

Feedback ist in den PDF-Kommentaren
(Chrome Viewer hat oft Probleme
mit Umlauten, daher besser in z.B.
Acrobat Reader öffnen)

50 Punkte

Super Arbeit, nur ein paar
Kleinigkeiten sind mir
aufgefallen :)

AB 1

Link zur Web Version dieser Notion page: <https://www.notion.so/AB-1-816446f762064cb382bc9c13113cb248>

Aufgabe 1

- a)
- b)
- c)

Aufgabe 2

Aufgabe 3

- a)
- b)
- c)
- d)
- e)
- f)
- g)
- h)
- i)

Aufgabe 4

- a)
- b)

Aufgabe 5

- a)
- b)
- c)

Aufgabe 6

- a)
- b)

Aufgabe 7

- a)
- b)

Aufgabe 8

Aufgabe 9

Aufgabe 10

Aufgabe 11

Aufgabe 12

Aufgabe 13

- a)
- b)
- c)

Aufgabe 14

- a)
- b)

Aufgabe 15

Modellierung eines endlichen Automaten

- a)
- b)
- c)

Aufgabe 1 3 Punkte

Verlangt:

1. Zugrundeliegende Inferenzregel angeben → Gültigkeit bestimmen

- "Falls in der Wahrheitsbelegung I alle **Prämissen** wahr sind, dann ist auch die **Konklusion** wahr in I ."
(Kriterium für die Gültigkeit von Inferenzregeln)

(Wahrheitsbelegung = Interpretation)

- **gültig** wenn für alle Wahrheitsb. immer $\text{val}_I(F) = 1$
- **erfüllbar** wenn für mindestens eine Wahrheitsb. $\text{val}_I(F) = 1$
- **widerlegbar** wenn für mindestens eine Wahrheitsb. $\text{val}_I(F) = 0$
- **unerfüllbar** wenn für alle Wahrheitsb. immer $\text{val}_I(F) = 0$

2. Weiteres Beispiel / Gegenbeispiel

a)

```
Es gibt Menschen die kriminell sind (wahr)
Alle Asylanten sind Menschen (wahr)
-----
Alle Asylanten sind kriminell (falsch)
```

Formel der Inferenzregel: ungültig (Generalisierung)

$$(\forall x \in Y) \wedge (z \in Y) \not\Rightarrow (\forall z = x)$$

```
Alle x sind y           Prämisse
z ist ein y           Prämisse
-----
z ist ein x           Konklusion
```

Weiteres Gegenbeispiel:

```
Alle Bälle sind rund (wahr)
Die Sonne ist rund (wahr)
-----
Die Sonne ist ein Ball (falsch)
```



b)

```
Wenn alle Tober xetig sind
und ein Hituch ein Tober ist
-----
Dann ist Hituch xetig
```

Inferenzregel: (gültig)

$$(\forall T \subseteq X) \wedge (h \in T) \Rightarrow (\forall h \in X)$$

```
Alle t sind x
h ist ein t
-----
h ist ein x
```

Weiteres Beispiel aus dem Alltag:

```
Wenn alle Urgroßmütter Großmütter sind  
und eine Großmutter eine Mutter ist  
-----  
Dann ist eine Urgroßmutter eine Mutter
```



c)

```
Alle Katzen miauen  
Bello ist keine Katze  
-----  
Also miaut Bello nicht
```

Inferenzregel: (ungültig)

$$(\forall K \subseteq M) \wedge (b \notin K) \not\Rightarrow (\forall b \notin M)$$

```
Alle k sind m  
b ist kein k  
-----  
b ist kein m
```

Gegen-Beispiel aus dem Alltag:

```
Alle Männer sind Menschen  
Frauen sind keine Männer  
-----  
Frauen sind keine Menschen
```



Aufgabe 2 4 Punkte

Jeden Satz einzeln Modellieren:

Wichtig: Elementaraussagen zeichnen sich dadurch aus, dass sie entweder richtig oder falsch sein können.

Liebe Kundin, lieber Kunde!

Wir gratulieren zum Erwerb dieses Produkts! Legen Sie Hammer und Schraubenzieher oder Akkuschauber bereit. Packen Sie die Plastikteile aus, aber beschädigen Sie dabei die Beschichtung nicht. Nur wenn Sie die Version 2.0 des Modells erworben haben, stecken Sie nun alle Teile zusammen. Falls nicht, müssen Sie alle Teile verleimen. Falls Sie beim Aufbau Probleme haben oder Teile fehlen, wenden Sie sich bitte an unsere Service-Hotline. Gutes Gelingen!

Legen Sie Hammer und Schraubenzieher oder Akkuschauber bereit. Packen Sie die Plastikteile aus, aber beschädigen Sie dabei die Beschichtung nicht.

- A Kunde legt Werkzeuge (= Hammer / Schraubenzieher) bereit
- B Kunde packt Plastikteile aus
- C Kunde beschädigt die Beschichtung

Formel: $A \wedge B \wedge (\neg C)$

Nur wenn Sie die Version 2.0 des Modells erworben haben, stecken Sie nun alle Teile zusammen. Falls nicht, müssen Sie alle Teile verleimen.

A Die vom Kunden erhaltene Version des Modells entspricht 2.0

B Kunde steckt alle Teile zusammen

C Kunde verleimt alle Teile

Struktur: Wenn A dann B , ansonsten C wobei B und C gleichzeitig zutreffen dürfen aber wenn C dann immer auch B aber umgekehrt nicht zwingend.

Formel: $(A \equiv B) \wedge (\neg A \equiv C) \wedge (C \supset B)$

Unter der Annahme, dass:

- A (Modell) bestimmt ob B (zusammenstecken) oder C (leimen) gewählt wird
- C (leimen) führt immer zum B (zusammenstecken) aber umgekehrt ist es optional.

Falls Sie beim Aufbau Probleme haben oder Teile fehlen, wenden Sie sich bitte an unsere Service-Hotline.

A Kunde hat beim Aufbau Probleme

B Es Fehlen Teile vom Produkt

C Kunde nimmt mit Hotline Kontakt auf

Struktur: Falls A oder B dann C aber sonst auch C möglich.

Formel: $(A \vee B) \wedge C$

Unter der Annahme, dass:

- Man sich dem Service auch zuwenden kann, falls nicht die erwähnten Probleme bestehen

Aufgabe 3 4 Punkte

M Maria kommt mit ins Kino

C Clara kommt mit ins Kino

P Peter kommt mit ins Kino

a)

Maria kommt mit ins Kino.

M 


b)

Peter kommt vielleicht mit.

P Das Wort vielleicht lässt sich nicht modellieren. 

c)

Clara kommt mit, Peter aber nicht.

$C \wedge (\neg P)$ 

d)

Wenn Maria mitkommt, dann kommt Peter nicht mit.

$$M \uparrow P \quad \checkmark$$

e)

Nur wenn Maria mitkommt und Peter nicht, dann kommt Clara.

$$(M \wedge (\neg P)) \equiv C$$

Annahme: Ansonsten kommt sie nicht.

$$\text{Würde sie sonst auch kommen, dann: } (M \wedge (\neg P)) \supset C \quad \times$$

f)

Wenn Peter nicht kommt, dann kommt Clara nicht, und wenn Clara nicht kommt, dann kommt Peter nicht.

$$P \equiv C \quad \checkmark$$

g)

Mindestens eine(r) der drei kommt mit.

$$M \vee C \vee P \quad \checkmark$$

h)

Genau zwei kommen mit.

$$(M \wedge C) \vee (M \wedge P) \vee (C \wedge P) \quad \times$$

i)

Höchstens zwei kommen mit.

$$\neg(M \wedge C \wedge P) \quad \checkmark$$

Aufgabe 4

4 Punkte

a)



Notiz: Ich habe diese Aufgabe mit Tutoren (Herr Dr. Martin Riener) bei einer Tutorstunde besprochen und wir sind daraufgekommen, dass es einen Tippfehler bei der Angabe gab als ich dieses Blatt gelöst habe

Zeigen Sie, dass die Menge **{iff, xor}** vollständig ist für die Klasse der aussagenlogischen Funktionen.

Eine Menge von Funktionen heißt vollständig (für eine Funktionsklasse), wenn damit alle Funktionen (der Klasse) ausgedrückt werden können.

Beispiele

{not, and}, {nand}, {nor}, {not, or}, {not, implies}, {implies, false}

Man kann keine anderen Funktionen erzeugen, diese Menge ist nicht funktional vollständig.

Beweis: iff ist dual zu xor und man kann mit der Menge {xor} alleine beispielsweise nicht die Funktionen and sowie or bilden.

$$\neg(\neg a \text{ xor } \neg b) \equiv \text{iff}(a, b) \quad \checkmark$$

b)

Zeigen Sie, dass die Menge {iff, or} nicht vollständig ist.

Es reicht anzugeben dass eine einzige Funktion nicht darstellbar ist. - Ich nehme mir vor zu beweisen, dass die **not** Funktion nicht darstellbar ist, weil sie nur ein Argument beansprucht und ich intuitiv weiß, dass sie nicht mit **iff** oder **or** darstellbar ist.

Wir nutzen nur das Argument x (weil die **not** Funktion nur mit einem einzigen Argument arbeitet).

Hypothese

Jeder Ausdruck bestehend aus (n oder weniger) **iff**, **or** und **x** ist äquivalent zu **x** oder **true**.

Induktiver Beweis (wobei n = # der Anwendungen eines Operators)

Induktionsanfang: n := 0

Keine Operatoren - bedeutet x selbst.

Induktionsanfang: n := 1

Wenn wir eine beliebige Kombination der uns verfügbaren Funktionen nehmen:

- $or(f(x), g(x))$
- $iff(f(x), g(x))$

Dann dürfen nur Kombinationen aus den Argumenten der Menge {iff, or} verwendet werden.

1. $or(x, x) = x$
2. $iff(x, x) = true \rightarrow$ dadurch haben wir eine weitere Funktion in unserer Menge

Induktionsanfang: n := 2

Wenn wir eine beliebige Kombination der uns verfügbaren Funktionen nehmen:


- $or(f(x), g(x))$
- $iff(f(x), g(x))$

Dann dürfen nur Kombinationen aus den Argumenten der Menge {iff, or} verwendet werden.

1. $or(true, x) = or(x, true) = or(true, true) = iff(true, true) = true$
2. $iff(true, x) = iff(x, true) = x$

Induktionsschritt: n := n+1

Für jeden weiteren rekursiven Einsatz kann nur (effektiv) mit den bereits ausgewerteten Ausdrücken von den vorherigen Schritten gearbeitet werden.

Dadurch können wir aber nicht die Funktion $not(x)$ ausdrücken - wodurch die Bedingung für funktionale Vollständigkeit verletzt ist. 

Alternativerweise:

Dadurch, dass **iff** die duale Funktion zu **xor** ist: $\neg(\neg a \text{ xor } \neg b) \equiv iff(a, b)$

lässt sich dieses Problem reduzieren auf das Problem "Zeigen Sie, dass die Menge {or, xor} nicht vollständig ist."

Siehe: Lösungen vom vergangenen Jahr.



Aufgabe 5

3 Punkte

Sei die Formel

$$F := (((A \vee \neg B) \supset C) \equiv ((B \supset C) \vee \neg A))$$

a)

Syntaktische Korrektheit

Die Menge \mathcal{A} der aussagenlogischen Formeln F_i ist die kleinste für die gilt:

1. $\mathcal{V} \subseteq \mathcal{A}$ (eine Variable ist eine gültige Formel)
2. $\{\top, \perp\} \subseteq \mathcal{A}$
3. $\neg F \in \mathcal{A}$ wenn $F \in \mathcal{A}$
4. $(F * G) \in \mathcal{A}$ wenn $F, G \in \mathcal{A}$ und der Operator $\{\wedge, \vee, \uparrow, \downarrow, \equiv, \neq, \subset, \supset\}$

Ich kürze jede gültige Variable mit V ab und beweise die Gültigkeit von innen nach außen mit Hinweisen auf die jeweiligen Bedingungen die dabei erfüllt werden

$$\begin{aligned} & \underbrace{((A \vee \neg B) \supset C)}_{(3)} \equiv \underbrace{((B \supset C) \vee \neg A)}_{(3)} \\ & \underbrace{((A \vee V) \supset C)}_{(4)} \equiv \underbrace{((B \supset C) \vee V)}_{(4)} \\ & \underbrace{(V \supset C)}_{(4)} \equiv \underbrace{(V \vee V)}_{(4)} \\ & \underbrace{\hspace{10em}}_{(4)} \end{aligned}$$

b)

Schrittweise Berechnung des Wertes $\text{val}_I(F)$

$$I(A) = 1, I(B) = 0 \text{ und } I(C) = 0$$

Ich berechne erneut nach dem oberen Muster von innen nach außen:

$$\text{val}_I(\neg B) = \text{not}(\text{val}_I(B)) = \text{not}(\perp) = 1$$

$$\text{val}_I(\neg A) = \text{not}(\text{val}_I(A)) = \text{not}(\top) = 0$$

$$\text{val}_I(A \vee \neg B) = \text{and}(\text{val}_I(A), 1) = \text{and}(\top, 1) = 1$$

$$\text{val}_I(B \supset C) = \text{val}_I(B) \text{ implies } \text{val}_I(C) = 0 \text{ implies } 0 = 1$$

$$\text{val}_I(F) = \text{val}_I(((A \vee \neg B) \supset C) \equiv ((B \supset C) \vee \neg A))$$

$$= (1 \text{ implies } 0) \text{ iff } (1 \text{ or } 0) = 1 \quad \checkmark$$

c)

Wahrheitstafel der Formel

[https://tuwien2020.github.io/tgi-pages/#/truth-table?input=\(\(a+or+!b\)+=>+c\)+%3C=%3E+\(\(b+%3E+c\)+or+!a\)](https://tuwien2020.github.io/tgi-pages/#/truth-table?input=((a+or+!b)+=>+c)+%3C=%3E+((b+%3E+c)+or+!a))

(Wahrheitsbelegung = Interpretation)

- **gültig** wenn für alle Wahrheitsb. immer $\text{val}_I(F) = 1$
- **erfüllbar** wenn für mindestens eine Wahrheitsb. $\text{val}_I(F) = 1$
- **widerlegbar** wenn für mindestens eine Wahrheitsb. $\text{val}_I(F) = 0$

- **unerfüllbar** wenn für alle Wahrheitsb. immer $\text{val}_I(F) = 0$

Diese Formel ist erfüllbar und widerlegbar, dadurch aber ungültig und unerfüllbar.

a	b	c	$a \vee \neg b$	$(a \vee \neg b) \implies c$	$b \implies c$	$(b \implies c) \vee \neg a$	$((a \vee \neg b) \implies c) \Leftrightarrow ((b \implies c) \vee \neg a)$
0	0	0	1	0	1	1	0
0	0	1	1	1	1	1	1
0	1	0	0	1	0	1	1
0	1	1	0	1	1	1	1
1	0	0	1	0	1	1	0
1	0	1	1	1	1	1	1
1	1	0	1	0	0	0	1
1	1	1	1	1	1	1	1



✓ Aufgabe 6 3 Punkte

Zeigen Sie, dass die beiden Formeln äquivalent sind.

$$F_1 := \neg((P \wedge Q) \supset (\neg P \equiv Q))$$

$$F_2 := ((Q \equiv R) \supset P) \wedge \neg(P \uparrow Q)$$

a)

Wahrheitstafel

[https://tuwien2020.github.io/tgi-pages/#/truth-table?input=!\(\(P+and+Q\)+=>+\(not+P+<=>+Q\)\)\)+%3C=%3E+\(\(\(Q+%3C=%3E+R\)+=%3E+P\)+and+not\(P+nand+Q\)\)](https://tuwien2020.github.io/tgi-pages/#/truth-table?input=!((P+and+Q)+=>+(not+P+<=>+Q)))+%3C=%3E+(((Q+%3C=%3E+R)+=%3E+P)+and+not(P+nand+Q)))

p	q	r	$\neg((p \wedge q) \implies (\neg p \Leftrightarrow q)) \Leftrightarrow (((q \Leftrightarrow r) \implies p) \wedge \neg(p \text{ nand } q))$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

b)

Algebraische Umformungen

Wir wollen algebraisch beweisen, dass dieser Ausdruck immer *true* bzw \top ergibt.

$$\neg((P \wedge Q) \supset (\neg P \equiv Q)) \equiv (((Q \equiv R) \supset P) \wedge \neg(P \uparrow Q))$$

$$\neg((P \wedge Q) \supset (\neg P \equiv Q)) \equiv (((Q \equiv R) \supset P) \wedge P \wedge Q)$$

Kontrolle: [%3C=%3E+\(\(\(Q+%3C=%3E+R\)+=%3E+P\)+and+P+and+Q\)](https://tuwien2020.github.io/tgi-pages/#/truth-table?input=((P+and+Q)+>+(not+P+<=>+Q)))

$$\neg(\neg((P \wedge Q) \supset (\neg P \equiv Q))) \vee (((Q \equiv R) \supset P) \wedge P \wedge Q) \text{ weil } a \Rightarrow b \Leftrightarrow \neg a \vee b$$

$$((P \wedge Q) \supset (\neg P \equiv Q)) \vee (((Q \equiv R) \supset P) \wedge P \wedge Q)$$

Kontrolle: [\(Q+%3C=%3E+R\)+=%3E+P\)+and+P+and+Q\)](https://tuwien2020.github.io/tgi-pages/#/truth-table?input=((P+and+Q)+>+(not+P+<=>+Q))+or+(((Q+%3C=%3E+R)+=%3E+P)+and+P+and+Q))

Jetzt müsste nur eines der beiden Teile der \vee Funktion *true* sein damit die gesamte Gleichung immer *true* liefert.

Teil 1:

$$((P \wedge Q) \supset (\neg P \equiv Q))$$

$$(\neg(P \wedge Q) \vee (\neg P \equiv Q)) \text{ weil } a \Rightarrow b \Leftrightarrow \neg a \vee b$$

$$(\neg(P \wedge Q) \vee ((\neg P \wedge Q) \vee (P \wedge \neg Q))) \text{ weil } (a \Leftrightarrow b) \Leftrightarrow ((a \wedge b) \vee (\neg a \wedge \neg b))$$

$$\neg(P \wedge Q) \vee (\neg P \wedge Q) \vee (P \wedge \neg Q)$$

In allen cases *true* außer $P = 1, Q = 1$

Das bedeutet Teil 2 muss zumindest für diese Interpretation gültig sein damit der gesamte Ausdruck gültig ist.

Teil 2:

$$(((Q \equiv R) \supset P) \wedge P \wedge Q)$$

Wir setzen die Wahrheitsbelegung für P und Q ein.

$$(((1 \equiv R) \supset 1) \wedge 1 \wedge 1)$$

$$(1 \equiv R) \supset 1$$

Case 1: $R = 1$

$$(1 \equiv 1) \supset 1$$

$$1 \supset 1$$

$$1$$

Case 2: $R = 0$

$$(1 \equiv 0) \supset 0$$

$$0 \supset 0$$

$$1 \quad \checkmark$$

Aufgabe 7 3 Punkte

Wenn ich schuldig bin, muss ich bestraft werden. [...]

S Sokrates ist schuldig.

B Sokrates muss bestraft werden.

Wenn Äquivalenz gemeint ist ($S \equiv B$)

Wenn Implikation gemeint ist ($S \supset B$)

S	B	$S \equiv B$	$S \supset B$
0	0	1	1
0	1	0	1
1	0	0	0
1	1	1	1

a)

| ... Ich bin schuldig.

Wenn $S = 1$

- ~~dann ist $S \equiv B$ nur wahr wenn $B = 1$ also muss Sokrates bestraft werden~~
- ~~dann kann $S \supset B$ auch wahr sein wenn $B \neq 1$ also ist es unklar ob Sokrates bestraft werden muss.~~

b)

| ... Ich muss bestraft werden

Wenn $B = 1$

- ~~dann ist $S \equiv B$ nur wahr wenn $S = 1$ also ist er schuldig.~~
- dann kann $S \supset B$ auch wahr sein wenn $S \neq 1$ also ist es unklar ob er schuldig ist oder nicht.



Konsequenzbeziehung (logische Konsequenz)

Die Semantik \models (in der Aussagenlogik) ist durch \supset ausdrückbar.

$F_1, \dots, F_n \models G$

- Aus $\text{val}_I(F_1) = \dots = \text{val}_I(F_n) = 1$ folgt $\text{val}_I(G) = 1$
- "Falls in der Wahrheitsbelegung I alle Prämissen wahr sind, dann ist auch die Konklusion wahr in I ."
(Kriterium für die Gültigkeit von Inferenzregeln)
- "Die Formel G ist eine logische Konsequenz der Formeln F_1, \dots, F_n "

$F_1, \dots, F_n \models G$ genau dann wenn $F_1 \supset (F_2 \supset \dots (F_n \supset G) \dots)$

und weil $A \supset (B \supset C) = (A \wedge B) \supset C$:

$F_1, \dots, F_n \models G$ genau dann wenn $(F_1 \wedge \dots \wedge F_n) \supset G$.

In diesem Fall würde die Gültigkeit / Ungültigkeit der Formel ($S \equiv B$) bzw ($S \supset B$) wenn man alle möglichen Fälle kennt und Wissen über die Realität von Sokrates hat ermitteln können ob Sokrates' Schlussfolgerungen per se Gültigkeit haben oder nicht.

Aufgabe 8 2 Punkte

Definition von DNF und KNF der Funktion $f : \mathbb{B}^n \mapsto \mathbb{B}$.

Konj. / Disj. einer Menge von Variablen:

$$\bigwedge \{F, G, H, \dots\} = F \wedge G \wedge H \wedge \dots \quad \bigwedge \{\} = \top$$

$$\bigvee \{F, G, H, \dots\} = F \vee G \vee H \vee \dots \quad \bigvee \{\} = \perp$$

Wir haben eine Funktion f mit

- Variablen A_i die durch Interpretation belegt werden $I_{\vec{b}}(A_i) = b_i$
- Abbildung $\vec{b} = (b_1, \dots, b_n) \in \mathbb{B}^n$

Definition DNF

Charakteristisches Konjunkt für $\vec{b} = (b_1, \dots, b_n) \in \mathbb{B}^n$:

$$K_{\vec{b}} = \bigwedge \{A_i \mid b_i = 1, i = 1..n\} \wedge \bigwedge \{\neg A_i \mid b_i = 0, i = 1..n\}$$

Wobei A_i die Variablen sind die als Argumente der Funktion dienen.

Beispiel:

$$\vec{b} = (1, 0, 1, 1) \implies K_{\vec{b}} = A_1 \wedge \neg A_2 \wedge A_3 \wedge A_4$$

Für Wahrheitsbelegung $I_{\vec{b}}$ hat $K_{\vec{b}}$ den Wert 1 und sonst 0.

$$I_{\vec{b}} : A_1 \mapsto 1, A_2 \mapsto 0, A_3 \mapsto 1, A_4 \mapsto 1 \implies \text{val}_{I_{\vec{b}}}(K_{\vec{b}}) = 1$$

Die Konjunkte werden disjunktiv verbunden.

$$\text{DNF}_f = \bigvee \{K_{\vec{b}} \mid f(\vec{b}) = 1, \vec{b} \in \mathbb{B}^n\}$$

Dadurch repräsentiert die DNF die Funktion f .

$$\forall \vec{b} \in \mathbb{B}^n : \text{val}_{\vec{b}}(\text{DNF}_f) = f(\vec{b})$$

Definition KNF

Charakteristisches Disjunkt für $\vec{b} = (b_1, \dots, b_n) \in \mathbb{B}^n$:

$$D_{\vec{b}} = \bigvee \{A_i \mid b_i = 0, i = 1..n\} \vee \bigvee \{\neg A_i \mid b_i = 1, i = 1..n\}$$

Wobei A_i die Variablen sind die als Argumente der Funktion dienen.

Beispiel:

$$\vec{b} = (1, 0, 1, 1) \implies D_{\vec{b}} = \neg A_1 \vee A_2 \vee \neg A_3 \vee \neg A_4$$

Für Wahrheitsbelegung $I_{\vec{b}}$ hat $D_{\vec{b}}$ den Wert 0 und sonst 1.

$$I_{\vec{b}} : A_1 \mapsto 1, A_2 \mapsto 0, A_3 \mapsto 1, A_4 \mapsto 1 \implies \text{val}_{I_{\vec{b}}}(D_{\vec{b}}) = 0$$

Die Konjunkte werden disjunktiv verbunden.

$$\text{KNF}_f = \bigwedge \{D_{\vec{b}} \mid f(\vec{b}) = 0, \vec{b} \in \mathbb{B}^n\}$$

Dadurch repräsentiert die DNF die Funktion f .

$$\forall \vec{b} \in \mathbb{B}^n : \text{val}_{\vec{b}}(\text{KNF}_f) = f(\vec{b})$$

x	y	z	$f(x, y, z)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Disjunktive Normalform

$$K_{001} = \neg x \wedge \neg y \wedge z$$

$$K_{101} = x \wedge \neg y \wedge z$$

$$K_{110} = x \wedge y \wedge \neg z$$

$$K_{111} = x \wedge y \wedge z$$

$$\text{DNF}_f = \bigvee \{K_{\vec{b}} \mid f(\vec{b}) = 1, \vec{b} \in \mathbb{B}^n\}$$

Konjunktive Normalform

$$D_{000} = x \vee y \vee z$$

$$D_{010} = x \vee \neg y \vee z$$

$$D_{011} = x \vee \neg y \vee \neg z$$

$$D_{100} = \neg x \vee y \vee z$$

$$\text{KNF}_f = \bigwedge \{D_{\vec{b}} \mid f(\vec{b}) = 0, \vec{b} \in \mathbb{B}^n\}$$

Sehr genau aufgeschrieben, super!

✓ Aufgabe 9 2 Punkte

$$X := (((G \vee \neg F) \supset H) \vee (\neg F \supset G))$$

[https://tuwien2020.github.io/tgi-pages/#/truth-table?input=\(\(\(G+or+!F\)+=>+H\)+or+\(!F+=%3E+G\)\)](https://tuwien2020.github.io/tgi-pages/#/truth-table?input=(((G+or+!F)+=>+H)+or+(!F+=%3E+G)))

F	G	H	$G \vee \neg F$	$(G \vee \neg F) \implies H$	$\neg F \implies G$	$((G \vee \neg F) \implies H) \vee (\neg F \implies G)$
0	0	0	1	0	0	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	1	1	1	1
1	0	0	0	1	1	1
1	0	1	0	1	1	1
1	1	0	1	0	1	1
1	1	1	1	1	1	1

Semantisch zur DNF gelangen:

[https://www.wolframalpha.com/input/?i=DNF+\(\(\(G+||+~F\)+implies+H\)+||+~F+implies+G\)\)](https://www.wolframalpha.com/input/?i=DNF+(((G+||+~F)+implies+H)+||+~F+implies+G)))

Minimale Form:

$$F \vee G \vee H = \text{DNF}_X$$

Algebraisch zur KNF gelangen:

$$(((G \vee \neg F) \supset H) \vee (\neg F \supset G))$$

$$((G \vee \neg F) \supset H) \vee F \vee G \text{ weil } a \supset b \Leftrightarrow \neg a \vee b$$

$$(\neg G \wedge F) \vee H \vee F \vee G \text{ weil } a \supset b \Leftrightarrow \neg a \vee b$$

$F \vee G \vee H$ weil ~~$(\neg C \wedge F) = F$~~

$\text{KNF}_X = D_{000} = F \vee G \vee H$ ✓

✓ Aufgabe 10 4 Punkte

"Mein lieber Watson, meine intensiven Nachforschungen gestatten mir, folgende Schlüsse zu ziehen:

- Wenn sich Brown oder Cooper als Täter herausstellen sollte, dann ist Adams unschuldig.
- Ist aber Adams oder Cooper unschuldig, dann muss Brown ein Täter sein.
- Ist Cooper schuldig, dann wäre Adams Mittäter."

A Adam ist schuldig / der Täter

B Brown ist schuldig / der Täter

C Cooper ist schuldig / der Täter

Deklarierte Constraints die alle gültig sein müssen:

- $F_1 := (B \vee C) \supset \neg A$ ✓
- $F_2 := \neg(A \vee C) \supset B = (\neg A \wedge \neg C) \supset B$ ✗
- $F_3 := C \equiv A$ ✗

Wir suchen eine Interpretation für die $F_1 \wedge F_2 \wedge F_3$ gültig ist.

Wir vereinfachen F_1 und F_2 dadurch dass A äquivalent zu C ist.

- $G_1 := (B \vee A) \supset \neg A = B \wedge \neg A$
- $G_2 := \neg(A \vee A) \supset B = \neg A \supset B = A \vee B$ weil $a \Rightarrow b \Leftrightarrow \neg a \vee b$

Wir suchen also dadurch eine Interpretation für die $B \wedge \neg A \wedge (A \vee B)$ immer gültig ist.

Dadurch, dass A und $\neg A$ nicht gleichzeitig *true* sein können, folgt, daraus:

es kann nur $B \wedge \neg A$ gültig sein.

Wir erinnern uns an $F_3 := C \equiv A$.

Also lautet die Lösung $\neg A \wedge B \wedge \neg C$

Das bedeutet auf die "Wirklichkeit" übertragen:

Es kann nur B Brown schuldig sein - A Adam und C Cooper sind unschuldig.

Überprüfung mit Wahrheitstabelle:

[https://tuwien2020.github.io/tgi-pages/#/truth-table?input=\(\(B+or+C\)+implies+!A\)+and+\(!\(A+or+C\)+implies+B\)+and+\(C+%3C=%3E+A\)](https://tuwien2020.github.io/tgi-pages/#/truth-table?input=((B+or+C)+implies+!A)+and+(!(A+or+C)+implies+B)+and+(C+%3C=%3E+A))

A	B	C	
0	0	0	
0	0	1	
0	1	0	
0	1	1	1
1	0	0	
1	0	1	
1	1	0	
1	1	1	

$$(((B \vee C) \implies \neg A) \wedge (\neg(A \vee C) \implies B)) \wedge (C \Leftrightarrow A)$$

	0
	0
	1
	0
	0
	0
	0
	0
	0

✓ Aufgabe 11 4 Punkte

Professor John Frink organisiert eine internationale Konferenz und sucht zur Unterstützung Student Volunteers. Nach zahlreichen Vorstellungsgesprächen schränkt er die Auswahl auf Bart, Janey, Lisa, Milhouse und Richard ein,

wobei allerdings nur Lisa und Richard auch Fremdsprachen beherrschen.

Er stellt folgende Überlegungen an:

- Janey möchte ich auf jeden Fall, sie hat bei der letzten Konferenz schon erfolgreich mitgearbeitet.
- Ich kann höchstens drei Volunteers anstellen.
- Ich brauche jedenfalls mindestens einen Volunteer, der Fremdsprachen spricht.
- Richard und Milhouse kennen die Räume, in denen die Konferenz stattfinden soll. Einen der beiden sollte ich auf jeden Fall nehmen, aber beide zu nehmen ist nicht notwendig.
- Richard will nur mitmachen, wenn ich auch Bart anstelle.
- Milhouse und Lisa wollen nur gemeinsam genommen werden, da sie andernfalls zusammen auf Urlaub fahren wollen.

Variablen

- B* Bart wird vom Professor angestellt.
- J* Janey wird vom Professor angestellt.
- L* Lisa wird vom Professor angestellt.
- M* Milhouse wird vom Professor angestellt.
- R* Richard wird vom Professor angestellt.

Formeln

Die folgenden constraints müssen in unserer deklarativen Modellierung eingehalten werden - also alle diese Formeln müssen bei der Interpretation nach der wir suchen *true* sein.

- $F_1 := J$ Janey muss mit ✓
- $F_2 := \neg(\wedge \{B, J, L, M, R\}) \vee \neg(\wedge \{J, L, M, R\}) \vee \neg(\wedge \{L, M, R, B\}) \vee \neg(\wedge \{M, R, B, J\}) \vee \neg(\wedge \{R, B, J, L\}) \vee \neg(\wedge \{B, J, L, M\})$ Höchstens 3 ✓
- $F_3 := L \vee R$ Min. 1 Fremdsprachler ✓
- $F_4 := R \uparrow M$ Entweder Richard oder Milhouse ✗
- $F_4 := R \equiv B$ Richard nur mit Bart ✗
- $F_5 := M \equiv L$ Milhouse nur mit Lisa ✓

Algebraische Lösung

Zuerst lösen nutzen wir die Äquivalenzen um alle Formeln zu vereinfachen und setzen J überall auf $true$.

$$J = 1$$

RB steht für Richard und Bart zugleich

ML steht für Milhouse und Lisa zugleich

- $F_2 := \neg(\wedge \{ML, RB\}) \vee \neg(\wedge \{ML, RB\}) \vee \neg(\wedge \{ML, RB\}) \vee \neg(\wedge \{ML, RB\}) \vee \neg(\wedge \{RB, ML\}) \vee \neg(\wedge \{RB, ML\})$ Höchstens 3
- $F_3 := ML \vee RB$ Min. 1 Fremdsprachler
- $F_4 := RB \uparrow ML$ Entweder Richard oder Milhouse

wir vereinfachen weiter

- $F_2 := \neg(ML \wedge RB) = \neg ML \vee \neg RB$
- $F_3 := ML \vee RB$ Min. 1 Fremdsprachler
- $F_4 := RB \uparrow ML$ Entweder Richard oder Milhouse

Alle deklarierten constraints gemeinsam:

$$(\neg ML \vee \neg RB) \wedge (ML \vee RB) \wedge (RB \uparrow ML)$$

Das lässt sich weiter vereinfachen da: $(\neg ML \vee \neg RB) \wedge (ML \vee RB) \equiv (RB \uparrow ML)$

Es gibt 2 mögliche Interpretationen die alle constraints erfüllen.

Lösung übertragen auf die "Wirklichkeit"

Daraus folgt, dass der Professor unbedingt Lisa mitnimmt und zwischen Team ML und Team RB entscheiden muss.

Er könnte dafür beispielsweise eine faire Münze werfen. ✗

✓ Aufgabe 12 4 Punkte

Eine Kollegin und ich modellieren das gleiche Problem.

- Sie hat die Formel H
- Ich habe die Formeln F, G

- Unsere Formeln sehen unterschiedlich aus.

Überprüfung der semantischen Äquivalenz mit SAT-Solver

1. Automatisiert die Formel H und die Formel $X := F \wedge G$ in die DNF oder KNF umwandeln
2. $(DNF_H \equiv DNF_X)$ in den SAT-Solver eingeben
3. Falls der SAT-Solver in der Lage ist eine Interpretation zu finden mit der die Gleichung aus dem zweiten Schritt nicht erfüllbar ist (also die Gleichung zu widerlegen) können wir sicher sein, dass die Gleichungen nicht äquivalent sind.

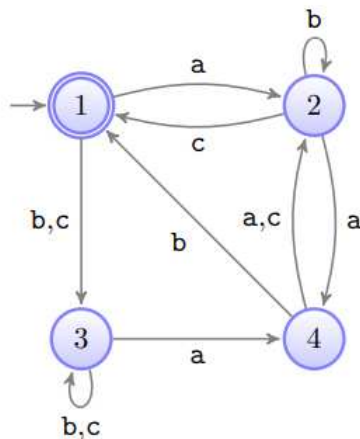
Die einzige Möglichkeit aber die Äquivalenz zu *beweisen* wäre es alle möglichen Interpretationen durchzugehen.

Was bedeutet es, wenn der SAT-Solver eine erfüllende Variablenbelegung findet?

Eine erfüllende Variablenbelegung für DNF_H oder DNF_X wäre eine Lösung unseres Problems \rightarrow Eine Lösung die sich an alle deklarierten constraints hält.

✓ Aufgabe 13 3 Punkte

Gegeben sei:



Dieser Automat ist ein DEA: von jedem Zustand gibt es einen eindeutigen Nachfolger mit jeder Eingabe aus dem Alphabet

$$\Sigma = \{a, b, c\}$$

$$Q = \{1, 2, 3, 4\}$$

$$q_0 = 1$$

$$F = \{4\}$$

a)

Alle Wörter aus *maximal 3 Zeichen* die akzeptiert werden (mit denen der Automat in einem *final state*) endet.

$$\mathcal{L}_{\leq 3}(\mathcal{A}) = \{w \in \Sigma^* \mid \delta^*(q_0, w) \in F, |w| \leq 3\}$$

1. ϵ

2. ac
3. abc
4. aab
5. bab
6. cab ✓

b)

Erweiterte Übergangsfunktion (auch für Sätze)

$$\delta^* : Q \times \Sigma^* \mapsto Q$$

$$\forall q \in Q, s \in \Sigma, w \in \Sigma^* : \quad \delta^*(q, \varepsilon) = q, \quad \delta^*(q, sw) = \delta^*(\delta(q, s), w)$$

$$\delta^*(1, bbacbc)$$

$$\delta^*(\delta^*(1, b), bacbc)$$

$$\delta^*(\delta^*(\delta^*(1, b), b), acbc)$$

...

$$\delta^*(\delta^*(\delta^*(\delta^*(\delta^*(1, b), b), a), c), b), c)$$

$$\delta^*(\delta^*(\delta^*(\delta^*(3, b), a), c), b), c)$$

$$\delta^*(\delta^*(\delta^*(3, a), c), b), c)$$

$$\delta^*(\delta^*(4, c), b), c)$$

$$\delta^*(\delta^*(2, b), c)$$

$$\delta^*(2, c) = 1 \in F$$

Dieses Wort ist Teil der akzeptierten Sprache

c)

Dieser Automat ist ein DEA: von jedem Zustand gibt es einen eindeutigen Nachfolger mit jeder Eingabe aus dem Alphabet

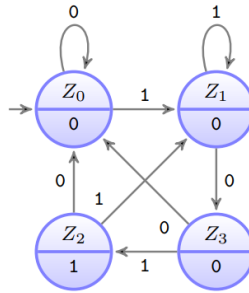
$$\delta : Q \times \Sigma \mapsto Q \quad \text{transition function}$$

Totale Funktion: Funktion, die für *jeden* Wert aus Definitionsmenge ein Abbild hat. Deshalb deterministisch.

δ	a	b	c
1	2	3	3
2	4	2	1
3	4	3	3
4	2	1	2



✓ **Aufgabe 14** 3 Punkte



a)

$$[\mathcal{A}](11001) = 000000$$

$$[\mathcal{A}](01011) = 000010$$

$$[\mathcal{A}](10101) = 000100$$



b)

$$\delta^*(Z_0, 01010)$$

$$\delta^*(\delta^*(Z_0, 0), 1010)$$

...

$$\delta^*(\delta^*(\delta^*(\delta^*(\delta^*(Z_0, 0), 1), 0), 1), 0)$$

$$\delta^*(\delta^*(\delta^*(\delta^*(Z_0, 1), 0), 1), 0)$$

$$\delta^*(\delta^*(\delta^*(Z_1, 0), 1), 0)$$

$$\delta^*(\delta^*(Z_3, 1), 0)$$

$$\delta^*(Z_2, 0) = Z_0$$

Erweiterte Ausgabefunktion (auch für Sätze)

$$\gamma^* : Q \times \Sigma^* \mapsto \Gamma^*$$

$$\forall q \in Q, s \in \Sigma, w \in \Sigma^* :$$

$$\gamma^*(q, \varepsilon) = \varepsilon$$

$$\gamma^*(q, sw) = \gamma(q) \cdot \gamma^*(\delta(q, s), w)$$

$$\gamma^*(Z_0, 01010)$$

$$\gamma^*(Z_0, 01010) = \gamma(Z_0) \cdot \gamma^*(\delta(Z_0, 0), 1010) = 0 \cdot \gamma^*(Z_0, 1010)$$

$$0 \cdot \gamma^*(Z_0, 1010) = 0 \cdot \gamma(Z_0) \cdot \gamma^*(\delta(Z_0, 1), 010) = 00 \cdot \gamma^*(Z_1, 010)$$

$$00 \cdot \gamma^*(Z_1, 010) = 00 \cdot \gamma(Z_1) \cdot \gamma^*(\delta(Z_1, 0), 10) = 000 \cdot \gamma^*(Z_3, 10)$$

$$000 \cdot \gamma^*(Z_3, 10) = 000 \cdot \gamma(Z_3) \cdot \gamma^*(\delta(Z_3, 1), 0) = 0000 \cdot \gamma^*(Z_2, 0)$$

$$0000 \cdot \gamma^*(Z_2, 0) = 0000 \cdot \gamma(Z_2) \cdot \gamma^*(\delta(Z_2, 0)) = 00001 \cdot \gamma^*(Z_0) = 000010$$



c)

Übersetzungsfunktion (gleich wie Mealy)

$$[\mathcal{A}] : \Sigma^* \mapsto \Gamma^*$$

$$[\mathcal{A}](w) = \gamma^*(q_0, w)$$

$$\begin{aligned} \gamma^* : Q \times \Sigma^* &\mapsto \Gamma^* \\ \forall q \in Q, s \in \Sigma, w \in \Sigma^* : \\ \gamma^*(q, \varepsilon) &= \varepsilon \\ \gamma^*(q, sw) &= \gamma(q) \cdot \gamma^*(\delta(q, s), w) \end{aligned}$$

✓ Aufgabe 15 4 Punkte

Elektronisches Tresorschloss

Display:

darstellbar := $\{0, 1, 2\}$

2 Stellen

initial: 00, linke stelle aktiv

Tasten:

Es gibt immer eine *aktive* Stelle

+ inkrementiert aktive Ziffer (ab 2 \rightarrow 0)

L aktive Stelle ist links (looped nicht im kreis)

R aktive Stelle ist rechts (looped nicht im kreis)

OK

Wenn 21 eingegeben und *OK* gedrückt dann öffnet sich Tresor. (alle Tasten außer *RESET* dann ignoriert)

Ansonsten nachdem *OK* gedrückt wurde in einem Fehlerzustand. (alle Tasten außer *RESET* dann ignoriert)

RESET

Kann immer gedrückt werden \rightarrow setzt zurück in den Anfangszustand

Modellierung eines endlichen Automaten

a)

Welche Informationen sind notwendig um den Zustand des Schlosses zu beschreiben?

Notwendige Informationen / Zustände:

- Zahlen am Display (also ein Wert für jede Stelle)
- Aktive Stelle
- Ob gerade der Tresor geöffnet wurde oder in einem Fehlerzustand ist ✓

Wie viele Zustände kann das Schloss annehmen?

- *OPEN*
- *ERROR*
- $2 \cdot 3^2 = 18$ Mögliche Zahlenkombinationen bei denen je eine Stelle aktiv ist ✓

Wie viele Zustände kann das Schloss im Allgemeinen mit n Ziffern (statt 3) und k Stellen (statt 2) annehmen?

- $|Stellen| \cdot (|Ziffer|^{Stellen})$ ✓

Anzahl der möglichen aktiven Stellen · Anzahl der Ziffer hoch Stellen.

b)

Welche Aktionen führen zu einem Zustandswechsel?

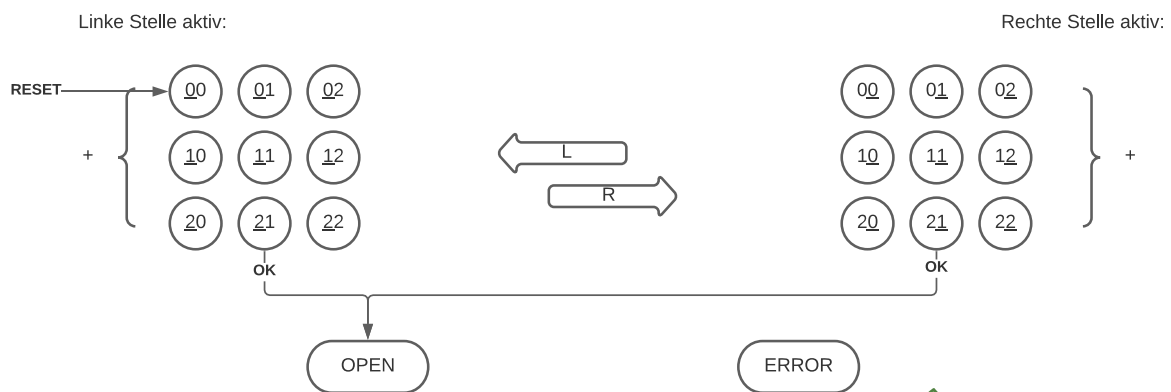
Jeder Tastendruck - siehe oben. ✓

c)

Geben Sie einen endlichen Automaten an, der das Verhalten des beschr. Schlosses vollständig beschreibt. (Auswahl zwischen Tabelle und Graph möglich)

($\mathcal{L}(\mathcal{A})$) sind alle Kombinationen die den Tresor öffnen)

Grobe Skizze:



Die Idee besteht daraus unseren Graphen in Gebiete zu unterteilen. ✓

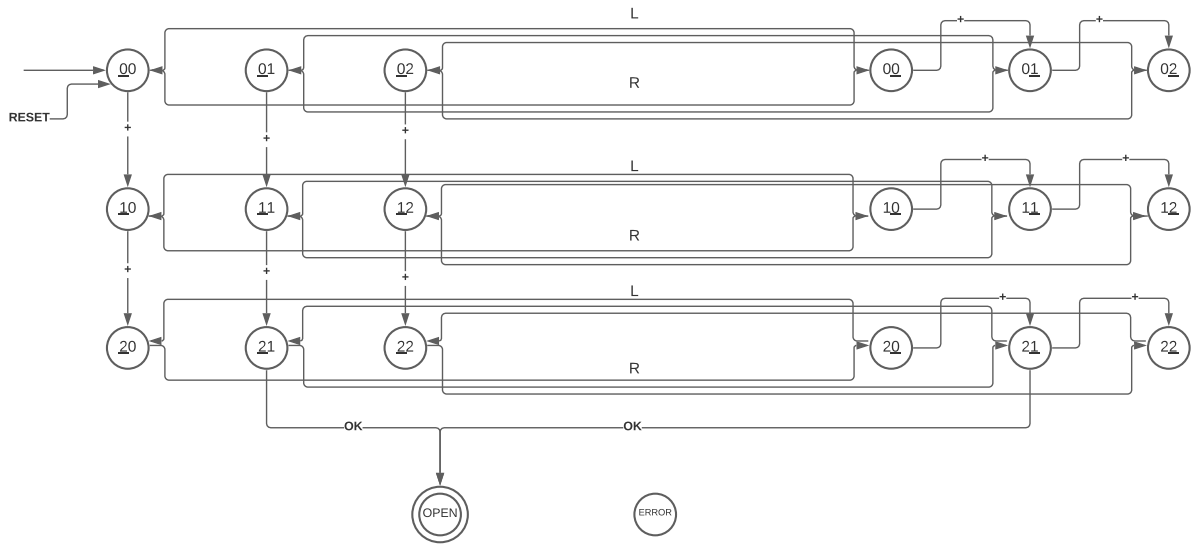
Links sind alle Zahlenkombinationen, wobei immer nur die linke Zahl aktiv ist und man weiß dass jede + Aktion nur einen Einfluss auf dieses Gebiet haben wird und rechts vice versa.

Weiters weiß man dass ein Wechsel zwischen den beiden Gebieten links und rechts nur mit *L* bzw *R* möglich ist.

Beim drücken von *OK* führt nur ein Zustand zu *OPEN* und alle anderen zum Zustand *ERROR*.

ERROR ist meine Falle, deshalb werde ich nicht alle Kanten dazu einzeichnen (zur besseren Lesbarkeit).

Weiters führt das Drücken von *RESET* immer wieder zum Zustand 00 weshalb man diese theoretisch auch nicht einzeichnen bräuchte. ✓



Vollständige Version:

