\}$. 
Working Example: Minimal Conflicts

- Assumption: $C = AC$, i.e., $B = \emptyset$
- $CS1 = \{c1, c4, c5\}$
- $CS2 = \{c1, c2, c5\}$
- $\text{inconsistent}(\{c1, c4, c5\} \cup B)$
- $\text{inconsistent}(\{c1, c2, c5\} \cup B)$
- Two algorithms to determine minimal conflicts:
  - Simple Conflict Detection
  - QuickXPlain
Simple Conflict Detection

**Algorithm 7.1 – SimpleConflictDetection**

```plaintext
1 func SimpleConflictDetection(C ⊆ AC, B = AC − C) : CS
2    CS ← ∅;
3    if inconsistent(B) or consistent(B ∪ C) return(∅);
4    else
5      repeat
6        Φ = CS;
7        repeat
8          c ← element(C − Φ);
9          Φ ← Φ ∪ {c};
10         until inconsistent(Φ)
11        CS ← CS ∪ {c};
12        until inconsistent(CS)
13      return(CS);
```

Best Case: \[2\]

Worst Case: \[\frac{(n \times (n+1))}{2} + n\]
Simple Conflict Detection: Execution

### Table 7.1 Example of the application of `SIMPLECONFLICTDETECTION`.

CS = \{c_1, c_4, c_5\} is returned as minimal conflict set (CS) for C = \{c_5, c_4, c_3, c_2, c_1\} and B = \{c_\alpha, c_\beta, c_\gamma, c_\delta, c_\epsilon, c_\phi, c_1\}.

<table>
<thead>
<tr>
<th>Step</th>
<th>CS</th>
<th>c</th>
<th>(\Phi)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>\emptyset</td>
<td>c_5</td>
<td>{c_5}</td>
</tr>
<tr>
<td>2</td>
<td>\emptyset</td>
<td>c_4</td>
<td>{c_5, c_4}</td>
</tr>
<tr>
<td>3</td>
<td>\emptyset</td>
<td>c_3</td>
<td>{c_5, c_4, c_3}</td>
</tr>
<tr>
<td>4</td>
<td>\emptyset</td>
<td>c_2</td>
<td>{c_5, c_4, c_3, c_2}</td>
</tr>
<tr>
<td>5</td>
<td>\emptyset</td>
<td>c_1</td>
<td>{c_5, c_4, c_3, c_2, c_1}</td>
</tr>
<tr>
<td>6</td>
<td>{c_1}</td>
<td>c_5</td>
<td>{c_1, c_5}</td>
</tr>
<tr>
<td>7</td>
<td>{c_1}</td>
<td>c_4</td>
<td>{c_1, c_5, c_4}</td>
</tr>
<tr>
<td>8</td>
<td>{c_1, c_4}</td>
<td>c_5</td>
<td>{c_1, c_4, c_5}</td>
</tr>
<tr>
<td>9</td>
<td>{c_1, c_4, c_5}</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>
QuickXPlain

Algorithm 7.2 – QuickXPlain

1. func QuickXPlain(C ⊆ AC, B = AC – C) : CS
2. if isEmpty(C) or inconsistent(B) return ∅;
3. else return QX(∅, C, B);
4. func QX(D, C = {c₁..c₉}, B) : CS
5. if D ≠ ∅ and inconsistent(B) return ∅;
6. if singleton(C) return C;
7. k = \[\frac{q}{2}\];
8. C₁ = {c₁..cₖ}; C₂ = {cₖ₊₁..c₉};
9. CS₁ = QX(C₂, C₁, B ∪ C₂);
10. CS₂ = QX(CS₁, C₂, B ∪ CS₁);
11. return(CS₁ ∪ CS₂);

Best Case:
\[\log_2\left(\frac{n}{k}\right) + 2k\]

Worst Case:
\[2k \times \log_2\left(\frac{n}{k}\right) + 2k\]
QuickXPlain

**Table 7.2** Example of QUICKXPLAIN: $\Gamma = \{c_\alpha, c_\beta, c_\gamma, c_\delta, c_\varepsilon, c_\phi, c_1\}$ is the (original) background knowledge and $CS = \{c_1, c_4, c_5\}$ is the returned conflict set. The sequence of the different QX activations is depicted in Figure 7.2.

<table>
<thead>
<tr>
<th>Step</th>
<th>D</th>
<th>C</th>
<th>B</th>
<th>$C_1$</th>
<th>$C_2$</th>
<th>Return</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\emptyset$</td>
<td>${c_1, \ldots, c_5}$</td>
<td>$\Gamma$</td>
<td>${c_1, c_2, c_3}$</td>
<td>${c_4, c_5}$</td>
<td>${c_1, c_4, c_5}$</td>
</tr>
<tr>
<td>2</td>
<td>${c_4, c_5}$</td>
<td>${c_1, c_2, c_3}$</td>
<td>$\Gamma \cup {c_4, c_5}$</td>
<td>${c_1, c_2}$</td>
<td>${c_3}$</td>
<td>${c_1}$</td>
</tr>
<tr>
<td>3</td>
<td>${c_3}$</td>
<td>${c_1, c_2}$</td>
<td>$\Gamma \cup {c_3, \ldots, c_5}$</td>
<td>${c_1}$</td>
<td>${c_2}$</td>
<td>${c_1}$</td>
</tr>
<tr>
<td>4</td>
<td>${c_2}$</td>
<td>${c_1}$</td>
<td>$\Gamma \cup {c_2, \ldots, c_5}$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>${c_1}$</td>
</tr>
<tr>
<td>5</td>
<td>${c_1}$</td>
<td>${c_2}$</td>
<td>$\Gamma \cup {c_1, c_3, c_4, c_5}$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>6</td>
<td>${c_1}$</td>
<td>${c_3}$</td>
<td>$\Gamma \cup {c_1, c_4, c_5}$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>7</td>
<td>${c_1}$</td>
<td>${c_4, c_5}$</td>
<td>$\Gamma \cup {c_1}$</td>
<td>${c_4}$</td>
<td>${c_5}$</td>
<td>${c_4, c_5}$</td>
</tr>
<tr>
<td>8</td>
<td>${c_5}$</td>
<td>${c_4}$</td>
<td>$\Gamma \cup {c_1, c_5}$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>${c_4}$</td>
</tr>
<tr>
<td>9</td>
<td>${c_4}$</td>
<td>${c_5}$</td>
<td>$\Gamma \cup {c_1, c_4}$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>${c_5}$</td>
</tr>
</tbody>
</table>
Runtime of Conflict Detection Algorithms

Table 7.3 Runtime evaluation: The average runtime in milliseconds (ms) needed by SIMPLECONFLICTDETECTION (SCD) and QUICKXPLAIN to calculate one minimal conflict set (on a standard PC). The basis for this evaluation are knowledge bases from www.splot-research.org.

<table>
<thead>
<tr>
<th>Domain</th>
<th>#con.</th>
<th>#var.</th>
<th>QUICKXPLAIN (ms)</th>
<th>SCD (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELL laptops</td>
<td>285</td>
<td>47</td>
<td>75.7</td>
<td>643.2</td>
</tr>
<tr>
<td>Smarthomes</td>
<td>73</td>
<td>55</td>
<td>42.6</td>
<td>89.6</td>
</tr>
<tr>
<td>Cars</td>
<td>150</td>
<td>73</td>
<td>42.9</td>
<td>406.2</td>
</tr>
<tr>
<td>Xerox printers</td>
<td>242</td>
<td>158</td>
<td>78.1</td>
<td>812.2</td>
</tr>
</tbody>
</table>
Definition (Diagnosis Task). A *diagnosis task* can be defined by the tuple \((C, AC)\) where \(AC = B \cup C\), \(B\) is the background knowledge, and \(C\) is the set of constraints to be analyzed.

Definition (Diagnosis). A *diagnosis* for a given diagnosis task \((C, AC)\) is a set of constraints \(\Delta \subseteq C\) such that \(B \cup C - \Delta\) is consistent. A diagnosis \(\Delta\) is *minimal* if there does not exist a diagnosis \(\Delta' \subset \Delta\) with the diagnosis property. Finally, a minimal diagnosis \(\Delta\) is denoted as *minimal cardinality diagnosis* if there does not exist a minimal diagnosis with \(|\Delta'| < |\Delta|\).
A diagnosis $\Delta$ is a subset of $C$ ($AC = C \cup B$) such that $B \cup C - \Delta$ is consistent. $\Delta$ is minimal if no subset of $\Delta$ fulfills the diagnosis property. $B$ again represents the background knowledge. An example diagnosis is $\Delta_1 = \{c_1\}$. 

**FIGURE 7.3**
Breadth-first based search for diagnoses on the basis of the minimal conflict sets $CS_1 = \{c_1, c_4, c_5\}$ and $CS_2 = \{c_1, c_2, c_5\}$. The resulting minimal diagnoses are $\Delta_1 = \{c_1\}$, $\Delta_2 = \{c_5\}$, and $\Delta_3 = \{c_2, c_4\}$. 
Duality of Conflicts and Diagnoses

FIGURE 7.5
Breadth-first based search for conflicts on the basis of the minimal diagnoses $\Delta_1 = \{c_1\}$, $\Delta_2 = \{c_5\}$, and $\Delta_3 = \{c_2, c_4\}$. The resulting minimal conflict sets are $CS_1 = \{c_1, c_2, c_5\}$, $CS_2 = \{c_1, c_4, c_5\}$. 
FastDiag

Algorithm 7.3 – FastDiag

1. func FastDiag($C \subseteq AC$, $AC = \{c_1..c_t\}$) : diagnosis $\Delta$
2. if isEmpty($C$) or inconsistent($AC - C$) return $\emptyset$
3. else return FD($\emptyset$, $C$, $AC$);

4. func FD($D$, $C = \{c_1..c_q\}$, $AC$) : diagnosis $\Delta$
5. if $D \neq \emptyset$ and consistent($AC$) return $\emptyset$;
6. if singleton($C$) return $C$;
7. $k = \frac{q}{2}$;
8. $C_1 = \{c_1..c_k\}$; $C_2 = \{c_{k+1}..c_q\}$;
9. $D_1 = FD(C_2, C_1, AC - C_2)$;
10. $D_2 = FD(D_1, C_2, AC - D_1)$;
11. return($D_1 \cup D_2$);

Best Case:
$\log_2\left(\frac{n}{d}\right) + 2d$

Worst Case:
$2d \times \log_2\left(\frac{n}{d}\right) + 2d$
## Runtime of Diagnosis Algorithms

### Table 7.5  Runtime evaluation: The average runtime in milliseconds (ms) needed by HSDAG and FASTDIAG to calculate one minimal diagnosis (on a standard PC). The basis for this evaluation are knowledge bases from www.splot-research.org (Dell laptops (laptops), smarthomes (homes), cars, and Xerox printers (printers)).

<table>
<thead>
<tr>
<th>Domain</th>
<th>#con.</th>
<th>#var.</th>
<th>FASTDIAG (ms)</th>
<th>HSDAG (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>#Δ_{i} 1</td>
<td>#Δ_{i} 5</td>
</tr>
<tr>
<td>Laptops</td>
<td>285</td>
<td>47</td>
<td>1638.7</td>
<td>2792.3</td>
</tr>
<tr>
<td>Homes</td>
<td>73</td>
<td>55</td>
<td>593.1</td>
<td>2433.5</td>
</tr>
<tr>
<td>Cars</td>
<td>150</td>
<td>73</td>
<td>1404.4</td>
<td>2730.8</td>
</tr>
<tr>
<td>Printers</td>
<td>242</td>
<td>158</td>
<td>2871.9</td>
<td>6927.2</td>
</tr>
</tbody>
</table>
Diagnosis as Optimization Problem

- Each constraint $c_i$ is represented by a variable $v_i \in [0,1]$
- Conflict sets $CS_j$ are represented by constraints $cs_j$
- Example: $cs_1: x_1 + x_4 + x_5 \geq 1$

\[

cs_1 : x_1 + x_4 + x_5 \geq 1.
\]
\[

cs_2 : x_1 + x_2 \geq 1.
\]
\[

cs_3 : x_1 + x_3 \geq 1.
\]
\[

cs_4 : x_2 + x_3 \geq 1.
\]

- Optimization function:

\[
\text{minimize} : x_1 + x_2 + x_3 + x_4 + x_5
\]
Diagnosis as Optimization Problem

Table 7.6  Representation of a diagnosis task as optimization problem. In this case, all minimal conflict sets (CS$_1$, ..., CS$_4$) have to be determined before the optimization can start (1 (0) denotes the fact that $c_i$ is part (not part) of the minimal conflict set).

<table>
<thead>
<tr>
<th>Conflict Set</th>
<th>$c_1$</th>
<th>$c_2$</th>
<th>$c_3$</th>
<th>$c_4$</th>
<th>$c_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS$_1$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CS$_2$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CS$_3$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CS$_4$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**cs1:** $x_1 + x_4 + x_5 \geq 1$
Solution Space: Predefined Instances

Table 7.7 A simple configuration problem defined by the variables $V = \{v_1, v_2, v_3\}$, $\text{dom}(v_i) = \{1, 2, 3\}$, and the constraint $c_p = conf_1 \lor conf_2 \lor conf_3 \lor conf_4 \in C_{KB}$.

<table>
<thead>
<tr>
<th>Variables</th>
<th>$conf_1$</th>
<th>$conf_2$</th>
<th>$conf_3$</th>
<th>$conf_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1$</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$v_2$</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>$v_3$</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 7.8 Example user requirements $C_R$ and their relationship to the configurations $conf_1, \ldots, conf_4$ ($1 = \text{requirement supported}, \ 0 = \text{not supported}$).

<table>
<thead>
<tr>
<th>User Requirements</th>
<th>$conf_1$</th>
<th>$conf_2$</th>
<th>$conf_3$</th>
<th>$conf_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1 = 1$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$v_2 = 1$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$v_3 = 1$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>support</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
Exercises

1. Given the following set of constraints $AC=\{x_1=1, x_1=2, x_2=x_1, x_3=x_2, x_3>2\}$ ($\text{dom}(x_i)=[1,2,3]$). Determine the complete set of minimal conflicts on the basis of Simple Conflict Detection.

2. For the identified minimal conflict sets determine the corresponding complete set of minimal diagnoses (on the basis of HSDAG).

3. Show how to use the HSDAG concept to determine the complete set of minimal conflicts from the diagnoses determined in 2 (duality of diagnoses and conflicts).

4. For the minimal conflict sets (from 1. and 3.), show how to represent diagnosis as optimization problem.
Thank You!
References (1)


References (2)


References (3)


