Lista subiecte subiecte Algoritmi Avansaţi

Ştefan Popescu

1 (30p) O clasă de elevi este formată din m copii. La această clasă Moș Crăciun aduce n cadouri, fiecare dintre ele având o valoare notată $val_1, val_2, ..., val_n$. Moșul, grăbit fiind, lasă task-ul împărțirii cadourilor pe umerii profesorului de informatică. Acesta își alege următorul obiectiv pentru împărțirea cadourilor: sa maximizeze valoarea cadourilor primite de către copilul care primește cel mai puțin. (Altfel spus, copilul cel mai "vitregit" în urma împărțirii să primească totuși cadouri de o valoare cât mai bună / Să existe un "spread" cât mai bun al cadourilor). În timp ce se gândește la o soluție de implementare, profesorul își amintește de la cursurile de algoritmică din facultate că astfel de probleme de optim sunt NP-hard! Fiind cuprins de groază (dar și de o nostalgie pentru cursurile de algoritmică din facultate) el vă cere vouă ajutorul.

Exemplu: [1] Pentru 3 copii și 6 cadouri cu valorile $3, 4, 12, 2, 4, 6$ o împărțire echitabilă arată astfel: primul copil primește cadourile $1, 2, 4$. Al doilea copil primește cadoul 3. Iar ultimul copil primește cadourile 5 și 6. În acest caz cel mai "vitregit" copil este cel dintâi, care primește cadouri în valoare totală de 9 unități, în timp ce ceilalți doi primesc cadouri în valoare totală de 12, respective 10 unități. Nu există nicio altă configurație mai reușită.

Notații:

OPT – soluția optimă care există: Valoarea totală a cadourilor primite de copilul cel mai "vitregit" în o configurație optima

ALG – soluția oferită de algoritmul vostru: Valoarea totală a cadourilor primite de copilul cel mai "vitregit" în urma algoritmului vostru

$C[k]$ - lista cadourilor primite de către copilul k la finalul algoritmului vostru

$W(K)$ – valoarea totala a cadourilor primite de către copilul k la finalul algoritmului vostru

$val(i)$ – valoarea cadoului i

$Val = \sum_{1 \leq i \leq n} val(i)$ – valoarea totală a cadourilor

Copiii vor fi indexați folosind variabile de forma k, q, k', q', etc... Cadourile vor fi indexate folosind variabile de forma i, j, i', j', etc....

Restricții: Considerăm că $n \geq m$ și că $val(i) \leq \frac{Val}{2m}$ pentru oricare cadou i.

Cerințe:

a) Descrieți un algoritm $\frac{1}{2}$ aproximativ pentru problema cadourilor in complexitate $\mathcal{O}(n \log m)$ **(10p)**

b) Fie k acel copil pentru care $ALG = W(K)$. Fie i ultimul cadou primit de un copil oarecare q ($q \neq k$). Care este relația între $W(K)$ și $W(Q) - val(i)$? Justificaţi. **(5p)**

c) Pe baza punctului b) arătați că $ALG \geq \frac{Val}{2m}$ **(5p)**

d) Demonstrați că algoritmul descris la punctul a) este $\frac{1}{2}$ aproximativ **(5p)**

e) Dați un exemplu format din minimum 2 copii și 4 cadouri pentru care algoritmul vostru nu găsește soluția optima. Spuneți care este soluția optima. Spuneți care este soluția dată de algoritmul vostru. **(5p)**

.....

[1] În acest exemplu nu se ţine cont de restricţia $val(i) \leq \frac{Val}{2m}$

2 (15p) ***Load Balancing with Restrictions***

Se consideră un proiect format din n task-uri ce trebuiesc efectuate de către m mașini de calcul. Fiecare task poate fi procesat doar de către una din două mașini dintre cele m. Task-urile sunt caracterizate ca fiind un triplete de forma (T_i, x_i, y_i), unde T_i este timpul necesar pentru a procesa task-ul i, iar x_i și y_i sunt indicele celor două mașini ce pot efectua task-ul i (celelalte $m-2$ mașini sunt incompatibile cu efectuarea taskului i). Se dorește planificarea fiecărui task pe câte o mașină compatibilă cu task-ul astfel încât întregul proiect să se termine cât mai repede. (Altfel spus, se dorește minimizarea timpului de lucru a mașinii celei mai solicitate.)

Cerințe

a) Să se scrie problema anterioară sub forma unei ***Probleme de Programare Liniară cu Numere Întregi*** (en. *Integer Linear Programming Problem*). Apoi această problemă să fie *relaxată* și adusă sub forma unei ***Probleme de Programare Liniară***. **(10p)**

b) Folosindu-vă de Problemele de Programare Liniară descrise la punctul a), propuneți un algoritm 2-aproximativ pentru problema inițială. Justificați de ce algoritmul propus are factorul de aproximare 2. **(5p)**

Notații și indicații:

OPT - încărcătura mașinii celei mai solicitate în configurația optimă.

ALG - încărcătura mașinii celei mai solicitate în urma algoritmului propus de voi.

LP, respectiv ILP - expresiile ce trebuiesc maximizate sau minimizte pentru problemele voastre de programare liniară, respectiv programare liniară cu numere întregi.

Task-urile vor fi indexate cu variabile de forma "i, j, k". Mașinile vor fi indexate cu variabile de forma "q, p, r".

$Comp(q)$ - lista task-urilor compatibile cu mașina q

Variabilele de tipul A_q^i vor indica dacă task-ul i este alocat mașinii q sau nu.

3 (15p) ***Probability Knapsack***

Se dau n obiecte, fiecare dintre ele fiind caracterizat de o valoare, respectiv de o probabilitate (valoare subunitară) de a putea fi transportate intact. Se dorește alegerea unor obiecte dintre cele n pentru efectuarea unui transport de o valoare cât mai mare, dar cu o probabilitate de cel puțin P ca întreg conținutul să ajungă intact la destinație.

Exemplu: Pentru $n = 3$, $P = 1/2$ și obiectele cu valoarea, respectiv probabilitatea de a ajunge întregi la destinație $(4, 4/5), (6, 3/5), (3, 4/5)$ transportul va include obiectele 1 și 3 cu o valoare totală de 7 unități și o probabilitate de a ajunge la destinație intacte de 16/25

Cerințe: În elaborarea unui algoritm genetic pentru rezolvarea acestei probleme

a) Descrieți cum ați codifica un cromozom? (care este lungimea cromozomului? Ce ar reprezenta valoarea fiecărei gene?) **(5p)**

b) Descrieți cum ați modela o funcție de fitness pentru această problemă **(10p)**

Notații și indicații:

obiectele vor fi indexate folosind variabilele i, j

pentru un obiect i, $val(i)$ va reprezenta valoarea sa iar $prob(i)$ probabilitatea ca acesta să ajungă intact la destinație.

.....