Hands On 6: Most frequent item in a stream: deterministic case

Algorithm Design

Ferraro Domenico
559813

1 Problem

Suppose to have a stream of n items, so that one of them occurs $> n/2$ times in the stream. Also, the main memory is limited to keeping just $O(1)$ items and their counters (where the knowledge of the value of n is not actually required). Show how to find deterministically the most frequent item in this scenario.

[Hint: the problem cannot be solved deterministically if the most frequent item occurs $\leq n/2$ times, so the fact that the frequency is $> n/2$ should be exploited.]

2 Solution

As we know, the problem cannot be solved deterministically if the most frequent item occurs $\leq n/2$ times. Before starting with the implementation of the solution, let's describe the idea behind it. Let's say that at a given moment the most frequent item is X. Instead of counting how many times X occurs, we can count how much X occurs more than any other item. The algorithm to do it is straightforward and uses just one counter: when an item arrives, increase the counter if the item is X, decrease it otherwise. It may happen that, given a new item, X is not the most frequent one anymore. This happens when we know that X does not occur more than the other items (i.e., the counter is zero) and the next item is not X. In this case, we are not interested in X anymore, so we can start counting how much the new value occurs more than the others. The following is the pseudocode of the algorithm when a new item arrives:

```
if (counter == 0)
  mostFrequent = newItem;
  counter = 1;
else if (newItem == mostFrequent)
  counter++;
else
  counter--;
```

The counter must be initialized to zero. When a new item arrives, the algorithm checks if it is equal to the current most frequent one. If it is, then the counter is increased, otherwise the counter is decreased. When a new item arrives, if the counter is zero then this new item is eligible to be the most

frequent one, so the counter is set to one and we keep in memory the newly arrived item as the most frequent.

This algorithm is known as *Boyer–Moore majority vote algorithm*. With this approach and known that the most frequent number occurs more than $n/2$ times, we can say, at a given moment, which item of the stream is the most frequent one. When an item occurs more than $n/2$ times, it is for sure the most frequent one and the other items will decrease the counter at most less than $n/2$ times. So, because the number of increments is higher than the decrements, the counter will always be greater than zero.

The algorithm uses one counter and stores also the most frequent item, so it takes $O(1)$ space.