An Application: Hashing

In this lecture, we will see a "killer app" of elementary probability in Computer Science:

- Suppose a hash function distributes keys evenly over a table of size n. How many (randomly chosen) keys can we hash before the probability of a collision exceeds (say) $\frac{1}{2}$?

As we shall see, this question can be tackled by an analysis of the balls-and-bins probability space which we have already encountered.

As you may recall, a hash table is a data structure that supports the storage of sets of keys from a (large) universe U (say, the names of all 250m people in the US). The operations supported are ADDing a key to the set, DELETEing a key from the set, and testing MEMBERship of a key in the set. The hash function h maps U to a table T of modest size. To ADD a key x to our set, we evaluate $h(x)$ (i.e., apply the hash function to the key) and store x at the location $h(x)$ in the table T. All keys in our set that are mapped to the same table location are stored in a simple linked list. The operations DELETE and MEMBER are implemented in similar fashion, by evaluating $h(x)$ and searching the linked list at $h(x)$. Note that the efficiency of a hash function depends on having only few collisions — i.e., keys that map to the same location. This is because the search time for DELETE and MEMBER operations is proportional to the length of the corresponding linked list.

The question we are interested in here is the following: suppose our hash table T has size n, and that our hash function h distributes U evenly over T.[1] Assume that the keys we want to store are chosen uniformly at random and independently from the universe U. What is the largest number, m, of keys we can store before the probability of a collision reaches $\frac{1}{2}$?

Let's begin by seeing how this problem can be put into the balls and bins framework. The balls will be the m keys to be stored, and the bins will be the n locations in the hash table T. Since the keys are chosen uniformly and independently from U, and since the hash function distributes keys evenly over the table, we can see each key (ball) as choosing a hash table location (bin) uniformly and independently from T. Thus the probability space corresponding to this hashing experiment is exactly the same as the balls and bins space.

We are interested in the event A that there is no collision, or equivalently, that all m balls land in different bins. Clearly $\Pr[A]$ will decrease as m increases (with n fixed). Our goal is to find the largest value of m such that $\Pr[A]$ remains above $\frac{1}{2}$. [Note: Really we are looking at different sample spaces here, one for each value of m. So it would be more correct to write $\Pr_m$ rather than just Pr, to make clear which sample space we are talking about. However, we will omit this detail.]

Let's fix the value of m and try to compute $\Pr[A]$. Since our probability space is uniform (each outcome has probability $\frac{1}{n^m}$), it's enough just to count the number of outcomes in A. In how many ways can we arrange m balls in n bins so that no bin contains more than one ball? Well, this is just the number of ways of choosing

[1] I.e., $|U| = \alpha n$ (the size of U is an integer multiple α of the size of T), and for each $y \in T$, the number of keys $x \in U$ for which $h(x) = y$ is exactly α.

m things out of n *without* replacement, which as we saw in Note 10 is

$$n \times (n-1) \times (n-2) \times \cdots \times (n-m+2) \times (n-m+1).$$

This formula is valid as long as $m \le n$: if $m > n$ then clearly the answer is zero. From now on, we'll assume that $m \le n$.

Now we can calculate the probability of no collision:

$$\begin{aligned}
\Pr[A] &= \frac{n(n-1)(n-2)\dots(n-m+1)}{n^m} \\
&= \frac{n}{n} \times \frac{n-1}{n} \times \frac{n-2}{n} \times \cdots \times \frac{n-m+1}{n} \\
&= \left(1-\frac{1}{n}\right) \times \left(1-\frac{2}{n}\right) \times \cdots \times \left(1-\frac{m-1}{n}\right). \qquad (1)
\end{aligned}$$

Before going on, let's pause to observe that we could compute $\Pr[A]$ in a different way, as follows. View the probability space as a sequence of choices, one for each ball. For $1 \le i \le m$, let A_i be the event that the ith ball lands in a different bin from balls $1, 2, \dots, i-1$. Then

$$\begin{aligned}
\Pr[A] = \Pr[\textstyle\bigcap_{i=1}^n A_i] &= \Pr[A_1] \times \Pr[A_2|A_1] \times \Pr[A_3|A_1 \cap A_2] \times \cdots \times \Pr[A_m|\textstyle\bigcap_{i=1}^{m-1} A_i] \\
&= 1 \times \frac{n-1}{n} \times \frac{n-2}{n} \times \cdots \times \frac{n-m+1}{n} \\
&= \left(1-\frac{1}{n}\right) \times \left(1-\frac{2}{n}\right) \times \cdots \times \left(1-\frac{m-1}{n}\right).
\end{aligned}$$

Fortunately, we get the same answer as before! [You should make sure you see how we obtained the conditional probabilities in the second line above. For example, $\Pr[A_3|A_1 \cap A_2]$ is the probability that the third ball lands in a different bin from the first two balls, *given that* those two balls also landed in different bins. This means that the third ball has $n-2$ possible bin choices out of a total of n.]

Essentially, we are now done with our problem: equation (1) gives an exact formula for the probability of no collision when m keys are hashed. All we need to do now is plug values $m = 1, 2, 3, \dots$ into (1) until we find that $\Pr[A]$ drops below $\frac{1}{2}$. The corresponding value of m (minus 1) is what we want.

But this is not really satisfactory: it would be much more useful to have a formula that gives the "critical" value of m directly, rather than having to compute $\Pr[A]$ for $m = 1, 2, 3, \dots$. Note that we would have to do this computation separately for each different value of n we are interested in: i.e., whenever we change the size of our hash table.

So what remains is to "turn equation (1) around", so that it tells us the value of m at which $\Pr[A]$ drops below $\frac{1}{2}$. To do this, let's take logs: this is a good thing to do because it turns the product into a sum, which is easier to handle. We get

$$\ln(\Pr[A]) = \ln\left(1-\frac{1}{n}\right) + \ln\left(1-\frac{2}{n}\right) + \cdots + \ln\left(1-\frac{m-1}{n}\right), \qquad (2)$$

where "ln" denotes natural (base e) logarithm. Now we can make use of a standard approximation for logarithms: namely, if x is small then $\ln(1-x) \approx -x$. This comes from the Taylor series expansion

$$\ln(1-x) = -x - \frac{x^2}{2} - \frac{x^3}{3} - \dots.$$

So by replacing $\ln(1-x)$ by $-x$ we are making an error of at most $(\frac{x^2}{2} + \frac{x^3}{3} + \cdots)$, which is at most $2x^2$ when $x \le \frac{1}{2}$. In other words, we have

$$-x \ge \ln(1-x) \ge -x - 2x^2.$$

And if x is small then the error term $2x^2$ will be much smaller than the main term $-x$. Rather than carry around the error term $2x^2$ everywhere, in what follows we'll just write $\ln(1-x) \approx -x$, secure in the knowledge that we could make this approximation precise if necessary.

Now let's plug this approximation into equation (2):

$$\begin{aligned}
\ln(\Pr[A]) &\approx -\frac{1}{n}-\frac{2}{n}-\frac{3}{n}-\ldots-\frac{m-1}{n} \\
&= -\frac{1}{n}\sum_{i=1}^{m-1} i \\
&= -\frac{m(m-1)}{2n} \\
&\approx -\frac{m^2}{2n}. \qquad (3)
\end{aligned}$$

Note that we've used the approximation for $\ln(1-x)$ with $x = \frac{1}{n}, \frac{2}{n}, \frac{3}{n}, \ldots, \frac{m-1}{n}$. So our approximation should be good provided all these are small, i.e., provided n is fairly big and m is quite a bit smaller than n. Once we're done, we'll see that the approximation is actually pretty good even for modest sizes of n.

Now we can undo the logs in (3) to get our expression for $\Pr[A]$:

$$\Pr[A] \approx \mathrm{e}^{-\frac{m^2}{2n}}.$$

The final step is to figure out for what value of m this probability becomes $\frac{1}{2}$. So we want the largest m such that $\mathrm{e}^{-\frac{m^2}{2n}} \geq \frac{1}{2}$. This means we must have

$$-\frac{m^2}{2n} \geq \ln(\tfrac{1}{2}) = -\ln 2, \qquad (4)$$

or equivalently

$$m \leq \sqrt{(2\ln 2)n} \approx 1.177\sqrt{n}. \qquad (5)$$

So the bottom line is that we can hash approximately $\underline{m = \lfloor 1.177\sqrt{n} \rfloor \text{ keys}}$ before the probability of a collision reaches $\frac{1}{2}$.

Recall that our calculation was only approximate; so we should go back and get a feel for how much error we made. We can do this by using equation (1) to compute the exact value $m = m_0$ at which $\Pr[A]$ drops below $\frac{1}{2}$, for a few sample values of n. Then we can compare these values with our estimate $m = 1.177\sqrt{n}$.

n	10	20	50	100	200	365	500	1000	10^4	10^5	10^6
$1.177\sqrt{n}$	3.7	5.3	8.3	11.8	16.6	22.5	26.3	37.3	118	372	1177
exact m_0	4	5	8	12	16	22	26	37	118	372	1177

From the table, we see that our approximation is very good even for small values of n. When n is large, the error in the approximation becomes negligible.

Why $\frac{1}{2}$?

Our hashing question asked when the probability of a collision rises to $\frac{1}{2}$. Is there anything special about $\frac{1}{2}$? Not at all. What we did was to (approximately) compute $\Pr[A]$ (the probability of no collision) as a function of m, and then find the largest value of m for which our estimate is smaller than $\frac{1}{2}$. If instead we were interested in keeping the collision probability below (say) 0.05 (= 5%), we would just replace $\frac{1}{2}$ by 0.95 in equation (4). If you work through the last piece of algebra again, you'll see that this gives us the critical value $m = \sqrt{(2\ln(20/19))n} \approx 0.32\sqrt{n}$, which of course is a bit smaller than before because our collision probability is now smaller. But no matter what "confidence" probability we specify, our critical value of m will always be $c\sqrt{n}$ for some constant c (which depends on the confidence).

The birthday paradox revisited

Recall from a previous lecture the birthday "paradox": what is the probability that, in a group of m people, no two people have the same birthday? The problem we have solved above is essentially just a generalization of the birthday problem: the bins are the birthdays and the balls are the people, and we want the probability that there is no collision. The above table at $n = 365$ tells us for what value of m this probability drops below $\frac{1}{2}$: namely, 23.

Using the union bound

Here's a cruder way to do the same calculation we did earlier. There are exactly $k = \binom{m}{2} = \frac{m(m-1)}{2}$ possible pairs among our m keys. Imagine these are numbered from 1 to $\binom{m}{2}$ (it doesn't matter how). Let A_i denote the event that pair i has a collision (i.e., both are hashed to the same location). Then the event $\overline{A}$ that *some* collision occurs can be written $\overline{A} = \bigcup_{i=1}^{k} A_i$. What is $\Pr[A_i]$? We claim it is just $\frac{1}{n}$, for every i. (Why?) So, using the union bound from the last lecture, we have

$$\Pr[\overline{A}] \le \sum_{i=1}^{k} \Pr[A_i] = k \times \frac{1}{n} = \frac{m(m-1)}{2n} \approx \frac{m^2}{2n}.$$

This means that the probability of having a collision is less than $\frac{1}{2}$ provided $\frac{m^2}{2n} \le \frac{1}{2}$, i.e., provided $m \le \sqrt{n}$. This is a somewhat more restrictive condition than the one in equation (5) that we derived earlier, and it gives answers that are less accurate than those in our earlier table. However, in terms of the dependence on n both conditions are the same (both are of the form $m = O(\sqrt{n})$).