Obligatorisk oppgave 2 i INF2080, våren 2012

22. mars 2012

Oppgave 1

Vi skal lage en turingmaskin som krypterer beskjeder over alfabetet $\mathcal{A} = \{a, b, c\}$. For å kunne anvende konvensjonelle funksjoner, ønsker vi å oversette beskjedene til alfabetet $\mathcal{B} = \{0, 1\}$ og tilbake. Vi lar derfor tapealfabetet til maskinen være $\mathcal{S} = \mathcal{A} \cup \mathcal{B} \cup \{\#\}$. Vi tar med # som et ekstra tegn som feks. kan brukes til å holde orden på hvlilke symoler som er lest.

a) Lag en maskin som oversetter et ord $w \in \mathcal{A}^*$ til et ord $w' \in \mathcal{B}^*$, etter følgende tabell:

a	b	c
10	01	00

Ved start vil leserhodet stå på første tegn fra høyre i input-ordet. Rydd opp på tapen etter at oversettelsen er gjort. Når maskinen terminerer skal det kun stå det oversatte ordet w' på strengen. La også leserhodet avslutte på det første tegnet fra høyre i w'. Kall denne maskinen $\mathtt{Trans}$.

b) Lag en maskin som beregner XOR-produktet $\oplus$ av to tall over alfabetet $\mathcal{B}$. Bitvis-XOR er gitt ved følgende tabell:

$\oplus$	0	1
0	0	1
1	1	0

Ved start vil de to input-tallene stå intill hverandre, adskilt av en tom rute. Leserhodet står da helt til høyre i input-tallet til høyre. Når maskinen terminerer skal kun resultatet stå igjen på tapen, og leserhodet skal stå på første siffer i resultatet. Kall denne maskinen XOR.

c) Vi antar at vi har en maskin $\mathtt{wrt}_n$ som skriver tallet n på tapen, for et vilkårlig tall n. Denne maskinen starter å skrive der lesehodet står i det maskinen blir kjørt. Maskinen avslutter med lesehodet stående på første siffer i det tallet som er skrevet. Anta også at vi har en maskin $\mathtt{Trans}^{-1}$ som oversetter tall tilbake til strenger fra alfabetet $\mathcal{A}$, tilsvarende som $\mathtt{Trans}$.

Vi ønsker nå å sette sammen disse maskinene til en krypteringsmaskin, som krypterer strenger over alfabetet $\mathcal{A}$, med en nøkkel n. Først må vi oversette strengen til alfabetet $\mathcal{B}$ med $\mathtt{Trans}$, deretter skriver vi nøkkelen n på tapen med wrt_n (merk at tallet n her må ha like mange bits som tallet vi får etter oversettelsen), sender disse to tallene gjennom XOR, og til sist oversetter resultatet tilbake med $\mathtt{Trans}^{-1}$. Lag en skisse av hvordan denne maskinen vil se ut, ved å bruke maskinene nevnt her. (Du trenger selvfølgelig ikke tegne maskinene som er gitt over, bare bruk navnene, og sørg for at lesehodet står riktig plassert hver gang en ny maskin kjøres.).
Den samme maskinen kan dekryptere meldingen igjen, ved kun å sende den krypterte teksten tilbake gjennom maskinen. Denne pene symmetrien har vi på grunn av at $n \oplus n = 00...0$ og $n \oplus 00...0 = n$, for alle $n \in \mathbb{N}$. Dermed får vi $(w \oplus n) \oplus n = w \oplus (n \oplus n) = w \oplus 00...0 = w$.

Oppgave 2

Vi skal i denne oppgaven se litt nærmere på hvordan maskiner kan simulere andre maskiner. Vi vet at det finnes universielle turingmaskiner, maskiner som kan simulere alle andre maskiner med vilkårlig input. En datamskin er et eksempel på en tilnærming til en universiell turingmaskin.
Vi ønsker å simulere DFA'er med opp til 2 tilstander over alfabetet $\{0, 1\}$. Vi kan, uten tap av generalitet, anta at maskinens tilstander heter a og b. Vi trenger en måte å lagre transisjoner, aksepterte tilstander, starttilstand, og input til DFA'en på tapen. La transisjonene være lagret på formen

...	a	1	b	...

Dette er da en transisjon fra tilstand a, med input-symbol 1, som går til tilstand b. Vi lar det være en tom rute mellom hver transisjon. La videre starttilstanden være det første symbolet fra venstre, etterfulgt av inputen til DFA'en som skal simuleres. Deretter har vi en # som skiller inputen fra transisjonene. Etter tranisjonene har vi en ny #, etterfulgt av en opplistning av de aksepterte tilstandene. Her er et eksempel på hvordan en tape kan se ut

...	ϵ	a	1	1	0	#	a	1	a	ϵ	a	0	a	#	a	ϵ	...

Dette vil da være en DFA med en tilstand a, hvor begge transisjonene går tilbake til a. a er også askeptert.

Lag en turingmaskin med alfabet $\{0, 1, a, b, \#\}$ som simulerer en vilkårlig DFA opp til 2 tilstander og med vilkårlig input over $\{0, 1\}$, slik som beskrevet over. Dersom maskinen din består av mange nesten like deler, kan du la være å tegne alle de nesten like delene opp. Tegn en eller to og forklar hvordan de resterende ser ut.