AB: gekwoteerde oefenzitting 2
3 dec 2018 8u30/9u tot 10u25

1 Beslisbaarheid

Bewijs voor de volgende talen of dat ze 1) beslisbaar, 2) niet-beslisbaar maar wel herkenbaar, of 3) niet-beslisbaar maar wel co-herkenbaar zijn. Geef zeker een beschrijving van een beslisser (1), een bewijs van onbeslisbaarheid en een beschrijving van een herkenner (2), of een bewijs van onbeslisbaarheid en een beschrijving van een co-herkenner (3).

Duidt het juiste antwoord aan en geef daaronder een bewijs. M is steeds een Turing machine, Σ het invoeralfabet en L_M de taal bepaalt door M.

1. $\{\langle M\rangle \mid \forall s \in L_M : |s| \mod 2 == 0\}$

   ☐ beslisbaar ☐ niet-beslisbaar maar wel herkenbaar ☐ niet-beslisbaar maar wel co-herkenbaar

   Antwoord: Deze eigenschap is niet triviaal en taalinvariant, dus voldoet het aan de stelling van Rice. Dus is dit niet beslisbaar.

   Herkenbaar???

   Co-herkenbaar? Ja, bouw een machine die input M simuleert op elke eindige string van oneven lengte, voor onbegrensde tijd; indien een string gevonden wordt die door M aanvaard wordt, antwoord 0. Deze machine stopt asa M een string van oneven lengte aanvaardt. In dit geval wordt M gereject.

   Hoe M simuleren op elke eindige string van oneven lengte voor onbegrensde tijd? De verzameling van strings van oneven lengte is regulier en er bestaat een enumereermachine voor. Zij $st_0, st_1, st_2, \ldots$ de enumerering ervan. Simuleer M 1 stap op st_0, vervolgens 2 stappen op st_0 en 2 stappen op st_1, enzovoort, n stappen op elk van $st_0, \ldots, st_{n-1}$ voor toenemende n, totdat een string aanvaard wordt of tot oneindig. Zo wordt uiteindelijk M voor een onbegrensde tijd gesimuleerd op elk van deze strings.

   Herkenbaar? Nee, want co-herkenbaar en niet beslisbaar.

2. $\{\langle M\rangle \mid M$ heeft exact 5 toestanden$\}$

   ☐ beslisbaar ☐ niet-beslisbaar maar wel herkenbaar ☐ niet-beslisbaar maar wel co-herkenbaar

   Antoord: beslisbaar, de codering van M expliciteert de toestanden ervan. Deze eigenschap is niet taal-invariant.

3. $\{\langle M, s\rangle \mid$ als je M uitvoert op s komt M maar een keer in zijn begintoestand$\}$

   ☐ beslisbaar ☐ niet-beslisbaar maar wel herkenbaar ☐ niet-beslisbaar maar wel co-herkenbaar

   Antwoord: niet beslisbaar: we reduceren het Halting problem naar dit probleem. We bouwen een Turing transformatie-machine die voor input X eerst test of X van de vorm $< M, s >$ is met M de codering van een Turing machine. Indien niet dan zet de transformatie machine X op de output en stopt. Indien wel, dan berekent de transformatie-machine een output $< M', s >$ en stopt. Hierbij is M' de codering van een Turing machine met begintoestand q_0' zodat M stopt op input s asa M' bij input s tweemaal door q_0' gaat.

   De transformatie-machine transformeert M tot M' door eerst een nieuwe starttoestand q_0' in te voeren met dezelfde vertrekkende transities als de oorspronkelijke starttoestand q_0 van M; dus heeft q_0' tot dusver geen inkomende transities vanuit toestanden van M. Voor de accept en reject toestanden q_a, q_r van M voegen we elk een nieuwe transitie naar q_0' toe. Dus, M bij invoer s stopt (in q_a of q_s) asa M' passeert tweemaal door q_0'. Om het werk af te maken moeten de accept en reject toestand van M' gedefinieerd worden, anders hebben we strikt genomen geen Turing machine. We voegen daarvoor twee nieuwe (onbereikbare) toestanden q_a', q_r' toe. Het resultaat is M'.

   Wel co-herkenbaar: Simuleer M op s; stop bij tweede doorgang door begintoestand.

2 Delphi

1. Beschouw $L_S = \{\langle M, n, s\rangle \mid M \text{ is een TM}, n \in \mathbb{N}, s \in \Sigma^*, M \text{ aanvaardt } s \text{ in exact } n \text{ stappen}\}$ voor een alfabet Σ. Stel dat je een orakel O_S ter beschikking hebt om te bepalen of dat een string in L_S zit, is het dan mogelijk om een beslisser voor het halting probleem H_{TM} te construeren? Duidt het juiste antwoord aan en geef daaronder een bewijs.

   ☐ ik kan een beslisser bouwen ☐ ik kan nog steeds geen beslisser bouwen

   Antwoord: ik kan nog steeds geen beslisser bouwen, want L_S is beslisbaar met Turing machine: simuleer M voor n stappen op s. Dus als we het halting probleem zouden kunnen oplossen met dit orakel, dan ook zonder dit orakel.

2. Kan je een beslisser bouwen voor E_{TM}, gebruikmakende van een orakel O_H voor H_{TM}?

   ☐ ik kan een beslisser bouwen ☐ ik kan nog steeds geen beslisser bouwen

   Antwoord: E_{TM} is co-herkenbaar: bouw een machine die input M voor onbegrensde tijd simuleert op alle strings (hoe? zie vraag 1.1!); stop indien een string gevonden wordt die door M aanvaard wordt. Deze machine stopt niet voor invoer M asa L_M leeg is. Geef deze machine aan het orakel.

3 Hoe meer talen, hoe meer vreugd

1. Gegeven twee niet-lege talen L_1 en L_2 en de veronderstelling dat de taal $L_1 \times L_2$ herkenbaar is, bewijs dat de talen L_1 en L_2 dan ook altijd herkenbaar zijn **en** beschrijf een herkenner voor L_1.

   Ter verduidelijking: $L_1 \times L_2 = \{(s_1, s_2) \mid s_1 \in L_1, s_2 \in L_2\}$.

   Antwoord. L_2 is niet leeg, dus neem $S_2 \in L_2$. Dan geldt $s_1 \in L_1$ asa $(s_1, S_2) \in L_1 \times L_2$.

   Bouw een machine die voor invoer s_1, het tuppel (s_1, S_2) op de tape zet en vervolgens de herkenner van $L_1 \times L_2$ oproept.

   Hier zit een staartje aan vast: wat indien we geen $S_2 \in L_2$ *kennen*? Dan kunnen we de bovenstaande machine niet construeren. Wat we wel kunnen doen is vooraf een paar $(S_1, S_2) \in L_1 \times L_2$ berekenen door de herkenner van $L_1 \times L_2$ te simuleren voor alle paren (S_1, S_2) voor onbegrensde tijd. Aangezien $L_1 \times L_2$ niet leeg is, zal ooit een paar gevonden worden, en dan is een string S_2 gekend die we vervolgens kunnen gebruiken in de machine van de eerste oplossing.

   Stel dat we weten dat L_2 niet leeg is, maar niet of L_1 leeg is. Dan zitten we vast. Dan levert bovenstaande constructie niet met zekerheid een herkenner op voor L_1. Maar als L_1 niet leeg is, zal het wel een herkenner zijn. Kortom, in dit geval weten we niet a priori of de tweede machine een herkenner is of niet.

2. Gegeven twee herkenbare talen L_1 en L_2, is de taal $L_C = \{s_1 s_2 \mid s_1 \in L_1, s_2 \in L_2\}$ (concatenatie van strings) 1) altijd herkenbaar of 2) niet altijd herkenbaar? Duidt jouw antwoord aan en bewijs en beschrijf een herkenner (1) of geef (tegen)voorbeelden (2).

   ☐ altijd herkenbaar ☐ some wel, soms niet

   Antwoord: altijd herkenbaar. Bouw een machine die voor input s alle mogelijke splits (s_1, s_2) berekent en vervolgens simultaan voor elke split (s_1, s_2) de herkenner van L_1 oproept over s_1 en de herkenner van L_2 op s_2. Deze machine stopt en accepteert als voor een split de beide herkenners stoppen en accepteren. Deze machine stopt en accepteert s asa s behoort tot L_C.

3. Gegeven een niet beslisbare taal L_1 en een oneindige beslisbare taal L_2 die verschillend is van Σ^*, is de taal $L_1 \cap L_2$ dan 1) altijd beslisbaar, 2) nooit beslisbaar of 3) hangt het af van de talen in kwestie? Duidt jouw antwoord aan en bewijs of geef (tegen)voorbeelden.

   ☐ altijd beslisbaar ☐ nooit beslisbaar ☐ hangt af van de talen

Antwoord: hangt af van de talen. Stel dat L_1 een deel is van L_2 , dan is $L_1 \cap L_2 = L_1$ niet beslisbaar. Stel dat L_2 een deel is van L_1, dan is $L_1 \cap L_2 = L_2$ wel beslisbaar.

Is het mogelijk dat een onbeslisbare L_1 een deel is van een beslisbare L_2. Ja. aangezien Σ^* beslisbaar is bestaat er minstens 1 string $s \notin L_1$. Neem $L_2 = \Sigma^* \setminus \{s\}$. L_2 is beslisbaar en voldoet aan de conditie van niet gelijk aan Σ^* te zijn. Er geldt $L_1 \subset L_2$.

Een concreet voorbeeld: neem $L_1 = H_{TM}$, en neem $L_2 = L_{TM} = \{< M, s > |\ M$ is de codering van een Turing machine, s een string $\}$; dan geldt $L_1 \subset L_2$, en L_2 is beslisbaar.

Is het mogelijk dat een onbeslisbare L_1 een beslisbaar oneindige subtaal L_2 heeft? Wel, ik kan niet bewijzen dat elke onbeslisbare L_1 een oneindige beslisbare deeltaal L_2 heeft. Maar het is wel in te zien dat er vele oneindige onbeslisbare L_1 bestaan die een oneindige beslisbare deeltaal L_2 bezitten. Vertrek daarvoor van een willekeurige oneindige beslisbare L_2 zodat $(L_2)^c$ ook oneindig is. Bv. de taal $\{ss \mid s \in \Sigma^*\}$. Dan zijn er overaftelbaar veel talen L_1 zodat $L_2 \subset L_1 \subset \Sigma^*$. Aangezien er maar aftelbaar veel Turing machines zijn, moeten er overaftelbaar veel onbeslisbare talen L_1 bestaan zodat $L_2 \subset L_1$.

Een concreet voorbeeld: neem $L_2 = (L_{TM})^c$ en $L_1 = H_{TM} \cup L_2$. Dan geldt dat L_2 beslisbaar is, $L_2 \subseteq L_1$ en L_1 is niet beslisbaar (aangezien $H_{TM} = L_1 \cap L_{TM}$ anders ook beslisbaar zou zijn).