Tutoraggio Ricerca Operativa 2020/2021 1. Programmazione Dinamica (I)

Aurora Rossi, Alice Raffaele, Romeo Rizzi

Università degli Studi di Verona

27 febbraio 2021

Programmazione Dinamica

- Introdotta da Bellman nel 1957, consente di risolvere un problema scomponendolo in sottoproblemi più piccoli e facili da risolvere;
- Le soluzioni di questi sottoproblemi sono riutilizzate per la costruzione (e risoluzione) dei problemi più grandi.
- Confronto con Divide et Impera:
 - Analogia: entrambi suddividono il problema in sottoproblemi più piccoli e combinandone le soluzioni;
 - Differenza: nel DI i sottoproblemi sono tutti indipendenti → Se ci sono sottoproblemi comuni, sono risolti più volte; nella PD, ogni sottoproblema è invece risolto una sola volta, salvandone la soluzione in apposite strutture dati, in modo da poterla riusare → PD è indicata anche per problemi non indipendenti tra loro.
- Come procedere? Individuando in primis il sottoproblema... Vediamo qualche esercizio!

TE 26/02/2019 - Es. 2: Sottosequenza comune più lunga (I)

Trovare la più lunga sottosequenza comune tra le stringhe s = ATGTCAGAAGAGTCGTA e t = GTACTGACTGAAGGTAT. Fare lo stesso con alcuni suffissi di s e t.

- 2.1(1pt) quale è la più lunga sottosequenza comune tra s e t?
- 2.2 (1pt) e nel caso sia richiesto che la sottosequenza comune incominci con 'C'?
- **2.3 (1pt)** quale è la più lunga sottosequenza comune tra s e il suffisso $t_9 = TGAAGGTAT$ di t?
- **2.4 (1pt)** quale è la più lunga sottosequenza comune tra t e il prefisso $s^{14} = ATGTCAGAAGAGTC$ di s?

tipo di sottosequenza comune	lunghezza	sottosequenza
qualsiasi		
parte con 'C'		
$\operatorname{tra} s \in t_9$		
$\operatorname{tra} s^{14} e t$		

TE 26/02/2019 - Es. 2: Sottosequenza comune più lunga (II)

- **Sottoproblema**: consideriamo una sottostringa di *s* e una sottostringa di *t*... Il problema è lo stesso!
- Caso banale: quando s o t ha lunghezza zero
- Caso generale: ?

Cominciamo da un esempio con due stringhe un po' più corte:

$$s = TAGTCACG$$
 e $t = AGACTGTC$

TE 26/02/2019 - Es. 2: Sottosequenza comune più lunga (III)

Sfruttiamo una tabella a doppia entrata dove mettiamo s in verticale a sinistra e t in orizzontale in alto.

$$s = TAGTCACG$$
, $t = AGACTGTC$

	-	Α	G	Α	С	Т	G	Т	С
-									
Т									
A G									
G									
Т									
С									
C A C									
С									
G									

TE 26/02/2019 - Es. 2: Sottosequenza comune più lunga (IV)

Caso banale: inizializziamo la tabella mettendo un trattino dove la lunghezza della sottosequenza comune (LCS, i.e., *Longest Common Subsequence*) sarebbe zero.

	-	Α	G	Α	С	Т	G	Т	С
-	-	-	-	-	-	-	-	-	-
Т	-								
Α	-								
G	-								
Т	-								
С	-								
C A	-								
C G	-								
G	-								

Indichiamo rispettivamente con s_n il **suffisso** della stringa s dal carattere n (incluso) in poi e con s^n il **prefisso** di s, cioè la sottostringa di s fino al carattere n (incluso).

TE 26/02/2019 - Es. 2: Sottosequenza comune più lunga (V)

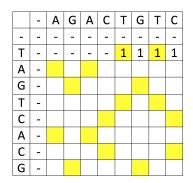
Match: consideriamo le celle dove il carattere della stringa s coincide con il carattere della stringa t ed evidenziamole in giallo:

	-	Α	G	Α	С	Т	G	Т	С
-	-	-	-	-	-	-	-	-	-
Т	-								
A G	-								
G	-								
Т	-								
С	-								
Α	-								
C A C G	-								
G	-								

In queste celle dove abbiamo i match possiamo considerare di incrementare la soluzione che stiamo costruendo.

TE 26/02/2019 - Es. 2: Sottosequenza comune più lunga (VI)

Consideriamo la stringa s un carattere alla volta e interamente la stringa t, procedendo riga per riga (sarebbe analogo proseguire viceversa per colonne, mantenendo tutta s e invece valutando t un carattere alla volta).



- Prima del primo match, la lunghezza di LCS sarà sempre 0
- Al match, incrementiamo di 1
- Fino alla fine della riga inseriamo altri 1 (stiamo considerando s^1)

TE 26/02/2019 - Es. 2: <u>Sottosequenza comune più lunga (VII)</u>

Passiamo alla seconda riga:

	-	Α	G	Α	С	Т	G	Т	С
-	-	-	-	-	-	-	-	-	-
Т	-	-	-	-	-	1	1	1	1
Α	-	1	1	1	1	1	1	1	1
G	-								
Т	-								
С	-								
Α	-								
C A C	-								
G	-								

Abbiamo un match tra s[2] e t[1], perciò LCS tra s^2 e t^1 varrà 1 e la stessa cosa vale per s^2 e t^3 : l'unico carattere in comune è solo la A, che sia t[1] o $t[3] \rightarrow$ Ci sono più soluzioni

TE 26/02/2019 - Es. 2: Sottosequenza comune più lunga (VIII)

Compiliamo ora la terza riga:

	-	Α	G	Α	С	Т	G	Т	С
-	-	-	-	-	-	-	-	-	-
Т	-	-	-	-	-	1	1	1	1
Α	-	1	1	1	1	1	1	1	1
G	-	1	2	2	2	2	2	2	2
Т	-								
С	-								
Α	-								
С	-								
G	-								

- s[3] e t[2] fanno match, quindi LCS ora sarà lunga 2
- Anche s[3] e t[6] fanno match, ma LCS sarà ancora lunga 2 (non 3)

Ogni cella rappresenta un sottoproblema che sfrutta il risultato di uno o più altri sottoproblemi già risolti: a quale/i ci riferiamo?

TE 26/02/2019 - Es. 2: Sottosequenza comune più lunga (IX)

Consideriamo la cella (i,j) e chiamiamo L[i,j] la lunghezza di LCS quando stiamo considerando s^i e t^j . Distinguiamo due casi:

- Cella match: L[i,j] = L[i-1,j-1] + 1
- Cella senza match: $L[i,j] = \max\{L[i-1,j], L[i,j-1]\}$

TE 26/02/2019 - Es. 2: Sottosequenza comune più lunga (X)

Possiamo ora completare il resto della tabella:

	-	Α	G	Α	С	Т	G	Т	С
-	-	-	-	-	-	-	-	-	-
Т	-	-	-	-	-	1	1	1	1
Α	-	1	1	1	1	1	1	1	1
G	-	1	2	2	2	2	2	2	2
Т	-	1	2	2	2	3	3	3	3
С	-	1	2	2	3	3	3	3	4
Α	-	1	2	3	3	3	3	3	4
С	-	1	2	3	4	4	4	4	4
G	-	1	2	3	4	4	5	5	5

- L'ultima cella in basso a destra conterrà il valore della lunghezza di LCS tra s e t
- Nota Fate attenzione ai conti: potrebbe essere sufficiente sbagliare il valore di una singola cella per compromettere gli altri a seguire e il risultato finale!

TE 26/02/2019 - Es. 2: Sottosequenza comune più lunga (XI)

Ora conosciamo quanto è lunga LCS, ma che valore ha? Dobbiamo ricostruire la soluzione. Procediamo a ritroso dall'ultima cella dove è massima (ricordiamoci che cresceva quando eravamo sulle celle match)

	-	Α	G	Α	С	Т	G	Т	С
-	-	-	-	-	-	-	-	-	-
Т	-	-	-	-	-	1	1	1	1
Α	-	1	1	1	1	1	1	1	1
G	-	1	2	2	2	2	2	2	2
Т	-	1	2	2	2	3	3	3	3
С	-	1	2	2	3	3	3	3	4
Α	-	1	2	3	3	3	3	3	4
С	-	1	2	3	4	4	4	4	4
G	-	1	2	3	4	4	5	5	5

- Sulle celle match risaliamo lungo la diagonale a sinistra
- Sulle celle non match ragioniamo da dove siamo giunti (da sinistra o dall'alto)
- Risaliamo così fino alla prima cella
- Il valore di LCS è dato dai caratteri delle celle match che abbiamo attraversato risalendo (in questo esempio, LCS = AGACG)

TE 26/02/2019 - Es. 2: Sottosequenza comune più lunga (XII)

Vediamo ora altre possibili richieste:

	-	Α	G	Α	С	Т	G	Т	С
-	-	-	-	-	-	-	-	-	-
Т	-	-	-	-	-	1	1	1	1
Α	-	1	1	1	1	1	1	1	1
G	-	1	2	2	2	2	2	2	2
Т	-	1	2	2	2	3	3	3	3
С	-	1	2	2	3	3	3	3	4
Α	-	1	2	3	3	3	3	3	4
С	-	1	2	3	4	4	4	4	4
G	-	1	2	3	4	4	5	5	5

- Qual è la più lunga sottosequenza comune che inizia con C?
 Consideriamo il primo match con la lettera C, cioè s₅ e t₄ e compiliamo un'altra tabella
- Qual è la più lunga LCS tra t e il suffisso s₅? Compiliamo ancora un'altra tabella
- Qual è la più lunga LCS tra s e il prefisso t⁵? Questo possiamo leggerlo direttamente dalla tabella di s e t

Ora potete provare da soli a risolvere l'esercizio del tema d'esame e anche a implementare in Python gli algoritmi per calcolare e stampare LCS

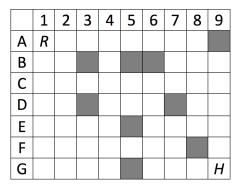
Un robot R, inizialmente situato nella cella A-1, deve portarsi nella sua home H situata nella cella G-9.

	1	2	3	4	5	6	7	8	9
Α	Ĥ								愛
В			愛		愛	妙			
С									
D			愛				愛		
Е					愛				
F								愛	
G					愛				^

I movimenti base possibili sono il passo verso destra e il passo verso il basso. Tuttavia il robot non può visitare le celle occupate dal pericolo chimico. Quanti sono i percorsi possibili?

- Quanti sono i percorsi possibili se la partenza è in A-1?
- e se la partenza è in C-3?
- e se con partenza in A-1 il robot deve giungere in F-6?
- e se con partenza A-1 ed arrivo in G-9 al robot viene richiesto di passare per la cella D-5?

- Sottoproblema: consideriamo una griglia di dimensioni più piccole
- Mosse consentite: verso destra e verso il basso (non in diagonale)
- Mosse proibite: non si può passare sopra i pericoli



Stavolta risolviamo direttamente l'esercizio del TE (la griglia è già data dal testo)

Procediamo al contrario partendo dalla fine, proprio dalla cella H:

	1	2	3	4	5	6	7	8	9
Α	R								
В									1
С									1
D									1
Ε									1
F									1
G	-	-	-	-		1	1	1	Н

- La cella H può essere raggiunta dalla cella a sinistra oppure da quella sopra, dove inseriamo quindi due 1
- Ogni cella contiene quindi il numero di percorsi possibili da lei fino a H

Anche qui ogni cella rappresenta un sottoproblema che sfrutta il risultato di uno o più altri sottoproblemi già risolti: a quale/i ci riferiamo?

Consideriamo la cella (i,j) e chiamiamo P[i,j] il numero di percorsi da essa fino a H. Distinguiamo

• Cella generica: P[i,j] = P[i+1,j] + P[i,j+1]

• Cella mina: P[i,j] = 0

Riempiamo quindi la penultima riga (da destra verso sinistra) e la penultima colonna (dal basso verso l'alto):

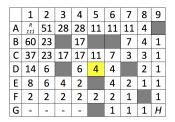
	1	2	3	4	5	6	7	8	9
Α	R								
В									1
С									1
D									1
Ε									1
F	2	2	2	2	2	2	1		1
G	-	-	-	1		1	1	1	Н

Possiamo ora completare tutta la tabella, procedendo sempre dal basso verso l'alto per le righe e da destra verso sinistra per le colonne:

	1	2	3	4	5	6	7	8	9
Α	R 111	51	28	28	11	11	11	4	
В	60	23		17			7	4	1
С	37	23	17	17	11	7	3	3	1
D	14	6		6	4	4		2	1
Ε	8	6	4	2		4	2	1	1
F	2	2	2	2	2	2	1		1
G	-	-	-	-		1	1	1	Н

- Quanti sono i percorsi possibili se la partenza è in A1? Lo leggiamo direttamente in A1: 111
- E se la partenza è in C3? Lo leggiamo in C3: 17
- E se con partenza in A1 il robot deve giungere in F6?
 Bisogna ricompilare un'altra griglia, più piccola, dove la nuova cella H è F6

Ultima richiesta del TE:



	1	2	3	4	5
Α	8	4	2	2	-
В	4	2		2	
С	2	2	2	2	1
D	-	-		1	Н

- E se con partenza A1 e arrivo in H al robot viene richiesto di passare per la cella D5?
 Per considerare il passaggio in D5, spezzo in due il problema:
 - il numero di percorsi da D5 a H lo leggo direttamente dalla griglia: 4
 - invece il numero di percorsi per arrivare in D5 lo calcolo compilando un'altra tabella: 8

Il numero complessivo è dato da $4 \cdot 8 = 32$.