21/03/2024

PROGRAMMAZIONE DINAMICA

TECNICA MOLTO POTENTE

SOMMARIO LEZIONE:

- TECNICA P.D. ALL'OPERA → APP. SU UN PROBLEMA.
- PROBLEMA: SET INDIPENDENTE DI PESO MAX
- APP. ALGO.
- PRINCIPI DELLA PROG. DINAMICA.

DEF PROBLEMA

INPUT

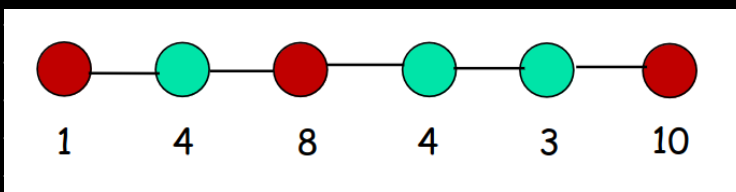
GRAFO G CON n NODI CON VALORE w_i.

PROB. FEASIBILE

INDIP. SET = SUBSET CON NODI ∄ ADIACENTI TRA LORO (∄ ARCO CHE LI COLLEGA)

OUTPUT: ∅ INDIP. SET DI PESO MAXIMO
(DA MAXIMIZZARE)

ESEMPIO



→ $\Pi = \{1, 8, 10\}$

$w_{OPT} = 19$

COME RISOLVERE?

BRUTE FORCE

· CALCOLIAMO X TUTTE OGNI POSS. SOTTOINSIEME

COSTO = $O(2^n)$ NOP.

ALGO GREEDY

· P IN MODO INCREMENTALE, 1 NODO + HEAVY X, EVITANDO NODI ADIACENTI

CORRETT: - SU ALCUNE ISTANZE DA SOL. OTT.

- MA NON SEMPRE!

DIVIDE ET IMPERA

POSSIBILE, MA DIFFICILE RICOSTRUIRE I SOTTOPROBLEMI

PUÒ SUCCEDERE

CHE 2 SOTTOPROBLEMI ABBIANO 2 NODI VICINI

nessuno di questi funziona davvero, cosa stiamo sbagliando?

NON CAPIAMO LA STRUTTURA DEL PROBLEMA

CAPENDOLO POSSIAMO TROVARE A NEW APPROACH

NUOVO APPROCCIO

- ? DI RAGIONARE SULLA STRUTTURA DEL PROBLEMA, COME IN DIVIDE ET IMPERA.
- COL P.D, COME ↑ :
  - SI / IN SOTTOPROBLEMI + SEMPLICI,
  - RAGIONARE SULLA LORO UNIONE

prendiamo la soluzione del problema

SIA S^* SOL. OTTIMA, QUINDI II CON PESO MASSIMO.
PRENDIAMO V_n ⟶ LAST NODO:

OSS: O $V_n \in S^*$ O $V_n \notin S^*$

CASO 1 $V_n \notin S^*$

CONSID. $G' = G - \{V_n\}$
> ALLORA S^* = SOL. OTTIMA × G'
> E SE ∃ SOL + MIGL. × S^*, ALLORA LO SAREBBE × G
ASSURDO

CASO 2 : $V_n \in S^*$ → SICURO $V_{n-1} \notin S^*$

CONS. $G'' = G - \{V_{n-1}, V_n\}$
> ALLORA $S^* \setminus \{V_n\}$ È SOL. OTTIMA × G''
> SEMPRE × PROVA → SE CONSIDERO SOL. S MIGLIORE SI ARRIVA AD UNA CONTRADDIZIONE.

QUINDI IL PROBLEMA HA 2 PROP.:

II DI PESO MAX IN G :
- O = II PESO MAX PER G'
- O II PESO MAX PER G'' + V_n

a prima botta uno potrebbe pensare, 'calcolo ricorsivamente tutti e due e prendo il massimo'.

MA COSTEREBBE

$T(n) = T(n-1) + T(n-2) + O(1)$ FIBONACCI 2

$\Downarrow$

$\boxed{T(n) = O(\phi^n)}$ → × IL METODO NON VA BENE

DUE + PROBLEMI UGUALI VENGONO RISOLTI

POSSIAMO MIGLIORARLO!

POSSIAMO CONSIDERARE MENO PROBLEMI DA RISOLVERE

INFATTI ABBIAMO n PROBLEMI DA RISOLVERE SE DISTINTI

IDEA = RISOLVERE ITERATIVAMENTE DA SOTT. PROBLEMI + PICCOLI A GRANDI → SU UN ARRAY

CONSIDERIAMO QUINDI

- G_J = SUBSET CON PRIMI J VERTICI
- $OPT[J]$ = PESO MAX DI IS PER G_J

QUINDI:

- $OPT[1] = w_1$ → $OPT[2] = \max\{w_1, w_2\}$

$$OPT[J] = \max\{OPT[J-1],\ OPT[J-2] + w_J\}$$

ALGO

```
OPT[1]=w1; OPT[2]= max {w1, w2}
for j=3 to n do
    OPT[j]= max {OPT[j-1], wj+OPT[j-2]}
return OPT[n]
```

→ CALCOLA IL VALORE OTTIMO MA NON LA SOLUZIONE

↓

CIOÈ L'INSIEME

RICERCA SOLUZIONE

PSEUDOCODICE

1. $S^* = \varnothing$; $j = n$;
2. **while** $j \geq 3$ **do**
3. **if** $OPT[j-1] \geq w_j + OPT[j-2]$
   then $j = j-1$;
   else $S^* = S^* \cup \{ v_j \}$; $j = j-2$;
4. **if** $j = 2$ e $w_2 > w_1$ **then** $S^* = S^* \cup \{ v_2 \}$
   else $S^* = S^* \cup \{ v_1 \}$;
5. **return** S^*

COSTO => $\Theta(n)$

SFRUTTO LA PROPRIETÀ CHE

$\forall\ v_j$ t.c. $\in \mathcal{I}$ DI PESO MAX DI G

$$\Downarrow$$

$$w_j + OPT[j-2] \geq OPT[j-1]$$