

Videregående algoritmik

Job-schedulering

David Pisinger, *Projekt opgave 2*, blok 2, DIKU, 2005-06

Dette er den anden obligatoriske projektopgave på kurset “Videregående algoritmik”. Opgaven stilles mandag 12. december 2005 og skal afleveres senest mandag 9. januar 2006 kl. 12.00 i DIKU’s førstedelsadministration. For at blive godkendt skal der være gjort et reelt forsøg på at løse samtlige spørgsmål. Besvarelsen skal udarbejdes i grupper på to til tre deltagere. Grupper med én deltager kræver accept fra instruktoren. Læs venligst hele opgaveformuleringen igennem inden du går i gang. Hints til opgaverne kan fås ved øvelserne, hvor der er afsat tid til at arbejde med projektopgaven.

Schedulering på identiske parallelle maskiner

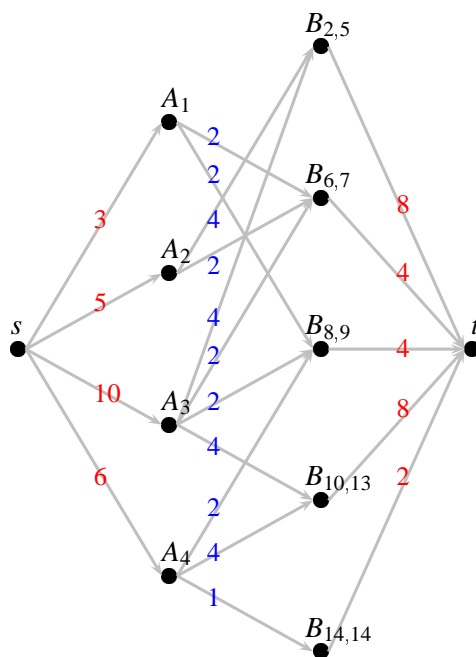
Job-schedulering går ud på at fordele et antal job på et antal givne maskiner, således at alle job bliver udført til tiden. Der er mange eksempler i litteraturen på at en god schedulering af job på maskiner kan føre til signifikant forøgelse af produktiviteten i en virksomhed, hvorfor der forskes meget i gode løsningsmetoder.

Der findes et utal varianter af job-schedulerings problemer afhængigt af om der er flere maskiner til rådighed, om job må afbrydes undervejs eller skal behandles hele, om det tager tid at omstille maskinerne fra et job til et andet, etc. Samtidig kan der arbejdes med et utal af objektfunktioner, f.eks. om man ønsker at maksimere antal behandlede job, eller minimere den samlede udførselstid. Nogle gange kan det også være relevant at betragte hvornår det sidste job bliver færdigt, idet dette måske afgør hvornår sendingen kan afsendes.

I denne opgave vil vi betragte et simpelt scheduleringsproblem SCHED hvor vi har M identiske (*uniforme*) maskiner der arbejder parallelt og en mængde J af job som skal udføres på maskinerne. Hvert job $j \in J$ har en behandlingstid p_j (*process time*) som angiver hvor lang tid der skal bruges på en enkelt maskine for at udføre jobbet. Endvidere har hvert job en starttid (*release time*) r_j som angiver hvornår jobbet må påbegyndes, og en sluttid (*due date*) d_j som angiver hvornår jobbet skal være afsluttet. Med afsluttet menes at vi må benytte maskinen indtil tidsinterval d_j (d_j ikke iberegnet). Vi antager endvidere at en maskine højst kan arbejde på et job ad gangen, og at hvert job højst kan behandles på en maskine ad gangen. Til gengæld må et job godt afbrydes på ethvert tidspunkt og fortsætte på en anden maskine. Dette kaldes også *preemption*.

Job j	1	2	3	4
processing time p_j	3	5	10	6
release time r_j	6	2	2	8
due date d_j	10	8	14	15

Figur 1: En instans af SCHED med $M = 2$ maskiner



Figur 2: Transformation af SCHED til en instans af MAXIMUM-FLOW

Scheduleringsproblemet SCHED er et afgørlighedsproblem, hvor vi alene skal besvare om det er muligt at schedulere de givne job eller ej.

Opgave 1 Find en løsning til scheduleringsproblemet i figur 1. Angiv for hvert job hvilke maskiner det skal udføres på, samt de tilhørende starttider og procestider. ■

Opgave 2 Formuler SCHED som et afgørlighedsproblem. ■

Det er muligt at transformere SCHED til et maksimalt strømningsproblem MAXIMUM-FLOW.

For en instans af SCHED med job $J = \{1, \dots, n\}$ og tilhørende behandlingstider p_j , starttider r_j og sluttider d_j , bemærker vi først at vi alene behøver at betragte tidspunkter svarende til en starttid eller en sluttid, idet tilstanden ikke ændres mellem disse. Derfor lader vi $T = \{t_1, \dots, t_m\}$ være de forskellige relevante tidspunkter sorteret i voksende rækkefølge. For eksemplet i figur 1 har vi de relevante tidspunkter $T = \{2, 6, 8, 10, 14, 15\}$.

Vi konstruerer nu en orienteret graf $G = (V, E)$. For hvert job $i \in J = \{1, \dots, n\}$ opretter vi en knude A_j . For hvert tidsinterval (t_i, t_{i+1}) svarende til to på hinanden følgende tider i T opretter vi en knude $B_{t_i, t_{i+1}-1}$. Endelig har vi knuderne $\{s, t\}$.

For alle A_j knuder oprettes en kant (s, A_j) med kapacitet p_j . For alle knuder $B_{a,b}$ oprettes en kant $(B_{a,b}, t)$ med kapacitet $M(b - a + 1)$. Endelig oprettes en kant mellem hver A_j knude og $B_{a,b}$ knude med kapacitet $(b - a + 1)$ såfremt $r_j \leq a$ og $b < d_j$.

Transformationen er illustreret i figur 2.

Opgave 3 Vis at SCHED har en lovlig løsning hvis og kun hvis den tilhørende instans af MAXIMUM-FLOW har en strømning af værdi $\sum_{j \in J} p_j$. ■

Vi antager nu at det ikke er tilladt at afbryde job undervejs (*no preemption*), dvs. hvis et job er påbegyndt på en maskine skal det afsluttes på samme maskine uden pauser. Vi vil betegne dette problem SCHED-NOP.

Opgave 4 Løs afgørlighedsproblemet SCHED-NOP for instansen beskrevet i figur 1. ■

Opgave 5 Formuler afgørlighedsproblemet SCHED-NOP formelt, og vis at SCHED-NOP er i \mathcal{NP} . ■

Opgave 6 Vis at SCHED-NOP er \mathcal{NP} -fuldstændigt ved reduktion fra TWO-PARTITION. ■

Vi betragter nu optimeringsversionen af SCHED-NOP, hvor man ønsker at maksimere maskinernes udnyttelsesgrad, dvs. at summen af procestiderne p_j for de udførte job maximeres. Dette problem betegnes MAX-SCHED-NOP.

Opgave 7 Beskriv en øvre grænseværdi for MAX-SCHED-NOP. Bevis at grænseværdien er lovlig idet den opfylder kravene for en relaxering. ■

Opgave 8 Angiv hvor hurtigt den foreslåede øvre grænseværdi for MAX-SCHED-NOP kan beregnes. ■

Opgave 9 Find en øvre grænseværdi for MAX-SCHED-NOP for instansen fra figur 1. ■

Opgave 10 Skitser en branch-and-bound algoritme for MAX-SCHED-NOP. ■

Noter

TWO-PARTITION er defineret som følgende afgørlighedsproblem: Givet et antal ikke-negative heltal $S = \{s_1, s_2, \dots, s_n\}$. Afgør om der findes en opdeling af S i to mængder, T og $S \setminus T$, således at $\sum_{i \in T} s_i = \sum_{i \in S \setminus T} s_i$. \mathcal{NP} -fuldstændighed af TWO-PARTITION bliver vist i Cormen opgave 34-2 c).

Den benyttede transformation af SCHED til et MAXIMUM-FLOW problem er foreslået i [1]. Der findes et utal af oversigtsartikler om schedulerings problemer. De to artikler [2] og [3] er tilgængelige på nettet.

Litteratur

- [1] A.K. Ahuja, T.L. Magnanti, J.B. Orlin, Network Flows – Theory, Algorithms, and Applications, Prentice Hall, Englewood Cliffs, New Jersey, 1993.
- [2] David Karger, Cliff Stein, Joel Wein, “Scheduling Algorithms”, available online <http://dimacs.rutgers.edu/~tshaheen/SchedulingSurvey.ps>
- [3] Albert Jones, Luis C. Rabelo “Survey of Job Shop Scheduling Techniques”, available online <http://www.mel.nist.gov/msidlibrary/doc/luis.pdf>