

DM517:Supplerende noter om uafgørlighedsbeviser:

Jørgen Bang-Jensen

September 29, 2011

Abstract

Formålet med denne note er at give en form for kogebogsopskrift på, hvorledes man bygger et uafgørlighedsbevis op. Noterne er på dansk, da de er en omskrivning af et gammelt notesæt¹.

1 Notation

- decidable (afgørligt)=Turing-computable (Turing beregnelig)=rekursive(rekursiv).
- semi-decidable (semi-afgørligt)=partially Turing computable= Turing-enumerable= recursively enumerable (rekursivt enumerabel)

2 Reduktioner

Lad os starte med at genkalde Definition 5.20 fra Lærebogen:

Lad $L_1, L_2 \subseteq \Sigma^*$ være sprog. En **reduktion fra L_1 til L_2** er en Turing beregnelig funktion $\tau : \Sigma^* \rightarrow \Sigma^*$ som opfylder at $x \in L_1$ hvis og kun hvis $\tau(x) \in L_2$.

Bemærk, at vi kræver at der findes en Turing Maskine som beregner τ (dvs som starter med input x og stopper med $\tau(x)$ på sit bånd). Dette er meget vigtigt og det betyder, at vi, før vi kan bruge en reduktion, skal kunne gøre rede for, at den tilsvarende funktion τ faktisk kan beregnes af en Turing maskine. Typisk gøres dette ved en diskution i ord af, hvad den Turing maskine der beregner τ , skal kunne gøre og argumentere for, at disse ting er mulige (se eksemplerne nedenfor).

Lemma 1 *Hvis L_2 er afgørligt og L_1 kan reduceres til L_2 vha en Turing beregnelig funktion τ , så er også L_1 afgørligt.*

Bevis: Ideen i beviset kan fint illustreres skematisk som vist i Figur 1. Her er \mathcal{A} en Turing maskine, der omdanner en streng $x \in \Sigma^*$ til strengen $\tau(x)$. Denne streng gives så videre til M_2 , som afgør L_2 . Heraf fås, at M_2 svarer ja på input $\tau(x)$, præcis hvis

¹If non-danish reading persons have trouble with this, please ask for explanation where needed

$\tau(x) \in L_2$ og nej ellers. Dvs den samlede maskine M^* , som er beskrevet i diagramform i Figur 1, har egenskaben, at den altid stopper og den svarer ja præcis når $x \in L_1$. Altså M^* afgør L_1 . \diamond

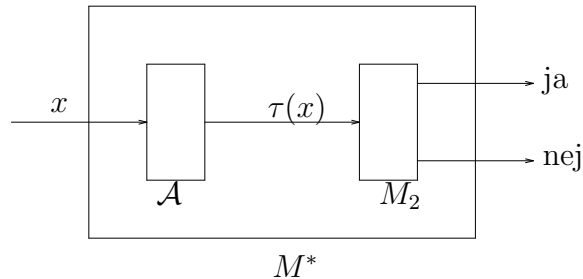


Figure 1: Turing maskinen M^* afgør L_1 , forudsat at M_2 findes.

Hvordan kan vi bruge Lemma 1 til uafgørligheds beviser?

Hvis vi kender et sprog L , som er uafgørligt og kan vise at L kan reduceres til sproget L' vha en Turing beregnelig funktion τ , så kan vi bruge Lemma 1 til at konkludere, at sproget L' er uafgørligt: lemmaet siger, at hvis L' er afgørligt, så er også L afgørligt, hvilket er en modstrid, da vi allerede ved, at L er uafgørligt.

3 Opskrift på et uafgørlighedsbevis:

Lad K være et sprog, vi ønsker at vise er uafgørligt. Det vil sige, vi vil vise, at der ikke kan findes en Turing maskine, som givet en streng x over samme alfabet som K , kan afgøre om $x \in K$.

1. Vælg et sprog R som vides at være uafgørligt.
2. Beskriv en reduktion fra R til K vha en Turing beregnelig funktion. Lad \mathcal{A} være en Turing maskine der beregner denne reduktion. Dvs givet x beregner \mathcal{A} strengen $\tau(x)$, som har egenskaben at $x \in R$ hvis og kun hvis $\tau(x) \in K$. Der skal redegøres for, hvordan \mathcal{A} kan realiseres.
3. Lad M_K være en hypotetisk Turing maskine der afgør K (vi ønsker at vise at M_K ikke findes).
4. Brug \mathcal{A} og M_K som vist i Figur 1 (med $M_2 = M_K$) til at lave en Turing maskine M^* der afgør R . M^* afgør R thi vi har argumenteret for at $x \in R$ hvis og kun hvis $\tau(x) \in K$.
5. Konkluder ud fra ovenstående modstrid at M_K ikke kan findes, hvorfor K er uafgørligt.

Bemærkninger:

- Det er ikke altid oplagt hvilket sprog man skal vælge som R . Pointen er, at man skal vælge noget, for hvilket det kan lade sig gøre at finde en reduktion. Selvfølgelig skal man bruge egenskaber ved sproget K til at hjælpe med at vælge det rette sprog R .

Kun træning med eksempler kan hjælpe med at blive god til denne disciplin!

- Af og til er det lettere at beskrive en reduktion af R til \bar{K} (komplementet af sproget K). Så skal man blot lade $M_{\bar{K}}$ være en hypotetisk maskine som afgør \bar{K} . En sådan Turing maskine findes hvis og kun hvis K er afgørligt, da afgørlige sprog er lukket under komplement. I nogle af mine diagrammer ved forelæsningerne bruger jeg stadig en hypotetisk Turing maskine for K og bytter så om på ja og nej tilsidst (output fra den store kasse). Det er stadig i orden som bevis (selvom det er bedre at gøre som ovenfor) og I er velkommen til at gøre dette, forudsat at I stadig kan argumentere for korrektheden!

4 Eksempler på uafgørlighedsbeviser.

Her følger tre eksempler, som er repræsentative, men selvfølgelig ikke udtømmende. Husk f.eks at Rice's sætning (Opgave 5.28 i bogen) er et andet nyttigt værktøj, som I må bruge medmindre det eksplicit forbydes i den pågældende opgave. **Husk at Rice's sætning kun kan bruges på spørgsmål som omhandler egenskaber ved sprog for Turing maskiner!** Dvs vi kigger på egenskaber ved $L(M)$ for en Turing maskine M som er givet ved sin universelle kodning $\langle M \rangle$. Et eksempel hvor Rice's sætning kan anvendes er til at vise at følgende sprog er uafgørligt: $L = \{ \langle M \rangle \mid L(M) \text{ indeholder to strenge af samme længde} \}$.

(A) Lad $H_\epsilon = \{ \langle M \rangle \mid M \text{ stopper på den tomme streng} \}$. Vi viser at H_ϵ er uafgørligt vha opskriften ovenfor.

1. Lad R være sproget $HALT_{TM} = \{ \langle M, w \rangle \mid M \text{ stopper på } w \}$. Vi ved fra Theorem 5.1, at $HALT_{TM}$ er uafgørligt.
2. Lad \mathcal{A} være en algoritme som givet koden $\langle M \rangle$ for en Turing maskine M og koden $\langle w \rangle$ for en streng w laver koden $\langle M_w \rangle$ for Turing maskinen M_w , der når den startes på den tomme streng, først simulerer M på w og stopper hvis (og kun hvis) M stopper på w . Denne Turing maskine er let at lave: M_w har koden $\langle U \rangle$ for den universelle TM indbygget i sig og på input ϵ starter M_w blot med at kopiere w op på sit input bånd, hvorefter den bruger U til at simulere M på w . M_w har nu følgende egenskab: **M_w stopper på den tomme streng hvis og kun hvis M stopper på w .**
3. Antag at M_{H_ϵ} afgør H_ϵ .
4. Brug \mathcal{A} og M_{H_ϵ} (med $M_2 = M_{H_\epsilon}$) til at lave en Turing maskine M^{HALT} der afgør $HALT_{TM}$.
5. Da $HALT_{TM}$ er uafgørligt kan M_{H_ϵ} ikke findes og det følger, at H_ϵ er uafgørligt.

(B) Lad $L = \{ \langle M \rangle \mid M \text{ stopper på alle strenge af længde } 22 \}$. Vi viser at L er uafgørligt vha opskriften ovenfor.

1. Lad R være tomstrenghaltingsproget H_ϵ som vi lige har vist er uafgørligt.
2. Lad \mathcal{B} være algorithmen, som givet koden for en Turing maskine M , laver koden for en Turing maskine M_{22} , som stopper på en given streng w , hvis og kun hvis M stopper på den tomme streng og $|w| = 22$. Mere præcist kan M_{22} beskrives som i Figur 2.

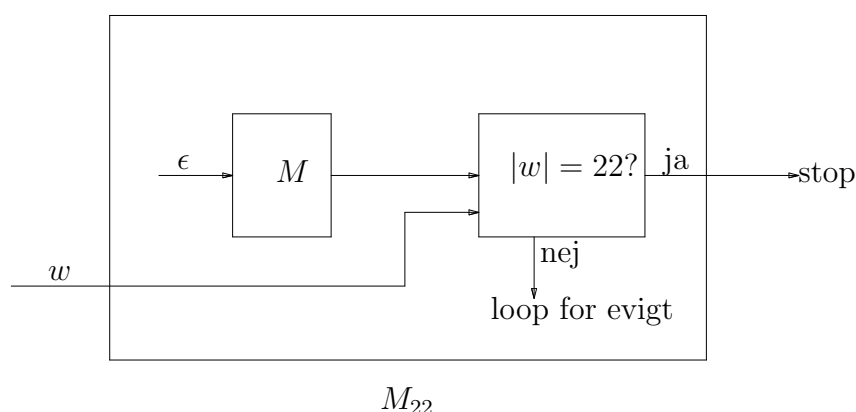


Figure 2: Skematisk beskrivelse af Turing maskinen M_{22} .

M_{22} simulerer først M på den tomme streng. Hvis M accepterer ϵ så (og først da!) undersøger M_{22} om dens input har længde 22. Hvis $|w| = 22$ stopper M_{22} og ellers looper den uendeligt. Dvs

$$L(M_{22}) = \begin{cases} \emptyset & \text{hvis } \langle M \rangle \notin H_\epsilon \\ \{w \in \Sigma^* : |w| = 22\} & \text{hvis } \langle M \rangle \in H_\epsilon \end{cases}$$

Det følger af ovenstående at $\langle M \rangle \in H_\epsilon$ hvis og kun hvis $\langle M_{22} \rangle \in L$. Fra ovenstående beskrivelse af M_{22} (samt lidt flere detaljer i ord om hvorledes man givet M kan lave M_{22}) er det klart at algoritmen \mathcal{B} som beregner en funktion τ der sender $\langle M \rangle$ over i strengen $\langle M_{22} \rangle$ kan realiseres vha en Turing maskine.

3. Antag at M_L afgør L .
4. Sæt \mathcal{B} og M_L sammen som vist i Figur 3.
Den resulterende Turing maskine M^* afgør H_ϵ da vi har vist at $\langle M \rangle \in H_\epsilon$ hvis og kun hvis $\langle M_{22} \rangle \in L$.
5. Konkluder at M_L ikke kan findes og dermed at L er uafgørligt.

Bemærk at uafgørligheden af dette L også kan udledes af Rice's sætning!

(C) Lad $Q = \{ \langle M \rangle \mid \exists w \text{ så } M \text{ vil gennemløbe alle sine tilstande på input } w \}$. Vi viser at Q er uafgørligt vha opskriften ovenfor.

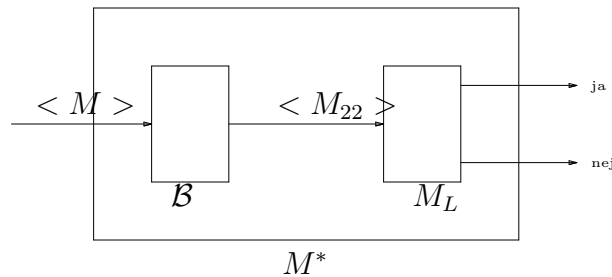


Figure 3: Konstruktionen af M^* der afgør H hvis M_L findes.

1. Lad igen R være (tomme strengs) haltingsproget H_ϵ . Vi ved fra (A) at dette sprog er uafgørligt.
2. Lad \mathcal{C} være algoritmen, som givet koden for en Turing maskine M , laver koden for en Turing maskine M_V , der gennemløber alle sine tilstande, præcis hvis M stopper på den tomme streng. Dette gøres ved at lade M_V starte med at simulere M på ϵ ved hjælp af en ægte delmængde af M_V 's tilstande (let at realisere, tilføj blot en special tilstand q^* , der først bruges efter at M er stoppet, hvis den gør det). Det er også let at sikre at en vi får gennemløbet alle tilstande f.eks ved at lave M om så den i stedet for at stoppe skriver et specielt symbol α på M_V 's bånd og så indrette M_V således at den på symbolet α fra en vilkårlig tilstand vil gennemløbe alle sine tilstande og derefter stoppe. Lad τ være den funktion der omdanner $\langle M \rangle$ til $\langle M_V \rangle$. Det er let at argumentere for at τ kan beregnes af en Turing maskine \mathcal{C} . Vi har nu at $x \in H_\epsilon$ hvis og kun hvis $\tau(x) \in Q$, thi special tilstanden q^* vil blive gennemløbet af M_V på input w hvis og kun hvis den vil blive det for et vilkårlig andet input w' og dette sker hvis og kun hvis M stopper på ϵ .
3. Antag at M_Q afgør Q .
4. Sæt \mathcal{C} og M_Q sammen som vist i Figur 4.

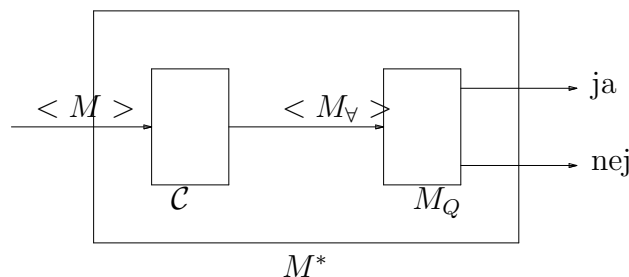


Figure 4: Konstruktionen af M^* der afgør H_ϵ hvis M_Q findes.

Den resulterende Turing maskine M^* afgør H_ϵ da vi har vist at $\langle M \rangle \in H_\epsilon$ hvis og kun hvis $M_V \in Q$.

5. Konkluder at M_Q ikke kan findes, dvs Q er uafgørligt.