Hands On 7: Deterministic data streaming

Algorithm Design

Ferraro Domenico
559813

1 Problem

Consider a stream of n items, where items can appear more than once. The problem is to find the most frequently appearing item in the stream (where ties are broken arbitrarily if more than one item satisfies the latter). For any fixed integer $k \geq 1$, suppose that only k items and their counters can be stored, one item per memory word; namely, only $O(k)$ memory words of space are allowed.

Show that the problem cannot be solved deterministically under the following rules: any algorithm can only use these $O(k)$ memory words and read the next item of the stream (one item at a time). You, the adversary, have access to all the stream, and the content of these $O(k)$ memory words: you cannot change these words and the past items, namely, the items already read, but you can change the future, namely, the next item to be read. Since any algorithm must be correct for any input, you can use any amount of streams and as many distinct items as you want.

Hints:

1. This is "classical" adversarial argument based on the fact that *any* deterministic algorithm A using $O(k)$ memory words gives the *wrong* answer for a *suitable* stream chosen by the adversary.

2. The stream to choose as an adversary is taken from a candidate set sufficiently large: given $O(k)$ memory works, let $f(k)$ denote the maximum number of possible situations that algorithm A can discriminate. Create a set of C candidate streams, where $C > f(k)$: in this way there are two streams $S1$ and $S2$ that A cannot distinguish, by the pigeon principle.

2 Solution

Let's define a function $MostFreq$ which gives the exact most frequent item of the stream. Given any deterministic algorithm A, we denote as $A(S_1)$ what the algorithm A returns as the most frequent item in the stream S_1.

Create a set of candidate streams

As an adversary, our goal is to find two streams S_1, S_2 such that $MostFreq(S_1) \neq MostFreq(S_2)$ but $A(S_1) = A(S_2)$.

Let's take a universe U and its subsets $\Sigma_i \subseteq U$ with cardinality $|\Sigma_i| = \left\lfloor \frac{|U|}{2} \right\rfloor + 1$. All the possible subsets are:

$$\binom{|U|}{\left\lfloor \frac{|U|}{2} \right\rfloor + 1} \cong 2^{|U|} > f(k)$$

$\forall i, j$ with $i \neq j$ we also have that

1. $|\Sigma_i \cap \Sigma_j| \geq 1$, at least one item is in common
2. $|\Sigma_i \setminus \Sigma_j| \geq 1$, at least one item is not in common

Choose two streams indistinguishable for A

Given the above subsets, we have that $\exists S_i \in \Sigma_i,\ S_j \in \Sigma_j,\ S_i \neq S_j$ candidate streams that are indistinguishable for the algorithm A. We can build those two streams like the following

$$S_i = s_1^{(i)} s_2^{(i)} s_3^{(i)} \dots s_{\frac{|U|}{2}+1}^{(i)}$$

$$S_j = s_1^{(j)} s_2^{(j)} s_3^{(j)} \dots s_{\frac{|U|}{2}+1}^{(j)}$$

If we pick $x \in \Sigma_i \setminus \Sigma_j$ and $y \in \Sigma_j \setminus \Sigma_i$ (in other words $x \in \Sigma_i,\ x \notin \Sigma_j$ and $y \notin \Sigma_i,\ y \in \Sigma_j$) and we concatenate both to the streams, obtaining $S_i xy$ and $S_j xy$, we will have that $MostFreq(S_i xy) = x$ because x was already present in the stream S_i, and for the same reason we have that $MostFreq(S_j xy) = y$. Then, we can conclude that

$$MostFreq(S_i xy) \neq MostFreq(S_j xy)$$

But, because the streams S_i and S_j are indistinguishable for A, so $S_i xy$ and $S_j xy$ are. Then, we can also conclude that

$$A(S_i xy) = A(S_j xy)$$

So, the algorithm A gives the wrong answer.