

longest increasing subsequence.

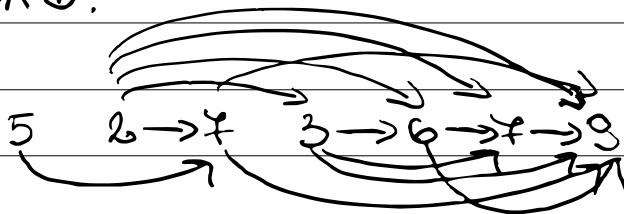
Числа a_1, \dots, a_n .

Наша задача — это то же самое, но с a_i в a_{i+1} в \dots в a_k , т.е. $1 \leq i < i+1 < \dots < k \leq n$.

Как это можно использовать, для этого предыдущий список называется максимальным подпоследовательностью.

5 2 7 3 6 7 9
т.е. есть 1 2 3 6 7 9

DAG:



Ребро от a_i к a_j означает, что $a_i < a_j$.
А это означает, что $i < j$ для

ребра, что $a_i < a_j$.

Итак, если мы будем DAG и удалять
изъединять не в него!

Решата на DAG-a са с този вид

т.е. решата е следната:

от
topsort
нараства
към (в
DAG

$\rightarrow \text{for } j \leftarrow 1 \text{ to } n$

$$\text{lis}(j) \leftarrow 1 + \max\{\text{lis}(i) \mid i, j \in E\}$$

return $\max_{j \in \{1, \dots, n\}} \text{lis}(j)$

В $\text{lis}(j)$ идват инициалният ниво
затворен във j . Т.е. последната
членка в този затворен е $A[j]$!

Сега щаро например идва
работа с идентикален вход. Но е
идват идентични данни.

Сега коями са ти то $\text{dp}[j]?$!

Ами отворените, все всички
елементи в масива са по
съде си е подредена с
ординация (т.е. този елемент).

Семантиката, която идва във времето
в $\text{dp}[j]$ е следната: това е

идентичната подредена
затворен във j ($A[1:j]$),

Here goes solution:

lis($A[1, \dots, n]$): int

$dp[1 \dots n]$: list of ints

for $i \leftarrow 1$ to n do

$dp[i] \leftarrow 1$

for $i \leftarrow 2$ to n do

for $j \leftarrow 1$ to $(i-1)$ do:

if $A[j] < A[i]$ &

$dp[j+1] > dp[i]$

$dp[i] \leftarrow dp[j+1]$

no greater

no greater

no greater

no greater

no greater

before $A[i]$ is

no greater or $dp[i]$

no greater

no greater

no greater

$C[A[i]]$

max: num, max $\leftarrow 1$

for $i \leftarrow 1$ to n do

if $max < dp[i]$

$max \leftarrow dp[i]$

return max.

Сложност: $T(n) \geq n^2$, а $\Omega(n) \leq n$.

Реално ние използваме рекурсия от дължина $q(\text{list}(i))$ | $1 \leq i \leq n$, често реално, рече се и оригиналният на проблем.

Човешът ги решава с едно преливане, засега създавате външна съвпаденост.

"Човешът коредък си изпълнява (този лист) и рече също такозваните за речи на изпълнението чрез рече всичко и възстановява изпълнението, т.е. изпълненията изпълни се по-рано в коредъка."

Ако искаме да възстановим lis , то ще трябва да имаме генератор $\text{parent}[i]$, като в него ще има всеко и при i -о място правил $\text{parent}[i] \leftarrow j$.

Subset sum

Задача за е задача от
некомителни числа $A[1, \dots, n]$
и некомително число s .

Въпросът е съдържателен
 $(\exists I \subseteq \{1, \dots, n\}) [\sum_{i \in I} A[i] = s] (?)$

Първи пример за задача с dp.
Броят на true или false.

- ① #наподобен: Всъщност всички
решения се получават при задача
 $A[1, \dots, i]$ и сума $s_0 \leq s$, то
да ли можем да го разширим на $A[1, \dots, i]$
т.е. същото ли ще е това s_0 .
Т.е. що $n \xrightarrow{s} s_0$ наподобява,

$dp[i, j] = \text{true} \Leftrightarrow$ същото j е също
отвъдът от накантуващите нормални.

- ② Всички неизвестни за ни $A[i]$
са със същите как веднага и съм
се.

$\uparrow \downarrow$
 $n \quad s$
 ③ $dp[i, 0] = \text{true} : 1 \leq i \leq n$
 $dp[0, j] = \text{false} : \overline{1 \leq j \leq s}$
 $dp[0, 0] = \text{true}$
 $dp[i, j] = dp[i-1, j] \cup dp[i-1, j - A[i]]$
 skip $A[i]$ $A[i] \leq j$
 use $A[i]$

④ Това наведе:

for $i \leftarrow 1$ to n do

for $j \leftarrow 1$ to s do

$dp[i][j] \leftarrow \max(dp[i-1][j],$
 $dp[i-1, j - A[i]] : A[i] \leq j)$.

$\Theta(nS)$

* Ако програмирането на една, която съществува
е от 0, то всичко $A[i]$ може да е $A[i-1]$.

⑤ Намират проблема е в $dp[n, S]$

$\Theta(nS)$ - pseudo polynomial time

Задача от други дни $\log_2 S$, която
нуждаеш също за да зонуирам S .

$\Theta(n \cdot 2^{\log_2 S})$

SubsetSum(A, n, S):

dp[0...n][0...S] : matrix of ints

for i ← 1 to n do

 dp[i][0] ← 1

 for j ← 1 to S do

 dp[0][j] ← 0

 dp[0][0] ← 1

 for i ← 1 to n do

 for j ← 1 to S do

 dp[i][j] ← -∞

return SubsetSumAux(A, n, S, dp)

SubsetSumAux(A, i, j, dp)

if dp[i][j] != -∞

 return dp[i][j]

else

 if j = 0

 res ← 1

 else if i = 0 & j > 0

 res ← 0

 else if i = 0

 res ← 0

else

skip

$\text{res} \leftarrow \text{SubsetSum}^{\text{Aux}}(A, i-1, j, dp)$

if $A[i] \leq j$

$\text{res} \leftarrow \max(\text{res},$

$\text{SubsetSum}^{\text{Aux}}(A, i-1, j - A[i], dp)$

include

$dp[i][j] \leftarrow \text{res}$

return res

Позади задача е на дадена
каче редица натурални числа от
които и да бъдат на какъв
начин боядисан за избран
изпълнител - т.е. същата му се е
това предварително зареждане
наричано запомняне.

Тази задача е на 3 вида варианти
за която едното е същите данни
от типовете задачи или те ѝ
са върнати.

В заниските на конегата има
мисия на него, а в pdf на Nine е
заглавък. Другият не е тялото си правилно.
Subset sum е свидетелски
knapsack с ограничено
условие.

Edit distance

Дадени са ви две думи u и v и искате да разберете какъв е възможен начин, по който да преобразувате думата u в думата v. Извънъважливи операции възможно:

- Добавяне
- Извличане
- Замяна

Самите различията между резултатните думи да са резултат от различни операции.

Тогава ще съществува определен начин да минимизирате, но и спрямо цените на операциите.

① # Изпълним ли така също?:

Потърпим неподходящи резултати, т.е. решението е чисто и съвсем иначе е на стапка и възможността за редицата на операции.

Виждате какъвъв във всички начини имате какъвъв преобразъци $u[1 \dots i]$ и $v[1 \dots j]$ и искате да конструирате $u[1 \dots i]$ в $v[1 \dots j]$.

т.е. имате $\Theta(|u|, |v|)$ подобрение

(2) Guess за част от решението:

Имете предикса $u[1..i]$ и
 $v[1..j]$ и сър цвото им за думите
 $u[i:j]$ и $v[i:j]$:

u ————— i
 v ————— j



Имете да си убедите че т.е. след
което ние $u[i:j] = v[i:j]$. Имете
зато опции

a) да направим insert на думата
 $v[i:j]$ на i-та позиция в дума.

Тогава ще зависи от оптималното
решение за членови $u[1..i]$ с

$v[1..(i-1)]$ (този едър улов. думата
от v : $v[i:j]$, но членуването е предположено
и не зиди как да "дели" една

предфикс на u и v , останалите
поке на едно от тях трябва е

страго на-десет от този едър улов.
предфикс $u[1..i]$, $v[i:j]$.

- b) от корректнм delete на букву $v[i]$. Тогда не останется
чтобы в $u[1..(i-1)]$ и
 $v[1..i]$, а это значит оптимального
меня не тут.
- c) от корректнм replace $u[i] \rightarrow v[i]$
кто, что са единицы в deshift ,
иначе са некие добавки и удали
меня за вставка $v[i]$ из буквы.
Тогда ни trivial оптимального
решение $\rightarrow u[1..(i-1)]$ и
 $v[1..(i-1)]$.

③ Так в report на языковые переп.

уп-е:

$$\text{dp}[i][j] = \min \left(\begin{array}{l} \text{cost of insert } v[j] + \text{dp}[i][j-1], \\ \text{cost of delete } u[i] + \text{dp}[i-1][j], \\ \text{cost of replace } u[i] \text{ with } v[j] + \text{dp}[i-1][j-1] \end{array} \right)$$

for $i=1..|u|$ and $j=1..|v|$

Пренето за пресъединение на нодове е $O(1)$.

(4) Типологична квадратика е от всички нодове на дървата преди всички геометрични

for $i \leftarrow 0$ to $|u|$ do

 for $j \leftarrow 0$ to $|v|$ do

Пример:

$u = \text{snowy}$

$v = \text{sunny}$

$s(n(o(w)y$
 ↑ ↓ ↓ ↓
 s n o w y

Червей във $L \rightarrow u$ знае insert u ,

$O \rightarrow v$ знае замени $O \in u, Q$

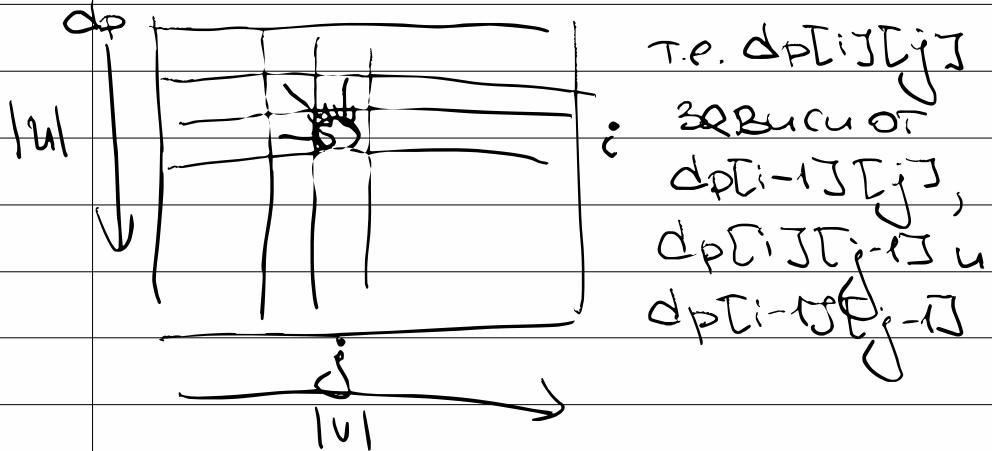
$w \rightarrow v$ знае delete w .

Това е съмнително често. Друга конфигурация:

$(s(n(o(w)y) y) y$ често

Суръю. Определение знакоу ТОКа
 иже речиоо же иогъзъжите зо
 згение зноуы и озънъи:
 $\langle e, e \rangle, \langle e, s \rangle, \langle e, su \rangle, \dots$,
 $\langle s, e \rangle, \langle s, s \rangle, \langle s, su \rangle, \dots$,
 $\langle su, e \rangle, \langle su, s \rangle, \langle su, su \rangle, \dots$
 УТН. Кто иогъ зноу иогъзъ
 и озънъато зноу.

Кто иД тоинъекъ:



ТОКъ възбъдяется въмъзъ, че не иу
 тръбъ иено морпхъ, зе ѹчи иогъзъ
 изънъленъто зо $\langle i, j \rangle$, съ иу тредъбр
 дънъзъни съзъ $(i-1)$ иа рез и $(j-1)$ иа
 зънъка.

⑤ Нашето решение се наема за в
 $dp[i][j][l][v]$

Задорвихме, че имена базови
слуги, която сае зърни
и розни предикси!

Ако $dp[0][0] \leftarrow 0$, залага
се предикс от розни никакви
имени, за да убедявам е с е.

Ако се създават
предикси и и розни предикси
на U , то v и w е съвместното
ко предикси на U , тъкмо delete
операции предикс до извличане
на U , за да го консервираме е с
т.е. $for i \leftarrow 1 to l do$

$dp[i][0] \leftarrow i$

Аналогично, ако предикси на U
е розни, а предикси на U не е.

Тогава блоки delete, insert
операции розни в U , за да
получим предикси на U .

for $j \leftarrow 1$ to $|v_1|$ do
 $dp[0:j][j] \leftarrow j$

Take ∞ значение для $dp[i:j]$ если
не insert, replace и delete
1, как replace не является единицей
единицей. Составляем
заправляя с ненулевыми. А сколько?
Примера есть $O(|u_1| \cdot |v_1|)$ костюм и
ночные (хотя это же можно сделать
 $O(|u_1| + |v_1|))$. $|u_1|$ $|v_1|$
единица (u, u^*, v, v^*)
 $dp[0 \dots n][0 \dots m]$
 $dp[0:j][0] \leftarrow 0$
for $i \leftarrow 1$ to n do
 $dp[i][0] \leftarrow i$
for $j \leftarrow 1$ to m do
 $dp[0][j] \leftarrow j$
for $i \leftarrow 1$ to n do
for $j \leftarrow 1$ to m do
 $dp[i][j] \leftarrow +\infty$

$\text{edMemoAux}(u, 1, n, v, 1, m, dp)$
return $dp[u][v]$

$\text{edMemoAux}(u, i, n, v, j, m, dp)$

if $dp[i][j] \neq +\infty$

return $dp[i][j]$

else

$res \leftarrow +\infty$

if $i = n$

$res \leftarrow j$

elseif $j = m$

$res \leftarrow i$

else

$res \leftarrow \min(1 + dp[i][j-1],$
 $1 + dp[i-1][j], \text{diff}(u[i], v[j]) +$
 $dp[i-1][j-1])$

$dp[i][j] \leftarrow res$

return res

if zero | $\text{diff}(c_1, c_2)$
if $c_1 = c_2$
return 1
return 0

Correspondenzen:

edIter(u, n, v, m)

$dp[0..n][0..m]$

$dp[0][0] \leftarrow 0$

for $i \leftarrow 1$ to n do

$dp[i][0] \leftarrow i$

for $j \leftarrow 1$ to m do

$dp[0][j] \leftarrow j$

for $i \leftarrow 1$ to n do

for $j \leftarrow 1$ to m do

$dp[i][j] \leftarrow \min ($

$1 + dp[i][j-1], 1 + dp[i-1][j],$

$\text{diff}(u[i], v[j]) + dp[i-1][j-1])$

return $dp[n][m]$.