Forums

Select a Forum ▼ Search | Watch Thread | My Post History | My Watches | User Settings

View: Flat (oldest first) | Threaded | Tree

Previous Thread | Next Thread

Forums Algorithm Discussions Skill-building and Educational Discussion for Algorithm Advanced dynamic programming techniques

Re: Advanced dynamic programming techniques (response to post by **anastasov.bg**) | 1 edit | Sat, Aug 1, 2009 at 8:44 PM EDT

Reply

FameofLight

534 posts

I have recently started practicing problem from TOpcoder on DP . I have encountered some amazing problem which teach me something.

Apart from above high level stuff , there are quite easy techniques which used with dp wonders in problem solving .

Like this http://forums.topcoder.com/?module=Thread&threadID=625789&start=0&mc=11
Instead of Going to any bottom up or top down if you go to all possible next state from current state it is more easy to think

Also this one http://www.topcoder.com/stat?c=problem_statement&pm=8538 which beautifully uses bit masking with dp.

I have also a list of problems from here http://forums.topcoder.com/?module=Thread&threadID=625789&start=0&mc=11 . I am currently solving problem in green and yellow.

My question you guys have much experience then me , can post some question , analysis or tutorial which will be really helpful to understand other techniques used with dp.

Re: Advanced dynamic programming techniques (response to post by Duc) | Reply Sat, Jul 11, 2009 at 10:43 PM EDT

felix_halim
203 posts

The algorithm has been described by ardiankp above. It's a bit high level but it will make sense after several hours ;). Just to add more details so that it's easier to understand:

Initially, you have a simple DP O(N^2 * K):

```
s[i] = sum of weights from w[i] to w[n]

for i:=1 to N do
  dp[1][i] = 0;

for k:=2 to K do
  for a:=1 to N do
    for b:=a+1 to N do
      dp[k][a] = max(dp[k][a], (b-a) * s[b] + dp[k-1][b]);

all = sum of s from s[1] .. s[n]
minCost = all - w[1] - dp[K][1];
```

We want to eliminate the innermost loop "for b:=a+1 to N do".
Each value **b** here is actually a line equation:

```
dp[k][a] = (b-a)*s[b] + dp[k-1][b];
dp[k][a] = (-s[b]) * a + s[b] * b + dp[k-1][b];
is a line y = m * x + c
```

We know all the value for m[b] = -s[b] and c[b] = s[b] * b + dp[k-1][b] from the pre-calc and the previous DP iteration. Hence, we have N line equations which are represented by m[b] and c[b].

Since we want the y = dp[k][a] value to be maximum, we are interested only t has the highest y value for all x. The good thing is all lines is already sorted by it's gradient m[b] in increasing order. Therefore we can omit the sorting and do the convex-hull-like algorithm in O(N).

Having this, we can improve the DP above to O(N * K):

```
s[i] = sum of weights from w[i] to w[n]

for i:=1 to N do
  dp[1][i] = 0;

for k:=2 to K do {
  select the top-most lines (m,c)
  from all line equations (1..N)
  using convex-hull-like algorithm

  i = 0;
  for a:=1 to N do {
    dp[k][a] = max(dp[k][a], m[i] * a + c[i]);

    advance i as necessary to maintain
    line i as the top-most line
```

```
        }
    }

    all = sum of s from s[1] .. s[n]
    minCost = all - w[1] - dp[K][1];
```

Hope that helps

Re: Advanced dynamic programming techniques (response to post by **ardiankp**) | Reply Fri, Mar 13, 2009 at 10:08 PM EDT

Duc
215 posts

given set of points and gradients, find the minimum y-intercept for each gradient. Ignoring the sorting, this problem can be solved in O(n).

Can you describe the algorithm... I dont know how to solve this in O(n)

Re: Advanced dynamic programming techniques (response to post by **anastasov.bg**) | Reply Thu, Jan 8, 2009 at 4:01 PM EST

nima.ahmadi
116 posts

If I remember correctly, the task `Cutting A Greed' from IOI 2006 practice, is a good problem to practice on. The sad thing is that it doesn't come with test cases!
Anyway here you can find the problem statement.

Re: Advanced dynamic programming techniques (response to post by **ardiankp**) | Reply Wed, Jan 7, 2009 at 11:31 PM EST

dragoon
130 posts

Hmm... Ok I will code it and let you know my condition. Anyway thanks for the nice problem and nice solution too :)

Re: Advanced dynamic programming techniques (response to post by **dragoon**) | Reply Wed, Jan 7, 2009 at 12:13 PM EST

ardiankp
677 posts

Support

it's O(N) because of the lines are automatically sorted (the gradient is in decre
increasing depend on the direction you see)).

Anyway you got the idea :)

Re: Advanced dynamic programming techniques (response to post by **ardiankp**) | Reply Wed, Jan 7, 2009 at 9:24 AM EST

dragoon
130 posts

Well wont it take O(nlgn) in each step? so total run time according to me should be, knlgn? am i wrong? if so then how to do that in O(n) instead of nlgn?

Re: Advanced dynamic programming techniques (response to post by **dragoon**) | Reply Wed, Jan 7, 2009 at 5:50 AM EST

Let the number of leaves in location i be wi. First, I ignore the first hole since it is always there.

If we do not put any hole, then the total cost = w1 + 2w2 + 3w3 + ... + wn. If we put one hole in position 4, then the total cost will be reduced by 4*(w4 + w5 + w6 + ... + wn). If we put another

ardiankp
677 posts

hole in position 6, then the total cost will be reduced by (6-4)*(w6 + w7 + ... + wn).

Therefore, let si be the sum of wi from i to n, if we put holes in position a,b,c,d then the total reduced cost is
a.sa +
(b-a).sb +
(c-b).sc +
(d-c).sd,
and our objective is to maximize this sum.

Now, let DP[K-1,b] be the optimal placing of (K-1) holes when the first is in b (e.g. DP[3,b] = (c-b).sc + (d-c).sd, for optimal choice of c and d), then DP[K,a] = the minimum of (b-a).sb + DP[K-1,b], for all possible b.

We can see that (b-a).sb + DP[K-1,b] form a line equation with gradient -sb, and offset b.sb + DP[K-1,b].

Hence, to calculate DP[K,a] for all a, first we built the line equation of all b, extract the "promising lines" (gi < gj and oi > oj, where g and o are gradient and offset respectively). Having all the lines, we can do line-sweep to find the optimal b for all a in O(n).

Re: Advanced dynamic programming techniques (response to post by forest) | Reply 1 edit | Wed, Jan 7, 2009 at 1:44 AM EST

dragoon
130 posts

May be it will sound stupid, but will BS on the max limit for each pile would work?

Edit: Ignore it, ArdianKp's solution seems to be interesting. Can you please enlight us more?

Re: Advanced dynamic programming techniques (response to post by forest) | Reply Tue, Jan 6, 2009 at 7:57 PM EST

forest
82 posts

up :). Any hints, please, anyone.

Support

Re: Advanced dynamic programming techniques (response to post by gawry) | Reply Fri, Apr 4, 2008 at 2:26 AM EDT

forest
82 posts

Another example problem is: http://vn.spoj.pl/IOITRAIN/problems/NKLEAVES/en

Really nice problem! Its strict timelimit made me a little nervous but I still like it :]

Could you give a hint how to solve it ?

Re: Advanced dynamic programming techniques (response to post by ardiankp) | Reply Tue, Mar 25, 2008 at 5:34 PM EDT

gawry
207 posts

Another example problem is: http://vn.spoj.pl/IOITRAIN/problems/NKLEAVES/en

Really nice problem! Its strict timelimit made me a little nervous but I still like it :]

Re: Advanced dynamic programming techniques (response to post by **anastasov.bg**) | Reply Tue, Mar 25, 2008 at 1:11 PM EDT

ardiankp
677 posts

I am no expert in both techniques, but I just try to give few insight.

1) There has been a discussion in similar problems applying same technique in http://forums.topcoder.com/?module=Thread&threadID=579321. You can check that thread :)

2) Try to reduce the problems into: given set of points and gradients, find the minimum y-intercept for each gradient. Ignoring the sorting, this problem can be solved in O(n). Another example problem is: http://vn.spoj.pl/IOITRAIN/problems/NKLEAVES/en/ (thanks to **ktuan** who introduce me this problem :))

Advanced dynamic programming techniques | Reply Tue, Mar 25, 2008 at 11:10 AM EDT

anastasov.bg
312 posts

I learned dynamic programming mainly from TopCoder (mostly by the tutorial). I find DP to be really helpful in solving a lot of problems and I have encountered some problems, which require some advanced techniques to be solved:

1) Reducing the inner cycle (or fully eliminating the inner cycle). As an example of such one is this **neal_wu**'s solution.

2) Using 'Convex hull' to speed up the transitions. An example of such a problem is 'acquire' from USACO Gold 2008 March.

I don't fully understand the first technique and I don't understand the second at all. I have looked over the solution to 'acquire', but I don't get it.

So here is my request:

Can someone put a link, or give a good description of these techniques (if there are others, mention them) and some problems to practice them on.

Forums Algorithm Discussions Skill-building and Educational Discussion for Algorithm Advanced dynamic programming techniques RSS
Previous Thread | Next Thread

Support

OTHERS
SITEMAP
ABOUT US
CONTACT US
HELP CENTER
PRIVACY POLICY
TERMS

topcoder is also on

Support