Longest Monotonically Increasing Subsequence Size (N log N)

After few months of gap posting an algo. The current post is pending from long time, and many readers (e.g. here, here, here may be few more, I am not keeping track of all) are posting requests for explanation of the below problem.

Given an array of random numbers. Find *longest monotonically increasing subsequence* (LIS) in the array. I know many of you might have read recursive and dynamic programming (DP) solutions. There are few requests for O(N log N) algo in the forum posts.

For the time being, forget about recursive and DP solutions. Let us take small samples and extend the solution to large instances. Even though it may look complex at first time, once if we understood the logic, coding is simple.

Consider an input array A = {2, 5, 3}. I will extend the array during explanation.

By observation we know that the LIS is either {2, 3} or {2, 5}. ***Note that I am considering only strictly increasing monotone sequences***.

Let us add two more elements, say 7, 11 to the array. These elements will extend the existing sequences. Now the increasing sequences are {2, 3, 7, 11} and {2, 5, 7, 11} for the input array {2, 5, 3, 7, 11}.

Further, we add one more element, say 8 to the array i.e. input array becomes {2, 5, 3, 7, 11, 8}. Note that the latest element 8 is greater than smallest element of any active sequence (*will discuss shortly about active sequences*). How can we extend the existing sequences with 8? First of all, can 8 be part of LIS? If yes, how? If we want to add 8, it should come after 7 (by

Google™ Custom Search

GeeksforGeeks

53,521 people like GeeksforGeeks.

Interview Experiences
Advanced Data Structures
Dynamic Programming
Greedy Algorithms
Backtracking
Pattern Searching
Divide & Conquer
Mathematical Algorithms
Recursion
Geometric Algorithms

replacing 11).

Since the approach is *offline (what we mean by offline?)*, we are not sure whether adding 8 will extend the series or not. Assume there is 9 in the input array, say {2, 5, 3, 7, 11, 8, 7, 9 …}. We can replace 11 with 8, as there is potentially *best* candidate (9) that can extend the new series {2, 3, 7, 8} or {2, 5, 7, 8}.

Our observation is, assume that the end element of largest sequence is E. We can add (replace) current element A[i] to the existing sequence if there is an element A[j] (j > i) such that E < A[i] < A[j] or (E > A[i] < A[j] – for replace). In the above example, E = 11, A[i] = 8 and A[j] = 9.

In case of our original array {2, 5, 3}, note that we face same situation when we are adding 3 to increasing sequence {2, 5}. I just created two increasing sequences to make explanation simple. Instead of two sequences, 3 can replace 5 in the sequence {2, 5}.

I know it will be confusing, I will clear it shortly!

The question is, when will it be safe to add or replace an element in the existing sequence?

Let us consider another sample A = {2, 5, 3}. Say, the next element is 1. How can it extend the current sequences {2,3} or {2, 5}. Obviously, it can’t extend either. Yet, there is a potential that the new smallest element can be start of an LIS. To make it clear, consider the array is {2, 5, 3, 1, 2, 3, 4, 5, 6}. Making 1 as new sequence will create new sequence which is largest.

The observation is, when we encounter new smallest element in the array, it can be a potential candidate to start new sequence.

From the observations, we need to maintain lists of increasing sequences.

In general, we have set of **active lists** of varying length. We are adding an element A[i] to these lists. We scan the lists (for end elements) in decreasing order of their length. We will verify the end elements of all the lists to find a list whose end element is smaller than A[i] (*floor* value).

Our strategy determined by the following conditions,

1. If A[i] is smallest among all *end* candidates of active lists, we will *start* new active list of length 1.

2. If A[i] is largest among all *end* candidates of active lists, we will clone the *largest*

Popular Posts

All permutations of a given string
Memory Layout of C Programs
Understanding “extern” keyword in C
Median of two sorted arrays
Tree traversal without recursion and without stack!
Structure Member Alignment, Padding and Data Packing
Intersection point of two Linked Lists
Lowest Common Ancestor in a BST.
Check if a binary tree is BST or not
Sorted Linked List to Balanced BST

active list, and extend it by A[i].

3. If A[i] is in between, we will find a list with *largest end element that is smaller than* A[i]. Clone and extend this list by A[i]. We will discard all other lists of same length as that of this modified list.

Note that at any instance during our construction of active lists, the following condition is maintained.

"end element of smaller list is smaller than end elements of larger lists".

It will be clear with an example, let us take example from wiki {0, 8, 4, 12, 2, 10, 6, 14, 1, 9, 5, 13, 3, 11, 7, 15}.

```
A[0] = 0. Case 1. There are no active lists, create one.
0.
-----------------------------------------------------------------------------
A[1] = 8. Case 2. Clone and extend.
0.
0, 8.
-----------------------------------------------------------------------------
A[2] = 4. Case 3. Clone, extend and discard.
0.
0, 4.
~~0, 8~~. Discarded
-----------------------------------------------------------------------------
A[3] = 12. Case 2. Clone and extend.
0.
0, 4.
0, 4, 12.
-----------------------------------------------------------------------------
A[4] = 2. Case 3. Clone, extend and discard.
0.
0, 2.
~~0, 4~~. Discarded.
0, 4, 12.
```

```
0, 4, 12.
-----------------------------------------------------------------------------
A[5] = 10. Case 3. Clone, extend and discard.
0.
0, 2.
0, 2, 10.
~~0, 4, 12~~. Discarded.
-----------------------------------------------------------------------------
A[6] = 6. Case 3. Clone, extend and discard.
0.
0, 2.
0, 2, 6.
~~0, 2, 10~~. Discarded.
-----------------------------------------------------------------------------
A[7] = 14. Case 2. Clone and extend.
0.
0, 2.
0, 2, 6.
0, 2, 6, 14.
-----------------------------------------------------------------------------
A[8] = 1. Case 3. Clone, extend and discard.
0.
0, 1.
~~0, 2~~. Discarded.
0, 2, 6.
0, 2, 6, 14.
-----------------------------------------------------------------------------
A[9] = 9. Case 3. Clone, extend and discard.
0.
0, 1.
0, 2, 6.
0, 2, 6, 9.
~~0, 2, 6, 14~~. Discarded.
-----------------------------------------------------------------------------
A[10] = 5. Case 3. Clone, extend and discard.
```

705 Subscribe

Recent Comments

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 18 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 21 minutes ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 46 minutes ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 48 minutes ago

newCoder3006 If the array contains negative numbers also. We...

Find subarray with given sum · 1 hour ago

newCoder3006 Code without using while loop. We can do it...

Find subarray with given sum · 1 hour ago

```
0.
0, 1.
0, 1, 5.
~~0, 2, 6~~. Discarded.
0, 2, 6, 9.
----------------------------------------------------------------------------
A[11] = 13. Case 2. Clone and extend.
0.
0, 1.
0, 1, 5.
0, 2, 6, 9.
0, 2, 6, 9, 13.
----------------------------------------------------------------------------
A[12] = 3. Case 3. Clone, extend and discard.
0.
0, 1.
0, 1, 3.
~~0, 1, 5~~. Discarded.
0, 2, 6, 9.
0, 2, 6, 9, 13.
----------------------------------------------------------------------------
A[13] = 11. Case 3. Clone, extend and discard.
0.
0, 1.
0, 1, 3.
0, 2, 6, 9.
0, 2, 6, 9, 11.
~~0, 2, 6, 9, 13~~. Discarded.
----------------------------------------------------------------------------
A[14] = 7. Case 3. Clone, extend and discard.
0.
0, 1.
0, 1, 3.
0, 1, 3, 7.
~~0, 2, 6, 9~~. Discarded.
```

```
0, 2, 6, 9, 11.
-----------------------------------------------------------------------------
A[15] = 15. Case 2. Clone and extend.
0.
0, 1.
0, 1, 3.
0, 1, 3, 7.
0, 2, 6, 9, 11.
0, 2, 6, 9, 11, 15. <-- LIS List
-----------------------------------------------------------------------------
```

It is required to understand above strategy to devise an algorithm. Also, ensure we have maintained the condition, "*end element of smaller list is smaller than end elements of larger lists*". Try with few other examples, before reading further. It is important to understand what happening to end elements.

Algorithm:

Querying length of longest is fairly easy. Note that we are dealing with end elements only. We need not to maintain all the lists. We can store the end elements in an array. Discarding operation can be simulated with replacement, and extending a list is analogous to adding more elements to array.

We will use an auxiliary array to keep end elements. The maximum length of this array is that of input. In the worst case the array divided into N lists of size one (*note that it does't lead to worst case complexity*). To discard an element, we will trace ceil value of A[i] in auxiliary array (again observe the end elements in your rough work), and replace ceil value with A[i]. We extend a list by adding element to auxiliary array. We also maintain a counter to keep track of auxiliary array length.

Bonus: You have learnt Patience Sorting technique partially :).

Here is a proverb, "*Tell me and I will forget. Show me and I will remember. Involve me and I will understand.*" So, pick a suit from deck of cards. Find the longest increasing sub-sequence of cards from the shuffled suit. You will never forget the approach. 🙂

Given below is code to find length of LIS,

```
#include <iostream>
#include <string.h>
#include <stdio.h>

using namespace std;

#define ARRAY_SIZE(A) sizeof(A)/sizeof(A[0])
// Binary search (note boundaries in the caller)
// A[] is ceilIndex in the caller
int CeilIndex(int A[], int l, int r, int key) {
    int m;

    while( r - l > 1 ) {
        m = l + (r - l)/2;
        (A[m] >= key ? r : l) = m; // ternary expression returns an l-
    }

    return r;
}

int LongestIncreasingSubsequenceLength(int A[], int size) {
    // Add boundary case, when array size is one

    int *tailTable   = new int[size];
    int len; // always points empty slot

    memset(tailTable, 0, sizeof(tailTable[0])*size);

    tailTable[0] = A[0];
    len = 1;
    for( int i = 1; i < size; i++ ) {
        if( A[i] < tailTable[0] )
            // new smallest value
            tailTable[0] = A[i];
        else if( A[i] > tailTable[len-1] )
            // A[i] wants to extend largest subsequence
            tailTable[len++] = A[i];
        else
            // A[i] wants to be current end candidate of an existing s
            // It will replace ceil value in tailTable
            tailTable[CeilIndex(tailTable, -1, len-1, A[i])] = A[i];
    }

    delete[] tailTable;

    return len;
}
```

```
int main() {
    int A[] = { 2, 5, 3, 7, 11, 8, 10, 13, 6 };
    int n = ARRAY_SIZE(A);

    printf("Length of Longest Increasing Subsequence is %d\n",
            LongestIncreasingSubsequenceLength(A, n));

    return 0;
}
```

Complexity:

The loop runs for N elements. In the worst case (what is worst case input?), we may end up querying ceil value using binary search (log *i*) for many A[i].

Therefore, T(n) < O(log N!) = O(N log N). Analyse to ensure that the upper and lower bounds are also O(N log N). The complexity is THETA (N log N).

Exercises:

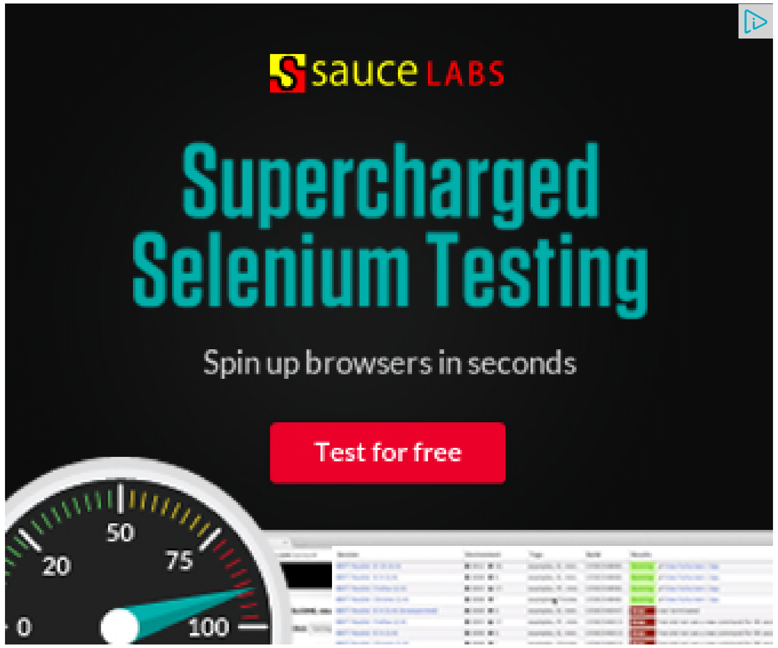
1. Design an algorithm to construct the longest increasing list. Also, model your solution using DAGs.

2. Design an algorithm to construct **all** monotonically increasing list**s of equal longest size**.

3. Is the above algorithm an *online* algorithm?

4. Design an algorithm to construct the longest *decreasing* list..

— Venki. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.



Related Tpoics:

- Remove minimum elements from either side such that 2*min becomes more than max
- Divide and Conquer | Set 6 (Search in a Row-wise and Column-wise Sorted 2D Array)
- Bucket Sort
- Kth smallest element in a row-wise and column-wise sorted 2D array | Set 1
- Find the number of zeroes
- Find if there is a subarray with 0 sum
- Divide and Conquer | Set 5 (Strassen's Matrix Multiplication)
- Count all possible groups of size 2 or 3 that have sum as multiple of 3

31 Tweet 2 1

Writing code in comment? Please use **ideone.com** and share the link here.

83 Comments GeeksforGeeks

Sort by Newest

lol • 23 days ago
I stopped reading after the invalid statement
1. If A[i] is smallest among all end candidates of active lists, we will start new a

Why should it being less than all end candidates mean it is also less than the

Reply • Share ›

Prateek Jain • 2 months ago
Have made it very Simple , Think of above implementing like a Binary Tree (Not Binary Search Tree) and the implementation becomes very simple and keep discarding the child nodes with larger value and same depth in tree.

Root is an empty container to hold the reference values, as there can be more

Below is the Java code implementation.

http://ideone.com/e.js/W1rcXp

Reply • Share ›

Prateek Jain • 2 months ago
Have made it very Simple , Think of above implementing like a Binary Tree (No
implementation becomes very simple and keep discarding the child nodes wit

Root is an empty container to hold the reference values, as there can be more

<script src="http://ideone.com/e.js/W1rcXp" type="text/javascript"></script>

Reply • Share ›

kraken • 4 months ago
Will we get all the longest subsequence?

Because if we take the example {2 5 3}

Then

2 : Clone =>
2

5 : Clone and Extend =>
2
2 5

3 : Clone Extend Discard
2
2 3

We dont get 2 and 5 using this approach?

Reply • Share ›

Robin Keskisärkkä kraken • 3 months ago
That's right, we don't get [2,5] because any continuation sequence that
reachable from [2,3]. For example: 3, 4, 1, 2, 3 is solved as:
3
--
3
3, 4
--
1 (more "potential" than 3)
3, 4
--
1
1, 2 (more "potential" than 3, 4)
--
1

1, 2
1, 2, 3
Reply • Share ›

Vivekz • 6 months ago
I dont think the implementation is correct, because everytime we encounter a
smallest element, we are NOT creating a new list but modifying existing tailTa

check out this link .. http://ideone.com/8Qoi5i
2 • Reply • Share ›

Neha Garg • 7 months ago
in third step we are finding the ceilindex and replacing a[i] over there .... plz tell
len?????
2 • Reply • Share ›

OP → Neha Garg • 5 months ago
Because len is always equals to the maximum length of monotonic su
Reply • Share ›

Guest • 7 months ago
its very well explained ,i understood the method but can someone explain me
Reply • Share ›

OP → Guest • 5 months ago
Time complexity = O(nlogn)
for each element we are doing one binary searh, so for total n element

Space complexity = O(n)
Reply • Share ›

googlybhai • 7 months ago

```
#include <iostream>
#include <string.h>
#include <stdio.h>
using namespace std;

#define ARRAY_SIZE(A) sizeof(A)/sizeof(A[0])

// Binary search (note boundaries in the caller)

// A[] is ceilIndex in the caller

int CeilIndex(int *A, int *table, int l, int r, int key) {
int m;
while( r-l > 1 ) {
m = l + (r-l)/2;
(A[table[m]] >= A[key] ? r : l) = m; // ternary expression returns an
}
```

see more

^ | ˅ • Reply • Share ›

Guest • 7 months ago

```
/* Modified the code to return longest increasing subsequence */

#include <iostream>

#include <string.h>

#include <stdio.h>
```

```
using namespace std;



#define ARRAY_SIZE(A) sizeof(A)/sizeof(A[0])

// Binary search (note boundaries in the caller)
```

see more

Reply • Share ›

Surya Baghrecha • 8 months ago

tailTable wont be having result sub sequence can you tell me what changes tc
sequence in nlogn
2, 5, 3, 7, 11, 8, 10, 13, 6,1

1 3 6 8 10 13 it will give this as output

Reply • Share ›

Kellogs ➜ Surya Baghrecha • 7 months ago

No.

(For 2)

2

****

(For 5)

2

2,5

****

(For 3)

2

2,3

2,5 (Discard)

see more

^ | ⌄ • Reply • Share ›

Surya Baghrecha ➦ Kellogs • 7 months ago

if(A[i] < tailTable[0])
// new smallest value
tailTable[0] = A[i];
wont it go into this?? how is it discarded?

^ | ⌄ • Reply • Share ›

Sudipto • 9 months ago

@Venki: Suppose we have an array of pairs, like: (a, b) instead of integers, wh
b < c.
Let the array be : {{1, 3}, {5, 9}, {5, 7}, {8, 23}}. Here how are we going to comp
pairs can succeed the other pair. If we replace {5, 9} with {5, 7} following the g
3, else it will be 2. So, how can we handle this situation?

^ | ⌄ • Reply • Share ›

Sudipto ➦ Sudipto • 9 months ago

I think this approach will work only if the elements of array are "totally o
comparable.

^ | ⌄ • Reply • Share ›

Rashik Hasnat • 9 months ago

???????? ????????

Reply • Share ›

Dip Mazumder • 9 months ago

&#039that was helpful :)&#039

Reply • Share ›

Rashik Hasnat • 9 months ago

that was helpful :)

1 • Reply • Share ›

Rajkiran • 10 months ago

What is meant by discard all other lists of the same length? At any given point
length, right? Can you please clarify?

Reply • Share ›

Rajkiran Rajkiran • 10 months ago

It is also not clear why the binary search function is called with the left
you want to position it before the start of the array). But it seems unne
guess.

Reply • Share ›

SS • 10 months ago

Great Tutorial !!!!!!!!!!

Reply • Share ›

Asap • 10 months ago

Can we use this approach for this questions?
http://www.geeksforgeeks.org/d...

Reply • Share ›

why this code is not working ?? Please help...

```
 #include<stdio.h>
int main()
{
    int n,a[10],A[10],len,low,high,mid;
    scanf("%d %d",&n,&a[0]);
    A[0]=a[0];
    len=1;
    for(int i=1;i<n;i++)
    {
            scanf("%d",&a[i]);
            if(a[i]>A[len-1])
                             A[len++]=a[i];
            else if(a[i]<A[0])
                 A[0]=a[i];
            else
            {
```

see more

^ | ⌄ • Reply • Share ›

Swapnil R Mehta • a year ago

Can't we just write a binary search function in place of CeilIndex and find the ir
tailTable and replace it with A[i].As tailTable is always sorted---->"end element
elements of larger lists".

^ | ⌄ • Reply • Share ›

Venki ➦ Swapnil R Mehta • a year ago

CeilIndex is binary search function only. Observe it carefully. It returns
guaranteed (Why?).

Am I missing anything to understand?

Reply • Share ›

Swapnil R Mehta Venki • a year ago

Sir, pls make the parameter received in ceilIndex() from int A[] t
was been sent as parameter.It will help remove confusions.
Rest alls correct. Thanks for reply :)

Reply • Share ›

Venki Swapnil R Mehta • a year ago

I am not sir.

I follow some coding convention. A for array, I have add

Reply • Share ›

Swapnil R Mehta Venki • a year ago

thanks for your valuable time...

Reply • Share ›

Bansari Shah • a year ago

Thanks for such a nice explanation! :)

Reply • Share ›

abhishek08aug • a year ago

Intelligent :D

```
/* Paste your code here (You may delete these lines if not writing c
```

Reply • Share ›

Qiao • a year ago

Will the CeilIndex function return the smallest element in A[] that is larger than return the largest element in A[] that is smaller than key.. Am I wrong??

```
/* Paste your code here (You may delete these lines if not writing c
```

Reply • Share ›

Qiao Qiao • a year ago

hmm.. I got it. Thanks~

Reply • Share ›

coderAce • a year ago

A small observation: The above procedure will find that LIS with the minimum

To elaborate a little, there can be multiple LISs for a given array. The above pr
list with end element smaller than A[i], clones the list and adds A[i] to it and fin

Now the discarded list obviously had its end element greater than A[i] and hen
having the minimum sum.

Please correct me if I am wrong

Reply • Share ›

Venki coderAce • a year ago

Good observation. The algorithm is greedy in nature, yet, yields smalle
subsequence.

Reply • Share ›

ravi • a year ago

Code for longest sub sequences in O(nlog(n))

```
 #include <stdio.h>
#define MX 16
int main()
```

```
{
        int array[MX]={0,8,4,12,2,10,6,14,1,9,5,13,3,11,7,15};
        int index;
        int k,j;
        int table[6][6];
        int m,n;
        for(m=0;m<6;m++)
        {
                for(n=0;n<6;n++)
                {
                        table[m][n]=-1;
                }
        }
```

see more

Reply • Share ›

anirudh beria • a year ago

i dont think this solution is nlogn due to check for min and max elements. A ma
nlogn sol.

1 • Reply • Share ›

cooldude • a year ago

44,1,21,41,90,65,82,72,50,16
For this example,output should be 1,21,41,65,82
but the above algo gives output as 4
with table as follows(end=length)
[1=1, 44=1]
[1=1, 21=2, 44=1]
[1=1, 21=2, 41=3, 44=1]
[1=1, 21=2, 41=3, 44=1, 90=4]
[1=1, 65=2, 41=3, 44=1, 90=4]

```
[1=1, 65=2, 82=3, 44=1, 90=4]
[1=1, 65=2, 72=3, 44=1, 90=4]
[50=2, 1=1, 72=3, 44=1, 90=4]
[16=2, 1=1, 72=3, 44=1, 90=4]
```

1 • Reply • Share ›

Venki cooldude • a year ago

http://ideone.com/dgEymU

• Reply • Share ›

cooldude Venki • a year ago

but still it doesnt work for this example and some other which i

• Reply • Share ›

Venki cooldude • a year ago

Please be specific. I got the following output to your inpu

1 21 41 65 72

The only difference your expected output and actual out
best possible next higher number to find largest sequen
elements constituted in the sequence, we only concern
sequence or not. I have left your expected output in my

I would encourage you read all comments as this discu

• Reply • Share ›

_maverick • a year ago

hello Venki a grt one!
I thank you frst for making things easy to understand.

may i have ur blog to get regular updates from u.

Once again thanks for this one.

1 • Reply • Share ›

Venki _maverick • a year ago

@Maverick, I am not active on personal blog. You can keep visiting Ge

• Reply • Share ›

Nandan Badheka • a year ago

very nice.

• Reply • Share ›

viswanath • a year ago

love it

```
/* Paste your code here (You may delete these lines if not writing co
```

• Reply • Share ›

cherry • a year ago

What is the out put for the given array
A[] = { 10,22,9,33,21,21,21,21,21 }; ?

yours is returning just 3. which is (10,22,33). isnt this wrong?

```
/* Paste your code here (You may delete these lines if not writing co
```

• Reply • Share ›

Venki cherry • a year ago

Thanks for bringing it into attention. I have only considered strictly incre
increasing monotone subsequence, it needs content and code modific

Reply • Share ›

Ravi • a year ago

In the above explanation (as explained in boxes), we can find longest increasir
code, we are only able to find length of LIS not LIS exactly (in all possible case

Let's take an example of {3, 4, 1}... according to code, final output (LIS) comes
4}.

Reply • Share ›

Venki Ravi • a year ago

@Ravi, I have mentioned in the content that the code is meant to find s
LIS as an exercise, and later added code also.

I got correct oupput for the sequence {3, 4, 1}

LIS - http://ideone.com/9odmXL

Note that I have mentioned that the sequence will be printed in reverse

Reply • Share ›

Load more comments

Subscribe Add Disqus to your site