

Corectitudini

Laborator Greedy

Lupașcu Marian (235) | Tehnici Avansate de Programare | 3 – nov - 2017

# Problema 1

## Solutie

**Algoritm greedy**: Initial primul jucator calculeaza sumele de pe pozitii pare si cele de pe pozitii impare pentru a sti cu ce sa inceapa jocul cu inceputul vectorului sau cu capatul. In cazul in care, spre exemplu vede ca pe pozitii impare suma e mai mare, va alege primul element din vector, lasand loc celui de al-2 lea jucator pe pozitiile 2 si n (care e par), jucatorul 2 va alege unul din aceste pozitii iar jucatorul 1 va alege pozitia impara libera. Astfel la sfarsit jucatorul 1 va alege toate pozitiile impare, presupuse cu suma maxima, deci va gastiga in defavoarea jucatorului 2.

## Corectitudine

Presupun fara a restrange generalitatea ca suma de pe pozitii impare este mai mare ca suma de pe pozitiile pare din vectorul initial.

Voi demonstra prin inductie dupa pasul de mutare. Initial presupun ca algoritmul alege numai elemente de pe pozitii impare pana la pasul k < n/2 si demonstrez ca la pasul k + 1 se va alege tot un element de pe pozitie impara.

...

k

...

...

1

1

1

2

2

2

Cum pana la pasul k, adica la k-1 algoritmul a ales numai elemente de pe pozitie impara inseamna ca au fost alese 2k-2 elemente din vector, cu alegerea lui k avem 2k-1 elemente alese. Presupunem ca alegerea k este facuta in partea Stanga a vectorului atunci in partea stanga avem un numar impar de elemente deci in dreapta avem un numar par de elemente alese deci ultul element neales din partea dreapta se alfa pe pozitie para. In concluzie jucatorul doi este obligat sa aleaga un element de dupa elementul ales la pasul k, aflat pe pozitie para, sau ultimul element neales din partea dreapta care se alta pe pozitie para Deci un element de pe pozitie para, urmand ca la pasul k+1 jucatorul 1 sa aleaga pozitia impara descoperita de alegerea jucatorului 2. In mod analog tratam si cazul in care la pasul k se face o alegere in partea dreapta.

Din PIM rezulta ca oricare ar fin k <= n/2 jucatorul 1 va alege tot un element de pe pozitie impara, iar cum suma elementelor de pe pozitii impare era mai mare sau egala decat suma elementelor de pe pozitii pare, avem ca jucatorul 1 castiga in defavoarea jucatorului 2.

# Problema 2

## Solutie

**Algoritm greedy**: Printr-o parcurgerea DFS se introduc in lista finala de noduri toate frunzele apoi se marcheaza parintii acestor noduri pentru a nu mai fi selectate, lucru echivalent cu stergerea frunzelor si a tatilor acestora din arbore, rezultand un nou arbore pe care repetam procedeul prin recursivitatea data de DFS.

## Corectitudine

1

2

3

4

5

6

7

8

DFS

La revenire

Demonstrez ca ori ce frunza trebuie sa faca parte din lista finala pe care algoritmul trebuie sa o returneze, notata cu F.

Fie f o frunza din Arb(arborele initial), Daca f ⋲ F suntem in regula daca f nu apartine lui F atunci consideram singura si unica muchie (f,h), fiindca f este frunza. Daca h nu apartine lui F atunci F + {frunza f} este o lista de iesire in regula. Daca h ⋲ F atunci consideram F = F – {nodul h} + {frunza f} din nou o lista in regula cu acceiasi cardinalitate cu fostul F. Deci orice frunza trebuie inclusa in F. Facand aces lucru recursiv pentru noul arbore rezultat prin eliminarea frunzelor si s parintilor acestora obtinem F-ul final.

# Problema 3

## Solutie

**Algoritm greedy**: Se sortează intervalele crescător după extremitatea iniţială. Pentru fiecare interval I în această ordine execută: se adaugă I la o submulţime deja construită, dacă se poate (nu se intersectează cu niciun interval din ea), altfel se creează o nouă submulţime cu intervalul I.

### Contraexemplu b). [10, 14], [12, 16], [17, 18], [14, 100]

### Contraexemplu c). ???

### Contraexemplu d). [10, 14], [12, 16], [17, 18], [14, 100]

## Corectitudine pentru a).

Fie S mulţimea de intervale dată.

**Definiţie:** Numim *adâncimea lui S* numărul maxim de intervale din *S* care se suprapun peste un acelaşi punct (care au toate un acelaşi punct)

**Exemplu**

Adâncimea mulţimii intervalelor de mai sus este 4 (există cel mult 4 intervale care se suprapun peste un acelaşi punct).

Pentru simplificarea prezentării vom asocia fiecărei submulţimi o etichetă (un număr). Astfel, problema se reduce la a eticheta intervalele cu un număr minim de etichete astfel încât două intervale cu aceeaşi etichetă să fie disjuncte.

Vom demonstra corectitudinea în 3 etape:

1) Soluţia construită de algoritm este corectă (posibilă), deoarece toate intervalele sunt etichetate (adăugate într-o submulţime) şi prin modul de alegere al etichetelor, intervalele cu aceeași etichetă sunt disjuncte două câte două (un interval este adăugat la o submulțime doar dacă nu se intersectează cu alt interval din submulţime)

2) **Propoziție**: Numărul minim de etichete necesare etichetării intervalelor din S este mai mare sau egal cu adâncimea lui S.

**Demonstrație:**

Notăm cu d adâncimea lui S. Din definiție rezultă că există o mulțime de intervale I1, I2,...,Id ce se suprapun peste un același punct. Deoarece nu putem folosi aceeaşi etichetă pentru două sau mai multe intervale care se suprapun, rezultă că cele d intervale anterior menționate trebuie să aibă etichete distincte. Astfel, pentru etichetarea corectă a tuturor intervalelor din S trebuie să folosim minim d etichete.

3) Vom demonstra că algoritmul Greedy foloseşte exact d etichete, de unde va rezulta folosind propoziția că numărul de etichete folosite de algoritm (deci de submulţimi generate) este minim.

Fie t numărul de etichete folosite de algoritm la pasul curent. Trebuie să demonstrăm că la fiecare pas t  d (în final va avea loc chiar egalitatea). Iniţial t=0.

Fie Ij intervalul care este etichetat la pasul curent.

Dacă lui Ij i se poate asocia o etichetă deja existentă (se poate adăuga la o submulţime existentă) atunci t nu se modifică şi inegalitatea rămâne valabilă.

Dacă lui Ij nu i se poate asocia o etichetă deja existentă (nu se poate adăuga la o submulţime existentă), atunci există t intervale cu etichete distincte două câte două (câte unul din fiecare submulţime deja construită) cu extremitatea iniţială mai mică sau egală decât cea a lui Ij ce se intersectează cu intervalul curent.

...

1

2

...

t

Ij

Din faptul că toate cele t intervale au extremitatea iniţială mai mică sau egală decât cea a lui Ij, rezultă că există cel puţin un punct în care toate cele t+1 intervale (incluzând intervalul Ij) se suprapun, şi anume acest punct este chiar extremitatea iniţială a lui Ij. Rezultă că t+1 ≤ d. Deci, chiar dacă pentru Ij este folosită o a t+1 etichetă (este creată o nouă submulţime), în total algoritmul nu va folosi mai mult de d etichete.

Din 1), 2) şi 3) rezultă că algoritmul găsește o etichetare optimă.