Algorithm Design

M. T. Goodrich and R. Tamassia

John Wiley & Sons

Solution of Exercise C-1.25

Let B_{n-1}, B_{n-2},..., B_1, B_0 be the n bottles of wine. Consider any binary string $\mathbf{b} = b_{n-1}b_{n-2}...b_1b_0$ of length n. A string $\mathbf{b}$ corresponds to a test as follows. If $\mathbf{b}$ has k 1's, we give a taste tester to drink a sample that consists of exactly k drops: a drop of bottle B_i is included to the sample only if $b_i = 1$. The idea is to have that sufficient collection of tests $\mathbf{b_1}$, $\mathbf{b_2}$,..., $\mathbf{b_T}$ running in parallel, so that, when (after a month) their outcome is known, we can deterministically identify the poisoned bottle.

There are several ways for someone to construct the tests. A naive solution uses n tests, each of them having a drop from only one distinct bottle. We can do much better. An efficient construction of the tests will be in such a way so that a binary search is performed in the sequence of the bottles.

For simplicity assume that n is a power of 2, i.e., $n = 2^k$ for some integer k. We define $\mathbf{B}(t)$ to be the binary string of length t that consists of $\frac{t}{2}$ consecutive 0's and followed by $\frac{t}{2}$ 1's. Let the first test be $\mathbf{b_1} = \mathbf{B}(n)$. Given the output of this test, our search space is reduced by half: if $\mathbf{b_1}$ is positive then we know that the poisoned bottle is one of $B_{\frac{n}{2}-1}$,...,B_1,B_0, otherwise one of B_{n-1},B_{n-2},...,$B_{\frac{n}{2}}$. Let now the second test be $\mathbf{b_2} = \mathbf{B}(\frac{n}{2})||\mathbf{B}(\frac{n}{2})$, where $||$ denotes string concatenation. Clearly tests $\mathbf{b_1}$ and $\mathbf{b_2}$ reduce the search space to one forth of the initial. We proceed in the same way: $\mathbf{b_3} = \mathbf{B}(\frac{n}{4})||\mathbf{B}(\frac{n}{4})||\mathbf{B}(\frac{n}{4})||\mathbf{B}(\frac{n}{4})$ and, generally, $\mathbf{b_i} = \mathbf{B}(\frac{n}{i+1})||\mathbf{B}(\frac{n}{i+1})||\mathbf{B}(\frac{n}{i+1})||\mathbf{B}(\frac{n}{i+1})$, for $1 \leq i \leq k$, where $k = \log n$. Combining all k tests $\mathbf{b_1}$, $\mathbf{b_2}$, ..., $\mathbf{b_k}$ the search space is finally reduced to only one bottle. A similar reasoning can be followed in the case that n is not an exact power of 2; in that case, the number of tests are $\lceil n \rceil$.

We give an example for $n = 13$. The tests could be:

test 1: $\mathbf{b_1}$=11111100 0 0000

test 2: $\mathbf{b_2}$=11100011 1 0000

test 3: $\mathbf{b_3}$=10010010 0 1100

test 4: $\mathbf{b_4}$=11011011 0 1010, that is, we use 4 taste testers. Suppose that only test 2 is positive. Given the structure of the tests, we can infer that the 5th bottle B_4 is the poisoned one.

When n is not a power of 2, there are generally more that one possible test structures. They all can determine the poisoned bottle, but the king should choose that collection of tests that have the least number of 1's (why?)...

Why is the king not smiling? No, it's not because the rumour about the queen - everybody believes she organized this!... The king is unhappy because all the bottles of wine (some of them were really old) had to be opened.