6.897 Advanced Data Structures (Spring'05)

Prof. Erik Demaine TA: Mihai Pătraşcu

Problem 6 – Solution

Upper Bound. The idea is that every message divides the size of the integers by $n^{1/t}$. Say Alice must speak next. She breaks her integer into $n^{1/t}$ chunks. Then, she computes a hash function from every chunk to $\lg n$ bits, and outputs the hash codes, taking $O(n^{1/t} \lg n)$ bits. Bob also breaks his integer into $n^{1/t}$ chunks, and applies the same hash function to every chunk. He then compares his own hash codes with what he received. Up to some point, Alice's code for chunk i is equal to Bob's code for chunk i, so with good probability the original integers were also equal in chunk i. Then, hash codes for some chunk differ, meaning that the integers diverged somewhere in that chunk. From now on, the players only need to worry about comparing their integers restricted to that chunk. So the size of the integers was reduced by $n^{1/t}$. Of course, only Bob knows i, because Alice never sees Bob's hash codes. But Bob can include i in his next message, taking an additional $O(\lg n)$ bits, which is a lower order term.

After $t-1$ messages, we are down to $n/(n^{1/t})^{t-1} = n^{1/t}$ bits. In the last message, the player who should speak just output his remaining integer. The other player can now announce the result of the comparison, because he has the relevant part of both inputs. The probability that the announced value is wrong is upper bounded by the sum of the probabilities of an error in each comparison of hash codes. We have $n^{1/t} \cdot (t-1) = o(n)$ hash codes which are compared in total. For any pair, the probability that different integers look equal through the hash function is $\frac{1}{n}$, so the probability of an error is upper-bounded by $o(1) < \frac{1}{3}$. Observe that a more careful analysis shows we only need hash functions to $O(\lg t)$ bits, which improves the upper bound to $O(n^{1/t} \lg t)$.

Lower Bound. The proof is parallel to the upper bound. We must identify a k-fold problem hidden in the greater-than problem. Say Alice speaks next. We break her integer into $x_1, \dots, x_k$ equal-sized chunks (this is conceptual; Alice only sees a big integer). For arbitrary i and y, we make Bob's integer be the concatenation of $(x_1, \dots, x_{i-1}, y, 0, \dots, 0)$. Clearly, comparing these two integers is equivalent to comparing x_i and y. But this is a k-fold problem.

Now assume there is a solution for the *GT* problem on n-bit numbers with error probability δ. Then, this solution must also be solving the problem we created above. In other words, we considered some restricted class of inputs, but if there's a general solution, it also works for the particular inputs that we decided to consider. But the round elimination lemma assures us that if there exists a solution for the k-fold problem we considered, there exists a solution for comparing x_i to y in which there's one fewer message. The error probability increases to $\delta + O(\sqrt{m/k}) = \delta + \frac{1}{9t}$, for some appropriate choice of $k = \Theta(mt^2)$. Thus, we have a solution for integers which are k times smaller, but which uses one fewer message.

Now assume $n \geq 2k^t$. Then, we can apply round elimination t times, and we're left with nontrivial integers (at least two bits), but no messages. Without communication, the problem cannot be solved with error probability better than $\frac{1}{2}$ (if the player to announce the outcome is Alice, the adversary can give her the integer 1, and give Bob either 0 or 2 randomly, so Alice cannot do better than random guessing). But we obtained a protocol with error probability $\leq \frac{1}{3} + \frac{1}{9t} \cdot t = \frac{4}{9}$, which is a contradiciton. This shows that $n < k^t$, which means $(\Theta(mt^2))^t > n$, or $m = \Omega(n^{1/t}/t^2)$.