Ilie Petre Cristian
Grupa 233

Temă - Alg. Aproximativi

Knapsack

a) Fie dp o matrice ce are inițial pe coloana și linia 0 toate elementele egale cu 0.

Pentru un i de la 1 la n

Pentru un j de la 0 la K

Dacă $s_i > j$, atunci $dp[i][j] = dp[i-1][j]$

Altfel, $dp[i][j]$ = maximul dintre $dp[i-1][j]$ și $dp[i-1][j-s_i] + s_i$

$ALG(I) = dp[n][K]$

Algoritmul utilizează programarea dinamică pentru a afla, la fiecare pas, suma maximă $\leq j$, utilizând primele i elemente din șir și salvând-o în $dp[i][j]$

b) Fie s_p - numărul maxim din șir

Suma = 0

Pentru fiecare $s : S$

Dacă $s \le K$, atunci Suma += s, K -= s

$ALG(I) = \max(Suma, s_p)$

Demonstrație

Fie OPT valoarea optimă pentru problemă.

Notăm s_j primul element din șir ce nu va fi luat de ALG (deoarece prin adăugarea sa, suma ar depăși K)

$$OPT(I) \le \sum_{i=1}^{j} s_i$$

$$OPT(I) \le \sum_{i=1}^{j-1} s_i + s_j \le \sum_{i=1}^{j-1} s_i + s_p$$

Dar $ALG(I) = \max\left(\sum_{i=1}^{j-1} s_i, s_p\right) \Rightarrow \begin{cases} \sum_{i=1}^{j-1} s_i \le ALG(I) \\ s_p \le ALG(I) \end{cases}$

$\Rightarrow ALG(I) \le OPT(I) \le 2 \cdot ALG(I)$

$\Rightarrow$ A. ALG este un alg. de 1/2 aproximare.

Load Balance

1. a) Soluția optimă a problemei Load Balancing va echilibra încărcăturile de pe mașini ⇒ La finalul execuției ~~mașina cea mai~~ diferența de timp dintre mașina cea mai încărcată și cea cu cel mai mic load nu poate fi $> \max(t_{j\, 1 \le j \le n})$. Dacă încărcăturile nu au un timp de lucru > 100 ⇒ În cazul algoritmului studentului, diferența ar trebui să fie $\le 100 + \frac{1}{10} \cdot 100 = 110$

$120 - 80 = 40 \le 110$ ⇒ Este posibil ca algoritmul său să fie de 1.1 aproximare.

b) În acest caz, diferența trebuie să fie $\le 10 + \frac{1}{10} \cdot 10 = 11$. $120 - 80 = 40 > 11$ ⇒ Dacă încărcăturile au timpul de lucru ≤ 10 atunci algoritmul studentului nu poate fi de 1.1 aproximare.

3. Fie K - mașina cu loadul cel mai mare la sfârșitul asignării

Fie j - ultimul job asignat mașinii K

Notăm cu load'(M_i) - loadul mașinii M_i după ce s-au asignat primele $j-1$ activități

Teorema 3 dovedește că ALG este $2-\frac{1}{m}$ aproximativ $\Rightarrow$ Pt a dovedi că este $\frac{3}{2}-\frac{1}{2m}$ aproximativ voi afla că:

$$load'(M_K) \le \frac{m+1}{2m} \cdot \sum_{i=1}^{m} load'(M_i) = \frac{m+1}{2m} \cdot \sum_{p=1}^{j} t_p \le$$

$$\le \frac{m+1}{2m} \cdot \sum_{p=1}^{n} - \frac{m+1}{2m} \cdot t_j$$

$$ALG = load'(M_K) + t_j \le \frac{m+1}{2m} \sum_{p=1}^{n} t_p - \frac{m+1}{2m} t_j + t_j \le$$

$$\le OPT - \frac{m+1}{2m} \cdot t_j + t_j \le OPT - \frac{m+1}{2m} t_{max} + t_{max} \le$$

$$\le OPT + OPT - \frac{m+1}{2m} OPT = 2 - \frac{m+1}{2m} OPT = \frac{3m-1}{2m} OPT = \frac{3}{2} - \frac{1}{2m} OPT$$

$\Rightarrow$ A

TSP

1. a) Știm că HC-P este NP-Complete.

Fie un graf G oarecare pe care vrem să rezolvăm HC-P. Din G construiesc G' a. î. V(G) = V(G') iar muchiile din G se găsesc și în G' cu costul 1. Adaug muchii de cost 2 în G' până devine graf complet.

În acest moment, soluția la problema TSP va oferi un răspuns egal cu n (n. de vârfuri) <=> G are ciclu hamiltonian. Altfel, acesta va returna un răspuns mai mare sau egal cu $n-1+2 = n+1$, astfel reducându-se la găsirea unui ciclu hamiltonian. Cum HC-P este NP-complete => Varianta TSP cu ponderea 1 sau 2 este NP-hard.

Vertex Cover

a) Cazul worst case s-ar afla în situația în care, alegând aleator variabile, să fie setate la true exact acelea care nu se mai află în niciun alt predicat, astfel trebuind să fie alese m variabile (unde m = nr. de predicate). Exemplu: $(x_1 \vee x_2 \vee x_3) \wedge (x_2 \vee x_3 \vee x_4) \wedge (x_3 \vee x_4 \vee x_5)$ ⇒ Pe acest exemplu, ar putea fi alese pe rând x_1, x_2, x_3 astfel fiind setate 3 ~~var~~ variabile în loc de 1 (x_3). ⇒ ⇒ Algoritmul este m-aproximativ

b) Pentru a ajunge la un algoritm 3-aproximativ, am putea ca în momentul alegerii unui predicat aleator, să-i setăm toate 3 variabilele la true. Astfel, worst case ar fi situația în care toate predicatele au variabile diferite ⇒ ⇒ soluția optimă ar alege ~~o~~ variabilă din fiecare predicat ⇒ m, în timp ce algoritmul acesta ar face toate variabilele true ⇒ 3m.

c) $X = \{x_1, \dots, x_n\}$, $C = \{C_1, \dots, C_m\}$

$0 \leq x_i \leq 1$; $1 \leq i \leq n$

$x_a, x_b, x_c \in C_i \ (\forall)\ i \in \{1, \dots, m\} \Rightarrow x_a + x_b + x_c \geq 1$

Trebuie să minimizăm

$$\sum_{i=1}^{n} x_i$$

d)

$$ALG = \sum_{i=1}^{n} \begin{cases} 1 & \text{dc. } x_i \geq \frac{1}{3} \\ 0 & \text{dc. } x_i < \frac{1}{3} \end{cases} \leq \sum_{i=1}^{n} 3x_i =$$

$$= 3 \sum_{i=1}^{n} x_i \leq 3 \cdot OPT \Rightarrow \text{soluție 3-aproximativă}$$