DM17:Supplerende noter om uafgørligheds beviser:

formålet med denne note er at give de studerende en form for kogebogsopskrift på, hvorledes man bygger et uafgørlighedsbevis op.

Lad os starte med at genkalde definition 5.4.1 fra Lærebogen:

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

Bemærk at vi kræver at τ er rekursiv, dvs at der findes en Turing Maskine som beregner τ. Dette er meget vigtigt og det betyder, at vi for at 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 0.1 *Hvis L_2 er afgørligt og L_1 kan reduceres til L_2 vha en rekursiv funktion τ, så er også L_1 afgørligt.*

Bevis: Ideen i beviset kan udemærket illustreres skematisk som vist i Figur **??**. 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 $\tau(x) \in L_2$ og nej ellers.

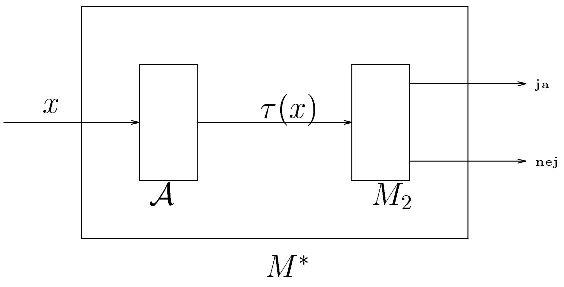


Figure 1: Turing maskinen M^* afgør L_1, forudsat at M_2 findes.

Dvs den samlede maskine M^* som er beskrevet i diagramform i Figur **??** har egenskaben at den altid stopper og svarer ja præcis når $x \in L_1$. Altså M^* afgør L_1. $\diamond$

Hvordan kan vi bruge Lemma ?? 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 **??** til at konkludere at sproget L' er uafgørligt, thi lemmaet siger at hvis L' er afgørligt, så er også L afgørligt, hvilket er en modstrid.

Opskrift på et uafgørlighedsbevis:

Lad K være et sprog vi ønsker at vise er uafgørligt, dvs 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 **??** (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!

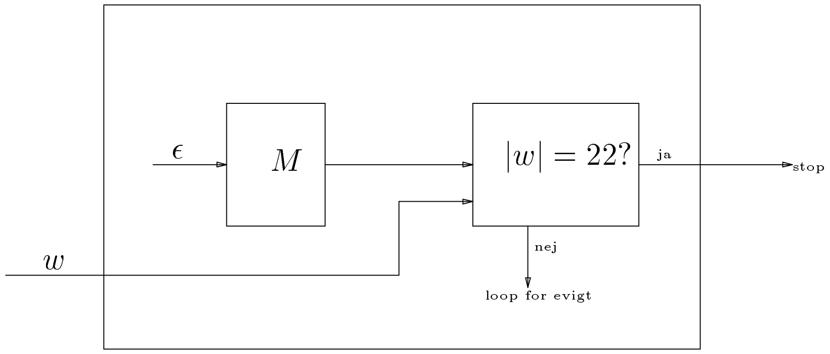
- Af og til er det lettere at beskrive en reduktion af R til $\bar{K}$. Så skal man blot lade $M_{\bar{K}}$ være en hypotetisk maskine som afgør $\bar{K}$. En sådan turing maskine findes hvis og kun hvis K er afgørligt, da afgørlige sprog er lukket under komplement. I nogle af mine diagrammer ved forelæsningerne 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 iorden 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!

Eksempler på uafgørlighedsbeviser.

Her følger to eksempler som er repræsentative, men selvfølgelig ikke udtømmende. Husk f.eks at Rice's sætning 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 sprog for turing maskiner!

- Lad $L = \{''M''|M$ stopper på alle strenge af længde 22 $\}$. Vi viser at L er uafgørligt vha opskriften ovenfor.

  1. Lad R være (tomme strengs) haltingsproget H_ϵ, dvs $R = H_\epsilon = \{''M'' : M$ stopper på den tomme streng$\}$. Vi ved fra Theorem 5.4.2(b) at dette sprog 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 **??**.



M_{22}

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

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

  Det følger af ovenstående at $''M'' \in H_\epsilon$ hvis og kun hvis $''M''_{22} \in L$. 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 $''M''$ over i strengen $''M''_{22}$ 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 **??**.
     Den resulterende turing maskine M^* afgør H_ϵ da vi har vist at $''M'' \in H_\epsilon$ hvis og kun hvis $''M''_{22} \in L$.
  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!

- Lad $Q = \{''M''|\exists w$ så M vil gennemløbe alle sine tilstande på input $w\}$. Vi viser at Q er uafgørligt vha opskriften ovenfor.

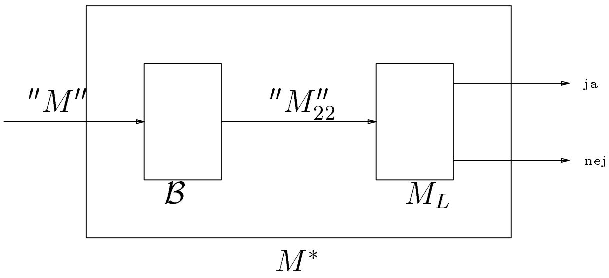


Figure 3: Konstruktionen af M^* der afgør H_ϵ hvis M_L findes.

1. Lad igen R være (tomme strengs) haltingsproget H_ϵ. Vi ved fra Theorem 5.4.2(b) 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 $M_\forall$, der gennemløber alle sine tilstande præcis hvis M stopper på den tomme streng. Dette gøres ved at lade $M_\forall$ starte med at simulere M på ϵ ved hjælp af en **ægte** delmængde af $M_\forall$'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 $M_\forall$'s bånd og så indrette $M_\forall$ 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 “M'' til “$M''_\forall$. 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 $M_\forall$ 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 **??**.

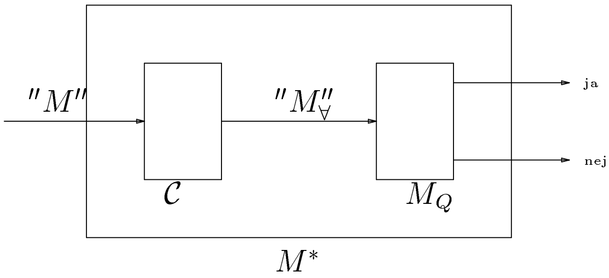


Figure 4: Konstruktionen af M^* der afgør H_ϵ hvis M_Q findes.

Den resulterende turing maskine M^* afgør H_ϵ da vi har vist at $''M'' \in H_\epsilon$ hvis og kun hvis $M_\forall \in Q$.

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