Institut for Matematik og Datalogi
Syddansk Universitet, Odense

27. januar 2013
RF

DM507 – Opgaver uge 5

Eksaminatorier

1. Cormen et al. problem 1.1 (side 14). Erstat dog "microseconds" med "nanoseconds" (dvs. 10^{-9} sekunder), da dette ca. er hvad en CPU-cykel tager på en moderne processor. Der er nok at udfylde søjlerne *second*, *hour*, *month*, *century*. For nogle af indgangene kan man finde svaret ved matematisk udregning, for andre må man prøve sig frem ved at indsætte forskellige værdier af n.

2. Brodal noter om puslespil, opgave 1.

3. Brodal noter om puslespil, opgave 2. "Optimal følge af ombytninger" betyder et antal ombytninger som angivet af sætning 1 i noterne. Opgaven viser at andre algoritmer end "grådige algoritmer" (dvs. algoritmer som altid bringer mindst eet element på plads) kan være optimale for dette puslespilsproblem.

4. Lav et Java-program (eller Python-program) som først genererer en tilfældig permutation af tallene 1 til n (for et n som er en input parameter). F.eks. kan man i Java bruge typen `ArrayList`, samt metoden `shuffle` fra `Collections` utility klassen.

   I en sådan permutation i et array kan man definere cykler på samme måde som for puslespillet fra første forelæsning: et tal x, som står på plads y i arrayet, giver en pil fra plads y til plads x (dvs. hvis tallet 2 står på plads 5, er der en pil fra plads 5 til plads 2), og en samling pile, der hænger sammen i en cyklisk kæde, kaldes en cykel (eller end kreds).

   Lav en algoritme (og en implementation heraf) som tæller antal cykler i permutationen. Hvad er køretiden for dit program som funktion af n? Brug (mange) gentagne kørsler af dit program til at give et bud på sandsynligheden for at der i en tilfældig permutation med $n = 12$ er k cykler, for $k = 1, 2, \ldots, 12$ (jf. figur 3 i noterne, hvor n dog er 64).

Find også det gennemsnitlige antal cykler i dine eksperimenter. Passer dit tal med (er tæt på) formlen på side 4 i noterne?

5. [Udfordrende] Brodal noter om puslespil, opgave 3. Opgaven handler om at lave en algoritme, som først og fremmest bruger et optimalt antal træk (et antal træk som i sætning 1), og derudover også maksimerer hvor mange af disse træk som *ikke* bringer en brik på plads (intuitivt set: find en optimal algoritme som er "længst muligt væk" fra den grådige virkemåde). Start med at besvare opgaven, som den står (hvor du har lov til at vælge et bestemt puslespil for hvert n). Forsøg derefter at løse opgaven mere generelt som beskrevet her (hvor algoritmen, og dit argument for dens egenskaber, skal virke for alle puslespil og alle n). Hint: fokuser på cykler af ulige længde.

Studiegrupper

Forslag til fokus for arbejde i studiegrupper: forbered dele af opgaverne til eksaminatorietimer, f.eks. på nedenstående måde.

- I opgave 1 ovenfor, lav hver især 2/3 af tallene, og sammenlign fælles resultater i gruppe (hvert tal burde blive regnet af mindst to, hvis der er mindst tre personer i gruppen).
- Gennemgå slides til Jul i Valhalla. Diskuter indholdet så I er sikker på at forstå hver påstand/lemma/sætning. Snak om svarene på opgaverne 2, 3 og 5 ovenfor. Men lav den endelige nedskrivning af svaret selvstændigt efter gruppemøde.
- Udvikl algoritmen i opgave 4 ovenfor sammen. Men lav selve programmeringen og programkørsel selvstændigt (eller evt. i par) efter gruppemøde.