TDT4125 Algoritmekonstruksjon

Eksamen, 29. mai 2020, 09:00–13:00

Faglig kontakt	Magnus Lie Hetland
Hjelpemiddelkode	A

Løsningsforslag

Løsningsforslagene i rødt nedenfor er *eksempler* på svar som vil gi full uttelling. Det vil ofte være helt akseptabelt med mange andre, beslektede svar, spesielt der det bes om en forklaring eller lignende. Om du svarte noe litt annet, betyr ikke det nødvendigvis at du svarte feil!

På grunn av koronapandemien er dette en hjemmeeksamen med alle hjelpemidler tillatt og med karakteruttrykk *bestått/ikke bestått*. Derfor er det ikke lagt vekt på å skille mellom prestasjoner i det øvre sjiktet av karakterskalaen, og utvalget av oppgavetyper er noe utenom det vanlige. Følgelig bør ikke eksamenssettet ses som representativt for ordinære eksamener.

1 Du har n rådgivere som gir deg råd om en serie med beslutninger. Før hver beslutning, må du bestemme deg for hvem av dem du vil lytte til. Etter beslutningen finner du ut hvem av rådgiverne som ga deg gode og dårlige råd. Anta at én av rådgiverne alltid gir deg gode råd – men du vet ikke hvilken. Hvordan kan du begrense antall ganger du følger dårlige råd? Hvor mange ganger vil du i verste fall ende med å følge dårlige råd?

Forklar og utdyp. Knytt til relevant teori, gjerne i ulike deler av pensum.

Dette er et standard-eksempel for å motivere multiplikativ vektoppdatering, der man følger en forenklet strategi: Følg alltid flertallet av dem som aldri har gitt dårlige råd. I hver iterasjon vil man da enten ha fulgt et godt råd eller få eliminert halvparten av rådgiverne. Antall ganger man kan følge dårlige råd er følgelig maksimalt logaritmisk. Her er det et poeng å også diskutere multiplikativ vektoppdatering, og ev. onlinealgoritmer generelt.

2 Du har en vektet urettet graf og vil velge ut et sett med kanter som er så tungt som mulig. Det eneste kravet du har er at ingen av kantene du velger skal dele noder. Hvis du velger grådig, hvor god blir løsningen din?

Forklar og utdyp. Knytt til relevant teori, gjerne i ulike deler av pensum.

Dette er grådig matching, som har ytelsesgaranti (*approximation ratio*) $\alpha = 1/2$. Man kan ganske enkelt argumentere for dette, ved å se at én kant i den grådige løsningen maksimalt kan blokkere to stk. fra optimum, og siden vi velger grådig vil den være minst like tung som disse – men ideelt sett bør man også koble dette opp mot snitt av matroider, som er eksplisitt diskutert i forelesninger og pensum. Dvs., det er snittet av en hode- og en hale-partisjons-matroide, der kantene er gitt vilkårlig retning. Eventuelt, om grafen er bipartitt (ikke spesifisert her), så er det snittet av to transversalmatroider. Her kan man gjerne også diskutere matching og approkismering generelt, i den grad det er relevant.

3 Anta at du har en metode for å finne en gyldige løsninger for lineærprogram. Diskuter hvordan du kan bruke en slik metode til å finne optimum. Hvorfor blir fremgangsmåten din korrekt?

Forklar og utdyp. Knytt til relevant teori, gjerne i ulike deler av pensum.

Dette knyttes direkte til en idé som presenteres i pensum og i forelesningene, om å kombinere primal- og dualprogrammet. Dvs., om man f.eks. maksimerer cx, gitt $Ax \leqslant b$, så er dualen å minimere yb, gitt $c \leqslant yA$. Ved optimum har man $yb = cx$. Man kan da finne en gyldig løsning (bestående av x og y) på programmet som krever $Ax \leqslant b$, $c \leqslant yA$, $cx = yb$, og man vil ha funnet optimum. Her kan man diskutere svak og sterk dualitet, nemlig nettopp at man generelt har $cx \leqslant yAx \leqslant yb$ og ved optimum har man $cx = yAx = yb$.

4 Anta at du har et lineærprogram med noen heltallsrestriksjoner av typen $x_j \in \{0,1\}$. Du endrer disse til $0 \leqslant x_j \leqslant 1$. Hva kan du si om optimum for dette nye programmet? Hva kan du si om optimum hvis du runder av hver slik x_j til 1 eller 0, randomisert, med sannsynligheter x_j og $1 - x_j$? Hva kan du si om restriksjoner $(Ax)_i \leqslant b_i$? Kan du få lignende resultater for målfunksjonen *uten* randomisering? Hva med restriksjonene?

Forklar og utdyp. Knytt til relevant teori, gjerne i ulike deler av pensum.

Her starter man med et heltallsprogram og slakker/relakserer til et vanlig lineærprogram. Man ender med å utvide mengden gyldige løsninger, så optimum m å være minst like bra. Ved slik randomisert avrunding blir forventningsverdien til avrundet x_j lik x_j, så forventet målverdi blir lik opprinnelig målverdi, og altså (super-)optimal. Det samme gjelder restriksjonene: De vil være oppfylt i forventning. Man kan derandomisere, og få minst like bra målverdi også deterministisk, med de betingede forventningsverdiers metode. Man kan naturligvis da ikke forvente å tilfredsstille alle restriksjoner; om man gjorde det, ville man ha en generell metode med polynomisk kjøretid for heltallsprogrammering (og følgelig ville P = NP).

5 I preferansevalg skal alle velgere rangere alle kandidater. Flere metoder for å avgjøre slike valg kombinerer rangeringene på en måte slik at man ender med en såkalt *turnering*, altså en komplett, rettet graf $G = (V, E)$: Mellom to noder $u, v \in V$, $u \neq v$, er nøyaktig én av kantene (u, v) og (v, u) i E. For å kunne kåre en vinner, ønsker vi oss at G skal ha følgende egenskap: For alle $u, v, w \in V$, hvis $(u, v) \in E$ og $(v, w) \in E$, så har vi $(u, w) \in E$. En vanlig strategi for å avgjøre slike valg er å snu retningen på så få kanter som mulig for å gi grafen den beskrevne egenskapen. Diskuter hvordan man kan gjøre dette. Hva om du heller velger å eliminere så få kandidater som mulig for å oppnå egenskapen?

Forklar og utdyp. Knytt til relevant teori, gjerne i ulike deler av pensum.

Problemet er å finne en minst mulig sykelkritisk kantmengde (*feedback arc set in tournaments*), som er omhandlet i pensum om parametriserte algoritmer. Metoden som beskrives der er for beslutningsversjonen av problemet. For å løse problemet slik det er beskrevet, kan man f.eks. bruke binærsøk etter riktig k-verdi. Kantene som snus i reduksjon FAST.1 blir med i løsningen, og de siste kantene finner man med rå makt i den gjenværende kjernen (med $k^2 + 2k$ noder).

Om man heller vil eliminere *kandidater*, er det et annet problem (*feedack vertex set in tournaments*), og det forventes ikke at man har en fullgod løsning på dette – men det kan være naturlig å diskutere kobling til det generelle problemed med sykelkritiske nodemengder (i urettede, generelle grafer), som er omhandlet i pensum om approksimering og som nevnes i pensum om begrensede søkeetrær (*bounded search trees*). Selv om det ikke kreves en løsning, vil fornuftige løsningskisser kunne gi poeng, dersom man ikke alt har full score.