CS4830: ORAM, Proof of Retrievability

Instructor: Elaine Shi, Weikai Lin

April 19, 2017

1 A Simple Markov Chain Analysis

In class, when we learned about Oblivious RAM, we made a claim about the queue length distribution of a simple queuing system, but without proving it. We will now prove it.

Imagine the following simple queuing system. In every time step, with probability p, an item arrives and enters the queue. With probability $q = 2p$, an item gets serviced and therefore is removed from the queue. This is often referred to as a discrete M/M/1 queue – the two Ms describe the fact that both the arrival process and the job service process are "memoryless", i.e., they do not depend on what happened in the past.

When $q > p$, this queuing system is said to be ergodic and has a steady state. This implies that when we run this queuing system for sufficiently many steps, the queue length has a stationary distribution. One way to analyze the stationary distribution is the follows: we draw a Markov Chain, where state i denotes the event that the queue length is i in some time step T. The arrows denote transition, and the number above/below the arrow denotes the probability of the transition.

For example, if the queue length is i in time T, then with probability $\alpha := p(1-q)$, the queue length becomes $i+1$ in time step $T+1$, and with probability $\beta := q(1-p)$, the queue length becomes $i-1$ in time $T+1$. Note that $\alpha < \beta, \alpha + \beta < 1$.

Suppose we are happy to assume that a stationary distribution exists, we can then derive the stationary distribution in the following manner. Note that stationary distribution means that for sufficiently large T, the queue length distribution in both T and $T+1$ follows the same distribution.

Suppose π_i is the probability (in the stationary distribution) that the queue length is i. I will now write a set of linear equations: For $i \geq 1$:

$$\pi_i := \alpha\pi_{i-1} + (1-\alpha-\beta)\pi_i + \beta\pi_{i+1}.$$

in addition, $\pi_0 = \frac{\beta}{\alpha}\pi_1$.

Now we want to solve this set of linear equations. One way to solve them is to make a guess and check that the guess is true. I will now guess that $\pi_i = \rho\pi_{i-1}$ for any $i \geq 1$ where $\rho := \alpha/\beta$. Plug this guess into the set of linear equations (including the one for π_0), and it is easy to see the guess is true. Solving for π_i using the fact that all the π_is should sum to 1, we have

$$\pi_0 = \frac{\beta - \alpha}{\beta}, \pi_i = \rho^i \pi_0.$$

Given $\rho \leq 1/2$, $\pi_i \leq 2^{-i}\pi_0$.

1.1 Binary-Tree ORAM: Analysis

Claim 1. *(Bucket size and overflow probability). If the bucket size Z is super-logarithmic in N, then over any polynomially many accesses, no bucket overflows except with negligible in N probability.*

- Root and level 1: the bucket will be chosen for eviction with probability 1. They are always empty.
- Now consider a bucket at level 2 of the ORAM tree. On average, one out of every four accesses (think about why), a block will enqueue in the bucket. With probability 1/2, the bucket will be chosen for eviction.
- In general, we can conclude that for any non-leaf level $i > 1$ of the ORAM tree, with each access, one out of every 2^i accesses, a block will enqueue, and with probability 2^{i-1}, the bucket is chosen for eviction.
- For the leaf nodes, we can apply a standard balls-and-bins analysis, that is, if we throw N balls into N bins at random, then by Chernoff bound, we have that for any super-constant function $\alpha(\cdot)$,

$$Pr[\text{max bin load} > \alpha \log N] \leq \exp(-\Omega(N))$$

Note that the Markov chain does not accurately model the queue of each bucket. For example, a bucket A in the 2nd level can store more than 1 block after some accesses (for some randomness). And then, a child B of A can receive more than 1 blocks when A is evicted (for some randomness). This happens with non-zero probability, where B receives more than 1 block in this time step. Thus, that case violates our assumption of the Markov chain, where there is exactly 1 item added with probability p, and 1 item removed with probability q.

2 Binary-Tree ORAM

Q: In the binary-tree ORAM, every time I remap a block, I choose a random path. What if instead of choosing a random path, I instead employ the procedure to choose a path:

- **Repeat:** sample a path at random,
- **Until** the new path isn't the same as the block's current (i.e., old) path.

Is the modified binary-tree ORAM scheme secure?

3 Proof of Retrievability

Q: In the proof of retrievability scheme based on $N, 2N$ codes, suppose we did the following variant. Instead of taking the entire database of N blocks and encoding it, we take each block of the database and encode it using a $k, 2k$ code for some value k. All codewords of the same block are stored adjacent to each other. Is the scheme secure?