Keith and the Giant Good Sequence

Christopher Xue

May 28, 2016

Sorry for the confusion: the problem statement should have asked for the smallest possible index that should be removed.

First, we establish the condition for checking whether a term can be added to an existing interesting sequence. The condition is: $|\text{new term} - \text{last term}| \leq 1$.

The idea is to try potential indices for removal and finding the maximum possible length in each case. If that length exceeds the existing maxLength, the new length becomes maxLength and the corresponding maxIndex is adjusted. To find the maximum possible length, we continue adding terms from the given sequence until the condition fails, or until the end of the given sequence.

It is inefficient to try to remove every single index in the sequence and then look for an interesting sequence there. Instead, we should limit the candidate indices for removal. Consider the sequence $\{3, 3, 3, 4, 5, 2, 6\}$. If we don't remove an index, we see that the first five terms constitute an interesting sequence, but that the next term, a 2, breaks the sequence. Thus, that 2 is a potential candidate for removal, because removing it potentially allows our sequence of five terms to connect with the terms after the 2.

On the other hand, consider the sequence $\{3, 3, 3, 4, 5, 3, 2, 1\}$. Again, the first five terms constitute an interesting sequence, but the next term, a 3, breaks that sequence. We might consider removing that 3, as we mentioned before, but it does not allow the original sequence to expand, since 5 does not connect with 2. We can instead consider the 5 a potential candidate for removal because it is a factor as to why the interesting sequence can not expand. This works in this case: by removing the 5, the 4 can now connect to the 3, and the interesting sequence expands. Thus we can characterize all candidates for the removed index as the following: either (1), they are one

element before the illegal element, or (2), they are the illegal element (here, "illegal element" denotes an element that cannot connect to the end of an interesting sequence). We do not have to consider indices in the middle of an interesting sequence, as it only shortens our sequence by 1 and does not allow it to expand. Moreoever, the way we iterate over the given sequence means we will not have to check whether the beginning of an existing interesting sequence is a candidate for removal.

However, there are some edge cases we may have to be worried about when thinking about the candidate indices. There is the case that the longest interesting sequence occurs when we don't remove anything from that subsequence. For example, given the list of integers $\{3, 5, 5, 4, 3, 4, 5, 6\}$, we should remove the 0^{th} index. To deal with this case, we loop over the sequence an initial time to determine the longest interesting sequence, without removing any indices, and setting this to be the initial maxLength. If this turns out to be the longest length, there are three cases to determine which index to remove:

1. The longest interesting sequence is exactly the given sequence. We remove the 0^{th} index in this case.

2. The longest interesting sequence contains the 0^{th} element but is shorter than the given sequence. We remove the index immediately after the end of this interesting sequence.

3. The longest interesting sequence does not contain the 0^{th} element. We remove the 0^{th} element in this case.

Now that we know which indices we should remove, we proceed as follows. We begin with some initial element of our to-be-built interesting sequence (at first, it is the 0^{th} element) and continue adding elements from the given sequence until the condition fails. Then, we try removing the element immediately before the illegal element, and check to see that the element two indices before the illegal element can pair with the illegal element. If so, we then continue until the condition fails, and this becomes our interesting sequence. We should keep track of where this condition has failed this time, because this becomes a candidate for removal and will save some computation. For our first illegal element, we also need to check if removing the illegal element itself is possible. To do this, we check to see whether the element immediately before the illegal element can connect with the element immediately after it. Then, we continue adding elements from the given sequence

until the condition fails - we can save some computation here by noting that this is almost identical to our first case, and will in fact be identical in most cases. It is those other cases that we have to worry about but can easily test for. Now, we determine which of the two cases gives the maxLength and adjust maxIndex accordingly. If the two cases give the same maxLength (which will be the case most times), then we take the smaller index as maxIndex. Now, we begin again with an initial element from the given sequence. We use the first illegal element for this purpose.

We keep repeating the above process, while adjusting maxLength and maxIndex as necessary, until the end of the given sequence, at which point we are done: we have found the maxLength and maxIndex values. This solution operates in linear time.