Mästarprov 2

André Josefsson 19940429-4176

8 december 2015

1 Flyktingplaceringsproblemet

Maximeringsproblemet omformuleras till ett beslutsproblem på följande vis:

Indata:
Antal platser n på boendet.
En lista $l = \{(a_i, p_i)\}$ om m st talpar, där a_i betecknar antalet flyktingar i grupp i och p_i betecknar värdet på deras prioritering av det aktuella boendet. Prioriteringen kan vara något av heltalen 1,2,3,4 eller 5.
Ett mål M som införts för att göra det aktuella problemet till ett beslutsproblem.

Beslutsproblemet:
Finns det någon lista s av flyktinggruper, med injektionen $f : \mathbf{N} \to \mathbf{N}, s_{f(x)} = l_y, \forall x \in \{1, ..., |s|\}$, sådan att: $\sum_{j=1}^{|s|} a_{f(j)} = n$ och $\sum_{j=1}^{|s|} a_{f(j)} p_{f(j)} \geq M$?

För att visa att detta beslutsproblem är NP-Fullständigt krävs att vi dels visar NP-tillhörighet och därefter att problemet är NP-svårt. Vi börjar med NP-tillhörigheten som visas genom att varje given ja-instants skall kunna verifieras i polynomisk tid. Detta visas genom följande algoritm:

```
VerifyFlyktingNP(l, s, n, M) =
for each pair p_s in s do
    if exists pair p_l in l identical to p_s then
        delete first instance of p_l from l
    else
        return false
    end if
end for
count ← 0
prio ← 0
for each pair (a, p) in s do
    count ← count + a
    prio ← prio + p * a
end for
if count = n and prio ≤ M then
    return true
end if
return false
```

Antag l och s lagras som länkade listor av par. Det tar då $\mathcal{O}(|s| * |l|)$ tid att hitta alla par ur s som finns i l. Det tar sedan $\mathcal{O}(|s|)$ tid att ta bort alla par. Vidare tar det $\mathcal{O}(|s|)$ tid att räkna ut slutvärdet på *count* och *prio*. Totalt sett är algoritmen av tidskomplexitet $\mathcal{O}(|s| * |l|)$ och problemet ligger därför i NP.

Nu ska problemet visas NP-svårt. Detta sker genom en reduktion $A \geq_p B$ där A är flyktingplaceringsproblemet och B är det känt NP-svåra problemet delmängdsumma. Genom en sådan reduktion kan vi visa att flyktingplaceringsproblemet är minst lika svårt som delmängdsumma, alltså minst NP-svårt.

Delmängdsummaproblemet går ut på att man givet en mängd heltal vill veta om det finns någon icke-tom delmängd vars element summeras till 0. Reduktionen sker på följande vis:
Den aktuella mängden delas upp i två partitioner, en med alla positiva element och en med de negativa. Om något av elementen är 0 tas det inte med i någon av de två partitionerna. Bilda ett talpar $(a_i, 1)$ för varje element a_i i den positiva partitionen. Låt listan av dessa talpar heta l_1. Bilda sedan ett talpar $(-b_i, 1)$ för varje element b_i i den negativa partitionen. Låt listan av dessa talpar heta l_2. Summera alla element i den positiva partitionen som k. Gör nu successivt anrop till flyktingproblemet med n satt till var och ett av värdena {1,2,..,k}, $l = l_1$, $s = l_1$ och $M = n$.e som n. För alla värden på n som returnerar ja görs anrop med samma n-värde men $l = l_2$, $s = l_2$ och $M = n$. Om något av dessa anrop returnerar ja ska delsummabeslutsproblemet returnera ja, annars ska det returnera nej.

Om den ursprungliga mängden består av q element kan partitioneringen göras i $\mathcal{O}(q)$ tid. Summeringen görs likaså i $\mathcal{O}(q)$ tid. Talparen bildas även de i $\mathcal{O}(q)$ tid. Totalt sett går alltså detta i polynomialt och tar $\mathcal{O}(q)$ tid. Därefter görs $\mathcal{O}(k)$ anrop till flyktingproblemet. Då detta är en polynomial mängd anrop blir alltså reduktionen totalt sett polynomial med $\mathcal{O}(k)$ anrop och $\mathcal{O}(q)$ förberedelseberäkningar.

Reduktionen måste vara korrekt enligt följande resonemang:
Om flyktingproblemet svarar ja för ett visst värde n' på n för både indatan där $l = l_1$ och den då $l = l_2$ gäller följande:

1. det måste finnas en delmängd positiva tal i den ursprungliga mängden som summerar till n'

2. det måste finnas en delmängd negativa tal i den ursprungliga mängden som summerar till $-n'$

1 och 2 ger då tillsammans att det finns någon delmängd vars element summerar till 0, en sådan (kanske den enda) är just sammanslagningen av de två disjunkta mängderna som gav upphov till 1 och 2.
Om vi istället tänker oss en ja-instans av delmängdsummaproblemet kan vi enkelt förstå att de negativa talen här summerar till något tal $-n''$ och de positiva talen summerar till n''. Detta motsvarar alltså precis 1 och 2.
Slutsatsen blir därför att reduktionen kommer svara ja för precis de instanser då det ursprungliga problemet svarar ja.

Flyktingproblemet är alltså även NP-svårt och därmed NP-fullständigt.