Typische Fehler bei der Bearbeitung der Klausur *Logik und Grundlagen der Informatik* 2024

Stephan Schulz Falko Kötter Jan Hladik

3. April 2024

1 Allgemeines

- Lesen Sie die Aufgabenstellung.
- Erfüllen Sie die gestellte Aufgabe. Nicht mehr, nicht weniger.

2 KNF-Bildung

- Die Reihenfolge *Äquivalenzsymbole auflösen*, *Implikationen auflösen*, *Negationen nach Innen schieben* (NNF), dann *Ausmultiplizieren* ist zwar nicht zwingend, aber sehr sinnvoll - damit erspart man sich das Rechnen im Kreis, und man hat mit der NNF ein bewertbares Zwischenergebnis.
- Am Ausmultiplizieren führt in der Regel kein Trick vorbei. Speziell kann man nicht einfach ein weiteres Negationszeichen vor die Formel schreiben, und nach innen propagieren, um die Operatoren "umzukippen" - das ist schlicht falsch. Und wenn man zwei Negationszeichen vor die Formel schreibt, dann ist das zwar eine erlaubte Äquivalenzumformung, aber es bringt nichts.
- Das Distributivgesetz sollte nur angewendet werden, wenn die *Disjunktion außen* und die *Konjunktion innen* steht. Sonst nähert man sich nicht der KNF, sondern entfernt sich von ihr.
- Wer unbedingt den Überstrich als Negation verwenden möchte (also $\overline{a \vee b}$ statt $\neg(a \vee b)$ sollte trotzdem darauf achten, dass keine Negationen verloren gehen (oder dazu kommen).

3 Tableaux

- Häufigster Fehler ist das falsche Aufteilen der Formel. Beachten Sie die Präzedenz der Operatoren. Jede Formel hat eine *eindeutige* voll geklammerte Form, an der die anzuwendende Regel abzulesen ist. Im konkreten Fall war die Formel:

$$\varphi = \neg((a \to b) \land (b \to a) \leftrightarrow (a \land b) \lor (\neg a \lor \neg b))$$

Der oberste Operator ist die Negation $\neg$. In dem Fall müssen wir den nächsten Operator finden - das ist das Äquivalenzsymbol $\leftrightarrow$, denn das hat die niedrigste Präzedenz (alle anderen Operatoren binden stärker). Vollständig geklammert sieht die Formel so aus:

$$\varphi = \neg(((a \to b) \land (b \to a)) \leftrightarrow ((a \land b) \lor ((\neg a) \lor (\neg b))))$$

Ein anderes Beispiel: $A \land B \lor C$ bedeutet $(A \land B) \lor C$, also ist die $\lor-$ und nicht die $\land$-Regel anzuwenden.

Tipp: Wenn Sie sich nicht sicher sind, schreiben Sie die Formel selbst einmal mit allen Klammern.

- Ein andere Fehler, der mehrfach auftrat, war, dass Formeln von verschiedenen Ästen / Spalten zum Schließen / für einen Clash (dann beider Äste) verwendet wurden.
- Entscheidend für Erfüllbarkeit ist *nicht*, ob es mit jeder Variable einen Clash (Widerspruch) gibt, sondern in jeder Spalte / auf jedem Ast.
- Zum Schluss kein Fehler, aber ein Tipp: Wenn absehbar ist, dass das Tableaux zwei relativ große und lange parallele Äste hat, kann es sinnvoll sein, eine zweite Seite zu nehmen, um den zweiten Ast darzustellen.

4 Formalisierung

- Prädikate in Prädikaten sind nicht erlaubt, z.B. $DarfKaufen(Straße(x))$. Denken Sie daran, dass der Wert des inneren Ausdrucks (hier $Straße(x)$) schon ein Wahrheitswert ist - damit kann das äußere Prädikat nichts mehr anfangen.
- Quantoren über Konstanten sind nicht erlaubt, insbesondere braucht man keinen Existenzquantor für Konstanten, z.B. $\exists schlossallee$
- Konstanten sind auch keine Prädikate, z.B. $\exists x(schlossallee(x))$
- Die Position von Quantoren und Klammerung ist nicht egal, so unterscheiden sich z.B.
  - $\forall x \forall y(Spieler(x) \land Strasse(y) \to ...)$
  - $\forall x Spieler(x) \land \forall y(Strasse(y) \to ...)$

  Speziell bindet der Allquantor im zweiten Fall *nur* die Teilformel $Spieler(x)$ - Quantoren haben die höchste Präzedenz und binden die kleinstmögliche Formel. D.h. nach einem Quantor müssen Sie fast immer Klammern um die Formel setzen, auf die er wirken soll.

- Das gilt auch bei Existenzquantoren, so unterscheiden sich z.B.
  - $\forall x \exists y(Spieler(x) \wedge Erreicht(x, y) \rightarrow ...)$
  - $\forall x(Spieler(x) \wedge \exists y Erreicht(x, y) \rightarrow ...)$
  - Bei der oberen Formel reicht ein Wert für y, der die linke Seite der Implikation falsch macht (z.B. y als Spieler), also nicht erreicht werden kann, um die Formel als ganzes wahr zu machen
  - Bei der unteren Formel reicht ein y, das erreicht werden kann, um den linken Teil der Implikation für alle Spieler wahr zu machen
- Oft wurde aus $\forall$ und $\exists$ das falsche gewählt, Faustregel: eine allgemeingültige Regel mit Allquantor und Implikation
- Ein negierter Allquantor ist nicht äquivalent zu einer negierten Formel mit Allquantor
  - $\neg \forall x(Strasse(x) \rightarrow farbe(x) = blau)$
    Nicht alle Straßen sind blau (aber manche vielleicht schon)
  - $\forall x(Strasse(x) \rightarrow \neg(farbe(x) = blau))$
    Alle Straßen sind nicht blau (also ist gar keine blau)

5 Unifikation (und Substitutionen)

- Der mit Abstand häufigste Fehler war das Binden von anderen Termen an Konstanten. *Es können nur Variablen gebunden werden!*

  Wenn Sie ganz sicher gehen wollen, schauen Sie in die Signatur, die wir immer mitgeben. Konstanten sind als spezielle Funktionssymbole in der Menge F. Variablen sind in der Menge V. Wir halten uns in der Regel auch an die Konventionen: Variablen kommen vom Ende des Alphabets $\{x, y, z, u, v, w\}$, Konstanten eher vom Anfang $\{a, b, c, d\}$ oder haben sprechende Namen wie *john*, *peter*, *schloss*.

  Oft schreiben wir Variablen auch groß, Konstanten klein.

  Also: $\{X \leftarrow d\}$ / $x \mapsto d$ (lies: *Die Variable X bekommt den Wert d*) ist eine korrekte Substitution. $\{d \leftarrow X\}$ ist schon per Definition falsch. Und $\{f(x, d) \leftarrow g(d)\}$ ist natürlich auch falsch. Nur Variablen können Werte gegeben werden.
- Sanity Check: Sehen Sie sich nach der Unifikation (auch innerhalb der Resolution) Ihren Unifikator an. Wenn auf der linken Seite irgend etwas anderes als eine *einzelne Variable* steht, haben Sie einen Fehler gemacht.
- Beim Binden einer Variable muss die neue Bindung auf beiden Seiten der Tabelle angewendet werden - zum einen auf alle Terme in den verbleibenden Gleichungen in der linken Spalte der Tabelle, zum anderen auf alle Terme, die den Variablen in der schon bestehenden Substitution rechts

zugewiesen werden. Rechts wird formal die Substitution $\{X \leftarrow t\} \circ \sigma$ berechnet (wobei $\{X \leftarrow t\}$ die neue Bindung ist, σ die bisherige Substitution). Oft wurden eine oder beide dieser Anwendungen vergessen, und die neue Bindung einfach zur vorhandenen Substitution hinzugefügt (das muss natürlich *auch* passieren).

- Beim *Zerlegen* wurde oft etwas gehuddelt. Denken Sie daran, dass unser Verfahren einen Unifikator für eine Menge von Gleichungen (=Termpaaren) berechnet. Wir fangen nur mit einer Menge an, in der erst einmal nur das ursprüngliche Unifikationsproblem steht. Wenn eine Gleichung $f(s_1, s_2) = f(t_1, t_2)$ zerlegt wird, dann entstehen dabei *zwei* neue Gleichungen, $s_1 = t_1$ und $s_2 = t_2$. Wenn das oberste Funktionssymbol einstellig ist (z.B. $g(s_1) = g(t_1)$), dann entsteht auch nur eine Gleichung (in dem Fall $s_1 = t_1$), und wenn das Symbol 3-stellig ist, dann eben 3 Gleichungen.

  In der Klausur habe ich viele Versuche gesehen, den Prozess abzukürzen. Dabei entstanden oft komische Gleichungen zwischen ungeklammerten Listen von Termen. Das entspricht zum einen nicht dem Verfahren, vor allem kommt man dabei aber auch leicht durcheinander.

- Ein konkretes Ergebnis der Anwendung einer Substitution ist ein Term oder eine Klausel, in denen die entsprechenden Variablen substituiert wurden. Normalweise ist das Ergebnis nicht $\top$ oder $\bot$, und erst recht sind es keine Zahlen. Eine Substitution ist vor allem erst einmal eine *Funktion*. Deswegen bleibt Sie i.a. auch nach dem Anwenden nicht vor der Klausel oder dem Term stehen - wenn $\sigma = \{X \leftarrow a\}$, dann ist $\sigma(f(X)) = f(a)$. Dieses Ergebnis ist dann eine *Instanz* des Ausgangsterms oder der Ausgangsklausel.

- Resolution ist nur eine mögliche Anwendung von Unifikation. Bei der Resolution kommen die beiden Atome, die unifiziert werden, aus verschiedenen, unabhängig quantifizierten Formeln. Nur deswegen dürfen Variablen disjunkt umbenannt werden (und, falls die Variablenmengen der Klauseln nicht sowieso schon disjunkt sind, müssen sie es auch). Das heißt aber nicht, dass man immer, wenn man den Unifikationsalgorithmus anwendet, die Variablen so umbenennen darf, wie man möchte. In der Resolution werden die Variablen der ganzen Klausel schon *vor* der Unifikation umbenannt, im Unifikationsalgorithmus selbst geht das nicht mehr.

6 Resolution

- Oft wurden bei der Unifikation Konstanten oder Terme gebunden. Dazu siehe den vorherigen Abschnitt (*Unifikation*)

- Auch wenn es verlockend ist: Es ist nicht möglich, einfach $\neg G(b, c)$ gegen $G(c, b)$ zu resolvieren. Die Reihenfolge der Argumente ist wichtig. Speziell sind in dem Fall b und c Konstanten, d.h. die Atome sind auch nicht unifizierbar.

- Ein Beweis der *Erfüllbarkeit* eine Klauselmenge ist für die Prädikatenlogik nur selten möglich, vor allem mit Resolution. Dafür muss eine *faire* Ableitung zu einem saturierten System ohne leere Klausel führen. Fairness heißt dabei nicht, dass jede der Ausgangsklauseln einmal verwendet wurde, sondern dass *alle* Inferenzen (Resolution und Faktorisierung) zwischen *allen* Klauseln (nicht nur denen der Ausgangsmenge, auch den später erzeugten) durchgeführt wurden. Das führt bei der Prädikatenlogik sehr oft ins unendliche (z.B. kann man in der Klausuraufgabe aus Klausel 4 mit wiederholter Resolution mit Klausel 5 $\neg P(f(c))$, $\neg P(f(f(c)))$, $\neg P(f(f(f(c))))\ldots$ herleiten - die Menge dieser beiden Klauseln hat also schon keine endliche Saturierung.