LU Einführung in Wissensbasierte Systeme
Aufgabenblatt 1: Modellbasierte Diagnose

Michael Fink und Hans Tompits
Arbeitsbereich Wissensbasierte Systeme 184/3,
Technische Universität Wien

Wintersemester 2010/2011

19. Oktober 2010

Allgemeine Problemstellung

Die erste Übungsaufgabe stammt aus dem Bereich der modellbasierten Diagnose. Ziel dieses ersten Übungsblocks ist die Erstellung eines automatisierten Hilfsmittels zur Diagnose einer einfachen Klimaanlage. Es handelt sich dabei um einen Teil einer Klimaanlage, der die Außenluft beheizt oder abkühlt. Ob geheizt oder gekühlt werden soll und wie stark, wird von einem externen Regelungsmechanismus vorgegeben.

Die zu diagnostizierende Anlage ist in Abbildung 1 schematisch dargestellt. Sie besteht aus einem Eingangsventil `s`, einem Heizelement `h`, zwei Kühlelementen $\texttt{c}_1$ und $\texttt{c}_2$ und einem Ausgangsventil `v`. Luftleitungen sind durch graue Balken, Steuerleitungen durch Linien dargestellt. An Luftleitungen können zwei Größen gemessen werden: die Temperatur `T`, hier ganzzahlig zwischen 0 und 60, und ein Luftstrom `S`, hier vereinfacht binär (strömt oder nicht). An Steuerleitungen kann jeweils einer von vorher festgelegten, endlich vielen symbolischen Werten anliegen.
Die Anlage besitzt folgende Eingänge:

- Die Außenluft `air_in`,
- den Wert eines Dreiwegschalters `threeway_in`, der die symbolischen Werte `cool`, `heat` oder `off` annehmen kann, und
- den Wert einer vierstufigen Regelgröße `scale_in`, der `0`, `1`, `2`, `3` sein kann.

Sie besitzt einen Ausgang,

- einen Luftstrom `air_out`.

Die Funktionsweise der Anlage kann man sich wie folgt vorstellen: Der Wert von `threeway_in` bestimmt, ob die Klimaanlage heizen (`heat`), kühlen (`cool`) oder ausgeschaltet sein (`off`) soll. Der entsprechende Wert liegt sowohl am Eingangsventil `s` (Eingang `i2`) als auch am Ausgangsventil `v` an (`i2`). Der Wert von `scale_in` bestimmt die Leistung der Anlage (`3` entspricht maximaler Leistung). Die Stellung dieses Leistungsreglers liegt an dem Heizelement `h` und an den Kühlelementen `c1` und `c2` (jeweils Eingang `i2`) an.

Die Außenluft `air_in` strömt in das Eingangsventil `s`, das den Luftstrom entsprechend der Schalterstellung zu dem Heiz- oder einem Kühlelement leitet, oder ganz blockiert. Die beiden funktionsgleichen Kühlelemente `c1` und `c2` sind für eine erhöhte Kühlleistung in Serie geschalten. Nachdem die Luft durch ein Heiz- bzw. Kühlelement geströmt ist (falls

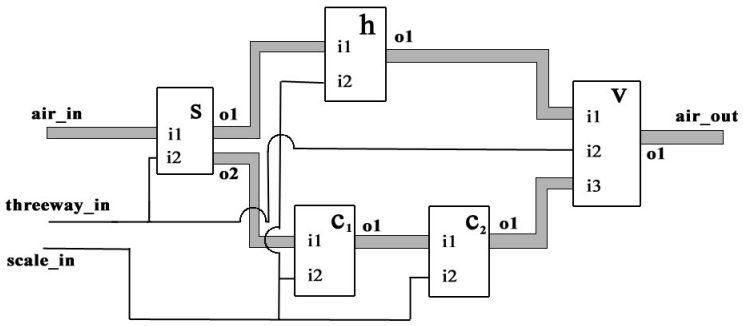

Abbildung 1: Aufbau der zu diagnostizierenden Klimaanlage

die Luft dort überhaupt ankam), bewegt sie sich weiter zum Ausgangsventil, das anhand der Schalterstellung den richtigen Luftstrom liefert bzw. den Luftstrom blockiert.

Der Luftstrom der Anlage ist durch einen "Temperaturstrom" und eine "Strömung" modelliert, d.h. an einem Eingang P einer Komponente C einer Luftleitung gibt es einen Temperaturwert $T(C, P)$ (ein Zahlenwert) und einen Strömungswert $S(C, P)$ (hier vereinfacht entweder 0 oder 1). Die Werte an Steuerleitungen am Punkt P werden mit $D(C, P)$ (im entsprechenden Wertebereich) modelliert. Die einzelnen Elemente verhalten sich dabei wie folgt (falls sie nicht defekt sind):

- Das Eingangsventil `s` erhält den eingehenden Luftstrom (Eingang `i1`) und leitet ihn danach abhängig von dem Wert an `i2` von `s` weiter, und zwar
  - zu Ausgang `o1` wenn $D(\texttt{s}, \texttt{i2}) = \texttt{heat}$ (d.h. in diesem Fall gilt $S(\texttt{s}, \texttt{o1}) = S(\texttt{s}, \texttt{i1})$ und $S(\texttt{s}, \texttt{o2}) = 0$),
  - zu Ausgang `o2` wenn $D(\texttt{s}, \texttt{i2}) = \texttt{cool}$ (d.h. in diesem Fall gilt $S(\texttt{s}, \texttt{o2}) = S(\texttt{s}, \texttt{i1})$ und $S(\texttt{s}, \texttt{o1}) = 0$),
  - zu keinem von den Zweien wenn $D(\texttt{s}, \texttt{i2}) = \texttt{off}$ (d.h. in diesem Fall gilt $S(\texttt{s}, \texttt{o1}) = S(\texttt{s}, \texttt{o2}) = 0$).

  Die Temperatur wird bei vorhandener Strömung zum entsprechenden aktiven Ausgang propagiert ($T(\texttt{s}, \texttt{o1}) = T(\texttt{s}, \texttt{i1})$ genau wenn $S(\texttt{s}, \texttt{o1}) = 1$, analog für `o2`). Sonst entspricht die Temperatur einer festen Umgebungstemperatur (durch das Prädikat `ambient_t(T)` definiert).

- Das Heizelement `h` erwärmt die durchströmende Luft bei vorhandener Strömung (also falls $S(\texttt{h}, \texttt{o1}) = 1$) und falls $T(\texttt{h}, \texttt{i1}) < 30$
  - um den Wert $2 * D(\texttt{h}, \texttt{i2}) + 1$, d.h. $T(\texttt{h}, \texttt{o1}) = T(\texttt{h}, \texttt{i1}) + 2 * D(\texttt{h}, \texttt{i2}) + 1$.

  Falls $T(\texttt{h}, \texttt{i1}) \geq 30$ wird die Temperatur der strömenden Luft unverändert weitergeleitet (d.h. $T(\texttt{h}, \texttt{o1}) = T(\texttt{h}, \texttt{i1})$ falls $T(\texttt{h}, \texttt{i1}) \geq 30$ und $S(\texttt{h}, \texttt{o1}) = 1$).

  Die Strömung wird von `h` nicht beeinflusst ($S(\texttt{h}, \texttt{o1}) = S(\texttt{h}, \texttt{i1})$). Ist die Strömung am Ausgang 0, so entspricht die Temperatur am Ausgang `ambient_t`.

- Ein Kühlelement `c` (`c1` oder `c2`) kühlt die durchströmende Luft bei vorhandener Strömung (also falls $S(\texttt{c}, \texttt{o1}) = 1$) und falls $T(\texttt{c}, \texttt{i1}) \geq 10$

  - um den Wert $2 * D(\texttt{c}, \texttt{i2}) + 3$ ab, falls $T(\texttt{c}, \texttt{i1}) \geq 25$, d.h. $T(\texttt{c}, \texttt{o1}) = T(\texttt{c}, \texttt{i1}) - 2 * D(\texttt{c}, \texttt{i2}) - 3$.
  - um den Wert $2 * D(\texttt{c}, \texttt{i2})$, falls $T(\texttt{c}, \texttt{i1}) \leq 24$, d.h. $T(\texttt{c}, \texttt{o1}) = T(\texttt{c}, \texttt{i1}) - 2 * D(\texttt{c}, \texttt{i2})$.

  Falls $T(\texttt{c}, \texttt{i1}) < 10$ wird die Temperatur der strömenden Luft unverändert weitergeleitet (d.h. $T(\texttt{c}, \texttt{o1}) = T(\texttt{c}, \texttt{i1})$ falls $T(\texttt{c}, \texttt{i1}) < 10$ und $S(\texttt{c}, \texttt{o1}) = 1$).

  Die Strömung wird von `c` nicht beeinflusst (d.h. $S(\texttt{c}, \texttt{o1}) = S(\texttt{c}, \texttt{i1})$). Ist diese am Ausgang 0, so entspricht die Temperatur am Ausgang `ambient_t`.

- Ist das Ausgangsventil `v` auf heizen gestellt ($D(\texttt{v}, \texttt{i2}) = \texttt{heat}$), wird der Luftstrom des Heizelements `h` abgegeben ($S(\texttt{v}, \texttt{o1}) = S(\texttt{v}, \texttt{i1})$) und, wenn Strömung herrscht ($S(\texttt{v}, \texttt{o1}) = 1$), auch dessen Temperatur weitergeleitet ($T(\texttt{v}, \texttt{o1}) = T(\texttt{v}, \texttt{i1})$), sonst entspricht die Temperatur `ambient_t`. Ist das Ausgangsventil `v` auf kühlen gestellt ($D(\texttt{v}, \texttt{i2}) = \texttt{cool}$) und herrscht am Eingang `i3` Strömung ($S(\texttt{v}, \texttt{i3}) = 1$), so wird Luftstrom am Eingang `i3` zum Ausgang `o1` weitergeleitet ($S(\texttt{v}, \texttt{o1}) = S(\texttt{v}, \texttt{i3})$). Wenn Strömung herrscht ($S(\texttt{v}, \texttt{o1}) = 1$), so wird auch dessen Temperatur weitergeleitet ($T(\texttt{v}, \texttt{o1}) = T(\texttt{v}, \texttt{i3})$), ansonsten entspricht die Temperatur `ambient_t`. Ist $D(\texttt{v}, \texttt{i2}) = \texttt{off}$, dann gilt $S(\texttt{v}, \texttt{o1}) = 0$, und die Temperatur entspricht der durch `ambient_t` definierten.

Aufgabe 1.1: Definition der Komponenten

Im ersten Teil der Aufgabe sollen Sie die Funktionsweise der einzelnen Komponenten in Answer-Set Programming, der "Kernsprache" von DLV, kodieren. Dazu sind folgende Prädikate zu verwenden:

- Prädikat $\texttt{t}(\texttt{C}, \texttt{I}, \texttt{V})$ repräsentiert den Temperaturwert an einem Anschluß einer Komponente:
  - `C` ist der Name der Komponente,
  - `I` ein "Messpunkt" (z.B. `i2`, `o1`), und
  - `V` ein "Messwert".

  Zum Beispiel bedeutet $\texttt{t}(\texttt{h}, \texttt{o1}, 30)$ dass am Anschluß `o1` der Komponente `h` die Temperatur 30 beträgt.

- Prädikat $\texttt{s}(\texttt{C}, \texttt{I}, \texttt{V})$ repräsentiert den Strömungswert an einem Anschluß einer Komponente (analog zur Temperatur). Zum Beispiel bedeutet $\texttt{s}(\texttt{s}, \texttt{o2}, 1)$ dass am Anschluß `o2` der Komponente `s` ein Luftstrom herrscht.

- Prädikat $\texttt{d}(\texttt{C}, \texttt{I}, \texttt{V})$ repräsentiert den Datenwert eines Anschlusses einer Komponente. Zum Beispiel bedeutet $\texttt{d}(\texttt{v}, \texttt{i2}, \texttt{cool})$ dass am Anschluß `i2` der Komponente `v` der Wert `cool` liegt.

- Prädikat $\texttt{ambient\_t}(\texttt{T})$ definiert die Umgebungstemperatur T.

- Prädikate $\texttt{heater}(\texttt{H})$, $\texttt{cooler}(\texttt{C})$, $\texttt{switch}(\texttt{S})$, $\texttt{valve}(\texttt{V})$ definieren, welche Komponenten Heizer, Kühler, Eingangs- oder Ausgangsventile sind.

Aufgabe 1.1.1: Testfälle für Komponenten

Überlegen Sie sich zuerst, für jede Komponente, fünf Testfälle, die die Spezifikationen der einzelnen Komponenten abdecken. Zum Beispiel:

```
heater(h). ambient_t(20).
s(h, i1, 1). t(h, i1, 10). d(h, i2, 3).
expect_s(h, i1, 1).expect_t(h, i1, 10).
expect_s(h, o1, 1). expect_t(h, o1, 17).
```

Dieser Testfall sagt aus, dass `h` ein Heizelement ist, und dass die Umgebungstemperatur 20 Grad beträgt. Weiters liegt an `i1` ein aktiver Luftstrom der Temperatur 10 an, und am Eingang `i2` der Wert 3. Somit erwarten wir, dass am Ausgang `o1` ein aktiver Luftstrom der Temperatur 17 (= $10 + 2 * 3 + 1$) anliegt.

Nennen Sie diese Files C`.test`n`.dl`, wobei $C \in \{$`s`, `h`, `c`, `v`$\}$ (für `switch`, `heater`, `cooler`, `valve`) und n eine fortlaufende Nummerierung ist ($1 \leq n \leq 5$).

Aufgabe 1.1.2: Modellierung der Komponenten

Schreiben Sie nun vier DLV Programme, `s.dl`, `h.dl`, `c.dl` und `v.dl`, die die Funktionsweisen der einzelnen Komponenten beschreiben. Wichtig: Benützen sie hier nicht die konkreten Konstanten aus der Abbildung (`h`, `c1`, etc.), sondern die Prädikate `heater(H)`, `cooler(C)`, etc. Verwenden Sie die Prädikate `ab(C)`, wie in der Einführungsvorlesung erklärt. Diese sollen die Regeln außer Kraft setzen, wenn eine Komponente `C` nicht normal funktioniert.

Hinweise:

- Verwenden Sie DLVs *built-in* Prädikate, z.B.: $X = Y + Z$. Um eine korrekte Interpretation arithmetischer built-in Prädikate zu gewährleisten, ist es notwendig einen maximalen Zahlenbereich festzulegen, für dieses Beispiel reicht der Zahlenbereich $[0, 60]$. Dieser wird festgelegt, indem Sie bei den Aufrufen auf der Kommandozeile $-$`N` $=$ `60` angeben.
- Beachten Sie, dass sich eine Subtraktion $X = Y - Z$ (im gegeben Zahlenbereich) mit Hilfe des built-in Prädikats für die Additon als $Y = X + Z$ ausdrücken läßt.
- Vergessen Sie nicht, dass bei der konsistenzbasierten Diagnose mittels DLV das unäre Prädikat `ab` (für abnormal) zu verwenden ist um Hypothesen zu spezifizieren. Weiters darf `ab` in der Theorie nur negativ (default-negiert) vorkommen, also z.B. als `not ab(H)`, und es ist das einzige Prädikat, das (default-) negiert vorkommen darf.

Aufgabe 1.1.3: Test der Modellierung

Verwenden Sie nun Ihre Testfälle um Ihre Modellierung zu testen. Nützlich dazu ist folgendes DLV Programm (`component.tester.dl`):

```
UNCOMPUTED_t(C, O, X) :- expect_t(C, O, X), not t(C, O, X).
UNCOMPUTED_s(C, O, X) :- expect_s(C, O, X), not s(C, O, X).
UNEXPECTED_t(C, O, X) :- t(C, O, X), not expect_t(C, O, X).
UNEXPECTED_s(C, O, X) :- s(C, O, X), not expect_s(C, O, X).
DUPLICATED_t(C, O, X, Y) :- t(C, O, X), t(C, O, Y), X < Y.
DUPLICATED_s(C, O, X, Y) :- s(C, O, X), s(C, O, Y), X < Y.
```

Das bedeutet, wenn an einem Ausgang `O` einer Komponente `C` ein Wert `X` für Temperatur bzw. Strömung erwartet, aber nicht berechnet wird, dann soll `UNCOMPUTED_t(C, O, X)`

bzw. UNCOMPUTED_s(C, O, X) gelten. Wenn andererseits ein Wert X berechnet, aber nicht erwartet wird, soll UNEXPECTED_t(C, O, X) bzw. UNEXPECTED_s(C, O, X) gelten. Wenn für den Ausgang O einer Komponente C zwei unterschiedliche Werte X, Y berechnet werden, dann soll DUPLICATED_t(C, O, X, Y) bzw. DUPLICATED_s(C, O, X, Y) gelten. Beachten Sie, dass die hier verwendeten Komponenten keine Datenausgänge haben, deshalb braucht man keine entsprechenden Regeln für d(C, O, X).

Mit folgendem DLV Aufruf können Sie Ihre Modellierung nun testen:

dlv C.dl C.test$n$.dl component.tester.dl -N=60

wobei wieder $C \in \{\mathtt{s}, \mathtt{h}, \mathtt{c}, \mathtt{v}\}$ und n die entsprechende Testnummer ist.

Aufgabe 1.2: Definition der Anlage

Im zweiten Teil der Aufgabe werden Sie aus den einzelnen Komponenten das Gesamtsystem bilden. Beachten Sie dabei unbedingt folgende Konventionen:

- in_t(air_in, T) definiert den Temperaturwert der Eingangsluft,
- in_s(air_in, S) definiert den Luftstrom der Eingangsluft,
- in_d(threeway_in, DT) definiert den Datenwert des Dreiwegschalters,
- in_d(scale_in, DS) definiert den Datenwert des Reglers,
- out_t(air_out, T) definiert den Temperaturwert der Ausgangsluft,
- out_s(air_out, S) definiert den Luftstrom der Ausgangsluft.

Aufgabe 1.2.1: Testfälle für das Gesamtsystem

Analog zu den Einzelkomponenten definieren Sie zehn Testfälle für das Gesamtsystem, die möglichst alle Möglichkeiten abdecken. Ein Beispiel:

```
ambient_t(20).
in_t(air_in, 28). in_s(air_in, 1). in_d(threeway_in, cool). in_d(scale_in, 3).
expect_out_t(air_out, 13). expect_out_s(air_out, 1).
```

Nennen Sie diese Files connect.test$n$.dl, wobei n eine fortlaufende Nummerierung ist ($1 \leq n \leq 10$).

Aufgabe 1.2.2: Modellierung des Gesamtsystems

Modellieren Sie jetzt das Gesamtsystem. Dazu brauchen Sie nur die Komponentennamen definieren (verwenden Sie die Konstanten s, h, c1, c2 und v als jeweilige Komponentennamen), die globalen Eingänge und Ausgänge mit den entsprechenden Komponenteneinein- und ausgängen in Verbindung bringen, und die Verschaltung der Komponenten untereinander definieren. Legen Sie diese Definitionen im File connect.dl ab.

Aufgabe 1.2.3: Test der Modellierung

Verwenden Sie nun Ihre Testfälle, um Ihre Modellierung zu testen. Nützlich dazu ist folgende Adaption des obigen DLV Programms (bezeichnen Sie das neue Programm mit

connect.tester.dl):

$$\texttt{UNCOMPUTED\_t}(\texttt{O},\texttt{X}) : -\texttt{expect\_out\_t}(\texttt{O},\texttt{X}), \texttt{not out\_t}(\texttt{O},\texttt{X}).$$
$$\texttt{UNCOMPUTED\_s}(\texttt{O},\texttt{X}) : -\texttt{expect\_out\_s}(\texttt{O},\texttt{X}), \texttt{not out\_s}(\texttt{O},\texttt{X}).$$
$$\texttt{UNEXPECTED\_t}(\texttt{O},\texttt{X}) : -\texttt{out\_t}(\texttt{O},\texttt{X}), \texttt{not expect\_out\_t}(\texttt{O},\texttt{X}).$$
$$\texttt{UNEXPECTED\_s}(\texttt{O},\texttt{X}) : -\texttt{out\_s}(\texttt{O},\texttt{X}), \texttt{not expect\_out\_s}(\texttt{O},\texttt{X}).$$
$$\texttt{DUPLICATED\_t}(\texttt{O},\texttt{X},\texttt{Y}) : -\texttt{out\_t}(\texttt{O},\texttt{X}), \texttt{out\_t}(\texttt{O},\texttt{Y}), \texttt{X} < \texttt{Y}.$$
$$\texttt{DUPLICATED\_s}(\texttt{O},\texttt{X},\texttt{Y}) : -\texttt{out\_s}(\texttt{O},\texttt{X}), \texttt{out\_s}(\texttt{O},\texttt{Y}), \texttt{X} < \texttt{Y}.$$

Wie müssen die Aufrufe hier aussehen?

Aufgabe 1.3: Diagnose

Bisher wurde ja nur modelliert, jetzt wollen wir wirklich Diagnose betreiben. Dafür müssen noch ein paar Constraints definiert werden: Nehmen Sie die Regeln für DUPLICATED_t und DUPLICATED_s aus Aufgaben 1.1.3 und 1.2.3 und wandeln Sie diese in Constraints um, die Sie ins File constraints.dl ablegen. Bei einem korrekt funktionierendem System sollten DUPLICATED_t bzw. DUPLICATED_s ja nie wahr werden, und genau diesen Umstand sollen die Constraints ausdrücken.

Bei der Diagnose wird dann geprüft, für welche Komponenten C man ab(C) annehmen muss, damit die Modellierung gemeinsam mit den Beobachtungen konsistent sind. In diesem Beispiel bedeutet das vor allem, dass die oben definierten Constraints erfüllt sind.

Formulieren Sie dann dazu geeignete Hypothesen (jede der fünf Komponenten kann abnormal sein) und legen Sie sie im File abnormal.hyp ab.

Vorbereitende Übung: Testfälle "diagnostizieren"

Verwenden Sie Ihre Testfälle für das Gesamtsystem als Beobachtungen in einem Diagnoseproblem. Kopieren Sie die Files dazu, ersetzen Sie den Suffix .dl durch .obs und entfernen Sie den Präfix expect_ von den Prädikatsnamen.

Bestimmen Sie nun alle konsistenzbasierten Diagnosen (DLV Option -FR), single-fault konsistenzbasierte Diagnosen (DLV Option -FRsingle), und teilmengenminimale konsistenzbasierte Diagnosen (DLV Option -FRmin). Interpretieren Sie die so erhaltenen Lösungen.

Hinweise:

- Diese Aufgabe fließt nicht in die Benotung mit ein. Trotzdem empfehlen wir, diese Übung zu machen, um ein besseres Verständnis über ihr System zu erlangen.
- Beachten Sie, dass die zuvor aufgestellten Hypothesen für unser Testsystem waren. Gegebenfalls müssen Sie für ihre eigenen Testfälle ihre Hypothesen anpassen.

Aufgabe 1.3.1: Fehlerfälle diagnostizieren

Repräsentieren Sie jetzt die in Abbildung 2 dargestellten Beobachtungen in einem neuen File fault.obs.

Bestimmen Sie nun wieder alle konsistenzbasierten Diagnosen (DLV Option -FR), single-fault konsistenzbasierte Diagnosen (DLV Option -FRsingle), und teilmengenminimale konsistenzbasierte Diagnosen (DLV Option -FRmin) und interpretieren Sie die so erhaltenen Lösungen.

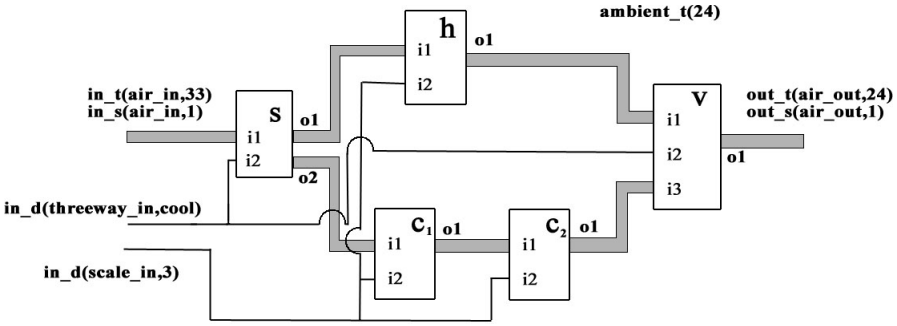

Abbildung 2: Ein beobachteter Fehlerfall

Aufgabe 1.3.2: Messpunkte

Betrachten Sie jetzt wieder die in Abbildung 2 dargestellte Situation und die minimalen Diagnosen. Sie haben den Verdacht, dass das Ausgangsventil `v` nicht defekt ist. An welchem Punkt würden Sie eine Messung vornehmen, um diesen Verdacht bestätigen zu können? Geben Sie eine Beispielmessung im File `fault.v.ok.obs` an, die zeigt, dass (gemeinsam mit den Beobachtungen aus Abbildung 2) `v` alleine nicht defekt ist.

Sie möchten jetzt weitere Messungen vornehmen, um zu sehen, was wirklich defekt ist, `s`, `c1`, `c2`, `v` oder mehrere Komponenten. Geben Sie Messwerte in einem File

`fault.s.c1.v.ok.obs`

an, sodass die einzige minimale Diagnose (gemeinsam mit den Beobachtungen aus Abbildung 2)

$$\{\texttt{ab(c2)}\}$$

lautet.

Hinweis: Es ist nicht notwendig, die Messergebnisse aus `fault.obs` mitzukopieren, da diese ja bereits in `fault.obs` modelliert sind.

Geben Sie ferner Messwerte in einem File `fault.c1.c2.ab.obs` an, sodass die einzige minimale Diagnose (gemeinsam mit den Beobachtungen aus Abbildung 2)

$$\{\texttt{ab(c1)}, \texttt{ab(c2)}\}$$

lautet.

Angenommen, das Ausgangsventil `v` ist defekt. Geben Sie Messwerte in einem File `fault.v.ab.obs` an, sodass die einzige minimale Diagnose (gemeinsam mit den Beobachtungen aus Abbildung 2)

$$\{\texttt{ab(v)}\}$$

lautet.