

Om matematisk logik

Henning Christiansen, Troels Andreasen

Contents

1	Indledning	3
2	Propositionel logik	5
2.1	Propositionelle logiksprog	5
2.1.1	Syntaks	5
2.1.2	Semantik	7
2.1.3	Entydig læsning af formler	10
2.2	Ræsonnering med sandhedsværdier	11
2.3	Aksiomatiske systemer	14
2.4	Ræsonnering ved modbevis	22
2.5	Resolution	23
2.5.1	Klausulform	24
2.5.2	Resolutionsystemer	27
3	Første-ordens prædikatlogik	31
3.1	Første-ordens prædikatlogiksprog	31
3.1.1	Syntaks	31
3.1.2	Semantik	33
3.1.3	Entydig læsning af formler	38
3.2	Ræsonnering med sandhedsværdier	39
3.3	Aksiomatiske systemer	39
3.4	Ræsonnering ved modbevis	40
3.5	Resolution	40
3.5.1	Klausulform	41
3.5.2	Særlige egenskaber ved klausuler	41
3.5.3	Resolutionssystemer	42
4	SLD-resolution og logikprogrammering	45
4.1	Om modeller for definite klausulprogrammer	47
4.2	Prolog	50
4.3	Om beregnede svar	51
5	Afsluttende bemærkninger	52

dummy side

1 Indledning

Logik er formalisme, der anvendes i formulering af og manipulation med udsagn, dvs. størrelser (eller sætninger) som, kan træffes at være sande eller falske.¹ En logik kan betragtes som et sprog og et sæt af regler, der tillader udledning af nye udsagn fra givne. Vi kan med almindelig sund fornuft forklare ræsonnementet

Hvis følgende gælder:

Alle mennesker er dødelige.

Sokrates er et menneske.

så kan vi slutte at:

Sokrates er dødelig.

Der er her oplagt tale om et logisk ræsonnement - slutningen som en sammentrækning af de to udsagn. I formel logik begrænser man sig imidlertid normalt til sprog med en veldefineret syntaks og semantik og en reformulering kunne f.eks. føre til formuleringer som:

$menneske(X) \rightarrow dødelig(X)$.

$menneske(sokrates)$.

Logikken kan så omfatte en regel, der tillader os at udlede endnu et udsagn fra ovenstående to, nemlig:

$dødelig(sokrates)$.

Logik (ofte præciseret som matematisk logik eller formel logik) søger at indfange en del af — eller give en abstraktion over en del af — menneskelig tænkning, som er

- immatriel,
- indholdsløs,
- sproglig i en meget abstrakt forstand,
- objektiv i den forstand, at logikkens ræsonnementer er sande i enhver verden.

¹indenfor den gren der undertiden præciseres som “boolske logikker”

Tænkning i almindelighed handler som regel om noget, som er i verden i og omkring os, f.eks. trillebører, følelser, mad. I logikken ser vi i princippet bort fra tilknytningen til verden. Der er i logikken ingen principiel forskel på trillebøren, kærlighed eller en vellagret Brie de Meaux.

Det sproglige aspekt er på et plan, hvor fonetiske og grammatiske særheder er pillet ud, altså rent symbolsk (prøv at konsultere forskellig opslagsværker, for at se, hvad de mener, et symbol er).

Det objektive aspekt er strengt taget blot en følge af det forhold at vi ser bort fra tilknytningen til verden. Et udsagn som “Sokrates er et menneske” bliver først subjektivt når vi lægger mening i symbolerne (Sokrates og menneske).

Hvad er da det nyttige ved logik, eller, hvorfor er så mange mennesker interesseret i logik. Her er nogle bud.

Det morsomt, sjovt at lege med (jvf. Holberg).

Når det kommer til en diskussion (et fredsommelig skænderi, en retsag osv.), så har “logiske” argumenter er stor vægt, de er ikke til at skyde igennem, det er “klart”, at der ikke er lusket skjulte antagelser ind.²

Vi tænker almindeligvis abstrakt; når det handler om dødeligheden af Sokrates, så giider vi ikke inddrage detaljer angående, om han gik i sandaler og/eller toga eller ej.³

Endelig er der det forhold at logikken er “mekaniserbar” – vi kan konstruere logik-maskiner, der automatisk kan gennemføre ræsonnementer for os. Prolog er et klassisk eksempel på en sådan, men også databasesystemer kan betragtes som logik-maskiner. Udsagn benyttes som struktur til repræsentation af data og ræsonnementer anvendes i evaluering af spørgsmål til de repræsenterede data.

Vi skal i det følgende kapitel se nærmere på detaljer om propositionel logik. Propositionel logik er en forenklet logik uden kvantorer og variable og har således en meget begrænset udtrykskraft. Til gengæld er den teknisk set en hel del nemmere at håndtere, og derfor er den velegnet som en overskuelig introduktion til logik. Med baggrund i grundbegreberne fra propositionel logik tager vi i kapitel 3 fat på på første-ordens logik. I kapitel 4 løfter vi så sløret

²Eller vi kan bruge logikken til at afsløre, når vor modpart lusker antagelser ind.

³Dog gik han ikke i toga, hvilket vi kan se på følgende måde. Sokrates oplyses almindeligvis at have levet ca. 470–399 f.Kr.; togaen blev først opfundet af romerne nogle hundrede år senere; ergo kan Sokrates ikke have båret toga.

for hvad man bl.a. kan bruge formel logik til, specielt ser vi på en forenklinger, der skaber grundlag for automatisk bevisførelse samt på logik-programmering og Prolog, som er baseret på disse forenklinger.

2 Propositionel logik

I propositionel logik arbejdes med simple udsagn såsom ”det regner”, ”solen skinner”, ”det er skyfrit”. Når man formulerer udtryk i propositionel logik søger man typisk at nedbryde disse så meget som muligt. F.eks. kan ”solen skinner og det er skyfrit” nedbrydes i ”solen skinner” og ”det er skyfrit”. Et propositionelt logikprog kan så benyttes til igen at samle de såkaldt ”atomare” udsagn, som f.eks. i (”solen skinner” \wedge ”det er skyfrit”) – eller i $(P \wedge Q)$ hvor P og Q repræsenterer hhv. ”solen skinner” og ”det er skyfrit”.

2.1 Propositionelle logikprog

Som ethvert formelt sprog, kan et propositionelt logikprog beskrives ved en syntaks og en semantik. Vi beskriver herunder syntaksen – eller rettere rammerne for propositionel syntaks og tager i næste afsnit fat på semantikken.

2.1.1 Syntaks

Sætninger i et propositionelt logikprog kaldes formler⁴. Syntaksen for et sprog defineres ved et sæt af symboler samt nogle regler for at sammensætte disse symboler. Symbolerne i et propositionelt logikprog kan opdeles i:

1. symboler for atomare formler
2. logiske konnektorer⁵
3. parenteser

1. er hvad man benytter til at notere *atomare formler*, f.eks. kan symbolerne P eller ”solen skinner” repræsentere det forhold at solen skinner. 2. kan f.eks. omfatte konnektorerne \neg , \wedge , \vee , (hhv. negation, konjunktion (logisk ”og”) og disjunktion (logisk ”eller”)) der tillader os at bygge sammensatte formler udfra atomare formler. 3. benyttes til at afgrænse virkefelter for de logiske konnektorer.

⁴På engelsk benyttes normalt mere specifikt ”well formed formulas”, der forkortes ”wff”.

⁵Konnektorer betegnes også konnektiver nu og da.

Atomare formler kaldes ofte blot *atomer*.⁶ Endelig skelnes et *literal* som et atom eller et negeret atom. P og $\neg P$ er altså begge literaler, men kun P er et atom.

Til en syntaksbeskrivelse hører også et sæt regler, som bestemmer, hvordan udvalget af symboler sættes sammen til formler. Et konkret sæt af symboler tilknyttet regler for sammensætning af disse udgør en syntaksbeskrivelse for et sprog. Reglerne kan f.eks. angives som i eksemplet nedenfor.

Eksempel 2.1 Tag som eksempel på et sprog følgende, der udover \wedge , \vee og \neg omfatter \rightarrow , \leftrightarrow der betegnes hhv. implikation og biimplikation. Sproget indeholder følgende symboler:

1. symboler for atomare formler: P, Q, R
2. logiske konnektorer: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$
3. parenteser: $(,)$

Formler i sproget defineres rekursivt som følger:

- Et atom er en formel.
- Hvis ϕ er en formel, så er (ϕ) og $\neg\phi$ formler.
- Hvis ϕ og ψ er formler, så er $\phi \vee \psi$, $\phi \wedge \psi$, $\phi \rightarrow \psi$ og $\phi \leftrightarrow \psi$ formler.

Følgende er altså eksempler på formler i sproget:

$$\begin{aligned} &P \\ &(((Q))) \\ &(P \rightarrow (P \vee Q) \wedge R) \end{aligned}$$

□

Bemærk, at vi har præsenteret konnektorerne \rightarrow og \leftrightarrow i et eksempel og ikke i en generel definition. Dette er for at understrege, at valget af konnektorer (ligesom notationen for disse) i princippet er frit. Imidlertid skal det understreges at det typisk netop er de fem konnektorer konjunktion, disjunktion, negation, implikation og biimplikation: $\wedge, \vee, \neg, \rightarrow, \overset{7}{\leftrightarrow}$, der betragtes i

⁶Der er desværre et overlap i terminologi. Dette begreb af atomer er forskelligt fra atomer i programmeringssproget Prolog; disse svarer i stedet til konstanter i prædikatlogikken, som vi kommer til senere.

⁷Vi vil dog undertiden benytte \leftarrow i stedet for \rightarrow , idet $P \leftarrow Q$ tillægges samme betydning som $Q \rightarrow P$.

logik. Dette skal dog ikke afholde os fra at se på logiksprøget som ikke omfatter alle konnektorerne eller for den sags skyld at indføre nogle nye. Som det vil fremgå når vi betragter semantikken forholder det sig iøvrigt også sådan at man udtrykke visse af konnektorerne v.h.j.a. andre.

2.1.2 Semantik

Semantikken af et propositionelt logiksprøget vedrører sandhedsværdien for formlerne i sproget. En sandhedsværdi er et af de to symboler *sand* og *falsk*. Vi præciserer, at sandhedsværdier er arbitrære symboler, vi kunne lige så godt have benyttet ♣ og ♠. De associationer, vi som brugere af logikken får ved et symbol som *falsk*, har ikke noget med logikken i sig selv at gøre, men snarere med det pragmatiske aspekt, som omhandler logikkens anvendelighed.

Hvis sandhedsværdien for sprogets atomer er givet på forhånd, kan betydningen af en vilkårlig formel bestemmes udfra en sandhedstabel.⁸

Eksempel 2.2 Semantikken for et sprøget med konnektorerne \wedge , \vee , \neg , \rightarrow , \leftrightarrow er bestemt ved følgende sandhedstabel:

ϕ	ψ	(ϕ)	$\phi \wedge \psi$	$\phi \vee \psi$	$\neg\phi$	$\phi \rightarrow \psi$	$\phi \leftrightarrow \psi$
<i>sand</i>	<i>sand</i>	<i>sand</i>	<i>sand</i>	<i>sand</i>	<i>falsk</i>	<i>sand</i>	<i>sand</i>
<i>sand</i>	<i>falsk</i>	<i>sand</i>	<i>falsk</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>	<i>falsk</i>
<i>falsk</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>	<i>sand</i>	<i>sand</i>	<i>sand</i>	<i>falsk</i>
<i>falsk</i>	<i>falsk</i>	<i>falsk</i>	<i>falsk</i>	<i>falsk</i>	<i>sand</i>	<i>sand</i>	<i>sand</i>

I tabellen er angivet sandhedsværdien for formler, der involverer formlerne ϕ og ψ og ved (eventuelt) gentagne anvendelser af tabellen, kan sandhedsværdien for enhver formel i sproget bestemmes udfra sandhedsværdien af atomerne i formelen. F.eks. er $\phi = P \vee \neg(Q \leftrightarrow \neg R)$ sand, hvis P er falsk, Q er sand og R er sand. \square

Generelt er sandhedsværdien af en formel relativ til sandhedsværdierne for atomerne. Eller omvendt, en tildeling af sandhedsværdier til sprogets atomer, bestemmer, for enhver af sprogets formler, en entydig sandhedsværdi⁹.

Har vi et sprøget med tre atomer P , Q , og R , er der $2^3 = 8$ forskellige muligheder:

⁸Bemærk, at semantikken således er *kompositionel*, også kaldet *induktiv* eller *homomorfisk*. Man kunne forestille sig andre "konteks-afhængige" principper for at udregne sandhedsværdier, men i propositionel logik benyttes det enkleste og mest "mekaniske".

⁹Dog forudsat at formelen har en entydig læsning – se næste afsnit.

I_1 :	$P := sand$	$Q := sand$	$R := sand$
I_2 :	$P := sand$	$Q := sand$	$R := falsk$
I_3 :	$P := sand$	$Q := falsk$	$R := sand$
I_4 :	$P := sand$	$Q := falsk$	$R := falsk$
I_5 :	$P := falsk$	$Q := sand$	$R := sand$
I_6 :	$P := falsk$	$Q := sand$	$R := falsk$
I_7 :	$P := falsk$	$Q := falsk$	$R := sand$
I_8 :	$P := falsk$	$Q := falsk$	$R := falsk$

Vi skal kalde disse muligheder for fortolkninger (af formler over atomerne P , Q , og R). F.eks. har formelen $\phi = P \vee \neg(Q \leftrightarrow \neg R)$, ovennævnte mulige fortolkninger. Kun nogle af disse opfylder ϕ , dvs. gør ϕ sand, nemlig I_1, I_2, I_3, I_4, I_5 og I_7 . (Prøv at verificere dette). Hvis vi nu antager, vi gives formelen ϕ sammen med den påstand, at den altså *er* sand, så udpeges herved nogle fortolkninger som mulige, de resterende som umulige.

Ved en model for ϕ forstås en fortolkning, der opfylder ϕ . Modellerne for $\phi = P \vee \neg(Q \leftrightarrow \neg R)$, er altså I_1, I_2, I_3, I_4, I_5 og I_7 .

Mere generelt defineres fortolkning og model for mængder af formler som følger.

Definition 2.1 En *fortolkning* for en mængde af formler Γ er en sandhedsværditildeling, der (mindst) omfatter samtlige atomer, der indgår i Γ . En *model* for Γ er en fortolkning, der gør alle formler i Γ sande. \square

Eksempel 2.3 F.eks. har vi for $\Gamma = \{\neg P, P \vee Q\}$, der indeholder to formler, fire forskellige fortolkninger,

I_1 :	$P := sand$	$Q := sand$
I_2 :	$P := sand$	$Q := falsk$
I_3 :	$P := falsk$	$Q := sand$
I_4 :	$P := falsk$	$Q := falsk$

men kun en model, nemlig I_3 . \square

Γ i eksemplet er informationsbærende i den forstand, at hvis dens formler ellers accepteres som sande, gør den det muligt at konkludere om fortolkningen af atomerne – atomernes sandhedsværdier afgrænses af kombinationerne i modellerne for Γ . Helt analogt til det forhold, at vi kan uddrage information om verden, når en person, vi stoler på, fortæller os noget.

Definition 2.2 En *gyldig formel* ϕ er en formel, der er sand i enhver fortolkning (dvs. enhver fortolkning for ϕ er en model for ϕ). En gyldig formel kaldes også en *tautologi*. \square

Eksempel 2.4 Formlerne

$$P \vee \neg P$$

$$P \rightarrow P$$

$$P \rightarrow P \vee Q$$

er eksempler på en gyldige formler. \square

Gyldige formler er universelle i den forstand, at de er sande uanset, hvordan de indgående atomer fortolkes. Det svarer på sin vis til sætninger, alle kan være enige om. Tager vi metaforen om information op igen, kan gyldige formler ikke siges at rumme nogen information, de fremhæver ikke nogle fortolkninger frem for andre. Omvendt kan de måske være til nytte, når vi ønsker at transformere formler.

Fortolkning og model for en mængde af formler var defineret ud fra sandhed for samtlige formler på én gang. Det fremgår af sandhedstabellen for \wedge , at sandhed af en mængde af formler er det samme som sandhed af deres konjunktion. For eksempel har $\{\phi, \psi, \xi\}$ de samme fortolkninger og modeller som $\{\phi \wedge \psi \wedge \xi\}$ for alle formler ϕ , ψ og ξ . Vi kan altså betragte en mængde af formler alternativt som konjunktionen af formlerne.

Definition 2.3 To formler siges at være *ækvivalente*, hvis de har samme sandhedsværdi i enhver fortolkning. Dette noteres $\phi \equiv \psi$. \square

Eksempel 2.5 Som eksempel har vi

$$P \rightarrow Q \equiv \neg P \vee Q$$

(Verificer dette). \square

Sådanne ækvivalenser vil vise sig at være nyttige, når vi kommer til at ræsonnere om formler. Hvis vi ønsker at udtale os om egenskaber ved $P \rightarrow Q$, kunne vi måske, hvis det var nemmere, bearbejde formlen $\neg P \vee Q$ i stedet for.

Eksempel 2.6 Prøv at betragte følgende ækvivalenser

$$\phi \wedge \psi \equiv \neg(\phi \rightarrow \neg\psi)$$

$$\phi \vee \psi \equiv \neg\phi \rightarrow \psi$$

$$\phi \leftrightarrow \psi \equiv (\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi) \equiv \neg((\phi \rightarrow \psi) \rightarrow \neg(\psi \rightarrow \phi))$$

Disse kunne f.eks. anvendes til at eliminere konnektorerne \wedge , \vee og \leftrightarrow fra enhver formel i sproget. Konnektorerne kan m.a.o. undværes uden at det går ud over udtrykskraften i sproget. Man ser undertiden sådanne ækvivalenser omtalt som *forkortelser*. \square

Endelig skal det bemærkes at kan man ved hjælp af en sandhedstabel forvise sig om, at $\phi \equiv \psi$ hvis og kun hvis $\phi \leftrightarrow \psi$ er en tautologi (Prøv).

2.1.3 Entydig læsning af formler

Det skal bemærkes at logikprog som defineret ovenfor efterlader et problem - formler kan ofte læses på flere måder. Formlen $P \wedge Q \vee R$ kan eksempelvis læses på to måder, som en konjunktion sat sammen af P og en disjunktion eller som en disjunktion sat sammen af en konjunktion og R . Ved at indføre en præcedens (eller prioritet) for konnektorerne, kan vi fjerne denne flertydighed. Vi placerer konnektorerne i et hierarki, med højest præcedens øverst.

\neg
 \wedge
 \vee
 $\rightarrow, \leftrightarrow$

Udtrykket $P \wedge \neg Q \vee R$ skal hermed forstås som en disjunktion af en konjunktion og atomet R ; den nævnte konjunktion består af atomet P og det negative literal $\neg Q$. Bemærk, at disse konventioner ikke er fuldstændige i den forstand, at de opløser enhver tvetydighed, f.eks. er \rightarrow og \leftrightarrow er på samme niveau. Her må parenteser benyttes for at opnå entydige udtryk. Konventionerne siger heller ikke noget om, hvorvidt eksempelvis $P \vee Q \vee R$ skal læses som $(P \vee Q) \vee R$ eller $P \vee (Q \vee R)$, men det kan vi se stort på, da \vee (og også \wedge) er associativ.¹⁰

Det skal advares om, at ikke alle forfattere er enige om konnektorernes indbyrdes præcedens, og det samme gælder om de analoge operatorer, som optræder i programmeringssprog.¹¹

¹⁰Associativitet af \vee vil sige, at betydningerne af udtrykkene $(P \vee Q) \vee R$ og $P \vee (Q \vee R)$ er identiske, m.a.o. de er ækvivalente. Vi kan således nøjes med at skrive $P \vee Q \vee R$.

¹¹Bemærk at selv om vi har et flertydigt sprog kan vi altid notere entydige formler. Derfor kunne vi vælge at indføre en notations-princip for "Sikker logik", nemlig: Brug altid parentes!

2.2 Ræsonnering med sandhedsværdier

Ved at jonglere med tildelinger af sandhedsværdier, dvs. ved at betragte fortolkninger og modeller, kan man ræsonnere på en mængde af formler.

Det er vigtigt at være opmærksom på at sådan ræsonnering grundlæggende er ren symbolmanipulation. Har vi nogle udsagn, som vi mener at kunne betragte som sande eller falske i vores virkelighed, og ønsker vi at manipulere med disse i logik, må vi opfinde symboler for udsagnene for at kunne notere dem, men vi noterer jo kun symbolerne og ikke deres forbindelse til virkeligheden, og det står enhver frit for at tillægge nye betydninger til symbolerne. Eller med andre ord, de samme konstellationer af symboler kan fremkomme ved at formalisere helt andre dele af virkeligheden, som så tilfældigvis har de samme logiske egenskaber.

Hvis vi f.eks. vil ræsonnere på vejr-situationen, kan vi identificere følgende atomare udsagn og notere dem ved atom-symboler.

P betyder ”vi har nedbør”
 Q betyder ”det er koldt”
 R betyder ”det er varmt”
 S betyder ”det sneer”
 T betyder ”det er regnvejr”

Følgende formler:

$$\Gamma = \{P \rightarrow S \vee T, P \wedge Q \rightarrow S, P \wedge R \rightarrow T\}$$

udtrykker at nedbør er sne eller regn afhængigt af, om det er hhv. koldt eller varmt. Har vi formaliseret vor viden om vejret i form af den logiske beskrivelse Γ , kan vi foretage ræsonnementer ved ren symbolmanipulation — vi kunne sågar sætte en datamaskine til det. Ræsonnementerne, vil være lige så anvendelige, hvis atomerne istedet (af det menneskelige subjekt) tillægges følgende betydninger.

P betyder ”vi er sultne”
 Q betyder ”vi har æg i køleskabet”
 R betyder ”køleskabet er tomt”
 S betyder ”vi skal have omelet”
 T betyder ”vi skal have pizza”

Logikken (eller datamaskinen) er ligeglad, den kender alligevel ikke forskel på regnvejr og en pizza.

Vi skal i det følgende gentagne gange se på et simpelt eksempel, som vi derfor her skal introducere nogle ”huske-symboler” på samt postulere noget mening med:

Is betyder ”Ingen skyer”
 Ds betyder ”Delvist skyet”
 Os betyder ”Overskyet”

Formlerne $\Gamma = \{Is \vee Ds \vee Os, \neg(Is \vee Ds)\}$ betyder hermed at det enten er skyfrit, delvist skyet eller overskyet, samt at det ikke gælder at det er skyfrit eller delvist skyet.

Ræsonnering med sandhedsværdier vedrører først og fremmest undersøgelse af opfyldelighed og logisk konsekvens som vi skal definere herunder. At en mængde af formler Γ er *opfyldelig* kan ses som et udtryk for at Γ ikke indeholder en modstrid. At en formel ϕ er en *logisk konsekvens* af en mængde af formler Γ udtrykker at ϕ følger af Γ eller at Γ godtgør at ϕ gælder.

Definition 2.4 En mængde af formler Γ siges at være *opfyldelig*, hvis Γ har en model. \square

Vi kan altså afgøre opfyldelighed ved at lede efter en model blandt fortolkningerne for Γ . Hertil kan anvendes en sandhedstabel over fortolkningerne for Γ sammenholdt med formlerne i Γ .

Eksempel 2.7 F.eks. kan vi konstatere at $\Gamma = \{Is \vee Ds \vee Os, \neg(Is \vee Ds)\}$ er opfyldelig ud fra sandhedstabellen, hvor vi for mulige fortolkninger af atomerne har udregnet sandhedsværdien for formlerne.

Is	Ds	Os	$Is \vee Ds \vee Os$	$\neg(Is \vee Ds)$	Γ
<i>sand</i>	<i>sand</i>	<i>sand</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>
<i>sand</i>	<i>sand</i>	<i>falsk</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>
<i>sand</i>	<i>falsk</i>	<i>sand</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>
<i>sand</i>	<i>falsk</i>	<i>falsk</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>
<i>falsk</i>	<i>sand</i>	<i>sand</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>
<i>falsk</i>	<i>sand</i>	<i>falsk</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>
<i>falsk</i>	<i>falsk</i>	<i>sand</i>	<i>sand</i>	<i>sand</i>	<i>sand</i>
<i>falsk</i>	<i>falsk</i>	<i>falsk</i>	<i>falsk</i>	<i>sand</i>	<i>falsk</i>

Vi ser, at begge formler i Γ er sande for $Is := falsk, Ds := falsk, Os := sand$, dvs. $\{Is := falsk, Ds := falsk, Os := sand\}$ er en model for Γ . Ud fra tabellen kan samtidigt konstateres, at den er den eneste model for Γ . Tilsvarende kan konstateres at $\Gamma' = \{Is \vee Ds \vee Os, \neg(Is \vee Ds), \neg Os\}$ ikke er opfyldelig. \square

Bestemmelse af opfyldelighed for Γ kan betragtes som undersøgelse af konsistens, for hvad der er udtrykt med Γ . Formlerne Γ kan ses som beskrivende

egenskaber for mulige verdener, hvor verden her svarer til model, og spørgsmålet om opfyldelighed svarer til, om der overhovedet kan eksistere en verden, i hvilken Γ gælder.

Definition 2.5 En formel ϕ er en *logisk konsekvens* af en mængde af formler Γ , hvis den er opfyldt i enhver model for Γ , dvs. hvis den er sand i enhver sandhedsværditildeling, der opfylder Γ . Logisk konsekvens noteres $\Gamma \models \phi$. \square

En sandhedstabel over fortolkningerne for atomerne i Γ sammenholdt med formlerne i Γ og en formel ϕ kan anvendes i bestemmelsen af logisk konsekvens.

Eksempel 2.8 Betragt som eksempel $\phi = Os \wedge \neg Is$ og $\Gamma = \{Is \vee Ds \vee Os, \neg(Is \vee Ds)\}$. At $\Gamma \models \phi$ følger af tabellen:

Is	Ds	Os	$Is \vee Ds \vee Os$	$\neg(Is \vee Ds)$	Γ	$Os \wedge \neg Is$
<i>sand</i>	<i>sand</i>	<i>sand</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>	<i>falsk</i>
<i>sand</i>	<i>sand</i>	<i>falsk</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>	<i>falsk</i>
<i>sand</i>	<i>falsk</i>	<i>sand</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>	<i>falsk</i>
<i>sand</i>	<i>falsk</i>	<i>falsk</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>	<i>falsk</i>
<i>falsk</i>	<i>sand</i>	<i>sand</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>	<i>sand</i>
<i>falsk</i>	<i>sand</i>	<i>falsk</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>	<i>falsk</i>
<i>falsk</i>	<i>falsk</i>	<i>sand</i>	<i>sand</i>	<i>sand</i>	<i>sand</i>	<i>sand</i>
<i>falsk</i>	<i>falsk</i>	<i>falsk</i>	<i>falsk</i>	<i>sand</i>	<i>falsk</i>	<i>falsk</i>

Vi ser, at der er en (og iøvrigt kun en) model for Γ , nemlig $\{Is := falsk, Ds := falsk, Os := sand\}$ og at denne opfylder ϕ . Altså er ϕ opfyldt i enhver model for Γ . \square

Bemærk specielt, at hvis Γ ikke er opfyldelig, dvs. må opfattes som inkonsistent, så er en hvilken som helst formel en logisk konsekvens af Γ .¹² Vi har tilsvarende, at tautologier er logiske konsekvenser af alle Γ 'er, opfyldelige som ikke opfyldelige.

Vi lister nogle andre egenskaber ved logisk konsekvens, hvoraf nogle vil komme os til nytte senere.

1. For en opfyldelig Γ gælder at $\Gamma \models \phi$ hvis og kun hvis $\Gamma \cup \{\phi\}$ er opfyldelig.
2. $\Gamma \models \phi$ hvis og kun hvis $\Gamma \cup \{\neg\phi\} \models falsk$, hvor *falsk* er et atom, hvis sandhedsværdi altid er *falsk*.

¹²Intuitivt, hvis en taler er begyndt at modsige sig selv, opfatter vi vedkommende som utilregnelig, og vi kan forvente hvad som helst af udtalelser.

3. $\Gamma \models (\phi \rightarrow \psi)$ hvis og kun hvis $\Gamma \cup \phi \models \psi$.
4. Antag $\Gamma \models \psi$. Da gælder $\Gamma \models \phi$ hvis og kun hvis $\Gamma \cup \psi \models \phi$.
5. $\Gamma \models \phi \wedge \psi$ hvis og kun hvis $\Gamma \models \phi$ og $\Gamma \models \psi$.
6. $\Gamma \models \phi \vee \psi$ hvis og kun hvis $\Gamma \models \phi$ eller $\Gamma \models \psi$.
7. $\Gamma \models \neg\phi$ hvis og kun hvis der ikke gælder $\Gamma \models \phi$.

Ad. 1: For at se, om ϕ er logisk konsekvens af Γ , kan vi undersøge sundhedstilstanden for det samlede system $\Gamma \cup \{\phi\}$

Ad. 2: Hvis vi ønsker at verificere ϕ , kunne vi antage, at det modsatte gælder, og vise, at det fører til en modstrid.

Ad. 3: For at undersøge en påstand ”hvis ϕ , så ψ ”, kan vi lade som om ϕ gælder — og så undersøge om det har ψ som konsekvens.

Ad. 4: Hvis vi ønsker at verificere ϕ , og allerede har verificeret ψ , så kunne vi prøve at tilføje ψ til vore grundantagelser og se om det kan være til nogen nytte.

Ad. 5–7: Illustrerer, at de logiske konnektorer, fra starten af er defineret hensigtsmæssigt.

Disse egenskaber kan bruges, hvis man ønsker at bevise egenskaber om de formaliserede bevissystemer, vi vil se på i det følgende. Det, at undersøge opfyldelighed og logisk konsekvens, vha. beregning af sandhedstabeller, som vi har gjort her, har kun en begrænset anvendelighed idet antallet af mulige sandhedsværditildelinger vokser eksponentielt med antallet af atomer — n atomer giver 2^n mulige sandhedsværditildelinger. Det skal understreges, at ræsonnering med sandhedsværdier under normale omstændigheder ikke er praktisk muligt i prædikatlogikken, som vi generaliserer til i næste kapitel.

2.3 Aksiomatiske systemer

Vi skal nu gå bort fra at betragte sandhedsværdier og i stedet fokusere på såkaldte aksiomatiske systemer. Formålet med et aksiomatisk system er at formalisere ræsonnering alene baseret på regler for at drage slutninger ud fra præmisser, uden hensyn til sandhedsværdier — med andre ord en slags syntaktisk ræsonnering, som kunne lægge op til en mekaniseret ræsonnering, f.eks. med brug af en datamaskine.

Et aksiomatisk system ¹³ $\mathcal{S} = \langle \mathcal{L}, \mathcal{D}, \mathcal{A} \rangle$ består af følgende:

- et logisk sprog \mathcal{L} ,

¹³også kaldet deduktionssystem eller bevissystem

- et sæt af slutningsregler \mathcal{D} ,
- et sæt af logiske aksiomer \mathcal{A} .

Man kan slutte nye formler ud fra givne formler i sproget ved at anvende slutningsreglerne og de logiske aksiomer. Vi har allerede, i afsnit 2.1.1, set et eksempel på et sprog, så næste skridt er at forklare slutningsregler og logiske aksiomer.

Slutningsregler er udsagn, som udtaler sig *om* formler, helt på samme måde som teksten i det foregående ikke er en formel, men der optræder formler i den, og den handler om formler. Uden at gøre opmærksom på det, har vi benyttet græske bogstaver som meta-variable i udtryk som " $\phi \vee \neg\phi$ er en tautologi". Domænet for variabelen ϕ er altså formler, og egenskaben "... er en tautologi" gælder for alle instanser, som fås ved at proppe en formel ind i stedet for ϕ . Et udtryk som $\phi \vee \neg\phi$ kaldes et formelskema og kan defineres syntaktisk som en formel, hvis atomer (her ϕ) ligger i en udvidelse til det logiske sprog, vi aktuelt taler om — signaleret ved brugen af græske bogstaver.

Vi definerer generelt en slutningsregel på følgende måde, som giver os en vis fleksibilitet, når det handler om at beskrive sådanne regler.

Definition 2.6 En *slutningsregel* er en regel, der kan anvendes til at producere en ny formel ud fra givne formler. Den nye formel kaldes *slutningen* og de givne formler kaldes *præmisserne*. \square

Som oftest angives præmisser og slutninger vha. formelskemaer adskilt af en vandret streg med præmisserne for oven og slutningen for neden.

Eksempel 2.9 Betragt som eksempel "modus ponens": "Fra ϕ og $\phi \rightarrow \psi$ slut ψ " altså givet præmisserne ϕ og $\phi \rightarrow \psi$ kan sluttes ψ , hvilket også noteres på følgende måde.

$$\frac{\phi \quad \phi \rightarrow \psi}{\psi}$$

En anvendelse af modus ponens opnås ved at sætte formler ind for meta-variablene. Her et eksempel, hvor vi indsætter et atom for den ene meta-variabel og et negativt literal for den anden.

$$\frac{\text{"det er skyfrit"} \quad \text{"det er skyfrit"} \rightarrow \neg\text{"det regner"}}{\neg\text{"det regner"}}$$

Eller udtrykt i ord, ud fra formlerne

”det er skyfrit”, og

”det er skyfrit” \rightarrow \neg ”det regner”

kan vi slutte formelen \neg ”det regner” ved brug af modus ponens.

Tilsvarende kan vi udfra formlerne

”det regner” \vee ”det sner”, og

(”det regner” \vee ”det sner”) \rightarrow \neg ”det er skyfrit”

med modus ponens slutte formelen \neg ”det er skyfrit”, ved instansen:

$$\frac{\text{”det regner”} \vee \text{”det sner”} \quad \text{”det regner”} \vee \text{”det sner”} \rightarrow \neg \text{”det er skyfrit”}}{\neg \text{”det er skyfrit”}}$$

□

Udover slutningsregler, kan der være brug for i et bevis at referere til logiske aksiomer, som indfanger fundamentale, semantiske egenskaber ved logikken, som det kan være nødvendigt at referere til i en bevisførelse.

Definition 2.7 Et *logisk aksiom* er en gyldig formel, som vi vælger at betragte som et formelskema, dvs. vi ophæver de variable til metavariable. Ved en *instans* af et logisk aksiom, forstås en formel i sproget, som fremkommer ved at erstatte metavariablene med formler. □

Eksempel 2.10 Betragt det logiske aksiom $A = \phi \vee \neg\phi$. I et sprog, hvor ϕ er et atom, er A gyldig, dvs. A er sand uanset om ϕ er fortolket som *sand* eller *falsk*. Benyttes A som logisk aksiom sammen med et sprog, som indeholder atomet ”at være”, har vi bl.a. følgende instans $A' = \text{”at være”} \vee \neg \text{”at være”}$. Uanset den aktuelle fortolkning af ”at være”, så er A' sand, dvs. A' er en gyldig formel. □

Instanser af logiske aksiomer er altid i sig selv gyldige formler.

Teknisk set kan logiske aksiomer forstås som slutningsregler med nul præmisser, men vi opretholder skelnenen af historiske grunde. Kravet om, at logiske aksiomer skal være gyldige, er egentlig ikke nødvendigt. Vi kunne have valgt at udelade forudsætningen for at understrege det arbitrære og mekanistiske i et aksiomatisk system. Men aksiomatiske systemer med ugyldige, logiske aksiomer ville hurtigt vise sig at være unyttige ved ikke at være sunde i den forstand, vi vil definere nedenfor.

Eksempel 2.11 Med et logisk aksiom som

$$\phi \rightarrow (\phi \vee \psi),$$

fås ved at substituere ”det er skyfrit” for ϕ og \neg ”det regner” for ψ , en gyldig formel

$$\text{”det er skyfrit”} \rightarrow (\text{”det er skyfrit”} \vee \neg \text{”det regner”}).$$

□

I det aksiomatiske system udledes nye formler ved beviser. Vi definerer herunder hvad der skal forstås ved et bevis i det aksiomatiske system, men understreger at det svarer ganske godt til hvad vi normalt opfatter som et ”bevis” – altså et forløb med konstateringer af hvad der gælder og ”lovlige” konklusioner udfra hvad der gælder og hvad vi allerede har konkluderet.

Definition 2.8 Lad $\mathcal{S} = \langle \mathcal{L}, \mathcal{D}, \mathcal{A} \rangle$ være et aksiomatisk system. Et *bevis* for en formel ϕ er en sekvens af formler ϕ_1, \dots, ϕ_n i \mathcal{L} , hvor $\phi = \phi_n$ og for alle i :

1. ϕ_i er en instans af et logisk aksiom i \mathcal{A} , eller
2. ϕ_i kan udledes fra formler udvalgt blandt $\phi_1, \dots, \phi_{i-1}$ ved anvendelse af en slutningsregel i \mathcal{D} .

Såfremt der findes et bevis for en formel ϕ , siges ϕ at være et *teorem* i \mathcal{S} , hvilket vi noterer $\mathcal{S} \vdash \phi$ (eller blot $\vdash \phi$, når \mathcal{S} er underforstået). □

Eksempel 2.12 Et aksiomatisk system kan f.eks. bestå af $\mathcal{S}_1 = \langle \mathcal{L}, \mathcal{D}, \mathcal{A} \rangle$ med:

\mathcal{L} : et logiksprøeg med konnektorerne \rightarrow og \neg ,

\mathcal{D} : een slutningsregel: modus ponens ,

\mathcal{A} : to logiske aksiomer

$$A_1 : \phi \rightarrow \phi \text{ og}$$

$$A_2 : \phi \rightarrow (\neg\phi \rightarrow \psi)$$

Med dette tilfældigt valgte system \mathcal{S}_1 kan vi f.eks. bevise $\phi = \neg(P \rightarrow P) \rightarrow Q$ som følger

$$\phi_1 = P \rightarrow P; \text{ instans af } A_1$$

$$\phi_2 = (P \rightarrow P) \rightarrow (\neg(P \rightarrow P) \rightarrow Q); \text{ instans af } A_2$$

$$\phi_3 = \neg(P \rightarrow P) \rightarrow Q; \text{ af } \phi_1 \text{ og } \phi_2 \text{ ved modus ponens}$$

Vi har dermed vist ϕ idet $\phi = \phi_3$. Det bemærkes at jonglering alene med gyldige formler (såsom logiske aksiomer) har en tendens til at gøre eksempler både banale og forvirrende. \square

Et aksiomatisk system, hvis det ellers opfylder passende kvalitetskrav, kan kun bruges i forhold til et vise gyldige formler, dvs. tautologier eller ”informationsløse” udsagn, som altid gælder. Vi udvider derfor begrebet til teorier, som i en eller anden forstand siges at handle om noget, nemlig, hvad der er udtrykt i de såkaldte ægte aksiomer.

Definition 2.9 Lad $\mathcal{S} = \langle \mathcal{L}, \mathcal{D}, \mathcal{A} \rangle$ være et aksiomatisk system. En teori \mathcal{T} baseret på \mathcal{S} er en udvidelse af \mathcal{S} med en mængde formler Γ kaldet *ægte aksiomer* (eller *ikke-logiske aksiomer*); vi noterer dette $\mathcal{T} = \langle \mathcal{L}, \mathcal{D}, \mathcal{A}, \Gamma \rangle$ eller $\mathcal{T} = \mathcal{S} + \Gamma$. Begrebet af beviser udvides til at gælde for teorier, ved udover instanser af ægte aksiomer, også at tillade ægte aksiomer i beviset. Såfremt der findes et bevis for en formel ϕ , siges ϕ at være et *teorem* i \mathcal{T} , hvilket vi noterer $\mathcal{T} \vdash \phi$ eller $\Gamma \vdash_{\mathcal{S}} \phi$ (eller blot $\Gamma \vdash \phi$, når \mathcal{S} er underforstået).

Ved *afslutningen* af en teori \mathcal{T} forstås den (evt. uendelige) mængde af alle teoremer i \mathcal{T} . \square

I litteraturen er der beklageligvis en terminologisk forvirring omkring noget så fundamentalt som aksiomer. Nogen steder benyttes ”aksiomer” kollektivt om (instanser af) logiske aksiomer og ægte aksiomer. Andre steder benyttes ”aksiomer” om logiske aksiomer og de ægte aksiomer kaldes ”antagelser”, eller slet og ret Γ . For at fuldende forvirringen, så støder man engang i mellem på forfattere, som benytter betegnelsen ”teori” om det, vi har kaldt teoriens afslutning.

Har vi en teori bestående af et aksiomatisk system \mathcal{S} og et sæt ægte aksiomer Γ , kan Γ intuitivt opfattes som et program, og \mathcal{S} som fortolkeren, der udfører det. Eller i en anden kontekst, Γ er en vidensbase og \mathcal{S} maskinen, som ræsonnerer over den repræsenterede viden.

Eksempel 2.13 Lad os se på en variant af \mathcal{S}_1 fra eksempel 2.12, hvor \mathcal{A} kun omfatter et logisk aksiom. Vi kalder systemet \mathcal{S}_2 :

\mathcal{L} : et logiksprøng med konnektorerne \rightarrow og \neg ,

\mathcal{D} : een slutningsregel: modus ponens ,

\mathcal{A} : et logisk aksiom

$$A_1 : \phi \rightarrow (\neg\phi \rightarrow \psi)$$

Hvis vi nu danner en teori baseret på \mathcal{S}_2 ved at udvide med de ægte aksiomer

$$\Gamma: \{Is, (\neg Is \rightarrow Ds) \rightarrow \neg Os\}$$

kan vi f.eks. vise at Os ikke gælder (det er ikke overskyet) som følger

Vi sætter $\phi = \neg Os$ og får

$$\phi_1 = Is; \text{ givet i } \Gamma$$

$$\phi_2 = Is \rightarrow (\neg Is \rightarrow Ds); \text{ instans af } \mathcal{A}_1$$

$$\phi_3 = (\neg Is \rightarrow Ds); \text{ af } \phi_1 \text{ og } \phi_2 \text{ ved modus ponens}$$

$$\phi_4 = (\neg Is \rightarrow Ds) \rightarrow \neg Os; \text{ givet i } \Gamma$$

$$\phi_5 = \neg Os; \text{ af } \phi_3 \text{ og } \phi_4 \text{ ved modus ponens}$$

Hermed vist at i teorien $\mathcal{T} = \mathcal{S}_2 + \Gamma$ gælder $\neg Os$ idet $\phi = \phi_5 = \neg Os$. \square

Det er vigtigt at holde sig for øje, at aksiomatiske systemer i princippet opstilles helt adskilt fra den logiske semantik, dvs. semantikken baseret på sandhedsværdier, hvilket også gav anledning til begrebet logisk konsekvens. Men når det kommer til at vurdere det hensigtsmæssige i et givet aksiomatisk system, så bør der naturligvis være en overensstemmelse mellem det aksiomatiske system, og det sprog, den skal forestille at implementere.

Definition 2.10 Et aksiomatisk system $\mathcal{S} = \langle \mathcal{L}, \mathcal{D}, \mathcal{A} \rangle$ siges at være *sundt* såfremt, for enhver mængde af formler Γ og formel ϕ i \mathcal{L} med $\Gamma \vdash_{\mathcal{S}} \phi$, at der gælder $\Gamma \models \phi$.

Et aksiomatisk system $\mathcal{S} = \langle \mathcal{L}, \mathcal{D}, \mathcal{A} \rangle$ siges at være *fuldstændigt* såfremt, for enhver mængde af formler Γ og formel ϕ i \mathcal{L} med $\Gamma \models \phi$, at der gælder $\Gamma \vdash_{\mathcal{S}} \phi$.

Et aksiomatisk system $\mathcal{S} = \langle \mathcal{L}, \mathcal{D}, \mathcal{A} \rangle$ siges at være *afgørligt* såfremt, der eksisterer en procedure, som for enhver Γ og formel ϕ i et endeligt antal skridt afgør, hvorvidt $\Gamma \vdash_{\mathcal{S}} \phi$ er tilfældet eller ej. \square

Man kan læse sundheds-kriteriet som “Hvad der kan bevises er sandt” og fuldstændighedskriteriet som “Hvad der er sandt kan bevises”. I denne læsning synes det oplagt at det kan det give mening at slække på fuldstændighedskriteriet, men ikke på sundheds-kriteriet. Det er i det mindste svært at forestille sig hvad et system, hvori man kan bevise noget sludder, skulle kunne anvendes til.

Et aksiomatisk system kan konstrueres på mange måder svarende til forskellige valg af slutningsregler og logiske aksiomer. F.eks. klare sig uden logiske aksiomer eller klare sig uden med meget på regler og mange aksiomer.

Eksempel 2.14 Vi skal herunder angive et eksempel på et sundt og fuldstændigt aksiomatisk system. Vi modificerer endnu engang systemet \mathcal{S}_1 fra eksempel 2.12 alene ved at ændre på de logiske aksiomer og danner systemet \mathcal{S}_3 omfattende:

\mathcal{L} : et logikprog med konnektorerne \rightarrow og \neg ,

\mathcal{D} : een slutningsregel: modus ponens ,

\mathcal{A} : tre logiske aksiomer

$$A_1 : (\neg\phi \rightarrow \phi) \rightarrow \phi$$

$$A_2 : \phi \rightarrow (\neg\phi \rightarrow \psi)$$

$$A_3 : (\phi \rightarrow \psi) \rightarrow ((\psi \rightarrow \xi) \rightarrow (\phi \rightarrow \xi))$$

Vi lader sundhed og kompletthed hænge i luften som påstande og undlader at føre "bevis" herfor.

Det bemærkes at det begrænsede sprog strengt taget ikke er noget problem. Vi kan jo blot formulere os frit med de fem konnektorer og herefter benytte ækvivalenserne (forkortelserne) fra eksempel 2.6 til omskrivning. \square

Eksempel 2.15 Hvis vi ønsker at danne en teori baseret på \mathcal{S}_3 (fra eksempel 2.12) med skydække-eksemplet: $\Gamma = \{Is \vee Ds \vee Os, \neg(Is \vee Ds)\}$ som ægte aksiomer, må vi først omskrive disse aksiomer til sproget i \mathcal{S}_3 . Ved brug af ækvivalenserne (forkortelserne) fra eksempel 2.6 fås $\Gamma = \{\neg Is \rightarrow (\neg Ds \rightarrow Os), \neg Is, \neg Ds\}$

Med denne omskrivning er $\mathcal{S}_3 + \Gamma$ et eksempel på en teori og f.eks $\neg Ds$ er helt trivielt et eksempel på et teorem. \square

Eksempel 2.16 For at bevise Os (overskyet) i det aksiomatiske system \mathcal{S}_3 beskrevet ovenfor, omskriver vi først $\Gamma = \{Is \vee Ds \vee Os, \neg(Is \vee Ds)\}$ til sproget i \mathcal{S}_3 : $\Gamma = \{\neg Is \rightarrow (\neg Ds \rightarrow Os), \neg Is, \neg Ds\}$. Vi betragter en teori $\mathcal{T} = \mathcal{S}_3 + \Gamma$, dvs. en teori med Γ som ægte aksiomer, og udleder i \mathcal{T} at $\Gamma \vdash Os$ gælder. Beviset for $\Gamma \vdash Os$, i teorien \mathcal{T} , kan forløbe som:

$$\phi_1 = \neg Is \rightarrow (\neg Ds \rightarrow Os); \text{ givet i } \Gamma$$

$$\phi_2 = \neg Is; \text{ givet i } \Gamma$$

$$\phi_3 = \neg Ds; \text{ givet i } \Gamma$$

$$\phi_4 = \neg Ds \rightarrow Os; \text{ fra } \phi_2 \text{ og } \phi_1 \text{ ved modus ponens}$$

$$\phi_5 = Os; \text{ fra } \phi_3 \text{ og } \phi_4 \text{ ved modus ponens}$$

Vi har således godtgjort at Os er et teorem i $\mathcal{S}_3 + \Gamma$. \square

Konsistens svarer som nævnt til det semantiske begreb opfyldelighed, dvs. vi kan afgøre opfyldelighed for en mængde af formler ved i det aksiomatiske system at bestemme konsistens for mængden af formler. Vi vælger her at definere konsistens for en teori.

Definition 2.11 Hvis det for en teori \mathcal{T} gælder for en formel ϕ , at både ϕ og $\neg\phi$ kan udledes, dvs. at både $\vdash \phi$ og $\vdash \neg\phi$ holder, så siges \mathcal{T} at være *inkonsistent*, i modsat fald siges \mathcal{T} at være *konsistent*. \square

Hvis en teori \mathcal{T} er inkonsistent må det altså gælde for mindst en formel ϕ at både ϕ og $\neg\phi$ er med i afslutningen af \mathcal{T} , og vi kan derfor bestemme konsistens for en teori \mathcal{T} ved at fortsætte med at tilføje formler til \mathcal{T} , der kan udledes fra \mathcal{T} , indtil enten at ingen nye formler kan tilføjes \mathcal{T} , eller til vi når til en version af \mathcal{T} hvor både ϕ og $\neg\phi$ er med for en formel ϕ (m.a.o. vi behøver ikke at bestemme den fulde afslutning af \mathcal{T} hvis \mathcal{T} er inkonsistent).

Konsistens for en mængde af formler Γ kan bestemmes ved at tilføje Γ som de ægte aksiomer til en (konsistent) teori \mathcal{T} og herefter at bestemme konsistens for \mathcal{T} .

Eksempel 2.17 At $\Gamma = \{Is \vee Ds \vee Os, \neg(Is \vee Ds), \neg Os\}$ ikke er opfyldelig kan verificeres som følger. Γ omskrives til $\Gamma = \{\neg Is \rightarrow (\neg Ds \rightarrow Os), \neg Is, \neg Ds, \neg Os\}$ og vi lader Γ være de ægte aksiomer i en teori \mathcal{T} . At $\vdash Os$ gælder, følger af beviset fra eksempel 2.16 ovenfor. At $\vdash \neg Os$ gælder følger af Γ , hvori $\neg Os$ forekommer. Derfor må \mathcal{T} være inkonsistent og dermed må Γ være ikke opfyldelig. \square

Bemærk, at konsistens for $\{Is \vee Ds \vee Os, \neg(Is \vee Ds)\}$ ikke kan udledes fra teorier baseret på systemerne eksemplificeret ovenfor. Problemet er at det generelt forholder sig sådan, at et uendeligt antal formler kan udledes fra en teori \mathcal{T} (afslutningen af \mathcal{T} er uendelig), der omfatter logiske aksiomer¹⁴. F.eks. kan vi fra:

$$A_1 : (\neg\phi \rightarrow \phi) \rightarrow \phi$$

udlede:

$$(\neg P \rightarrow P) \rightarrow P$$

$$\neg((\neg P \rightarrow P) \rightarrow P) \rightarrow ((\neg P \rightarrow P) \rightarrow P) \rightarrow ((\neg P \rightarrow P) \rightarrow P)$$

¹⁴Visse – men ikke alle – eksempler på slutningsregler giver anledning til samme fænomen

$$\begin{aligned} & (\neg((\neg((\neg P \rightarrow P) \rightarrow P) \rightarrow ((\neg P \rightarrow P) \rightarrow P)) \rightarrow ((\neg P \rightarrow P) \rightarrow P)) \rightarrow \\ & ((\neg((\neg P \rightarrow P) \rightarrow P) \rightarrow ((\neg P \rightarrow P) \rightarrow P)) \rightarrow ((\neg P \rightarrow P) \rightarrow P))) \rightarrow \\ & ((\neg((\neg P \rightarrow P) \rightarrow P) \rightarrow ((\neg P \rightarrow P) \rightarrow P)) \rightarrow ((\neg P \rightarrow P) \rightarrow P)) \end{aligned}$$

osv.

Nedenfor skal vi se på det såkaldte resolutions-system, der ikke indeholder logiske aksiomer.

2.4 Ræsonnering ved modbevis

Konsistens-begrebet kan anvendes i et velkendt bevisprincip kaldet modbevis. Idéen er som følger. For at undersøge, om en formel ϕ kan udledes fra en mængde af formler Γ , så antages den modsatte, dvs. vi konstruerer et nyt sæt formler $\Gamma' = \Gamma \cup \{\neg\phi\}$ hvorefter vi søger en inkonsistens eller modstrid i Γ . Semantisk er dette begrundet ud fra følgende egenskab, som vi noterede om logisk konsekvens, $\Gamma \models \phi$ hvis og kun hvis $\Gamma \cup \{\neg\phi\} \models \text{falsk}$.

Det semantiske ræsonnement, dvs. ræsonnementet baseret på sandhedsværdier, kan præciseres på følgende måde:

1. Sæt $\Gamma' = \Gamma \cup \{\neg\phi\}$.
2. Undersøg Γ' for opfyldelighed ved at betragte en sandhedstabel for Γ' .
3. Hvis Γ' ikke er opfyldelig, så er ϕ en logisk konsekvens af Γ .

Det tilsvarende princip kan anvendes i forbindelse med aksiomatiske systemer. Antag, vi har et aksiomatisk system og en teori \mathcal{T} , som udvider systemet med et sæt ægte aksiomer Γ . Vores opgave er at bevise, at en given formel ϕ er en logisk konsekvens af Γ .

Modbevisprincippet kan præciseres som følger.

1. Modificer \mathcal{T} til \mathcal{T}' med ægte aksiomer $\Gamma' = \Gamma \cup \{\neg\phi\}$.
2. Undersøg \mathcal{T}' for inkonsistens, f.eks., ved at føre et bevis for $\mathcal{T}' \vdash \text{falsk}$.
3. Hvis \mathcal{T}' er inkonsistent, så er ϕ bevist i \mathcal{T} .

Vi kan tilpasse begreberne af sandhed og afgørighed til modbevisprincippet på følgende måde.

Definition 2.12 Et aksiomatisk system $\mathcal{S} = \langle \mathcal{L}, \mathcal{D}, \mathcal{A} \rangle$ siges at være *sundt* mht. modbevis såfremt, for enhver mængde af formler Γ og formel ϕ i \mathcal{L} med $\Gamma \cup \{\neg\phi\} \vdash_{\mathcal{S}} \text{falsk}$, at der gælder $\Gamma \models \phi$.

Et aksiomatisk system $\mathcal{S} = \langle \mathcal{L}, \mathcal{D}, \mathcal{A} \rangle$ siges at være *fuldstændigt* mht. modbevis såfremt, for enhver mængde af formler Γ og formel ϕ i \mathcal{L} med $\Gamma \models \phi$, at der gælder $\Gamma \cup \{\neg\phi\} \vdash_{\mathcal{S}} \text{falsk}$.

Et aksiomatisk system $\mathcal{S} = \langle \mathcal{L}, \mathcal{D}, \mathcal{A} \rangle$ siges at være *afgørligt* mht. modbevis såfremt, der eksisterer en procedure, som for enhver Γ og formel ϕ i et endeligt antal skridt afgør, hvorvidt $\Gamma \cup \{\neg\phi\} \vdash_{\mathcal{S}} \text{falsk}$ er tilfældet eller ej. \square

Frasen ”mht. modbevis” udelades ofte, når talen er om sundhed m.v. af et aksiomatisk system, som er tænkt udelukkende til modbeviser. Men det er en farlig sprogbrug, da fuldstændighed i den ene forstand ikke nødvendigvis medfører fuldstændighed i den anden. — Der behøves ikke et tilsvarende tilpasning af sundhedsbegrebet (overvej!)

Eksempel 2.18 Hvis vi ønsker at vise at $\neg P$ følger af:

$$\Gamma = \{\neg P \rightarrow Q, P \rightarrow \neg Q, Q\}$$

Så kan vi betragte en teori hvor vi først tager Γ som de ægte aksiomer, f.eks. $\mathcal{T} = \mathcal{S}_3 + \Gamma'$ og herefter vise at $\neg P$ er et teorem. Hertil udvides Γ med $\neg\neg P$ – altså med P – og vi får:

$$\Gamma' = \{\neg P \rightarrow Q, P \rightarrow \neg Q, Q, P\}$$

Nu anvendes modus ponens på $P \rightarrow \neg Q$ og P , hvoraf følger $\neg Q$. Vi udvider således til:

$$\Gamma' = \{\neg P \rightarrow Q, P \rightarrow \neg Q, Q, P, \neg Q\}$$

Da nu Q både forekommer negativt og positivt (Q og $\neg Q$) har vi vist inkonsistens for $\Gamma \cup \{\neg\neg P\}$ og dermed godtgjort at $\neg P$ er et teorem. Dvs. at $\neg P$ følger af Γ . \square

2.5 Resolution

Resolution er en specielt interessant slutningsregel, som ofte benyttes i forbindelse med automatisk bevisførelse. Et resolutions-system er et aksiomatisk system, der består af:

- et sprog med en simpel syntaks: klausul form,
- én slutningsregel: resolution,
- INGEN logiske aksiomer

Resolutions-systemer er specielt indrettet på modbevis-princippet.

2.5.1 Klausulform

Klausulform, sproget i et resolutionssystem, er en mere generel form af det, vi kender fra Prologs klausuler, så navneligheden er ikke tilfældig. Vi vil senere se på en tilsvarende klausulform for første-ordens-logik, herunder Prolog, men i dette afsnit ser vi kun på det propositionelle tilfælde.

Idéen i klausulform er at reducere vilkårlige formler til formler, som stadig bevarer semantikken (\equiv), men kun indeholder de to konnektorer \vee og \neg i passende konstellationer. Dette kommer os så til nytte i konstruktion af et aksiomatisk system, hvor der er langt færre tilfælde at tage højde for.

Definition 2.13 En *klausul* er en disjunktion af literaler. \square

Som eksempel på en klausul, betragt følgende.

$$P \vee \neg Q \vee \neg R \vee S$$

Analogien til Prolog ligger i, at klausuler kan skrives ækvivalent ved en implikation med de positive literaler på den ene side, og atomerne i de negative literalerne på den anden. Formlen ovenfor er ækvivalent med denneher.

$$P \vee R \leftarrow Q \wedge S$$

Dette kan nemt indses ved at opstille en sandhedstabel. Prologklausuler (hvori prædikaterne har nul argumenter) svarer altså til klausuler med netop ét positivt literal.

Vi vil også henregne den tomme disjunktion, som pr. konvention svarer til *falsk*, til klausuler. Det er egentlig blot et kunstgreb, som gør det enklere at formulere resolution, men den kan begrundes intuitivt på følgende måde.

Har vi en disjunktion af tre literaler, f.eks. $A \vee B \vee C$, så kan den være sand på tre måder, ved enten at A , B eller C er det. En disjunktion af to literaler, $A \vee B$, så kan den være sand på to måder, ved enten at A eller B er det. En disjunktion af ét literal, A , så kan den være sand på én måde, nemlig ved at A er sand. Ved en disjunktion bestående af nul literaler, er alt håb ude, den kan være sand på nul måder. Den tomme klausul, altså at fortolke som *falsk* eller modstrid, noterer vi som *falsk*. Man ser også ofte symbolerne \square og $\{ \}$ brugt.

For at bringe en vilkårligt sæt formler på klausulform, benytter vi følgende mellemform.

Definition 2.14 En formel er på *konjunktiv normalform*, hvis den er en konjunktion af (nul eller flere) klausuler. En mængde af formler er på konjunktiv normalform, hvis hver formel er på konjunktiv normalform. \square

Følgende formel er på konjunktiv normalform.

$$(P \vee Q) \wedge (\neg R \vee S) \wedge T$$

Vi har tre konjunker: $(P \vee Q)$, $(\neg R \vee S)$ og T , der alle er klausuler.

Analogt ovenfor, så betragtes også den tomme konjunktion (altså en konjunktion af nul formler – her disjunktioner) som *sand*. Intuitionen kan her støttes af følgende.

Har vi en konjunktion af tre formler, f.eks. $\phi \wedge \psi \wedge \xi$, så er der tre formler der skal være sande for at konjunktionen er sand, nemlig hver af ϕ , ψ , ξ . Tilsvarende er der i $\phi \wedge \psi$, to formler der skal være sande for at konjunktionen er sand og i “konjunktionen” af en formel ϕ er der en formel der må være sand. I “konjunktionen” af nul formler er der ingen formler, der skal være sande for at konjunktionen er sand – altså er konjunktionen af nul formler altid sand.

Som vi tidligere har observeret, er en konjunktion af formler ækvivalent med mængden af konjunkerne, i vores eksempel med følgende.

$$\{P \vee Q, \neg R \vee S, T\}$$

Dvs. kan vi få konverteret et sæt formler over i et ækvivalent sæt på konjunktiv normal form, kan vi også få det over i klausulform.

Enhver formel kan omskrives til konjunktiv normalform ved anvendelse af følgende ækvivalenser,

1. $\phi \leftrightarrow \psi \equiv \phi \rightarrow \psi \wedge \psi \rightarrow \phi$
2. $\phi \rightarrow \psi \equiv \neg\phi \vee \psi$
3. $\neg(\phi \vee \psi) \equiv \neg\phi \wedge \neg\psi$
 $\neg(\phi \wedge \psi) \equiv \neg\phi \vee \neg\psi$
 $\neg\neg\phi \equiv \phi$
4. $\phi \vee (\psi \wedge \xi) \equiv (\phi \vee \psi) \wedge (\phi \vee \xi)$
 $(\phi \wedge \psi) \vee \xi \equiv (\phi \vee \xi) \wedge (\psi \vee \xi)$

Herfra kan vi benytte 1 og 2 (fra venstre mod højre) til at eliminere hhv. \leftrightarrow og \rightarrow . 3 bruges tilsvarende til at reducere virkefeltet for \neg og 4 anvendes til at omskrive til konjunktions af disjunktioner.

Eksempel 2.19 Formlen:

$$P \leftrightarrow (Q \wedge R)$$

bliver med 1 omskrevet til:

$$(P \rightarrow (Q \wedge R)) \wedge ((Q \wedge R) \rightarrow P)$$

og med 2 til:

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

Med 3 fås:

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

og endelig med 4:

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

□

For at omskrive fra konjunktiv normalform til klausulform, skal vi blot opdele hver formel i et antal formler — én svarende til hver konjunkt.

Eksempel 2.20 Fra den konjunktive normalform:

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

opnår vi altså klausulform ved blot at ændre notationen til:

$$\{\neg P \vee Q, \neg P \vee R, \neg Q \vee \neg R \vee P\}$$

altså til en mængde af tre disjunker. □

Fordelen ved klausulform er, at vi opnår en simpel syntaks med kun to logiske konnektorer. Men der er vigtigt at pointere, at vi ikke mister udtrykskraft, forstået på den måde, at ethvert sæt formler, som kan skrives i den generelle propositionslogik, dvs. med alle konnektorer, kan skrives som ækvivalent sæt af formler i klausulform. Ækvivalens af to sæt formler betyder pr. definition, at de hver især udpeger de samme modeller, og at de dermed bevarer det uformelle begreb af information, vi har benyttet tidligere i intuitive forklaringer.

Vi har herved forenklet problemet med at opskrive regler for udledning af nye formler og dermed problemet med at konstruere et aksiomatisk system.

En ulempe ved klausulform kan dog være læseligheden. Den oprindelige form af eksemplet ovenfor, $P \leftrightarrow (Q \wedge R)$, synes mere overskuelig end den klausulform, vi nåede frem til.

Der findes flere forskellige varianter af notationer for klausulform f.eks. som allerede nævnt Prolog's notation. Vi kunne også vælge mængde notation for klausuler, idet rækkefølgen af disjunker i en disjunktion er ligegyldig (Forklar hvorfor). En mængde af formler kan altså noteres som en mængde af mængder af literaler, f.eks. $\{\{\neg P, Q\}, \{\neg P, R\}, \{\neg Q, \neg R, P\}\}$ svarende til eksemplet ovenfor. Imidlertid ønsker vi at fastholde den eksplícitte notation af disjunktionser med \vee -konnektoren af hensyn til læseligheden.

2.5.2 Resolutionsystemer

Et resolutionssystem er et aksiomatisk system $\langle \mathcal{K}, \{\mathcal{R}\}, \emptyset \rangle$ bestående af

- et sprog \mathcal{K} af alle klausuler bygget over en eller anden mængde atomer,
- kun en slutningsregel \mathcal{R} kaldet resolution, som vi præciserer nedenfor,
- den tomme mængde logiske aksiomer.

En teori baseret på resolution er således en udvidelse af et resolutionssystem med et sæt ægte aksiomer i klausulform.

Vi forklarer først resolutionsreglen i, hvad der er at betragte som, en forenklet form og udsætter den generelle definition.

$$\frac{\phi_1 \vee \neg\phi_2 \quad \phi_2 \vee \phi_3}{\phi_1 \vee \phi_3}$$

Vi vil analysere det hensigtsmæssige i denne slutningsregel ved at se på mulige sandhedsværdier for præmisserne. Der er to muligheder for ϕ_2 , enten er den *sand* eller også er den *falsk*, men ved anvendelse af bevisreglen, ved vi ikke hvilken. Vi analyserer først tilfældet $\phi_2 \equiv \textit{sand}$. Præmisserne kan her analyseres som følger,

$$\phi_1 \vee \neg\textit{sand} \equiv \phi_1 \vee \textit{falsk} \equiv \phi_1, \text{ og}$$

$$\textit{sand} \vee \phi_3 \equiv \textit{sand}.$$

Så hvis de to præmisser er sande og ϕ_2 er *sand*, så kan vi uddrage, at ϕ_1 må være sand, mens der ikke er information om ϕ_3 . Den anden mulighed, $\phi_2 \equiv \textit{falsk}$, leder frem til, at ϕ_3 må være sand, og ϕ_1 kan vi ikke sige noget om. Men da vi under brug af en slutningsregel ikke kan referere til sandhedsværdier, er den konklusion, vi på sund måde kan uddrage af de to præmisser netop $\phi_1 \vee \phi_3$.

Det er klart at reglen kan anvendes på $P \vee \neg Q$ og $Q \vee R$ med slutningen $P \vee R$, men kan den bruges på f.eks. de to klausuler $P \vee \neg Q$ og $R \vee Q$? Næ – ikke med mindre vi først omskriver sidste klausul til $Q \vee R$.

Problemet kan klares ved i stedet at vælge mængde notation for klausuler. Med reglen

$$\frac{\{\phi_1, \neg\phi_2\} \quad \{\phi_2, \phi_3\}}{\{\phi_1, \phi_3\}}$$

sluttes jo $\{Q, R\}$ både fra de to klausuler $\{P, \neg Q\}$ og $\{Q, R\}$ og fra klausulerne $\{P, \neg Q\}$ og $\{R, Q\}$, ganske enkelt fordi klausulerne er parvis identiske. Vi har altså gjort sagerne mere indviklede ved vores valg af notation.

En mulighed er nu blot at tage lidt løst på formaliteterne ved at skrive klausuler eksplicit som disjunktionser, men at læse disse som om de var mængde noterede. Vi skal dog for fuldstændighedens skyld herunder angive den generelle form af slutningsreglen til den valgte notation.

Definition 2.15 Slutningsreglen kaldet *resolution*, ser således ud.

$$\frac{\phi_1 \vee \cdots \vee \phi_n \quad \psi_1 \vee \cdots \vee \psi_k}{\phi_1 \vee \cdots \vee \phi_{i-1} \vee \phi_{i+1} \vee \cdots \vee \phi_n \vee \psi_1 \vee \cdots \vee \psi_{j-1} \vee \psi_{j+1} \vee \cdots \vee \psi_k},$$

hvor $\phi_i = \neg\psi_j$

Såfremt en klausul γ er konklusion ved anvendelse af resolution på klausuler α og β , siger vi, at α og β *resolverer* med *resolventen* γ . \square

Vi betragter det specialtilfælde, hvor de to præmisses indeholder netop et literal,

$$\frac{\neg\phi \quad \phi}{\text{falsk}}$$

Konklusionen er her (hvis vi tæller de små indices i definitionen) den tomme disjunktions, som vi definerede ækvivalent med *falsk* svarende til modstrid. Dette forekommer ganske hensigtsmæssigt, da et sæt formler af formen $\{\neg\phi, \phi\}$ ikke er opfyldeligt.

Det aksiomatiske system bestående af resolutionsreglen, kan man nemt overbevise sig om, er sundt. Det kan bevises ved en argumentation svarende til vor intuitive analyse af den forenklede regel ovenfor. Til gengæld er det ikke fuldstændigt i den strenge forstand (se definition 2.10). Det ses ved at betrage et eksempel.

Betragt følgende mængde af klausuler $\Gamma = \{P, Q\}$; ud fra disse kan man ikke vise nogen nye formler ved resolution, men de har andre klausuler som logisk konsekvens, f.eks. $P \vee Q$.

Det nyttige ved resolution ligger ene og alene i dens anvendelse i en bevis-procedure baseret på modbevis, idet vi kan vise følgende.

Sætning 2.1 Lad Γ være en mængde af klausuler, som ikke er opfyldelig, og \mathcal{T} teorien baseret på resolution med Γ som ægte aksiomer. Da indeholder afslutningen af \mathcal{T} den tomme klausul. \square

Vi kan således formulere spørgsmålet om logisk konsekvens indenfor propositionslogik vha. resolution. Betragt en arbitrær mængde af formler Γ og en formel ϕ og betragt spørgsmålet

”Er ϕ logisk konsekvens af Γ , dvs. gælder der $\Gamma \models \phi$?”

Vi kan omskrive dette i følge, hvad vi allerede ved, til¹⁵

”Er $\Gamma \cup \{\neg\phi\}$ uopfyldelig, dvs. gælder der $\Gamma \cup \{\neg\phi\} \models \text{falsk}$?”

og endelig til dette.

”Gælder der, at afslutningen af $\mathcal{T}_{\Gamma \cup \{\neg\phi\}}$ indeholder den tomme klausul, hvor $\mathcal{T}_{\Gamma \cup \{\neg\phi\}}$ er et teori baseret på resolution med ægte aksiomer, som er en omskrivning af $\Gamma \cup \{\neg\phi\}$ til klausulform?”

Vi mangler så blot en procedure, som afgør, hvorvidt teorier baseret på resolution er konsistente eller ej.

Benytter vi resolution som beskrevet, kan vi komme ud for, at afslutningen er uendelig. Betragt for eksempel mængden af klausuler $\{\neg Q, Q \vee Q \vee \neg Q\}$. Ved gentagen brug af resolution kan man resolve sig frem til vilkårlige store klausuler af formen $Q \vee \dots \vee Q \vee \neg Q$. Men det er klart (ud fra en semantisk overvejelse), at gentagelser af literal i en klausul ikke tilføjer ny information, og tilsvarende at to klausuler er ækvivalente, hvis de indeholder de samme literaler (i forskellig rækkefølge).

Også her kan vi løse problemet ved at skifte til mængdenotation for klausuler eller ved blot at indføre ”læsning” af klausuler som mængder. For et givet sæt af n atomer, er antallet af klausuler endeligt, nærmere bestemt 2^{2n} .¹⁶

Vi kan verificere opfyldelighed for en mængde af klausuler ved at undersøge om afslutningen indeholder den tomme klausul ved følgende simple procedure. Givet en mængde af klausuler Γ (i mængdeform):

1. Hvis Γ indeholder den tomme klausul, STOP, Γ er inkonsistent.
2. Hvis ingen klausuler i Γ resolverer med en resolvent, som ikke allerede er med i Γ , STOP, Γ er konsistent.
3. Tilføj en ny resolvent af to klausuler i Γ til Γ ; gå til 1).

Som nævnt føres et modbevis som undersøgelse af konsistens for den modsatte antagelse. For at udlede en klausul ϕ fra en mængde af klausuler Γ , udvider vi

¹⁵Bemærk, at omskrivningen her holder, hvad enten Γ i sig selv er opfyldelig eller ej.

¹⁶Konstruer argumentet for, at der netop findes dette antal mulige klausuler.

altså blot til $\Gamma' = \Gamma \cup \{\neg\phi\}$, hvorefter vi udfører ovennævnte konsistenskontrol. Er Γ' inkonsistent, så er ϕ en logisk konsekvens af Γ .

Denne procedure viser, at resolution i tilfældet propositionslogik er afgørlig: Hvert gennemløb af løkken forøger mængden Γ , og mængden af mulige klausuler (i mængdeform) over et endeligt antal givne atomer er i sig selv endeligt. Ergo må algoritmen terminere.

Eksempel 2.21 At Q er et teorem i en teori med de ægte aksiomer:

$$\Gamma = \{P, \neg P \vee Q\}$$

kan godtgøres ved modbevis som følger:

$$\begin{aligned}\Gamma' &= \{P, \neg P \vee Q, \neg Q\} \\ \Gamma' &= \{P, \neg P \vee Q, \neg Q, Q\} \\ \Gamma' &= \{P, \neg P \vee Q, \neg Q, Q, \textit{falsk}\}\end{aligned}$$

Ergo: Q er bevist.

Først tilføjes $\neg Q$ i trin 1), herefter tilføjes Q , idet Q fremkommer som resolvent af P og $\neg P \vee Q$, og endelig tilføjes den tomme klausul, *falsk*, som resolventen af Q og $\neg Q$. \square

Eksempel 2.22 Vi runder af med skydække-eksemplet og godtgør ved modbevis, at O_s er et teorem. Klausul-formen af $\Gamma = \{I_s \vee D_s \vee O_s, \neg(I_s \vee D_s)\}$ er $\Gamma = \{I_s \vee D_s \vee O_s, \neg I_s, \neg D_s\}$.

$$\begin{aligned}\Gamma' &= \{I_s \vee D_s \vee O_s, \neg I_s, \neg D_s, \neg O_s\} \\ \Gamma' &= \{I_s \vee D_s \vee O_s, \neg I_s, \neg D_s, \neg O_s, D_s \vee O_s\} \\ \Gamma' &= \{I_s \vee D_s \vee O_s, \neg I_s, \neg D_s, \neg O_s, D_s \vee O_s, O_s\} \\ \Gamma' &= \{I_s \vee D_s \vee O_s, \neg I_s, \neg D_s, \neg O_s, D_s \vee O_s, O_s, \textit{falsk}\}\end{aligned}$$

Ergo: O_s er bevist.

\square

3 Første-ordens prædikatlogik

Vi skal nu generalisere propositionslogikken, således at atomerne kan tage argumenter, ved at tilføje prædikat- og funktions-symboler. Hvor propositionslogikkens grundlæggende udsagn eller atomer er "monolitiske", som f.eks. "det regner", udvider vi nu, så atomerne har en struktur, f.eks. "vejr(dato(20,marts,1995), kreta, 19, skyfrit)".

Dette giver en langt større udtrykskraft, da vi kan parameterisere med variable og eksempelvis i aksiomerne beskrive egenskaber ved vejret, som ikke nødvendigvis er bundet til et bestemt tidspunkt og klokkeslet. Prisen er så selvfølgelig, at semantikken bliver mere kompliceret (men dog håndterlig).

Første-ordens prædikatlogik har stor betydning bl.a. som et grundlag for databaser, logikprogrammering og ekspertsystemer.

Ambitionen med dette kapitel er kun en meget kortfattet introduktion, der bygger på intuitionen, som er opbygget i kapitel 2.

3.1 Første-ordens prædikatlogiksprøg

Første-ordens logikken drejer sig, som propositionslogikken, om sproget af formuler opbygget efter en syntaksbeskrivelse og med en semantik defineret ud fra sandhedsværdier.

3.1.1 Syntaks

Symbolerne i et første-ordens prædikatlogiksprøg kan opdeles i følgende kategorier.

1. prædikatsymboler, hver med given aritet¹⁷, der er ≥ 0 .
2. funktionssymboler, hver med given aritet ≥ 0 ; et funktionssymbol med aritet 0 kaldes almindeligvis en konstant.
3. en mængde af variable.
4. kvantorer (typisk \forall og \exists).
5. konnektorer.
6. parenteser og komma.

¹⁷ariteten for et prædikat/en funktion, er antallet af argumenter til prædikatet/funktionen

Kvantoren \forall kaldes alkvantor og \exists for eksistenskvantor. Når udvalget af symboler er fastlagt, defineres mængderne af termer, atomer, literaler og formler rekursivt på følgende måde (i det vi antager de sædvanlige konnektorer \wedge , \vee , \neg , \rightarrow , \leftrightarrow).

- En konstant er en term.
- En variabel er en term.
- Hvis f er et funktionssymbol af aritet $n > 0$ og t_1, \dots, t_n er termer, så er $f(t_1, \dots, t_n)$ en term.
- Hvis p er et prædikatsymbol af aritet $n > 0$ og t_1, \dots, t_n er termer, så er $p(t_1, \dots, t_n)$ et atom.
- Hvis A er et atom, så er A og $\neg A$ literaler.
- Et atom er en formel.
- Hvis ϕ er en formel, så er (ϕ) og $\neg\phi$ formler.
- Hvis ϕ og ψ er formler, så er $\phi \vee \psi$, $\phi \wedge \psi$, $\phi \rightarrow \psi$ og $\phi \leftrightarrow \psi$ formler.
- Hvis K er en kvantor, x en variabel og ϕ en formel, så er $Kx\phi$ en formel.

Følgende er et eksempel på en formel.

$$p(x) \wedge \forall x(\exists y(\neg q(f(y, a) \vee (\forall x r(z))))))$$

De to udtryk $\forall x \exists y p(x) \wedge y$ og $\forall x \exists y y(x)$ er ikke formler.

På det syntaktiske niveau tjener kvantorer som erklæringer af variable, hvilket vi præciserer med følgende terminologi; igen en kompakt og rekursiv definition. Betragt en formel $Kx\phi$, hvor K er en kvantor, x en variabel og ϕ en formel. En forekomst af x i ϕ siges at være *bundet* ved (den markerede forekomst af) kvantoren K medmindre den er bundet af en kvantor i ϕ . Betragt eksempelvis følgende formel.

$$\forall x(p(x) \wedge (\exists x(\forall y q(x, y))))$$

Forekomsten af x i $p(x)$ er bundet af den yderste kvantor. I atomet $q(x, y)$ er x bundet af $\exists x$ og y af $\forall y$. En forekomst af en variabel, som ikke er bundet af en kvantor siges at være *fri*; en formel er *åben* hvis den har frie variable, ellers *lukket*.

Formlen overfor er således et eksempel på en lukket formel (alle variable er ”erklæret”). Formlen $\forall x(\text{elsker}(x, y))$ er åben.¹⁸ Man benytter skrivemåden $\forall x_1, x_2, \dots, x_n \phi$ som kort form for $\forall x_1 \forall x_2 \dots \forall x_n \phi$; notationen $\forall \phi$ som generel skrivemåde for formelen $\forall x_1, x_2, \dots, x_n \phi$, hvor x_1, \dots, x_n er de frie variable, som forekommer i ϕ . Tilsvarende for skrivemåderne $\exists x_1, x_2, \dots, x_n \phi$ og $\exists \phi$.

Endelig siger vi, at en formel eller en term er *grundet* hvis den ikke indeholder variable.

3.1.2 Semantik

Semantikken er defineret helt analogt til semantikken for propositionslogiske sprog, blot udvidet passende til at håndtere konstanter, variable, funktionssymboler og kvantorer. Bemærk, at fortolkning af funktionssymboler er temmelig generel.

Definition 3.1 En *fortolkning* for et første-ordens prædikatlogikprog består af følgende.

- En mængde U kaldet et *univers*.
- For hver konstant i sproget, en tilknytning af et element i U .
- For hvert funktionssymbol af aritet n , en funktion fra U^n over i U .
- For hvert prædikatsymbol af aritet n , en funktion fra U^n over i $\{\text{sand}, \text{falsk}\}$.

□

Umiddelbart giver en fortolkning en sandhedsværdi til grundede atomer på følgende måde.

- Værdien af en konstant fås direkte af fortolkningen.
- Værdien af en term $f(t_1, \dots, t_n)$ fås ved at anvende den funktion F , som fortolkningen tillægger f , på værdierne af t_1, \dots, t_n .
- Sandhedsværdien for et atom $p(t_1, \dots, t_n)$ fås ved at anvende den sandhedsfunktion P , som fortolkningen tillægger p , på værdierne af t_1, \dots, t_n .

En fortolkning kaldes også for en struktur¹⁹, da den del, der vedrører termer, svarer til det, vi kender som en algebraisk struktur.

¹⁸Ja, hva’ hulen er y for noget? Bemærk analogien til en ”pointer” i Pascal, som peger ud i luften, en såkaldt ”dangling pointer”.

¹⁹Hvilket, ak, deler navn med et andet begreb, vi bl.a. kender fra Prolog, som er noget helt, helt andet.

Eksempel 3.1 Betragt et sprog, med konstantsymboler $0, 1, 2, \dots$ og et binært funktionssymbol $+$, som skrives ved infiks notation, samt prædikatsymboler $\text{lige}(-)$ og $=$; den sidste ved infiks notation. Vi betragter en fortolkning \mathcal{N} med univers svarende til de naturlige tal, hvor eksempelvis symbolet 2 afbildes over i det naturlige tal to, symbolet $+$ over i en funktion, som afbilder to naturlige tal over i deres sum; \mathcal{N} knytter til prædikatsymbolet $\text{lige}(-)$, en funktion, som afbilder naturlige tal, som er multipla af to over i *sand*, alle andre naturlige tal over i *falsk*. Lighedstegnet fortolkes på sædvanlig måde. I fortolkningen \mathcal{N} gælder, at det grundede atom $\text{lige}(2 + 2)$ er sandt; det grundede atom $2+2=4$ er også sandt.

Vi kan betragte en anden fortolkning \mathcal{P} med univers svarende termer i Prolog, hvor eksempelvis 2 afbildes over i Prolog-konstanten 2 , hvor $+$ afbildes over i en funktion, som til to termer t_1 og t_2 tilknytter strukturen $+(t_1, t_2)$; \mathcal{P} knytter til prædikatsymbolet $\text{lige}(-)$, en funktion, som afbilder $0, 2, 4, \dots$ over i *sand*, alle andre Prolog-konstanter over i *falsk*. \mathcal{P} 's funktion for $\text{lige}(-)$ afbilder yderligere en term $+(t_1, t_2)$ over i *sand* hvis og kun hvis t_1 og t_2 enten begge afbildes over i *sand* eller begge over i *falsk*. Lighedstegnet afbildes over i en funktion, som returnerer *sand*, netop når den anvendes på to identiske termer. I fortolkningen \mathcal{P} gælder, at det grundede atom $\text{lige}(2 + 2)$ er sandt; det grundede atom $2+2=4$ er her falskt.

Vi betragter en tredje fortolkning \mathcal{X}_0 med univers $\{\text{høstblomst}\}$, hvor ethvert konstant symbol afbildes over i *høstblomst*, hvor funktionen svarende til $+$ afbilder $\langle \text{høstblomst}, \text{høstblomst} \rangle$ over i *høstblomst*, og endelig at funktionen til $\text{lige}(-)$ afbilder *høstblomst* over i *falsk*. Vi lader her $=$ altid returnere *falsk*. I fortolkningen \mathcal{X}_0 gælder, at det grundede atom $\text{lige}(2 + 2)$ er falskt; det grundede atom $2+2=4$ er tillige falskt.

Endelig definerer vi \mathcal{X}_1 , som er identisk med \mathcal{X}_0 , bortset fra, at $\text{lige}(-)$ og $=$ ændres til at stå for en funktioner, som afbilder alt over i *sand*. I fortolkningen \mathcal{X}_1 gælder, at det grundede atom $\text{lige}(2+2)$ er sandt; det grundede atom $2+2=4$ er her sandt. \square

At knytte en sandhedsværdi til en åben formel giver kun mening, når vi kender variablenes værdier. Det, at knytte værdier til variable, præciseres i følgende begreb af en tildeling.

Definition 3.2 Givet en logisk sprog og en fortolkning med univers U , så forstås ved en *tildeling* en afbildning fra en mængde variable til værdier i U . Hvis T er en tildeling, så noterer $T[x \mapsto v]$ en tildeling svarende til T undtagen, at variabelen x nu afbildes over i den nye værdi $v \in U$. \square

Givet en fortolkning og en tildeling fås på naturlig måde også sandhedsværdier for ikke-grundede atomer. Hvad angår konnektorerne, så defineres de på præcis samme måde som i propositionslogikken; se afsnit 2.1.2.

For at kunne bestemme sandhedsværdien for en vilkårlig formel i førsteordens prædikatlogik, skal vi blot tilføje følgende, som definerer betydningen af kvantorerne.

Givet en fortolkning med univers U og en tildeling T , så gælder

- En formel $\forall x\phi$ er sand under T såfremt, der for enhver værdi $v \in U$ gælder, at ϕ er sand under $T[x \mapsto v]$; i modsat fald er den falsk.
- En formel $\exists x\phi$ er sand under T såfremt, der findes en værdi $v \in U$, så ϕ er sand under $T[x \mapsto v]$; i modsat fald er den falsk.

Ser vi på lukkede formler, så giver det mening, at tale om, hvorvidt de er sande eller falske i forhold til en given fortolkning. Vi behøver ikke at antage en ”ekstern” tildeling, da samtlige variable er dækket af en kvantor.

Eksempel 3.2 Betragt formelen $\text{lige}(x+2)$ i sproget beskrevet i eksempel 3.1. Den er sand under fortolkning \mathcal{N} med tildelingen $[x \mapsto \text{to}]$, ditto under \mathcal{P} med tildelingen $[x \mapsto 2]$.

Følgende kvantificerede formler er sande i såvel \mathcal{N} , \mathcal{P} og \mathcal{X}_1 , $\exists x \text{lige}(x+2)$ og $\forall x \text{lige}(x+x)$. Under \mathcal{X}_0 er de begge falske. \square

Vi kan nu definere et modelbegreb for førsteordens prædikatlogik.

Definition 3.3 Givet et førsteordens prædikatlogikssprog, så er en *model* for en mængde lukkede formler Γ en fortolkning, der gør alle formler i Γ sande. \square

Eksempel 3.3 Vi betragter igen sproget og de tre fortolkninger fra de foregående eksempler, og lad

$$\Gamma = \{\forall x, y(\text{lige}(x+y) \leftrightarrow (\text{lige}(x) \leftrightarrow \text{lige}(y)))\}.$$

Da er \mathcal{N} , \mathcal{P} og \mathcal{X}_1 modeller for Γ .

Betragt nu

$$\Gamma' = \Gamma \cup \{\text{lige}(0), \neg \text{lige}(1), \text{lige}(2), \neg \text{lige}(3), \dots\}.$$

Kun \mathcal{N} og \mathcal{P} er modeller for Γ' . \square

Følgende egenskaber er defineret helt på samme måde som i det propositionelle tilfælde.

Definition 3.4 En *gyldig formel* ϕ er en formel, der er sand i enhver fortolkning (dvs. enhver fortolkning for ϕ er en model for ϕ). En gyldig formel kaldes også en *tautologi*.

To formler siges at være *ækvivalente*, hvis de har samme sandhedsværdi i enhver fortolkning. Dette noteres $\phi \equiv \psi$. \square

Eksempel 3.4 For vilkårligt sprog og vilkårlig formel ϕ heri gælder ækvivalenserne. $\neg(\forall x\phi) \equiv \exists x(\neg\phi)$ og $\neg(\exists x\phi) \equiv \forall x(\neg\phi)$.

Ækvivalenserne noteret under propositionslogikken, gælder også i førsteordens prædikatlogik. \square

En klasse af fortolkninger (og modeller), som er særligt vigtige i forbindelse med automatiseret bevisførelse, er Herbrand-fortolkninger (og modeller), som baserer sig på et univers af termer. Værdien af en konstant er konstanten selv og funktionen knyttet til et funktionssymbol f , er symbolet selv betragtet som en term-konstruktor. Herbrand-modeller udtrykker en semantik, som er baseret alene på syntaktiske størrelser (termer), og deri ligger relevansen i forhold til datamaskiner. Vi skal blot være helt klar på, at vi ikke umiddelbart kan (eksempelvis) begrænse os til at se på Herbrand-modeller bare fordi de nu engang er bekvemme. Vi vender tilbage til Herbrand-modeller senere i forbindelse med begrænsede logikprog, hvor vi kan gøre denne indskrænkning.

Eksempel 3.5 Betragt igen eksempel 3.3. Fortolkningen \mathcal{P} , som er model for såvel Γ som Γ' er ikke en Herbrand-model, men den er nært beslægtet med en sådan.

Vi kan definere en Herbrand-model \mathcal{H} helt analogt til \mathcal{P} ved at udskifte ”Prolog-termer” med ”termer i det logiske sprog”. \square

Det kan også være relevant at fiksere betydningen af visse symboler. Hvis vi f.eks. interesserer os for sprog, som indeholder konstanter $0, 1, 2, \dots$, funktionssymbolet $+$ og prædikaterne $\text{lige}(-)$ og \geq . Her kunne vi vælge at betragte kun fortolkninger, hvis univers indeholdt de naturlige tal, og hvor fortolkningen af de nævnte symboler skal være, som vi har lært det i skolen — og derved slippe for at skulle opskrive aksiomer for dem. Definitionerne af gyldighed m.v. skal så generaliseres derefter, og der er ingen væsentlige tekniske problemer forbundet med dette.

Hvad med åbne formler?

Ovenfor definerede vi sandhedsværdier af lukkede formler i forhold til en given fortolkning. Det, at kunne klassificere en lukket formel som sand eller falsk, er en intuitivt fornuftig måde til tillægge den en betydning.

Men hvad med åbne formler? For at kunne håndtere lukkede formler ovenfor, benyttede vi et sandhedsbegreb, som yderligere er relativt til en tilskrivning, for at håndtere åbne del-formler. Men hvad hvis vi betragter en åben formel isoleret, hvilken fornuftig mening kan vi tillægge den. Vi betragter et eksempel.

$$\phi = p(x, y) \wedge \neg q(y)$$

Antager vi en fortolkning til stede og en helt bestemt tilskrivning, får vi en sandhedsværdi, men omvendt, så bestemmer formlen en ny relation om mulige værdier for x og y . Vi kan definere en binær relation over fortolkningens domæne, bestående af de par $\langle v_x, v_y \rangle$, hvor ϕ er sand i forhold til skrivningen $[x \mapsto v_x, y \mapsto v_y]$.

En åben formel er analog til et polynomium indenfor matematikken, f.eks. $x^2 + y^2 + 2xy$, som definerer en funktion i to variable, $f(x, y) = x^2 + y^2 + 2xy$. Tilsvarende kan vi definere et nyt prædikat r svarende til vor åbne term ovenfor,

$$\forall (r(x, y) \leftrightarrow p(x, y) \wedge \neg q(y)).$$

Hvis vi har aksiomer (defineret nedenfor), som bestemmer prædikaterne $p(-, -)$ og $q(-)$, kan vi blot tilføje denne formel som et nyt aksiom, som definerer $r(-, -)$. I dette eksempel ligger en klar analogi til procedurdefinitioner i traditionelle programmeringssprog.

Eksempel 3.6 Vi fortsætter eksemplerne ovenfor og betragter den åbne formel

$$\phi = \exists x (y = (x + x) + (x + 7))$$

Vælger vi fortolkningen \mathcal{N} , ser vi, at tilskrivningerne $[x \mapsto \text{syv}]$, $[x \mapsto \text{ti}]$, $[x \mapsto \text{tretten}]$, osv. gør ϕ sand under \mathcal{N} .

Vi kan udvide vort sprog med et nyt prædikatsymbol syvplustretre og betragte formlen $\psi = \forall y (\text{syvplustretre}(y) \leftrightarrow \phi)$. Vi opnår en model for $\Gamma' \cup \{\psi\}$ ved at udvide \mathcal{N} med en fortolkningsfunktion for syvplustretre, som sender syv, ti, tretten, osv. over i *sand*, alle andre naturlige tal over i *falsk*. \square

Kvantorer og det (u)nævnelige

Bemærk, at eksistens- og alkvantoren defineres ved, at der findes passende mængder af værdier i universet U . Dette er væsensforskelligt fra en definition, hvor en tildeling var en afbildning fra variable til termer (hvis værdi, man så skulle regne ud). Det kan jo være tilfældet, at universet indeholder værdier,

som ikke kan beskrives i sproget, dvs. at der ikke findes grundede termer, som noterer dem (dvs. har dem som værdi).

Givet et sprog og en fortolkning F med univers U , så siger vi at et element $u \in U$ er *denoterbart*, hvis der findes en grundet term t , hvis værdi under F er lig med u ; i såfald siger vi, at t *denoterer* u . Vi så eksempler ovenfor på sprog og model, hvor det naturlige tal 'to' kan denoteres ved 2 og det reelle tal, som er kvadratroden af to, ikke er denoterbart.

Eksempel 3.7 Betragt igen det gennemgående eksempelsprog med naturlige talkonstanter og $+$. Vi lader sproget uændret og udvider modellen \mathcal{N} ved at universet nu også indeholder negative hele tal, og funktionerne for $+$ og lige($-$) udvides på naturlig måde; vi kalder denne udvidede model for \mathcal{Z} . Betragt nu formelen

$$\exists z (z + 2 = 0).$$

Denne formel er klart sand i \mathcal{Z} , fordi der findes en fortolkning, som tilordner *værdien* nemlig minus-to til z , som gør indmaden sand. Men der eksisterer ikke nogen term, som under \mathcal{Z} har værdien minus 2. (Thi sproget indeholder hverken en konstant -2 eller et monadisk eller binært funktionssymbol ' $-$ '.) Altså, på trods af, at formelen $\exists z (z + 2 = 0)$ er sand, så findes der ingen term t , for hvilken $t + 2 = 0$ er sand.

Man kunne så fortsætte med at udvide sproget med et minus-funktionssymbol, dernæst definere en multiplikation \times og så studere modeller for aksiomer som $\forall x \exists z (z \times x = x)$, hvor man skal huske på, at relevante værdier for z også indeholder to og minus-en. (Og derved bliver man tvunget til at genopdage en væsentlig del af ældre og nyere matematik). \square

Vi bemærker specielt Herbrand-fortolkninger. De har den særlige egenskab, at ethvert element i universet kan denoteres (nemlig ved sig selv), og hvert element kan denoteres på netop én måde. Tag for eksempel elementet $f(g(a), b)$; det denoteres ved termen $f(g(a), b)$ og ikke andre.

3.1.3 Entydig læsning af formler

Det eneste vi har brug for her, er at indflette kvantorerne i hierarkiet. Kvantorerne indplaceres med højeste prioritet sammen med negation.

$$\neg, \forall, \exists$$

$$\wedge$$

\vee

$\rightarrow, \leftrightarrow$

3.2 Ræsonnering med sandhedsværdier

Følgende begreber går igen fra det propositionelle tilfælde, idet vi husker på at model og sandhed kun gjaldt for lukkede formler.

Definition 3.5 En mængde af lukkede formler Γ siges at være *opfyldelig*, hvis Γ har en model.

En lukket formel ϕ er en *logisk konsekvens* af en mængde lukkede formler Γ , hvis den er opfyldt i enhver model for Γ , dvs. hvis den er sand i enhver fortolkning, der opfylder Γ . Logisk konsekvens noteres $\Gamma \models \phi$. \square

I propositionslogikken kunne vi afgøre, om ϕ er logisk konsekvens af Γ ved at studere mængden af alle modeller for Γ . Denne mængde var endelig og kunne karakteriseres vha. sandhedstabeller. Men prædikatlogikkens modeller er langt mere generelle, og findes der en model, så findes der også uendeligt mange modeller (man kan få en ny model ved at erstatte universet med en anden isomorf mængde, eller ved at udvide med nye, ikke-denoterbare værdier).

Eksempel 3.8 Vi betragter igen det gennemgående eksempel. Formlen $\forall x (x = x)$ er sand i modellerne \mathcal{N} og \mathcal{P} ; formelen $2 + 2 = 4$ er sand i \mathcal{N} , men ikke i \mathcal{P} .

Formlen $2 + 2 = 2 + 2$ er en logisk konsekvens af $\forall x (x = x)$, men $2 + 2 = 4$ er det ikke. \square

I førsteordensprædikatlogik gælder de samme egenskaber om logisk konsekvens som i propositionslogikken, samt nogle flere, som involverer kvantorer. Disse kan slås op i en bog, og vi kommer ikke nærmere ind på dem her.

3.3 Aksiomatiske systemer

Vi kan her genbruge alle definitioner af slutningsregler, aksiomer, aksiomatiske systemer, teorier og beviser, hvor vi blot holder os til lukkede formler. Ditto med begreberne om sundhed, fuldstændighed og afgørlighed.

Eksempel 3.9 Betragt et simpelt logisk sprog opbygget af konstanter a, b, \dots , monadiske prædikatsymboler p, q, \dots , og hvor formlerne enten er atomer eller af formen $\forall(\phi \rightarrow \psi)$, hvor ϕ og ψ er atomer. Vi kan opbygge et aksiomatisk system bestående af modus ponens (præcis den samme som for propositionslogikken) og følgende, vi kan kalde for *instantiering*.

$\frac{\forall\phi}{\phi'}$ hvor ϕ' fås fra ϕ ved at proppe grundtermer ind for variablene.

Et sådant system er sundt, men ikke fuldstændigt. Det er afgørligt i den generelle forstand. Til gengæld, kan det (med passende modifikation af begreberne) påstås at være fuldstændigt og afgørligt med hensyn til opgaven at bestemme mængden af grundede atomer, som er logisk konsekvens af en samling ægte aksiomer Γ . \square

Der findes aksiomatiske systemer for første-ordens prædikatlogik, som kan vises at være sunde og fuldstændige; der kan henvises til systemer, som tilskrives Gentzen og Hilbert. Det skal dog bemærkes, at aksiomerne i disse systemer forekommer ret groteske og ikke særligt intuitive.

I propositionslogikken kunne man opstille bevisprocedurer, som var afgørlige, men med første-ordens prædikatlogikken forholder det sig anderledes. Det kan vises, at selv om vi begrænser os til et logisk sprog, som kun indeholder

- én konstant c ,
- to funktionssymboler $f(-)$ og $g(-)$,
- ét binært prædikatsymbol $p(-, -)$,

så kan der ikke opstilles fuldstændige bevisprocedurer. (Den ihærdige læser kan more sig med at finde ud af, hvordan en vilkårlig Turing-maskine kan indkodes som ægte aksiomer vha. ovennævnte symboler, og hvordan man kan formulere et vilkårligt standsningsproblem). Med andre ord, der findes udsagn ϕ , man hverken kan bevise eller modbevise på trods af, at en sådan ϕ vil være sand eller falsk.

3.4 Ræsonnering ved modbevis

Alle begreber går igen fra propositionslogikken. Der gælder $\Gamma \models \phi$ hvis og kun hvis $\Gamma \cup \{\neg\phi\} \models \text{falsk}$, og følgelig, hvis vi har et deduktivt system, som måske ikke er egnet til at vise vilkårlige ϕ 'er, kunne det måske være det kunne bruges til at bevise *falskhed* ud fra samlinger af formler i passende format.

3.5 Resolution

Der gælder også i prædikatlogikken, at enhver formel kan omskrives til en (passende generaliseret) klausulform med færre konnektiver indblandet. Bevismetoden modbevis-ved-resolution kan også generaliseres.

3.5.1 Klausulform

Definitionerne er som før, blot skal der være mulighed for at kvantificere.

Definition 3.6 En *klausul* er af formen $\forall\phi$, hvor ϕ er en disjunktion af literaler. \square

For nemheds skyld udelades alkvantoren ofte i en klausul (i en kontekst, hvor alle ved, at talen kun er om klausuler), enhver variabel er alkvantoriseret.²⁰

Der gælder også her, at enhver samling lukkede formler kan omskrives til konjunktiv normal form eller ækvivalent med en mængde af klausuler. — Her benyttes ækvivalenserne fra afsnit 2.5.1 sammen med følgende til at få negationstegnene ind bag kvantorerne, $\neg(\forall x\phi) \equiv \exists x(\neg\phi)$ og $\neg(\exists x\phi) \equiv \forall x(\neg\phi)$. Derudover benyttes en ækvivalens, som handler om ombenævning af variable, så hver forekomst af en kvantor får sin egen variabel, og endelig en teknik kaldet skolemisering (efter Skolem), der vedrører indførelse af specielle funktionssymboler til at indfange hvad der svarer til eksistenskvantorisering i klausulformen (hvor alle variable er al-quantoriserede). De interesserede opfordres til at konsultere litteraturen. Vi vil ikke her komme nærmere ind på omformning til klausulform, men blot bemærke at det er en triviel og ikke særligt betydningsfuld diciplin. Når det er bekvemt skal vi herefter blot antage formler som givet på klausulform.

3.5.2 Særlige egenskaber ved klausuler

Klausuler har den helt særlig egenskab, at vi kan begrænse os til at se på Herbrand-modeller, når vi studerer opfyldelighed. Vi har tidligere nævnt Herbrand-modeller; vi opsummerer her definitionen, og det centrale resultat.

Definition 3.7 Lad et første-ordens prædikatlogiksprag \mathcal{L} være givet. *Herbrand-universet* for \mathcal{L} er mængden af alle grundede termer i \mathcal{L} . En *Herbrand-fortolkning* er en fortolkning over Herbrand-universet, hvor hver konstant og funktionssymbol afbildes over i sig selv (funktionssymboler forstået som term-konstruktorer).

En Herbrand-model for en mængde af formler Γ er en Herbrand-fortolkning, som er en model for Γ . \square

Vi vil alternativt forstå en given Herbrand-fortolkning eller -model, som mængden af alle grundede atomer, den påstår sande.

²⁰Helt analogt til, hvordan variable i en Prolog-klausul skal forstås.

Eksempel 3.10 Betragt Herbrand-modellen \mathcal{H} introduceret i eksempel 3.5 tidligere. Den kan karakteriseres ved mængden²¹

$$\{\text{lige}(0), \text{lige}(0+0)\text{lige}(2), \text{lige}(4), \dots, 0 = 0, 1 = 1, 1+0 = 1+0, 1+1 = 1+1, 2 = 2, 2+0 = 2+0, \dots, 3+(7+9) = 3+(7+9), \dots\}.$$

Vi kan definere en anden Herbrand-model for det samme sprog, som mere er i overensstemmelse med modellen \mathcal{N} , ved at den yderligere indeholder ligninger som $0 = 0 + 0$, $2 + 2 = 4$, $(1 + 2) + 5 = (0 + 0 + 4 + 0) + (1 + (1 + 2))$ osv. \square

Følgende egenskab er nyttig, for den, som måtte ønske at bevise korrektheden af de påstande, vi fremsætter om resolution i det følgende.

Sætning 3.1 Lad Γ være en mængde af klausuler. Γ har en model, dvs. Γ er opfyldelig hvis og kun hvis Γ har en Herbrand-model. \square

Et modstridsbevis for $\Gamma \models \phi$ (hvor Γ og ϕ er af passende natur), kan således forklares som en undersøgelse om, at $\Gamma \cup \{\neg\phi\}$ ikke har nogen Herbrand-model.

3.5.3 Resolutionssystemer

Som for propositionslogikkens vedkommende, er et resolutionssystem et aksiomatisk system $\langle \mathcal{K}, \{\mathcal{R}\}, \emptyset \rangle$ bestående af et sprog \mathcal{K} af klausuler, én slutningsregel, kaldet resolution, som præciseres nedenfor, samt det tomme sæt logiske aksiomer.

For at kunne generalisere resolutionsreglen, må vi kunne håndtere alkvanterede variable. Til det formål har vi brug for lidt begreber om substitution og unifikation.

Definition 3.8 En *substitution* er en afbildning σ fra en endelig mængde variable $\mathbf{X} = \{x_1, \dots, x_n\}$ til termer således, at for intet i , at $x_i\sigma$ indeholder variable fra \mathbf{X} .

En substitution generaliseres på sædvanlig måde til en afbildning for termer t og for kvantor-fri formler ϕ ; vi benytter notationen $t\sigma$ hhv. $\phi\sigma$ for den nye term hhv. formel, som fremkommer. Vi benytter en notation for substitutioner, svarende til notationen for tildelinger benyttet tidligere. \square

²¹Det er ikke lykkedes ved brug af prikker at antyde en fuldstændig nummerering af de atomer, som indgår. Meninger er, at der er samtlige atomer lige(t), hvor termen t svarer til et lige tal, samt alle atomer $t = t$.

Vi har her begrænset os til den slags substitutioner, som kaldes idempotente. Nogle forfattere foretrækker generelle substitutioner, som er vilkårlige afbildninger fra variable til termer. Men intuitivt og teknisk er de idempotente de nemmeste at håndtere.

De svar, som et Prolog-system producerer, er eksempler på idempotente substitutioner. Betragt f.eks. et svar

$$\begin{aligned} X &= f(a, Z) \\ Y &= g(f(a, Z)) \end{aligned}$$

Variablene X og Y forekommer ikke på højre-siderne, jvf. definition 3.8.

Hvad ville vi sige til et Prolog-system, som gav følgende svar i stedet?

$$\begin{aligned} X &= f(a, Z) \\ Y &= g(X) \end{aligned}$$

Uden at vi skal gå nærmere ind på matematikken bag, så er denne — ikke idempotente — substitution ækvivalent med den første, men intuitivt kan den forekomme som en slags halvfabrikata.

Betegnelsen idempotent skyldes den algebraiske egenskab, at for enhver sådan substitution σ gælder $\sigma\sigma = \sigma$; det følger umiddelbart af definitionen. Sammensætning af substitutioner er defineret som følger.

Eksempel 3.11 Lad σ være en substitution $[x \mapsto g(f(a, z), y \mapsto g(f(a, z))]$. Vi har da for en term $t = h(x, y, z)$, at $t\sigma = h(g(f(a, z), g(f(a, z))), z)$. \square

Definition 3.9 Lad θ og σ være substitutioner. En sammensat substitution $\theta\sigma$ er defineret således, at for alle x som ligger i grundmængden for θ eller σ , at

- $(\theta\sigma)(x) = t\sigma$ såfremt $\theta(x) = t$ er defineret,
- $(\theta\sigma)(x) = \sigma(x)$ ellers.

\square

Vi minder om det tidligere definerede begreb af tildelinger (i forhold til en givet fortolkning), vi definerede tidligere, def. 3.2. Hvis vi ser på Herbrandfortolkninger, så svarer tildelinger til grund-substitutioner, som er substitutioner, der afbilder variable over i grundede termer.

Definition 3.10 Lad A_1 og A_2 være to atomer. En *unifikator* for A_1 og A_2 er en substitution σ , så $A_1\sigma = A_2\sigma$.

En unifikator for A_1 og A_2 siges at være *mest generel* såfremt for enhver anden unifikator²² θ , at der findes en substitution γ , så $\theta = \sigma\gamma$. Vi benytter notationen $\text{mgu}(A_1, A_2)$ for en mest generel unifiator for A_1 og A_2 . \square

Eksempel 3.12 Lad $A_1 = p(x, a)$ og $A_2 = q(b, y)$ være to atomer. Substitutionen $[x \mapsto b, y \mapsto a]$ er en $\text{mgu}(A_1, A_2)$. Vi har, at $\text{mgu}(p(x), q(y)) = [x \mapsto z, y \mapsto z]$; substitutionen $[x \mapsto \text{abekat}, y \mapsto \text{abekat}]$ er også en unifier, men ikke en mgu.

De to atomer $p(x, x)$ og $q(b, a)$ har ingen mgu. \square

Ud fra eksemplet, kan man nemt forestille sig en algoritme, der kan finde mgu'er, ved at traversere de to atomer i parallel ned gennem deres termer og disses undertermer, og konstruere en substitution undervejs. Vi nøjes her med at påstå, at sådanne algoritmer eksisterer, dvs. at unifikation er afgørligt.

Unifikation, repræsenteret ved operationen $\text{mgu}(-, -)$, er hvad der skal til for at generalisere resolution til første-ordens logik.

Vi giver her reglen i sin korte form, og henviser til diskussionen i kapitel 2 angående associative og kommutative varianter, som skal håndteres også.

$$\frac{\forall(\phi_1 \vee \neg\phi_2) \quad \forall(\phi_3 \vee \phi_4)}{\forall((\phi_1 \vee \phi_4)\sigma)} \quad \text{hvor } \sigma = \text{mgu}(\phi_2, \phi_3).$$

Det antages, at variablene i den ene (f.eks., den første) præmis omdøbes således at de to præmisser har forskellige variable — ellers giver mgu-operationen ikke det ønskede resultat.²³

Modsat propositionslogikken, kan vi ikke forvente at de to udvalgte litteraler er eksakt identiske — anvendelsen af mgu-operationen sørger netop at specialisere de indgående formler lige præcist så meget, at litteralerne ϕ_2 og ϕ_3 lige akkurat bliver ens, men der specialiseres ikke mere end højst nødvendigt.

Som tidligere, så ligger det nyttige i resolution i forbindelse med modbevis. Antag, vi har en mængde formler Γ og en formel ϕ , hvor vi vil undersøge hvorvidt der gælder $\Gamma \models \phi$. Vi kører Γ over i en klausulform Γ' , og ditto $\neg\phi$ over i en klausulform Φ^- og benytter resolution til at verificere $\Gamma' \cup \Phi^- \vdash \text{falsk}$.

Det kan vises, at resolution er sund og fuldstændig med hensyn til modbevis inden for klausuler. Skal man konstruere et bevis for dette, er de

²²Betegnelsen unifikator er oversat fra det engelske "unifier".

²³Det er legalt at omdøbe kvantoriserede variable, der er f.eks. klart at $\forall x p(x)$ er ækvivalent med $\forall z p(z)$.

observationer, vi gjorde angående Herbrand-modeller ovenfor, særdeles anvendelige. Men jævnfør tidligere bemærkninger om afgørlighed (og Turing-maskiner og den slags), så er resolution ikke afgørlig — første-ordenslogikken er ikke afgørlig, selv om der var en ung mand ved navn Hilbert, som en gang led af den vrangforestilling.

4 SLD-resolution og logikprogrammering

Vi lægger nu yderligere en begrænsning på sproget, så vi når frem den type klausuler, man benytter i forbindelse med logikprogrammering.

Definition 4.1 En *Horn-klausul* er en klausul med højst ét positivt literal. Hvis der er netop ét positivt literal, er der tale om en *definit klausul*, ellers om et *definit mål*. En samling af definite klausuler kaldes et *definit program*. \square

Vi vil straks gøre begreberne mere tilgængelige gennem nogle eksempler. En definit klausul:

$$\forall(p(x) \vee \neg q(x) \vee \neg r(x))$$

Et definit mål:

$$\forall(\neg p(x) \vee \neg s(x))$$

Hvis vi nu omskriver klausulen til en medførerpil (og benytter den udgave, som peger mod venstre) og lader al-kvantoren være implicit, kommer den til at ligne en Prolog-regel i betænkelig grad.

$$p(x) \leftarrow q(x) \wedge r(x)$$

Omkring målet, så husker vi på, at det interessante her, er anvendelse i forhold til modbevis for en eller anden forespørgsel. Det definite mål tænkes at komme ind som negationen af den forespørgsel, vi gerne ville vise. Negerer vi målet ovenfor ”tilbage igen” får vi:

$$\exists(p(x) \wedge s(x))$$

Dette svarer i mistænkelig grad til den type forespørgsler, vi giver til et Prolog-system.²⁴ Vi definerer derfor en *forespørgsel* til at være et formel af formen $\exists\psi$, hvor ψ er en konjunktion af atomer.²⁵

²⁴At Prolog ud over at undersøge, hvorvidt der findes et passende x , også giver oplysninger om hvilke sådanne, der faktisk er tale om, er så en ekstra feature, vi vil kigge nærmere på om lidt.

²⁵Der er desværre lidt forvirring omkring terminologien, idet nogle forfattere i forbindelse med logikprogrammering bruger betegnelsen ”mål” (eng.: ”goal”) for det positive udsagn $\exists\cdots$; andre (som her) benytter det om det negerede.

Vi kan da benytte modbevis-ved-resolution, præcis som beskrevet ovenfor, til at undersøge en påstand $\Delta \models \phi$, hvor Δ er en mængde definite klausuler og ϕ en forespørgsel. Når vi negerer ϕ og tilføjer den som ægte aksiom, når vi frem til et problem, $H \vdash \text{falsk}$, hvor H er en mængde Horn-klausuler.

Men hvorfor begrænse os til Horn-klausuler, når der i resolution ved modbevis foreligger et aksiomatisk system, som er sundt og fuldstændigt? Vel, et aksiomatisk system kan læses som en specifikation for en mængde af beviser, men der ligger jo ikke heri nogen procedure for, hvordan disse beviser kan konstrueres.

Begrænsningen til Horn-klausuler skaber grundlag for en forenklet bevisprocedure. Vi skal se på en procedure, der af historiske (og ikke umiddelbart indlysende årsager) kaldes *SLD-resolution*.

Vi laver nu en specialiseret version af resolutionsreglen. Notationen forenkles ved at droppe kvantorerne (dvs. at lade dem implicite). Vi opskriver mål vha. den bagvendte medførerpil, og benytter notationen " $\leftarrow \dots$ " for "*falsk* $\leftarrow \dots$ ". Eksemplerne på definit klausul og mål ovenfor kommer nu til at se således ud.

$$p(x) \leftarrow q(x) \wedge r(x)$$

$$\leftarrow p(x) \wedge s(x)$$

Bevisreglen for SLD-resolution bliver nu som følger, hvor første præmis er en klausul (med passende, ombenævnte variable) og anden præmis et mål; konklusionen er igen et mål²⁶.

$$\frac{H \leftarrow K_1 \wedge \dots \wedge K_n \quad \leftarrow M_1 \wedge \dots \wedge M_m}{(\leftarrow M_1 \wedge \dots \wedge M_{i-1} \wedge K_1 \wedge \dots \wedge K_n \wedge M_{i+1} \wedge M_m)\sigma}$$

for et eller andet i og $\sigma = \text{mgu}(H, M_i)$.

Vi kan nu præsentere en bevisprocedure, omend med et vist gran af nondeterminisme i sig. Givet en definit program Δ og en forespørgsel, som negeres til et mål Φ_0 og lad $k = 0$. Gentag følgende indtil Φ_k er tom:

- Anvend reglen SLD-resolution med præmisser, hvor den ene er en klausul i Δ (med ombenævnte variable) og den anden er Φ_k ; vi kalder den anvendte mgu for σ_k og konklusionen for Φ_{k+1} .

²⁶Det er ganske analogt til, hvad der foregår i et program i et traditionelt programmeringssprog. Vi står med et række atomer (\approx procedurekald) M_1, \dots, M_m som skal udføres, Vi vælger et af dem, M_i , og kalder en procedure H med krop K_1, \dots, K_n . Allokering af nye instanser af lokale variable klares af ombenævningen og parameteroverførslen af $\text{mgu}(H, M_i)$. Dette medfører så kald af nye underprocedurer $K_1 \wedge \dots \wedge K_n$, med mindre $n = 0$, svarende til en simpel procedure eller et faktum i Prolog.

– $k := k + 1$.

Vi antager en ikke nærmere specificeret delprocedure til at vælge et atom i den aktuelle forespørgsel; en sådan kaldes traditionelt en *beregningsregel*.

For at opnå en bevisprocedure, som er sund og fuldstændig, kan vi forestille os en datastruktur, som på en bredde-først vis simulerer parallelitet i valget af klausul, dvs. samtlige mulige klausuler vælges i parallel. Denne datastruktur kunne være et træ, hvor så en vej ned gennem træet svarer til et bevis.

Man kan gøre rede for at denne procedure er sund og fuldstændig uafhængigt af beregningsregler. Interesserede kan konsultere f.eks. Lloyd: "Foundations of logic programming, Second, Extended Edition" for et bevis eller til en mere uformel argumentation hos Christiansen: "Logikprogrammering, et sprog og dets implementation".

Fuldstændighed betyder, at vi kan sætte proceduren igang med at kværne, og findes der et bevis for modstrid, så vil det blive fundet før eller senere. Uafgørligheden (som stadig er til stede med begrænsningen til Horn-klausuler) kommer til udtryk på den måde, at hvis vores procedure efter 10^6 år endnu ikke har fundet et bevis, så kan vi ikke vide, om det er fordi, der ikke er noget bevis, eller om det vil blive fundet på et eller andet senere tidspunkt.

4.1 Om modeller for definite klausulprogrammer

Vi beskrev tidligere, hvordan man med generelle klausuler kunne indskrænke sig til at se på Herbrand-modeller. For definite klausuler kan vi yderligere begrænse os til at se på én enkelt, udvalgt Herbrand-model.

Vi vælger her, at se Herbrand-modeller som mængder af grundede atomer, og det giver således mening, at betragte den *minimale* Herbrand-model for et definit klausul-program Δ , som er fællesmængden af alle Herbrand-modeller; vi kalder denne for M_Δ . Vi har altså pr. definition, at $\alpha \in M_\Delta$ hvis og kun hvis, for enhver Herbrand-model H_Δ for Δ , at $\alpha \in H_\Delta$. Nu følger det centrale resultat om logisk konsekvens for definite klausul-programmer.

Sætning 4.1 Lad Δ være et definit klausul-program og α et grundet atom. Da gælder, at $\Delta \models \alpha$ hvis og kun hvis $\alpha \in M_\Delta$. \square

Eksempel 4.1 Betragt følgende definite klausul-program.

$$\forall x : \text{menneske}(x) \rightarrow \text{dødelig}(x)$$

$$\text{menneske}(\text{Sokrates})$$

Vi påstår, at hver af de følgende mængder er Herbrand-modeller for programmet.

{menneske(Platon), dødelig(Platon), dødelig(Sokrates), menneske(Sokrates)}
 {kan-flyve(påfugl), dødelig(påfugl), menneske(Sokrates), dødelig(Sokrates), grøn-ost(månen)}

Ingen af disse er minimale; den minimale er følgende.

{dødelig(Sokrates), menneske(Sokrates)}

□

Problemet om implementation af logisk konsekvens kan formuleres som et spørgsmål om

1. at karakterisere M_Δ ,
2. at betragte egenskaben " $\in M_\Delta$ ".

En ekstensionel²⁷ repræsentation af M_Δ , som i eksemplet ovenfor, er i de færreste tilfælde relevant, da M_Δ meget vel kan være uendelig.

Følgende karakteristik giver os en idé om, hvordan M_Δ (endelig eller uendelig) ser ud.

Definition 4.2 Lad Δ være et definit klausul-program og lad \mathcal{H} referere til mængden af alle Herbrand-fortolkninger. Den *umiddelbare konsekvens-operator* for Δ , T_Δ , er afbildningen fra \mathcal{H} til \mathcal{H} defineret som følger.

$$T_\Delta(I) = \{A \mid A: \neg B_1, \dots, B_n \text{ er grundinst. af klausul i } \Delta, B_1, \dots, B_n \in I\}$$

□

Intuitivt, så er $T_\Delta(I)$ de nye udsagn eller fakta, vi kan aflede ved at proppe nogle allerede kendte (I) ind i klausulernes kroppe. Lloyd klarer igen matematikken for os, så vi kan være sikre på, at T_Δ opfører sig tilstrækkeligt pænt til, at fikspunktet i det følgende er veldefineret, og at sætningen faktisk holder; en intuitiv beskrivelse af dette fikspunkt følger.

Sætning 4.2 For ethvert definit klausul-program Δ gælder, at M_Δ er det mindste fikspunkt for T_Δ . □

²⁷Ekstensionel: angivet ved en opremsning af samtlige elementer. Modsat intensionel: angivet ved fælles egenskaber.

Udtrykt på en anden måde, starter vi med den tomme mængde udsagn og anvender programmets regler “baglæns” tilstrækkeligt længe, får vi indfanget netop alle de fakta, som er logiske konsekvenser af Δ . Vi kan skitsere dette som en (pseudo-) algoritme.

```

M:= ∅;
repeat
    M1:= M;
    M:= TΔ(M)
until M = M1

```

Hvis den mindste Herbrand-model for Δ er endelig, vil processen terminere med $M = M_1 = M_\Delta$; det er indeholdt i sætningen. Såfremt M_Δ ikke er endelig, må den beskrives som den grænseværdi, M konvergerer imod, når tiden går mod uendelig.²⁸

Eksempel 4.2 Kald programmet i eksempel 4.1 for Σ . Vi har da følgende.

$$T_\Sigma(\emptyset) = \{\text{menneske}(\text{Sokrates})\}$$

$$T_\Sigma^2(\emptyset) = T_\Sigma(T_\Sigma(\emptyset)) = \{\text{menneske}(\text{Sokrates}), \text{dødelig}(\text{Sokrates})\}$$

$$T_\Sigma^3(\emptyset) = \{\text{menneske}(\text{Sokrates}), \text{dødelig}(\text{Sokrates})\}$$

Denne generering af udsagn kan forstås således, at vi starter med at vide intet (dvs. \emptyset) for at være sikker på ikke at have nogen falsk viden. Derefter genererer vi ny viden ud fra, kun den viden, vi er sikker på, ikke er forurenset af falskhed.²⁹

I dette tilfælde terminerede algoritmen efter ganske få iterationer, og det fremgår, at den resulterende mængde af udsagn er et fikspunkt: Anvendes T_Σ på den, får vi den uændret tilbage som et resultat. At der er tale om et

²⁸Sammenlign med Newtons metode til at udregne kvadratrødder med, som man kan finde i de fleste bøger om matematisk analyse. Her er det et tal (og ikke en fortolkning = en mængde fakta), man pumper igennem en afbildning, og værdien konvergerer mod et (typisk) infinitesimalt tal (f.eks., $\sqrt{2}$). Det kan skrives som følgende algoritme.

```

k:= 1;
repeat
    k1:= k;
    k:= k/2 + 1/k
until |k - k1| < 10-6

```

At $\sqrt{2}$ faktisk er et fikspunkt, kan man overbevise sig om ved at indsætte $\sqrt{2}$ på k 's plads i ligningen $k = k/2 + 1/k$.

²⁹I Platons dialog ‘Menon’, findes en scene, hvor Sokrates belærer en slave efter dette princip.

mindste fikspunkt kan indses ved at, hvis man fjerner et element fra den, så er der ikke tale om et fikspunkt længere. \square

Eksempel 4.3 Betragt et program Π bestående følgende to klausuler.

$$p(a)$$

$$\forall x : p(x) \rightarrow p(f(x))$$

Her vil algoritmen producere følgende sekvens af fortolkninger (= mængder af grundede atomer).

$$T_{\Pi}(\emptyset) = \{p(a)\}$$

$$T_{\Pi}^2(\emptyset) = \{p(a), p(f(a))\}$$

$$T_{\Pi}^3(\emptyset) = \{p(a), p(f(a)), p(f(f(a)))\}$$

...

$$T_{\Pi}^n(\emptyset) = \{p(a), p(f(a)), p(f(f(a))), \dots, p(\overbrace{f(\dots f(a)\dots)}^{n-1})\}$$

I dette tilfælde terminerer iterationen ikke, og vi må forstå fikspunktet (= den mindste Herbrand-model), som den værdi, T_{Π}^n konvergerer imod.

$$M_{\Pi} = \lim_{n \rightarrow \infty} T_{\Pi}^n(\emptyset) = \{p(f^k(a)) \mid k = 0, 1, 2, \dots\}$$

\square

Rent spekulativt kunne man foreslå en implementation af logisk konsekvens på følgende måde. For at undersøge, hvorvidt $\Delta \models \alpha$, sætter vi algoritmen i gang og i hvert skridt kigger efter, om α skulle være dukket op. Såfremt der faktisk gælder $\Delta \models \alpha$, vil algoritmen definitivt efter rum tid rapportere dette. I modsat fald kan vi forvente en uendelig løkke, som aldrig giver noget svar fra sig (undtagen i de særlige tilfælde, hvor M_{Δ} er endelig).

4.2 Prolog

Bevissystemet Prolog kan beskrives således:

- Beregningsreglen siger ”vælg det første atom i målet”

- Klausulerne afprøves sekventiel ved backtracking ud fra den rækkefølge de fremtræder i programteksten.³⁰

Når vi forudsætter en eller anden effektiv måde at implementere unifikation på, kan vi se, at vi opnår en procedure, som er effektiv og sund, men prisen for effektivitet er manglende fuldstændighed, som giver sig udslag i, at Prolog er tilbøjelig til at gå i uendelige løkke, hvis klausulerne vender blot en lille smule forkert. I Christiansen: ”Sprog og abstrakte maskiner”, afsnit 9.3 og 9.4, ser vi på effekten af andre beregningsregler.

Men, ak, af effektivitetsgrunde, så er unifikation faktisk ikke implementeret korrekt i Prolog. Man kan undlade det såkaldte ”occur check”, som er nødvendigt når man forsøger at unificere en variabel x med en term t , hvori x forekommer. Der findes ingen $mgu(x, f(x))$, eller sagt på en anden måde, der findes ingen Herbrand-fortolkning, hvor en ligning $x = f(x)$ kan opfyldes.

Grunden til, at man udelader dette i Prolog, er, at det vil være for omkostningsbefængt at traversere en måske meget stor term, hver gang, man skulle binde en variabel. I Christiansen: ”Logikprogrammering, et sprog og dets implementation” beskrives en unifikationsalgoritme med tilhørende datastruktur, som svarer til Prologs, dvs. man snyder for ”occur check”.

4.3 Om beregnede svar

Vi kan benytte de mgu 'er, som opstår gennem et resolutionsbevis, til at sige noget om de variable, som indgår i forespørgslen. Givet program og forespørgsel Φ og et afsluttet modstridsbevis ved resolution, og antag mgu 'erne $\sigma_0, \dots, \sigma_k$ optræder i den nævnte rækkefølge. Den sammensatte substitution $\sigma_0\sigma_1 \dots \sigma_k$ skåret ned til de variable, som optræder i Φ , kaldes et *beregnet svar* for Φ .

I et Prolog-system svarer det til, at i hvert skridt skabes nye variabelbindinger, enten til variable i forespørgslen, eller til internt genererede variable. Herigennem bliver bindingerne til forespørgslens variable gradvist specialiseret, når mgu 'erne efterhånden sættes sammen.

Det, at vi skærer slut-substitutionen ned til kun de relevante variable, svarer til, at Prolog-systemet altså ikke skriver bindinger ud til alle mulige besynderlige, interne variable, men kun de, vi forespurgte med.

Der kan gøres rede for, at SLD-resolution er sund og fuldstændig med hensyn til generering af svar forstået på den måde, at mængden af mulige,

³⁰Dvs. der huskes kun ét bevis, som er under konstruktion. Beviset vokser, når der er klausuler, hvis hovede unifier med det valgte atom, og krymper når dette ikke er tilfældet og der backtracks. Et bevis er færdigkonstrueret, når målet er tomt, og alternative beviser konstrueres ved provokeret backtracking.

beregnete svar for en forespørgsel $\exists\phi$ er sammenfaldende med mængden af substitutioner σ , for hvilke $\forall(\phi\sigma)$ er logisk konsekvens af det foreliggende program.

5 Afsluttende bemærkninger

Vi har i dette skrift gennemgået propositionslogik og løftet sløret for den vigtige første-ordens prædikatlogik.

Vi har beskrevet et bevissystem i detaljer, SLD-resolution. Et system som de kendte, effektive implementationer af logikprogrammeringssproget Prolog, bygger på.

I mere generelle, automatiske bevissystemer kan generel resolution benyttes, men selvfølgelig med en stor omkostning i forbrug af plads og tid til at styre den iboende nondeterminisme.

Resolutionssystemer med den type bevisprocedurer, vi har beskrevet her, svarer til deduktiv ræsonneren. Den af sin samtid miskendte filosof Pierce opstillede sammen med deduktion to andre ræsonneringsformer, abduktion og induktion, som værende de tre fundamentale former.

Der er hverken plads eller tid til at komme ind på abduktion og induktion her, men det skal blot nævnes, at det også er noget, der i øjeblikket arbejdes ihærdigt på at implementere.

Endelig kunne man også nævne forskellige, alternative logikker, som også benyttes i forbindelse med automatisk ræsonneren (rækkende fra såkaldte programmeringssprog til påstået intelligente systemer). Vi nævner i flæng logik med ligninger, typet logik, højjordenslogik, metalogik, modallogik, temporal logik, fuzzy-logik, fler-værdilogik, lineær logik, default-logik, første-ordenslogik udvidet med kontrafaktisk viden, osv., osv.