Tema 1 Algoritmi Avansati

Moroianu Theodor

23 martie 2021

4 Vertex Cover / SAT

A

Analizam factorul de aproximare al urmatorului algoritm:

```
Greedy_3CNF(C, X):
    C := {C1, ..., Cm}          # mulțimea de predicate
    X := {x1, ..., xn}          # mulțime de variabile

    while C != { }:
        Cj <- C                   # extragem aleator un predicat din C
        Xi <- Cj                  # Extragem o variabila aleator din Cj
        xi := True
        C := C \ xi               # Eliminăm din C toate predicatele
                                  # ce îl conțin pe xi
    return X
```

Fie I_n, cu $n \in \mathbf{N}$ o familie de inputuri definite astfel:
$I_k = (X_1 \vee X_1 \vee X_1) \wedge (X_1 \vee X_2 \vee X_2) \wedge \ldots \wedge (X_1 \vee X_k \vee X_k)$.
Este trivial de aratat ca solutia optima pentru oricare element I_i este $x_i = \delta_{i,1}$, unde prin δ se intelege functia Kronecker.

In cel mai rau scenariu, algoritmul nostru *Greedy-CNF* seteaza variabilele $x_i = 1, \forall i$.

Asadar, observam ca pe multimea de inputuri I algoritmul dat nu respecta nicio constanta de optimizare:

$$lim_{n\to\infty}\frac{ALG(I_n)}{OPT(I_n)} = lim_{n\to\infty}\frac{n}{1} = \infty$$

Se observa usor ca pentru o problema de satisifabilitate formata din N clauze, solutia optima cat si cea generata de greedy sunt mereu in intervalul $[1, N]$: In cel mai bun caz putem satisface toate disjutiile cu o singura variabila, si in cel mai rau caz ne trebuie o variabila per disjunctie.

Cum greedy-ul *Greedy-3CNF* este pe multimea de inputuri $(I_k)_{k\in\mathbf{N}}$ de N ori mai prost decat OPT, si N este cel mai post factor de aproximare putem deduce ca factorul de aproximare al algoritmului *Greedy-3CNF* este N, unde N reprezinta numarul de clauze din input. □

B

Propunem urmatorul algoritm, si afirmam ca reprezinta o 3-aproximare a solutiei optime:

```
Good_Greedy_3CNF(C, X):
    C := {C1, ..., Cm}          # mulțimea de predicate
    X := {x1, ..., xn}          # mulțime de variabile

    while C != { }:
        Cj <- C                   # extragem aleator un predicat din C
        (Xa, Xb, Xc) := Cj        # extragem cele 3 variabile din Cj

        Xa := True                # setam cele 3 variabile ca adevarat
        Xb := True
        Xc := True


        C := C \ Xa \ Xb \ Xc    # Eliminăm din C toate predicatele
                                  # ce îl conțin pe Xa, Xb sau Xc
    return X
```

Lema 1. *Algoritmul Good_Greedy_3CNF reprezinta o 3-aproximare a solutiei optime.*

Demonstraţie. Fie o clauza $(X_a \vee X_b \vee X_c)$. Clauza fiind satisfacuta, cel putin una din variabilele X_a, X_b si X_c sunt adevarate. Algoritmul prezentat considera cele 3 variabile adevarate, si deci in cel mai rau caz are 3 variabile adevarate in loc de una. Dupa aceea, algoritmul elimina toate clauzele satisfactute si repeta pasii.

Putem sa privim asignarea variabilelor cu analiza amortizata, mai precis cu ajutorul metodei potentialelor:

- Gasim o solutie optima Y a asignarii variabilelor X.
- Aplicam algoritmul greedy descris mai sus, si pentru fiecare clauza $(X_a \vee X_b \vee X_c)$ tratata stim ca cel putin una din cele 3 variabile este in Y.
- Asociem fiecarei asignare $X_i := True$ costul 1.
- In plus de costul descris mai sus, asociem fiecarei asignare al unei variabile din Y un bonus de 3 (un cost de -3).

Cum pentru fiecare clauza suma bonusurilor minus cea a costurilor este pozitiva, suma totala este pozitiva.

Asadar, $3 * |Y| \ \geq \ |\{X_i \mid X_i = True\}|$ □

C

Fie $X = \{ x_1, x_2, \ldots, x_n \}$ o multime de variabile booleene, si C o formula in forma 3CNF.

Consideram urmatoarea instanta de programare liniara booleana (programare liniara pe intregi cu valori 0-1):

- Pentru fiecare i avem restrictia $x_i \geq 0$.
- Pentru fiecare clauza $(x_a \vee x_b \vee x_c)$ adaugam restrictia $x_a + x_b + x_c >= 1$.
- Formula pe care dorim sa o minimizam este $x_1 + x_2 + \cdots + x_n$.

Exemplu

Ilustram construirea instantei de programare liniara pentru urmatoarea expresie in 3CNF:

$$(X_1 \vee X_2 \vee X_3) \ \wedge \ (X_2 \vee X_4 \vee X_5) \ \wedge \ (X_1 \vee X_6 \vee X_7) \ \wedge \ (X_5 \vee X_7 \vee X_8)$$

Instanta de programare liniara pe care o obtinem este urmatoarea:

$$\begin{aligned} X_1 + X_2 + x_3 &\geq 1 \\ X_2 + X_4 + X_5 &\geq 1 \\ X_1 + X_6 + X_7 &\geq 1 \\ X_5 + X_7 + X_8 &\geq 1 \\ X_1, \ \ldots, \ X_8 &\geq 0 \end{aligned}$$

Ecuatia liniara pe care dorim sa o minimizam este suma variabilelor X:

$$S_{min} \ = \ X_1 + X_2 + X_3 + X_4 + X_5 + X_6 + X_7 + X_8$$

Se observa usor ca o solutie optima este de-a alege $X_i = \delta_{i,1} + \delta_{i,5}$. Altfel spus, alegem X_1 si X_5 adevarate si restul false.
Solutia este in mod evident optima, prima si a 3-a clauza fiind disjuncte ca multimi de variabile.
Se poate observa ca solutia instantei de programare liniara este si o solutie a problemei 3CNF.

Lema 2. *Solutiile ale instantei de programare liniara care minimizeaza suma S_{min} sunt aceleasi cu solutiile problemei 3CNF care minimizeaza numarul de variabile adevarate.*

Demonstraţie. Fie C o problema de 3CNF, si W problema corespunzatoare de programare liniara, construita folosind procedeul descris mai sus.

Fie Y o solutie a problemei C. Implicit, din constructia instantei de programare liniara, Y satisface toate integalitatile ale acesteia, si deci formeaza o solutie. Analog, orice solutie a problemei W este si o solutie a problemei C.

Pentru a demonstra ca o solutie minima a lui C este si o solutie minima a lui W observam ca atat W cat si C doresc sa minimizeze numarul de variabile cu valoarea 1 sau *Adevarat*. Cum solutiile celor doua probleme formeaza o bijectie (bijectia fiind chiar functia identitate), este clar ca solutiile minime sunt aceleasi. □

Asadar, prin procedeul descris mai sus putem reduce orice problema de tipul 3CNF cu toti termenii pozitivi (fara negatii) intr-o instanta a programarii lineare pe numere intregi. □

D

Rezolvarea instantelor generale de programare liniara pe numere intregi este cunoscuta ca fiind *NP-Complete*. Totusi, rezolvarea instantelor de programare liniara pe **R** este posibila in timp liniar prin algoritmi de tipul *Simplex*.

Asadar, prezentam urmatorul algoritm 3-aproximativ pentru instanta de programare liniara:

```
Good_Greedy_3Lin_Prog(C, X):
    C := {C1, ..., Cm}          # Mulțimea de inegalitati.
                                # Stim ca totate inegalitatile sunt de
                                # forma x_i + x_j + x_k >= 1
    X := {x1, ..., xn}          # Mulțime de variabile.
                                # Stim de asemenea ca ecuatia pe care
                                # dorim sa o minimizam este
                                # X_1 + ... + X_n

    Y <- Simplex(C, X)          # Rezolvam C pe reale.

    Z := {Z1, ..., Zn}          # Variabilele din raspuns.

    for i in {1, ..., n}:
        if Yi >= 1 / 3:         # Daca o variabila din simplex
            Zi = 1              # are o valoare mai mare de
        else:                   # 1/3, atunci o setam ca
            Zi = 0              # adevarata, daca nu falsa.

    return Z
```

Lema 3. *Algoritmul Good_Greedy_3Lin_Prog reprezinta o 3-aproximare a solutiei optime.*

Demonstraţie. Fie o variabila Y_i. Daca valoarea acesteia este mai mica de $\frac{1}{3}$, atunci raspunsul Z_i este 0, si daca valoarea acesteia este mai mare de $\frac{1}{3}$, atunci Z_i este 1. Asadar, se observa imediat ca:

$$Y_i \geq 3 * Z_i$$

$$\sum_{i=1}^{n} Y_i \geq 3 * \sum_{i=1}^{n} Z_i$$

Asadar, algoritmul prezentat este, presupunand ca acesta este corect, o 3-aproximare a solutiei Y.

Pentru a arata ca algoritmul intoarce o solutie valida, observam ca din principiul cutiei in fiecare inegalitate $X_i + X_j + X_k \geq 1$ exista cel putin un element mai mare decat $\frac{1}{3}$, care este transformat in 1.

Pe de alta parte, solutia optima reprezinta o restrangere a problemei calculate de simplex pe numere intregi, si deci este mai mare decat aceasta.

Asadar, avem inegalitatie:

$$SIMPLEX \leq ALG \leq 3 * SIMPLEX$$

$$SIMPLEX \leq OPT$$

$$OPT \leq ALG$$

Primele doua inegalitati provin din explicatiile de mai sus, si a treia inegalitate din minimalitatea solutiei optime.

Asadar, avem inegalitatea urmatoare:

$$ALG \leq 3 * OPT$$

Asadar, algoritmul prezentat este o 3-aproximare a solutiei optime. □