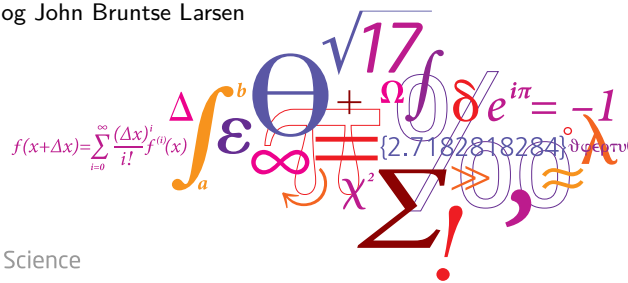
Formaliseret førsteordenslogik

Bachelorforsvar

Andreas Halkjær From

Vejleder: Jørgen Villadsen

Medvejledere: Anders Schlichtkrull og John Bruntse Larsen



DTU Compute
Institut for Matematik og Computer Science

Disposition

- Førsteordenslogik
- Naturlig deduktion
- Isabelle
- Sundhed
- Komplethed
- Åbne formler
- Konklusion

Motivation

Værktøj til ræsonnering, f.eks om softwarekorrekthed.
Software er alle steder, i vores telefoner, biler, fly, pacemakere osv.

Motivation

Værktøj til ræsonnering, f.eks om softwarekorrekthed.
Software er alle steder, i vores telefoner, biler, fly, pacemakere osv.

Syntaks

Hvordan vi skriver ræsonnementer ned.

Semantik

Hvad de betyder.

Syntaks

Termer

- Variable, $x, y, z, \ldots$
- Funktioner, $f, g, h, \ldots$ anvendt på lister af termer

Syntaks

Termer

- Variable, $x, y, z, \ldots$
- Funktioner, $f, g, h, \ldots$ anvendt på lister af termer

Formler

- Falskhed, $\bot$.
- Prædikater, $p, q, r, \ldots$, givet en liste af termer af bestemt længde.
- Konjuktion, $A \wedge B$, disjunktion, $A \vee B$, implikation, $A \to B$.
- Universel, $\forall x.A$, og eksistentiel, $\exists x.A$, kvantificering.

Syntaks

Termer

- Variable, $x, y, z, \ldots$
- Funktioner, $f, g, h, \ldots$ anvendt på lister af termer

Formler

- Falskhed, $\bot$.
- Prædikater, $p, q, r, \ldots$, givet en liste af termer af bestemt længde.
- Konjuktion, $A \wedge B$, disjunktion, $A \vee B$, implikation, $A \to B$.
- Universel, $\forall x.A$, og eksistentiel, $\exists x.A$, kvantificering.

Forkortelser

- $\top \equiv \bot \to \bot$
- $\neg A \equiv A \to \bot$

Semantik

Fortolkning

- Domæne, $\mathcal{D}$, f.eks. de naturlige tal eller personerne i dette lokale.
- Tildeling af funktionssymboler, $\mathcal{F} : id \to (\mathcal{D} \to \ldots \to \mathcal{D})$, f.eks. addition af tal.
- Tildeling af prædikatsymboler, $\mathcal{G} : id \to (\mathcal{D} \to \ldots \to \{T, F\})$, f.eks. ulighed.

Antagelse: Domænet er ikke tomt.

Semantik

Fortolkning

- Domæne, $\mathcal{D}$, f.eks. de naturlige tal eller personerne i dette lokale.
- Tildeling af funktionssymboler, $\mathcal{F} : id \to (\mathcal{D} \to \ldots \to \mathcal{D})$, f.eks. addition af tal.
- Tildeling af prædikatsymboler, $\mathcal{G} : id \to (\mathcal{D} \to \ldots \to \{T, F\})$, f.eks. ulighed.

Antagelse: Domænet er ikke tomt.

Miljø

Oprindelig tildeling af værdier til variable, eks. for $\mathcal{D} = \mathbb{N}$:

$$\sigma(x) = 2, \sigma(y) = 17$$

Notation: $\sigma[x \leftarrow d]$:

$$\sigma[x \leftarrow d](x) = d$$
$$\sigma[x \leftarrow d](y) = \sigma(y)$$

Semantik

Termer

- $v_\sigma\left(x\right) = \sigma(x)$.
- $v_\sigma\left(f(t_1, \ldots, t_n)\right) = (\mathcal{F}(f))(v_\sigma\left(t_1\right), \ldots, v_\sigma\left(t_n\right))$.

Semantik

Termer

- $v_\sigma(x) = \sigma(x)$.
- $v_\sigma(f(t_1, \ldots, t_n)) = (\mathcal{F}(f))(v_\sigma(t_1), \ldots, v_\sigma(t_n))$.

Formler

- $v_\sigma(\bot) = F$
- $v_\sigma(p(t_1, \ldots, t_n)) = \mathcal{G}(p)(v_\sigma(t_1), \ldots, v_\sigma(t_n))$
- $v_\sigma(A \wedge B) = T$ hviss $v_\sigma(A) = T$ og $v_\sigma(B) = T$
- $v_\sigma(A \vee B) = T$ hviss $v_\sigma(A) = T$ eller $v_\sigma(B) = T$
- $v_\sigma(A \to B) = T$ hviss $v_\sigma(A) = F$ eller $v_\sigma(B) = T$
- $v_\sigma(\forall x.A) = T$ hviss $v_{\sigma[x \leftarrow d]}(A) = T$ for alle $d \in \mathcal{D}$.
- $v_\sigma(\exists x.A) = T$ hviss $v_{\sigma[x \leftarrow d]}(A) = T$ for mindst et $d \in \mathcal{D}$.

Opfyldelighed og gyldighed

Opfyldelighed

A er opfyldelig hvis $v_\sigma(A) = T$ for mindst én fortolkning og miljø.
Fortolkningen kaldes en model for A.

Eks.

$$\forall x.\exists y.p(x, y)$$

er sand i domænet af de naturlige tal hvor $\mathcal{G}(p) = \leq$.

Opfyldelighed og gyldighed

Opfyldelighed

A er opfyldelig hvis $v_\sigma(A) = T$ for mindst én fortolkning og miljø.
Fortolkningen kaldes en model for A.

Eks.

$$\forall x.\exists y.p(x, y)$$

er sand i domænet af de naturlige tal hvor $\mathcal{G}(p) = \leq$.

Gyldighed

A er gyldig hvis $v_\sigma(A) = T$ i alle fortolkninger og miljøer.

Eks.

$$(\forall x.p(x)) \rightarrow (\exists x.p(x))$$

Bevissystem

Problem: Hvordan bestemmer vi at en formel er gyldig?

Bevissystem

Problem: Hvordan bestemmer vi at en formel er gyldig?
Løsning: Syntaktisk metode til at udlede formler.

Aksiomer

Formler der som udgangspunkt kan udledes.

Inferensregler

Regler til at udlede flere formler.

Bevissystem

Problem: Hvordan bestemmer vi at en formel er gyldig?
Løsning: Syntaktisk metode til at udlede formler.

Aksiomer

Formler der som udgangspunkt kan udledes.

Inferensregler

Regler til at udlede flere formler.

To vigtige egenskaber:

Sundhed

Kun gyldige formler kan udledes.

Komplethed

Alle gyldige formler kan udledes.

I en lærebog

$$\frac{\begin{array}{|c|}\hline \neg\phi \\ \vdots \\ \bot \\ \hline \end{array}}{\phi}\ PBC \qquad \frac{\phi \qquad \phi \to \psi}{\psi} \to E \qquad \frac{\begin{array}{|c|}\hline \phi \\ \vdots \\ \psi \\ \hline \end{array}}{\phi \to \psi} \to I$$

$$\frac{\phi \vee \psi \qquad \begin{array}{|c|}\hline \phi \\ \vdots \\ \chi \\ \hline \end{array} \qquad \begin{array}{|c|}\hline \psi \\ \vdots \\ \chi \\ \hline \end{array}}{\chi} \vee E \qquad \frac{\phi}{\phi \vee \psi} \vee I_1 \qquad \frac{\psi}{\phi \vee \psi} \vee I_2$$

$$\frac{\phi \wedge \psi}{\phi} \wedge E_1 \qquad \frac{\phi \wedge \psi}{\psi} \wedge E_2 \qquad \frac{\phi \qquad \psi}{\phi \wedge \psi} \wedge I$$

Syntaks

Baseret på HOL. Alle beviser reduceres til grundlæggende aksiomer og inferensregler.
Med al sandsynlighed sundt.

Syntaks

Baseret på HOL. Alle beviser reduceres til grundlæggende aksiomer og inferensregler.
Med al sandsynlighed sundt.

Termer

type-synonym *id* = ‹*char list*›

datatype *tm* = *Var nat* | *Fun id* ‹*tm list*›

Formler

datatype *fm* = *Falsity* | *Pre id* ‹*tm list*› |
Imp fm fm | *Dis fm fm* | *Con fm fm* | *Exi fm* | *Uni fm*

Semantik I

Termer

primrec
semantics-term :: ‹(*nat* ⇒ *'a*) ⇒ (*id* ⇒ *'a list* ⇒ *'a*) ⇒ *tm* ⇒ *'a*› **and**
semantics-list :: ‹(*nat* ⇒ *'a*) ⇒ (*id* ⇒ *'a list* ⇒ *'a*) ⇒ *tm list* ⇒ *'a list*› **where**
‹*semantics-term e f* (*Var n*) = *e n*› |
‹*semantics-term e f* (*Fun i l*) = *f i* (*semantics-list e f l*)› |
‹*semantics-list e f* [] = []› |
‹*semantics-list e f* (*t* # *l*) = *semantics-term e f t* # *semantics-list e f l*›

Semantik II

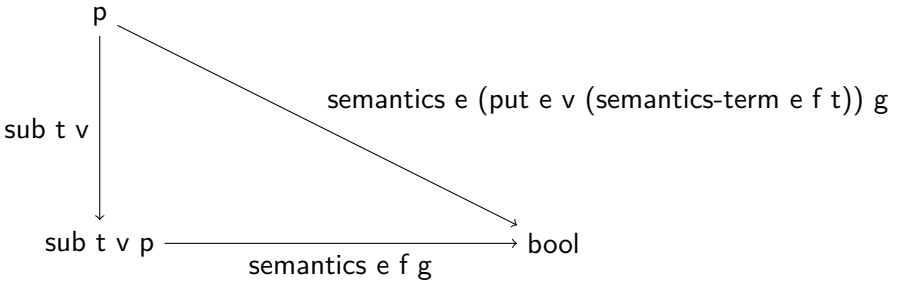
Formler

primrec
semantics :: ‹(*nat* ⇒ *'a*) ⇒ (*id* ⇒ *'a list* ⇒ *'a*) ⇒ (*id* ⇒ *'a list* ⇒ *bool*) ⇒ *fm* ⇒ *bool*› **where**
‹*semantics e f g Falsity* = *False*› |
‹*semantics e f g* (*Pre i l*) = *g i* (*semantics-list e f l*)› |
‹*semantics e f g* (*Imp p q*) =
(*if semantics e f g p then semantics e f g q else True*)› |
‹*semantics e f g* (*Dis p q*) =
(*if semantics e f g p then True else semantics e f g q*)› |
‹*semantics e f g* (*Con p q*) =
(*if semantics e f g p then semantics e f g q else False*)› |
‹*semantics e f g* (*Exi p*) =
(∃ *x*. *semantics* (λ*n*. *if n* = *0 then x else e* (*n* − *1*)) *f g p*)› |
‹*semantics e f g* (*Uni p*) =
(∀ *x*. *semantics* (λ*n*. *if n* = *0 then x else e* (*n* − *1*)) *f g p*)›

OK

inductive $OK :: \langle fm \Rightarrow fm\ list \Rightarrow bool \rangle$ **where**
$Assume: \langle member\ p\ z \Longrightarrow OK\ p\ z \rangle$ |
$Boole: \langle OK\ Falsity\ ((Imp\ p\ Falsity)\ \#\ z) \Longrightarrow OK\ p\ z \rangle$ |
$Imp\text{-}E: \langle OK\ (Imp\ p\ q)\ z \Longrightarrow OK\ p\ z \Longrightarrow OK\ q\ z \rangle$ |
$Imp\text{-}I: \langle OK\ q\ (p\ \#\ z) \Longrightarrow OK\ (Imp\ p\ q)\ z \rangle$ |
$Dis\text{-}E: \langle OK\ (Dis\ p\ q)\ z \Longrightarrow OK\ r\ (p\ \#\ z) \Longrightarrow OK\ r\ (q\ \#\ z) \Longrightarrow OK\ r\ z \rangle$ |
$Dis\text{-}I1: \langle OK\ p\ z \Longrightarrow OK\ (Dis\ p\ q)\ z \rangle$ |
$Dis\text{-}I2: \langle OK\ q\ z \Longrightarrow OK\ (Dis\ p\ q)\ z \rangle$ |
$Con\text{-}E1: \langle OK\ (Con\ p\ q)\ z \Longrightarrow OK\ p\ z \rangle$ |
$Con\text{-}E2: \langle OK\ (Con\ p\ q)\ z \Longrightarrow OK\ q\ z \rangle$ |
$Con\text{-}I: \langle OK\ p\ z \Longrightarrow OK\ q\ z \Longrightarrow OK\ (Con\ p\ q)\ z \rangle$ |
$Exi\text{-}E: \langle OK\ (Exi\ p)\ z \Longrightarrow OK\ q\ ((sub\ 0\ (Fun\ c\ [])\ p)\ \#\ z) \Longrightarrow$
$news\ c\ (p\ \#\ q\ \#\ z) \Longrightarrow OK\ q\ z \rangle$ |
$Exi\text{-}I: \langle OK\ (sub\ 0\ t\ p)\ z \Longrightarrow OK\ (Exi\ p)\ z \rangle$ |
$Uni\text{-}E: \langle OK\ (Uni\ p)\ z \Longrightarrow OK\ (sub\ 0\ t\ p)\ z \rangle$ |
$Uni\text{-}I: \langle OK\ (sub\ 0\ (Fun\ c\ [])\ p)\ z \Longrightarrow news\ c\ (p\ \#\ z) \Longrightarrow OK\ (Uni\ p)\ z \rangle$

Substitution



sub

sub-term v s (*Var n*) =
(*if n* < *v then Var n else if n* = *v then s else Var* (*n* − *1*))

put

fun *put* :: ‹(*nat* ⇒ *'a*) ⇒ *nat* ⇒ *'a* ⇒ *nat* ⇒ *'a*› **where**
‹*put e v x* = (λ*n*. *if n* < *v then e n else if n* = *v then x else e* (*n* − *1*))›

Sundhed

Kontekst

lemma *soundness'*:
‹*OK p z* ⟹ *list-all* (*semantics e f g*) *z* ⟹ *semantics e f g p*›

Uni-I: ‹*OK* (*sub 0* (*Fun c* []) *p*) *z* ⟹ *news c* (*p* # *z*) ⟹ *OK* (*Uni p*) *z*›

case (*Uni-I c p z*)
then have ‹∀ *x*. *list-all* (*semantics e* (*f*(*c* := λ*w*. *x*)) *g*) *z*›
by *simp*
then have ‹∀ *x*. *semantics e* (*f*(*c* := λ*w*. *x*)) *g* (*sub 0* (*Fun c* []) *p*)›
using *Uni-I* **by** *blast*
then have ‹∀ *x*. *semantics* (*put e 0 x*) (*f*(*c* := λ*w*. *x*)) *g p*›
by *simp*
then have ‹∀ *x*. *semantics* (*put e 0 x*) *f g p*›
using ‹*news c* (*p* # *z*)› **by** *simp*
then show ‹*semantics e f g* (*Uni p*)›
by *simp*

Overblik

Fittings bevis fra *First-Order Logic and Automated Theorem Proving*.
Formaliseret af Berghofer for en anden formulering af naturlig deduktion.
Oversat til Isar, tilpasset til NaDeA og udvidet til åbne formler af mig.

Abstrakt

- Konsistensegenskab, C
- Alternativ konsistensegenskab, C^+
- Finit karakter, C^*
- Maksimal udvidelse, H, er Hintikka og har en Herbrand model

Konkret

- Konsistens af formler hvorfra falsk ikke kan udledes
- Komplethed via modstrid
  - Antag p er gyldig men ikke kan udledes
  - Så er $\{\neg p\} \in C$ og har en model

Beviset er givet i rapporten. Fokus her på oversættelsen.

Apply til Isar I

```
theorem mk-alt-consistency-closed:
  subset-closed C ⟹
  subset-closed (mk-alt-consistency C)
  apply (unfold mk-alt-consistency-def
      subset-closed-def)
  apply (rule ballI allI impI)+
  apply (rule CollectI)
  apply (erule CollectE)
  apply (erule exE)
  apply (subgoal-tac
      psubst f ‘ S ⊆ psubst f ‘ S')
  apply blast+
  done
```

```
theorem mk-alt-consistency-closed:
  assumes ⟨subset-closed C⟩
  shows ⟨subset-closed (mk-alt-consistency C)⟩
  unfolding subset-closed-def
proof (intro ballI allI impI)
  fix S S'
  assume ⟨S ∈ mk-alt-consistency C⟩ ⟨S' ⊆ S⟩
  then obtain f where ∗: ⟨psubst f ‘ S ∈ C⟩
    unfolding mk-alt-consistency-def by blast
  moreover have ⟨psubst f ‘ S' ⊆ psubst f ‘ S⟩
    using ⟨S' ⊆ S⟩ by blast
  ultimately have ⟨psubst f ‘ S' ∈ C⟩
    using ⟨subset-closed C⟩
    unfolding subset-closed-def by blast
  then show ⟨S' ∈ mk-alt-consistency C⟩
    unfolding mk-alt-consistency-def by blast
qed
```

Apply til Isar II

Kontekst

theorem *close-consistency*:
assumes *conc*: ⟨*consistency C*⟩
shows ⟨*consistency* (*close C*)⟩

apply (*erule conjE′*)
apply (*rule allI impI*)+
apply (*erule allE impE*)+
apply (*erule subsetD*)
apply *assumption*
apply *blast*

{ fix *P*
assume ⟨*Exi P* ∈ *S′*⟩
then have ⟨*Exi P* ∈ *S*⟩
using ⟨*S′* ⊆ *S*⟩ **by** *blast*
then have ⟨∃ *c*. *S* ∪ {*sub 0* (*Fun c* []) *P*} ∈ *C*⟩
using ⟨*S* ∈ *C*⟩ *conc consistency-def* **by** *blast*
then show ⟨∃ *c*. *S′* ∪ {*sub 0* (*Fun c* []) *P*} ∈ *close C*⟩
using ⟨*S′* ⊆ *S*⟩ *subset-in-close* **by** *blast* **}**

Strategi

- Luk formel
- Brug oprindeligt komplethedsbevis
- Genudled oprindelig formel

Universel lukning

Binder alle frie variable til en alkvantor.

$$\forall p(0,1,2) \rightsquigarrow \forall\forall\forall p(0,1,2)$$

Direkte eliminering

$$(\forall\forall p(0,1,2))[2/0] \rightsquigarrow \forall((\forall p(0,1,2))[3/1]) \rightsquigarrow \forall\forall(p(0,1,2)[4/2]) \rightsquigarrow \forall\forall p(0,1,4)$$
$$(\forall p(0,1,4))[1/0] \rightsquigarrow \forall(p(0,1,4)[2/1]) \rightsquigarrow \forall p(0,2,3)$$
$$p(0,2,3))[0/0] \rightsquigarrow p(0,1,2)$$

Ny regel

Udvidelse

inductive *OK-star* :: ‹*fm* ⇒ *fm list* ⇒ *bool*› **where**
Proper: ‹*OK p z* ⟹ *OK-star p z*› |
Subtle: ‹*OK-star p* [] ⟹ *OK-star* (*subc c s p*) []›

Konstanter for kvantorer

fun *consts-for-unis* :: ‹*fm* ⇒ *id list* ⇒ *fm*› **where**
‹*consts-for-unis* (*Uni p*) (*c*#*cs*) =
consts-for-unis (*sub 0* (*Fun c* []) *p*) *cs*› |
‹*consts-for-unis p - = p*›

Variable for konstanter

primrec *vars-for-consts* :: ‹*fm* ⇒ *id list* ⇒ *fm*› **where**
‹*vars-for-consts p* [] = *p*› |
‹*vars-for-consts p* (*c* # *cs*) =
subc c (*Var* (*length cs*)) (*vars-for-consts p cs*)›

Teleskoperen

subc c_0 *(m-1) (subc* c_1 *(m-2) (...* *(subc* c_{m-1} *0 (sub 0* c_{m-1} *...))))*

Parvis annullering

lemma *subc-sub*: ‹$c \notin$ *params p* $\Longrightarrow$ *closed* (*Suc m*) *p* $\Longrightarrow$
subc c (*Var m*) (*sub m* (*Fun c* []) *p*) = *p*›
by (*induct p arbitrary*: *m*) *simp-all*

Annullering

lemma *vars-for-consts-for-unis*:
‹*closed* (*length cs*) *p* $\Longrightarrow$ $\forall$ *c* $\in$ *set cs*. *c* $\notin$ *params p* $\Longrightarrow$ *distinct cs* $\Longrightarrow$
vars-for-consts (*consts-for-unis* (*put-unis* (*length cs*) *p*) *cs*) *cs* = *p*›
using *sub-free-params-all subc-sub* **by** (*induct cs arbitrary*: *p*) *auto*

Udledning af oprindelig formel

```
lemma remove-unis-star:
  assumes ⟨sentence (put-unis m p)⟩ ⟨OK (put-unis m p) []⟩
  shows ⟨OK-star p []⟩
proof −
  obtain cs :: ⟨id list⟩ where ⟨length cs = m⟩
    and ∗: ⟨distinct cs⟩ and ∗∗: ⟨∀ c ∈ set cs. c ∉ params p⟩
    using assms fresh-constants by blast
  then have ⟨OK (consts-for-unis (put-unis (length cs) p) cs) []⟩
    using assms consts-for-unis by blast
  then have ⟨OK-star (vars-for-consts (consts-for-unis
      (put-unis (length cs) p) cs) cs) []⟩
    using Proper vars-for-consts-star by blast
  moreover have ⟨closed (length cs) p⟩
    using assms ⟨length cs = m⟩ closed-put-unis by simp
  ultimately show ⟨OK-star p []⟩
    using vars-for-consts-for-unis ∗ ∗∗ by simp
qed
```

Komplethed

theorem *completeness-star*:
  assumes ‹*infinite* (*UNIV* :: (*'a* :: *countable*) *set*)›
    and ‹∀ (*e* :: *nat* ⇒ *'a*) *f g*. *semantics e f g p*›
  shows ‹*OK-star p* []›
proof −
  obtain *m* **where** ∗: ‹*sentence* (*put-unis m p*)›
    using *ex-closure* **by** *blast*
  moreover have ‹∀ (*e* :: *nat* ⇒ *'a*) *f g*. *semantics e f g* (*put-unis m p*)›
    using *assms valid-put-unis* **by** *blast*
  ultimately have ‹*OK* (*put-unis m p*) []›
    using *assms completeness* **by** *blast*
  then show ‹*OK-star p* []›
    using ∗ *remove-unis-star* **by** *blast*
qed

Konklusion

NaDeA er

- Sundt.
- Komplet for lukkede formler.
  - Deklarativt bevis.
  - Eget valg af (tælleligt uendeligt) domæne.
- Komplet for åbne formler med en udvidelse.
  - Med vilkårlige antagelser.
  - Weakening delresultat.