INF2080 – Logikk og beregninger

Forelesning 18: Ufullstendighet

UiO : Institutt for informatikk

Sist oppdatert: 2012-03-22 10:03

18.1 Ufullstendighet

Setting

Setting

- Stoppeproblemet kan ikke avgjøres
- Vi kan simulere beregninger ved

Setting

- Stoppeproblemet kan ikke avgjøres
- Vi kan simulere beregninger ved
  - termer i første ordens logikk

Setting

- Stoppeproblemet kan ikke avgjøres
- Vi kan simulere beregninger ved
  - termer i første ordens logikk
  - logikk over endelige kjeder — spesielt automater

Setting

- Stoppeproblemet kan ikke avgjøres
- Vi kan simulere beregninger ved
  - termer i første ordens logikk
  - logikk over endelige kjeder — spesielt automater
  - logikk over binære trær

Setting

- Stoppeproblemet kan ikke avgjøres
- Vi kan simulere beregninger ved
  - termer i første ordens logikk
  - logikk over endelige kjeder — spesielt automater
  - logikk over binære trær
- Simuleringene kan ikke avgjøre stoppeproblemet

Setting

- Stoppeproblemet kan ikke avgjøres
- Vi kan simulere beregninger ved
  - termer i første ordens logikk
  - logikk over endelige kjeder — spesielt automater
  - logikk over binære trær
- Simuleringene kan ikke avgjøre stoppeproblemet
- Begrensninger for logikk

Simulering — termer i første ordens logikk

Simulering — termer i første ordens logikk

- Kan simulere beregning på turingmaskin på en automat med to stacker
- Hver stack kan representeres ved en term i språket

Simulering — termer i første ordens logikk

- Kan simulere beregning på turingmaskin på en automat med to stacker
- Hver stack kan representeres ved en term i språket
- Kan simulere en beregning på turingmaskin ved en formel bygd opp ved binære relasjoner og unære funksjonssymboler

Simulering — termer i første ordens logikk

- Kan simulere beregning på turingmaskin på en automat med to stacker
- Hver stack kan representeres ved en term i språket
- Kan simulere en beregning på turingmaskin ved en formel bygd opp ved binære relasjoner og unære funksjonssymboler
- Binær relasjon: representerer innholdet i de to stackene i en tilstand

Simulering — termer i første ordens logikk

- Kan simulere beregning på turingmaskin på en automat med to stacker
- Hver stack kan representeres ved en term i språket
- Kan simulere en beregning på turingmaskin ved en formel bygd opp ved binære relasjoner og unære funksjonssymboler
- Binær relasjon: representerer innholdet i de to stackene i en tilstand
- Unær funksjon: representerer et symbol i alfabetet

Simulering — termer i første ordens logikk

- Kan simulere beregning på turingmaskin på en automat med to stacker
- Hver stack kan representeres ved en term i språket
- Kan simulere en beregning på turingmaskin ved en formel bygd opp ved binære relasjoner og unære funksjonssymboler
- Binær relasjon: representerer innholdet i de to stackene i en tilstand
- Unær funksjon: representerer et symbol i alfabetet
- Formel: $\mathit{START} \wedge \mathit{TRANSISJON} \rightarrow \mathit{FINAL}$

Simulering — termer i første ordens logikk

- Kan simulere beregning på turingmaskin på en automat med to stacker
- Hver stack kan representeres ved en term i språket
- Kan simulere en beregning på turingmaskin ved en formel bygd opp ved binære relasjoner og unære funksjonssymboler
- Binær relasjon: representerer innholdet i de to stackene i en tilstand
- Unær funksjon: representerer et symbol i alfabetet
- Formel: $\mathit{START} \wedge \mathit{TRANSISJON} \rightarrow \mathit{FINAL}$
- Kan ikke avgjøre om formelen er gyldig eller falsifiserbar

Simulering — termer i første ordens logikk

- Kan simulere beregning på turingmaskin på en automat med to stacker
- Hver stack kan representeres ved en term i språket
- Kan simulere en beregning på turingmaskin ved en formel bygd opp ved binære relasjoner og unære funksjonssymboler
- Binær relasjon: representerer innholdet i de to stackene i en tilstand
- Unær funksjon: representerer et symbol i alfabetet
- Formel: $\mathit{START} \wedge \mathit{TRANSISJON} \rightarrow \mathit{FINAL}$
- Kan ikke avgjøre om formelen er gyldig eller falsifiserbar
- Kan ikke avgjøre om analysetreet er endelig eller uendelig

Simulering — termer i første ordens logikk

- Kan simulere beregning på turingmaskin på en automat med to stacker
- Hver stack kan representeres ved en term i språket
- Kan simulere en beregning på turingmaskin ved en formel bygd opp ved binære relasjoner og unære funksjonssymboler
- Binær relasjon: representerer innholdet i de to stackene i en tilstand
- Unær funksjon: representerer et symbol i alfabetet
- Formel: $START \wedge TRANSISJON \rightarrow FINAL$
- Kan ikke avgjøre om formelen er gyldig eller falsifiserbar
- Kan ikke avgjøre om analysetreet er endelig eller uendelig

Entscheidungsproblem vist uavgjørbart av Turing

Datastruktur — binære trær

Datastruktur — binære trær

Syntaks: Δ_0 — ikke ubegrensete kvantorer

Beregnbart: Σ_1 — $\exists$ utenfor Δ_0

Datastruktur — binære trær

Syntaks: Δ_0 — ikke ubegrensete kvantorer

Beregnbart: Σ_1 — $\exists$ utenfor Δ_0

Sannhet / usannhet av Δ_0-setninger kan avgjøres — bruk endelig tre med OG-noder og ELLER-noder.

Datastruktur — binære trær

Syntaks: Δ_0 — ikke ubegrensete kvantorer

Beregnbart: Σ_1 — $\exists$ utenfor Δ_0

Sannhet / usannhet av Δ_0-setninger kan avgjøres — bruk endelig tre med OG-noder og ELLER-noder.
Alternativ — bruk følgende aksiom system

Datastruktur — binære trær

Syntaks: Δ_0 — ikke ubegrensete kvantorer

Beregnbart: Σ_1 — $\exists$ utenfor Δ_0

Sannhet / usannhet av Δ_0-setninger kan avgjøres — bruk endelig tre med OG-noder og ELLER-noder.
Alternativ — bruk følgende aksiom system

1. $\forall x, y. \neg \mathbf{nil} = \mathbf{cons}(x, y)$

Datastruktur — binære trær

Syntaks: Δ_0 — ikke ubegrensete kvantorer

Beregnbart: Σ_1 — $\exists$ utenfor Δ_0

Sannhet / usannhet av Δ_0-setninger kan avgjøres — bruk endelig tre med OG-noder og ELLER-noder.
Alternativ — bruk følgende aksiom system

1. $\forall x, y. \neg \mathbf{nil} = \mathbf{cons}(x, y)$
2. $\forall x, y, u, v. (\mathbf{cons}(x, y) = \mathbf{cons}(u, v) \rightarrow x = u \vee y = v)$

Datastruktur — binære trær

Syntaks: Δ_0 — ikke ubegrensete kvantorer

Beregnbart: Σ_1 — $\exists$ utenfor Δ_0

Sannhet / usannhet av Δ_0-setninger kan avgjøres — bruk endelig tre med OG-noder og ELLER-noder.
Alternativ — bruk følgende aksiom system

1. $\forall x, y.\neg \mathbf{nil} = \mathbf{cons}(x, y)$
2. $\forall x, y, u, v.(\mathbf{cons}(x, y) = \mathbf{cons}(u, v) \rightarrow x = u \vee y = v)$
3. $\forall x.(x = \mathbf{nil} \vee \exists u, v.x = \mathbf{cons}(u, v))$

Datastruktur — binære trær

Syntaks: Δ_0 — ikke ubegrensete kvantorer

Beregnbart: Σ_1 — $\exists$ utenfor Δ_0

Sannhet / usannhet av Δ_0-setninger kan avgjøres — bruk endelig tre med OG-noder og ELLER-noder.
Alternativ — bruk følgende aksiom system

1. $\forall x, y.\neg \mathbf{nil} = \mathbf{cons}(x, y)$
2. $\forall x, y, u, v.(\mathbf{cons}(x, y) = \mathbf{cons}(u, v) \rightarrow x = u \vee y = v)$
3. $\forall x.(x = \mathbf{nil} \vee \exists u, v.x = \mathbf{cons}(u, v))$
4. $\forall x.\neg x \prec \mathbf{nil}$

Datastruktur — binære trær

Syntaks: Δ_0 — ikke ubegrensete kvantorer

Beregnbart: Σ_1 — $\exists$ utenfor Δ_0

Sannhet / usannhet av Δ_0-setninger kan avgjøres — bruk endelig tre med OG-noder og ELLER-noder.
Alternativ — bruk følgende aksiom system

1. $\forall x, y.\neg \mathbf{nil} = \mathbf{cons}(x, y)$
2. $\forall x, y, u, v.(\mathbf{cons}(x, y) = \mathbf{cons}(u, v) \rightarrow x = u \vee y = v)$
3. $\forall x.(x = \mathbf{nil} \vee \exists u, v.x = \mathbf{cons}(u, v))$
4. $\forall x.\neg x \prec \mathbf{nil}$
5. $\forall x, u, v.(x \prec \mathbf{cons}(u, v) \rightarrow x \preceq u \vee x \preceq v)$

Datastruktur — ufullstendighet

For datastrukturen binære trær gjelder

Datastruktur — ufullstendighet

For datastrukturen binære trær gjelder

- Σ_1-setninger for beregnbarhet

Datastruktur — ufullstendighet

For datastrukturen binære trær gjelder

- Σ_1-setninger for beregnbarhet
- Π_1-setninger for ikke-beregnbarhet

Datastruktur — ufullstendighet

For datastrukturen binære trær gjelder

- Σ_1-setninger for beregnbarhet
- Π_1-setninger for ikke-beregnbarhet
- Fins en (beregnbar) aksiomatisering av de sanne Δ_0-setningene

Datastruktur — ufullstendighet

For datastrukturen binære trær gjelder

- Σ_1-setninger for beregnbarhet
- Π_1-setninger for ikke-beregnbarhet
- Fins en (beregnbar) aksiomatisering av de sanne Δ_0-setningene
- Fins ingen (beregnbar) aksiomatisering av de sanne Π_1-setningene

Datastruktur — ufullstendighet

For datastrukturen binære trær gjelder

- Σ_1-setninger for beregnbarhet
- Π_1-setninger for ikke-beregnbarhet
- Fins en (beregnbar) aksiomatisering av de sanne Δ_0-setningene
- Fins ingen (beregnbar) aksiomatisering av de sanne Π_1-setningene

Tilsvarende gjelder ikke for datastrukturen unære tall — en må ha med addisjon og multiplikasjon for å beskrive syntaks

Datastruktur — ufullstendighet

For datastrukturen binære trær gjelder

- Σ_1-setninger for beregnbarhet
- Π_1-setninger for ikke-beregnbarhet
- Fins en (beregnbar) aksiomatisering av de sanne Δ_0-setningene
- Fins ingen (beregnbar) aksiomatisering av de sanne Π_1-setningene

Tilsvarende gjelder ikke for datastrukturen unære tall — en må ha med addisjon og multiplikasjon for å beskrive syntaks
Dette er en enkel versjon av Gödels ufullstendighets teorem