Member Count: 612,969 - February 18, 2014 [Get Time]

Hello, paulocezar | Logout

My TopCoder

Competitions
TopCoder Networks
Events
Statistics
Tutorials
Forums
Surveys
My TopCoder
Help Center
About TopCoder

Member Search:
Handle: Go
Advanced Search

Forums

Round Tables
News Discussions
Algorithm Matches
Marathon Matches
NASA Tournament Lab
Software Forums
TopCoder Cookbook
High School Matches
Sponsor Discussions

Search

Watch Thread | My Post History | My Watches | User Settings
View: Flat (newest first) | Threaded | Tree
Previous Thread | Next Thread

Forums ▸ Round Tables ▸ Educational Discussion ▸ Range update in BIT

Range update in BIT | Feedback: (+1/-1) | [+] [-] | Reply **Wed, Jul 27, 2011 at 8:18 AM BRT**

minib00m
25 posts

Hello,
I wonder how to make range update in BIT(binary indexed tree), because doing updates for every element in range [a,b] wouldn't be very good idea.

With segment tree it is easy to do that (range update), but how to do this with BIT ? I'm new to this structure, so thanks for any hints.

Cheers

Re: Range update in BIT (response to post by minib00m) | Feedback: (+1/-0) | [+] [-] | Reply **Wed, Jul 27, 2011 at 10:34 AM BRT**

VladBelous
30 posts

I don't see this often done with BITs (as segment trees are better for this), but I can see how you could use the same "lazy update"/"push downward" technique as with segment trees. That would need a top-bottom traversal, instead of the usual bottom-up.

Also, this (at least to me) seems to increase both query and update times to O(log^2 n), since when you "touch" a lazily updated interval (tree node), all it's subintervals (children) must be updated, but in BIT a node corresponding to an interval of length N has O(log N) children. In segment trees there are just 2 children.

Anyone knows if better than O(log^2 n) is possible?

Re: Range update in BIT (response to post by VladBelous) | Feedback: (+1/-2) | [+] [-] | Reply **Wed, Jul 27, 2011 at 6:17 PM BRT**

pt1989
290 posts

Yes you can

Re: Range update in BIT (response to post by pt1989) | Feedback: (+1/-0) | [+] [-] | Reply **Wed, Jul 27, 2011 at 9:54 PM BRT**

fushar
381 posts

I don't understand his approach, could someone explain that more thoroughly?

Re: Range update in BIT (response to post by fushar) | Feedback: (+3/-0) | [+] [-] | Reply **Thu, Jul 28, 2011 at 1:00 AM BRT**

lg5293
20 posts

I know how to change a BIT to range updating, but I don't know how to do both query and update ranges.

Basically, for a BIT, there are two modes available:

a) point updating, range querying
b) range updating, point querying

You are probably more familiar with BITs using (a).

For (b), it's just a very simple modification, so I'll try to explain it the best I can.

For (a), when you call update(x, v), it will add the value of v to the position at x, and calling query(x) will add up all the elements x' <= x and return that.

Now, to extend it to (b), we see that when we call update(x, v), it will affect all the queries x' >= x. Therefore, to update a range [a,b], we can call update(a,v) and update(b+1,v). Then, to get a point, we call query(p). Note that this will actually return the actual value at p, not the cumulative sum.

Now, to see why this works, see the following examples.
Suppose we just called update(a,v) and update(b+1,v).

Now, let's say we called query(p). We have three cases:
p < a. p will not be affected by the updates, so query(p) will not be affected and still return the correct result
p > b. p will be affected by the update(a,v) since p >= a, and update(b+1,-v) since p >= b+1, therefore, v-v=0 so everything cancels out and query(p) will mpt be affected and return the correct result
a <= p <= b. p is only affected by update(a,v), but not update(b+1,-v), therefore, query(p)'s value is increased by v, and will return the correct result

Hopefully, that's helpful. However, I don't know how to do both updating ranges and querying ranges, so I am also wondering if someone can explain that.

Re: Range update in BIT (response to post by lg5293) | Feedback: (+4/-0) | [+] [-] | Reply **Thu, Jul 28, 2011 at 4:02 AM BRT**

VladBelous
30 posts

As for {range update, point query} option, I can suggest a different view, which is easier (I think):

Suppose A[0..n] is the original array. Instead of storing A[0..n] in BIT, store it's "differentiated" array, i.e.

D[0] = A[0],
D[1] = A[1]-A[0],
D[2] = A[2]-A[1], etc.

Build a (usual) BIT on top of D[], instead of A[].
Now notice that A[k] = D[0]+D[1]+...+D[k], thus bit_query(D[],k) will in fact return your point value A[k].

To add v to range A[i..j], simply do bit_update(D[i],v) and bit_update(D[j+1],-v). Now any point query in [i..j] will return a value larger by v, and for outside of [i..j] the return value wont be affected.

Re: Range update in BIT (response to post by fushar) | Feedback: (+22/-3) | [+] [-] | Reply **3 edits | Fri, Jul 29, 2011 at 5:59 PM BRT**

AnilKishore
211 posts

While solving http://www.spoj.pl/submit/HORRIBLE , I came across that comment by **sicasli** and was thrilled to know that BIT can be used for Range Update and Range Query also. I didn't understand his approach though. The following is how I thought about it and solved. Its some what tricky to explain the working of BIT in simple text, I'll try my best.

```
Similar to Range Update - Point query, we maintain a BIT (say B1)
- Add v to [a..b] --> Update(a,v) and Update(b+1,-v) on the BIT B1
- Query(p) on B1 now gives the correct value of A[p]

The answer we want is ( Sum(0..b) - Sum(0..a-1) ), so lets design Sum(0..p). The thing with BIT is,
if you design it to work for one update ( which is easy to imagine ) and all possible queries on that
one update, mostly it should work for multiple updates ;)

Lets consider just one update : Add v to [a..b], rest all are 0


Now, consider Sum(0..p) for all possible p

1. 0 <= p <  a : 0
2. a <= p <= b : v * ( p - (a-1) )
3. b <  p <  n : v * ( b - (a-1) )

This suggests that, for a index p, if we have (v * p) we can get the Sum(0..p) by subtracting X from it

1. 0 <= p <  a : v = 0, X = 0
2. a <= p <= b : (v*p) - (v*(a-1)), X = v*(a-1)
3. b <  p <  n : v = 0, X = - v*b + v*(a-1)

So, we need to maintain something else too, to get that extra X factor and that should give
0 for p < a, v*(a-1) for p in [a..b], -v*b+v(a-1) for p > b.
Does this ring something ;) ? hoho ! one more BIT for keeping this X factor.

We need to maintain another BIT (say B2)
- Add v to [a..b] --> Update(a,v*(a-1)) and Update(b+1,-v*b) on the BIT B2
- Query(p) on B2 now gives the extra sum that should be subtracted from A[p]*p
```

[Complete Code in Edit]

Re: Range update in BIT (response to post by AnilKishore) | Feedback: (+3/-0) | [+] [-] | Reply **Fri, Jul 29, 2011 at 6:29 PM BRT**

minib00m
25 posts

THANKS BUDDY !

That's very well explained, i loved it.

Forums ▸ Round Tables ▸ Educational Discussion ▸ Range update in BIT
Previous Thread | Next Thread
RSS

Home | About TopCoder | Press Room | Contact Us | Careers | Privacy | Terms
Competitions | Cockpit