

100 PRISONERS AND A LIGHT BULB

WILLIAM WU

ABSTRACT. We present a variety of distributed protocols known to date for solving the 100 Prisoners and a Light Bulb riddle. Computations of expected runtimes are included. (This is an updated and corrected version of the original unpublished document released on December 5, 2002.)

1. THE RIDDLE

One hundred prisoners have been newly ushered into prison. The warden tells them that starting tomorrow, each of them will be placed in an isolated cell, unable to communicate amongst each other. Each day, the warden will choose one of the prisoners uniformly at random with replacement, and place him in a central interrogation room containing only a light bulb with an on/off switch. The prisoner will be able to observe the current state of the light bulb. If he wishes, he can toggle the light bulb. He also has the option of announcing that he believes all prisoners have visited the interrogation room at some point in time. If this announcement is true, then all prisoners are set free, but if it is false, all prisoners are executed.

The warden leaves, and the prisoners huddle together to discuss their fate. Can they agree on a protocol that will guarantee their freedom?

It may seem surprising at first that such a protocol could exist. We will present an assortment of such protocols that guarantee freedom, and analyze and compare the average number of days that each protocol requires. To compare the protocols fairly, we will express their average runtimes in terms of N , where n is the number of prisoners in general.

2. I'M FEELING LUCKY PROTOCOL

The days are split into n -day blocks. During each n -day block, each prisoner operates according to the following instructions upon entering the interrogation room:

- If it is day 1 for the current block:
 - If the bulb is OFF, turn the bulb ON.
 - If the bulb is already ON, and the first n -day block has already elapsed, announce that all prisoners have visited.
- On any other day of the current block:
 - If it is your first time visiting the room during the current block, do nothing.

- If it is your second time visiting the room during the current block, turn the light OFF.
- If it is your third or more time visiting the room during the current block, do nothing.

The general idea behind this strategy is that eventually, with probability 1, we will be lucky enough to have a block of n -days during which no prisoner enters the room twice, or in other words, during which every prisoner will enter the room exactly once. Then the bulb which was turned ON on day 1 will still be ON after n -days, since bulbs are only turned OFF upon a second return visit. Thus, if the bulb remains ON on the first day of a new block, we know that every prisoner must have visited the interrogation room during the block that had just elapsed.

We now compute the expected runtime of this protocol. Let X be the number of days the protocol requires. Let B be the number of n -day blocks required till the protocol succeeds. Then B is a geometric random variable with parameter

$$\frac{n}{n} \cdot \frac{n-1}{n} \cdot \frac{n-2}{n} \cdot \frac{n-3}{n} \cdots \frac{1}{n} = \frac{n!}{n^n}.$$

Since the expectation of a geometric random variable is the reciprocal of its parameter, and $X = nB$, the expected number of days required is

$$\mathbf{E}[X] = n\mathbf{E}[B] = n \frac{n^n}{n!} = \frac{n^{n+1}}{n!}.$$

Using Stirling's approximation $n! \sim \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$, and big-Oh notation,

$$\mathbf{E}[X] \sim \frac{1}{\sqrt{2\pi}} n^{1/2} e^n = O(n^{1/2} e^n).$$

When $n = 100$, $\mathbf{E}[X]$ equals 1.072×10^44 days. We can expect the prisoners to be long dead by then, and thus, it would behoove us to design a faster protocol.

3. SINGLE COUNTER PROTOCOL

One of the possible sources of difficulty in solving this riddle is the natural idea that every prisoner should follow the same instructions. I will call such a protocol *symmetric*; the previous protocol is an example of such. Realizing that such a constraint does not actually exist, we can design a simple asymmetric scheme that performs much better, allowing the prisoners to escape within a reasonable amount of time on average.

Letting prisoners have different roles, we assign one prisoner to be “the counter”. He will maintain an integer variable in his head that is initialized to 1. Call this variable T . Upon entering the room, prisoners adhere to the following instructions:

- If you are not the counter:
 - If the bulb is OFF, and you have never turned the bulb ON before, turn it ON.
 - If the bulb is ON, do nothing.
- If you are the counter:

- If the bulb is OFF, do nothing.
- If the bulb is ON, turn it OFF, and set $T=T+1$.
- If $T = n$, announce that all prisoners have visited.

The idea behind this protocol is that every prisoner besides the counter will turn ON the bulb exactly once, whenever he can. When the bulb is ON, no one can turn it OFF except for the counter. Eventually the counter will enter the room, turn this bulb OFF, and increment the count T . In this way, each prisoner indicates his presence in the room to the counter by leaving an ON bulb which is eventually recorded by the counter.

To analyze the runtime, we can split the process into epochs. Let X_i denote the number of days between the first day on which $T = i - 1$, and the first day on which $T = i$. Between these two days, two events must occur:

- (1) An unrecorded prisoner must be chosen, causing the bulb to be turned ON. Let Y_i denote the number of days between from when $T = i - 1$ until this event occurs.
- (2) The counter must then enter the room to record this ON bulb. Let Z_i denote the number of days from when the bulb is turned ON until this occurs.

Letting X be a random variable corresponding to the number of days the protocol requires in total, we have the following equations:

$$X_i = Y_i + Z_i,$$

$$X = \sum_{i=1}^{n-1} X_i = \sum_{i=1}^n (Y_i + Z_i).$$

Y_i is a geometric random variable with parameter $\frac{n-i}{n}$, and Z_i is a geometric random variable with parameter $\frac{1}{n}$. Hence, by linearity of expectation, the expected runtime is

$$\begin{aligned} \mathbf{E}[X] &= \sum_{i=1}^{n-1} (\mathbf{E}[Y_i] + \mathbf{E}[Z_i]) \\ &= \sum_{i=1}^{n-1} \left(\frac{n}{n-i} + n \right) \\ &= (n-1)n + \sum_{i=1}^{n-1} \frac{n}{n-i} \\ &= (n-1)n + n \sum_{i=1}^{n-1} \frac{1}{n-i} \\ &= (n-1)n + n \sum_{i=1}^{n-1} \frac{1}{i} \\ &= n((n-1) + H_{n-1}). \end{aligned}$$

In big-Oh,

$$\mathbf{E}[X] \sim n^2 + n \log n = O(n^2).$$

When $n = 100$, $\mathbf{E}[X]$ equals 10417.74 days, or 28.54 years, which is still within the span of a young prisoner’s lifetime.

This is the “standard solution” to the puzzle. In the sequel, we will describe some less well-known solutions that perform even better.

4. TWO STAGE COUNTING PROTOCOL

Observe that in the Single Counter Protocol, we will have long stretches of time where the bulb is ON and we are waiting for the counter to enter the room. This suggests that it maybe useful to have multiple alternative counters who are also authorized to record the ON bulb and turn it OFF. Furthermore, it would be nice if we could count faster to n . That is, rather than counting 1-by-1 to n , what if we counted in jumps of 10 instead?

The Two-Stage Counting Protocol improves on the Single Counter Protocol in both of the aforementioned aspects. Firstly, it divvies up the task of counting the prisoners amongst a group of assistant counters. Secondly, the head counter counts up to n more quickly by collecting the aggregated counts of the assistant counters.

To begin the protocol’s description, there are three different possible roles for a prisoner: head counter, assistant counter, and “drone”. There is exactly one head counter, and there is some number of assistant counters $a \ll n$, while the vast majority prisoners are still drones – regular prisoner with no counting tasks. The head counter and all assistant counters all have an integer variable in their heads, initialized to one.

The protocol has two stages, Stage I and Stage II. Each stage lasts for a certain number of preset days, which we will call s_1 and s_2 , respectively. In Stage I, each assistant counter is responsible for counting a quota of q drones. In Stage II, the head counter will be responsible for counting up the assistant counters who have reached their quota. In this way, the head counter counts toward n in jumps of size q . If the head counter does not succeed by the end of Stage II, then we repeat Stage I and Stage II again, still maintaining all the mental counts from before. In other words, we repeatedly alternate between Stages I and II until victory is declared. Notice that there were five parameters required by the protocol: n, a, q, s_1 , and s_2 .

Getting down to brass tacks, upon entering the interrogation room, each prisoner adheres to the following instructions:

- During the first $s_1 - 1$ days of Stage I:
 - If you are a drone:
 - * If the bulb is OFF, and you have not turned it ON before, turn it ON.
 - * If the bulb is ON, do nothing.
 - If you are an assistant counter:
 - * If the bulb is OFF, do nothing.
 - * If the bulb is ON, and you have not reached your quota yet, turn it OFF, and increment your count.
 - If you are the head counter: always do nothing.

- On the last day of Stage I:
 - If you are a drone:
 - * If the bulb is OFF, do nothing.
 - * If the bulb is ON, turn it OFF, and plan to turn the bulb ON one extra time in future invocations of Stage I.
 - If you are an assistant counter:
 - * If the bulb is OFF, do nothing.
 - * If the bulb is ON, turn it OFF, and increment your count.
 - If you are the head counter:
 - * If the bulb is OFF, do nothing.
 - * If the bulb is ON, turn it OFF, and plan to turn the bulb ON one extra time in future invocations of Stage I.
- During the first $s_2 - 1$ days of Stage II:
 - If you are a drone: always do nothing.
 - If you are an assistant counter:
 - * If the bulb is OFF, and you have reached your quota q and have not turned it ON before, turn it ON.
 - * If the bulb is ON, do nothing.
 - If you are the head counter:
 - * If the bulb is OFF, do nothing.
 - * If the bulb is ON, turn it OFF, and increment your count by q .
 - * If your count equals n , declare victory.
- On the last day of Stage II:
 - If you are a drone:
 - * If light is ON, turn it OFF. Then in future invocations of Stage I, turn the light ON q times.
 - * If light is OFF, do nothing.
 - If you are an assistant counter:
 - * If light is ON, turn it OFF. Then in future invocations of Stage II, turn the light ON once. (This task is in addition to the default task that each assistant counter has. So this assistant counter may have to turn on the light more than once in future Stage IIs.)
 - * If light is OFF, do nothing.
 - If you are the head counter:
 - * If the light is OFF, do nothing.
 - * If the light is ON, add q to count. If the count equals n , declare victory.

The algorithm is a little more complex than one might expect due to the care that must be taken on the last days of Stages. I am very indebted to Hans van Ditmarsch for revealing these special cases to me.

The average runtime of this algorithm is difficult to compute, and remains open for now. However, simulations with certain parameters for the case of $n = 100$ yield runtimes between 3500 and 4000 days, or 9.5 to 11 years.

5. BINARY TOKENS PROTOCOL

The basic idea behind the two stage counting protocol was that in order to speed things up, we should sometimes count in clumps rather than one-by-one. In the first stage, assistant counters were assigned to count one-by-one, and the second stage, the master counter counted the clumps collected by the assistant counters.

The head counter and assistant counters protocol can be thought of in terms of exchanging “tokens” with variable point values. To make the analogy clear, imagine that all prisoners not assigned any counting roles start with a token worth one point. During Stage 1, these prisoners deposit their one-point tokens into the central room by turning on the bulb when they can, and assistant counters collect them. Suppose assistant counters are ordered to count up to 10. Then, in Stage 2, assistant counters can deposit their 10-point tokens into the room by turning on the bulb, and the master counter collects these bigger tokens. Thus, a lighted bulb will represent a different number of points depending on the stage we are in, and we may reach 100 more quickly by counting in terms of higher denomination tokens.

The binary tokens scheme generalizes these ideas. Let n be the total number of prisoners, and for now, suppose n is a power of 2. Beforehand, we will define a sequence (P_k) that dictates the number of points a lighted bulb is worth on day k . Furthermore, every P_k will be some nonnegative power of 2. Then, rather than assigning different roles, all prisoners follow the same instructions:

- Keep an integer in your head; call it T . Initialize it to $T = 1$.
- Let T_m denote the m^{th} bit of T expressed in binary.
- Upon entering the room on day k , where $P_k = 2^m$, go through four steps:
 - (1) If the bulb is ON, set $T := T + P_{k-1}$, and turn it OFF.
 - (2) If $T \geq n$, declare Victory.
 - (3) If $T_m = 1$, turn the bulb ON, and set $T := T - P_k$.
 - (4) Else, if $T_m = 0$, leave the bulb OFF and do nothing.

Notice that Step 1 amounts to taking a token worth P_{k-1} points left over from the previous day, and Step 2 amounts to depositing a token worth P_k points. In short, all prisoners will collect and deposit tokens whenever they may legally do so, where the value of tokens are universally dictated by a prespecified sequence P_k that is *only* a function of what day it is. Whenever someone accumulates 100 points worth of tokens, the game is over.

It remains to specify what the point sequence (P_k) should be. The sequence should start with a block of consecutive ones, since everyone starts with only one point. If this block is long enough, there will be many prisoners who have collect more than one point, and perhaps a subsequent block of twos would be effective. Then, for reasons that will become apparent in the runtime analysis of the following section, we will choose the nondecreasing sequence

$$(P_k) = \left(\underbrace{1, 1, \dots, 1}_{n \ln n + n \ln \ln n}, \quad \underbrace{2, 2, \dots, 2}_{n \ln n + n \ln \ln n}, \quad \underbrace{4, 4, \dots, 4}_{n \ln n + n \ln \ln n}, \quad \dots, \quad \underbrace{\frac{n}{2}, \frac{n}{2}, \dots, \frac{n}{2}}_{n \ln n + n \ln \ln n} \right).$$

Notice that (P_k) consists of $\log_2 n$ blocks each of size $n \ln n + n \ln \ln n$, where the terms in the k^{th} block are set to 2^k , where k indexes from 0 to $(\log_2 n) - 1$.

Lastly, if we do not succeed by the time this sequence of length $(\log_2 n)(n \ln n + n \ln \ln n)$ expires, the prisoners still maintain the integers in their heads, and the $V(n)$ sequence restarts on itself. That is, the sequence $V(n)$ is periodic. So we can think of the protocol as going through cycles, where each cycle has $\log_2 n$ stages.

6. BINARY TOKENS RUNTIME ANALYSIS

The goal of this section is to prove that the average runtime of the binary tokens protocol is

$$O(n(\ln n)^2).$$

To outline our approach, we will first show that the binary tokens protocol can be reduced to a succession of coupon collector problems. (Recall the coupon collector problem: suppose there are n different possible coupons, and each day we receive one of them uniformly at random. The objective is then to collect all n coupons.) After having done so, we will modify the proof of the following well-known result to suit our needs (see Appendix A):

Lemma 1. *In a coupon collection problem with n coupons, after $n \ln n + cn$ draws, the probability of not having seeing all the coupons is less than $\frac{1}{e^c}$.*

Working through a simple example will illustrate why our problem is related to coupon collection. Suppose we have $n = 4$ prisoners labeled A, B, C, and D. Stage 0, in which the bulb is always worth 1 point, then lasts for $\lceil n \ln n + n \ln \ln n \rceil$ days. In the beginning, every prisoner starts with one point, and the bulb is OFF. We can represent this initial state by the table

Day 0:	<i>OFF</i>	2^1	2^0
	<i>A</i>	0	1
	<i>B</i>	0	1
	<i>C</i>	0	1
	<i>D</i>	0	1

where the bulb's status is indicated in the upper left, and the integers being mentally maintained by each of the prisoners is listed in binary in the lower right. Let us play out the following sequence of visitations in Stage 0: A, B, C, B, A, \dots, D .

On Day 1, A is chosen. Following the protocol, A will turn the bulb ON and decrement his number. The new state becomes:

End of Day 1:	<i>ON</i>	2^1	2^0
	<i>A</i>	0	0
	<i>B</i>	0	1
	<i>C</i>	0	1
	<i>D</i>	0	1

On Day 2, B is chosen. He sees the ON bulb, turns it off, and increments his count. He then checks if the zeroth bit of his newly incremented count is a 1, but it is not, so he does

not activate the bulb. The new state is:

End of Day 2:	<table style="border-collapse: collapse; text-align: center;"> <tr> <th style="border-right: 1px solid black; padding: 2px 5px;"><i>OFF</i></th> <th style="padding: 2px 5px;">2^1</th> <th style="padding: 2px 5px;">2^0</th> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>A</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>B</i></td> <td style="padding: 2px 5px;">1</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>C</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">1</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>D</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">1</td> </tr> </table>	<i>OFF</i>	2^1	2^0	<i>A</i>	0	0	<i>B</i>	1	0	<i>C</i>	0	1	<i>D</i>	0	1
<i>OFF</i>	2^1	2^0														
<i>A</i>	0	0														
<i>B</i>	1	0														
<i>C</i>	0	1														
<i>D</i>	0	1														

On Day 3, *C* is chosen. This leads to:

End of Day 3:	<table style="border-collapse: collapse; text-align: center;"> <tr> <th style="border-right: 1px solid black; padding: 2px 5px;"><i>ON</i></th> <th style="padding: 2px 5px;">2^1</th> <th style="padding: 2px 5px;">2^0</th> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>A</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>B</i></td> <td style="padding: 2px 5px;">1</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>C</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>D</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">1</td> </tr> </table>	<i>ON</i>	2^1	2^0	<i>A</i>	0	0	<i>B</i>	1	0	<i>C</i>	0	0	<i>D</i>	0	1
<i>ON</i>	2^1	2^0														
<i>A</i>	0	0														
<i>B</i>	1	0														
<i>C</i>	0	0														
<i>D</i>	0	1														

On Day 4, suppose that *B* is chosen again. *B* sees the bulb, still worth 1 point, and turns it OFF. He then increments his count to $2 + 1 = 3$, which is 11_2 in binary. Then he sees that the zeroth bit of his count so far is a 1, so he decrements his count back to 2, and turns the bulb ON again. So within Day 4, we have

Start of Day 4:	<table style="border-collapse: collapse; text-align: center;"> <tr> <th style="border-right: 1px solid black; padding: 2px 5px;"><i>OFF</i></th> <th style="padding: 2px 5px;">2^1</th> <th style="padding: 2px 5px;">2^0</th> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>A</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>B</i></td> <td style="padding: 2px 5px;">1</td> <td style="padding: 2px 5px;">1</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>C</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>D</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">1</td> </tr> </table>	<i>OFF</i>	2^1	2^0	<i>A</i>	0	0	<i>B</i>	1	1	<i>C</i>	0	0	<i>D</i>	0	1	→	End of Day 4:	<table style="border-collapse: collapse; text-align: center;"> <tr> <th style="border-right: 1px solid black; padding: 2px 5px;"><i>ON</i></th> <th style="padding: 2px 5px;">2^1</th> <th style="padding: 2px 5px;">2^0</th> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>A</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>B</i></td> <td style="padding: 2px 5px;">1</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>C</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>D</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">1</td> </tr> </table>	<i>ON</i>	2^1	2^0	<i>A</i>	0	0	<i>B</i>	1	0	<i>C</i>	0	0	<i>D</i>	0	1
<i>OFF</i>	2^1	2^0																																
<i>A</i>	0	0																																
<i>B</i>	1	1																																
<i>C</i>	0	0																																
<i>D</i>	0	1																																
<i>ON</i>	2^1	2^0																																
<i>A</i>	0	0																																
<i>B</i>	1	0																																
<i>C</i>	0	0																																
<i>D</i>	0	1																																

which is the same state as the previous day. In short, choosing *B* again has no effect on the system.

Now suppose that on Day 5, *A* (or equivalently, *C*) is chosen. The consequent behavior will again be identical to that of *B* on Day 4. Any prisoner with a zeroed count will simply add and immediately subtract out whatever the bulb is worth on that day to his count, resulting in no net state change. Thus, any prisoner whose count reaches zero can be thought of as being *inactive* for the rest of this stage.

Hence, we see that in the remaining days of Stage 0, no net state change will occur unless *D*, the only person unchosen so far, is chosen, which would lead to the last state in Stage 0:

	<table style="border-collapse: collapse; text-align: center;"> <tr> <th style="border-right: 1px solid black; padding: 2px 5px;"><i>ON</i></th> <th style="padding: 2px 5px;">2^1</th> <th style="padding: 2px 5px;">2^0</th> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>A</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>B</i></td> <td style="padding: 2px 5px;">1</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>C</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>D</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">1</td> </tr> </table>	<i>ON</i>	2^1	2^0	<i>A</i>	0	0	<i>B</i>	1	0	<i>C</i>	0	0	<i>D</i>	0	1	→		<table style="border-collapse: collapse; text-align: center;"> <tr> <th style="border-right: 1px solid black; padding: 2px 5px;"><i>OFF</i></th> <th style="padding: 2px 5px;">2^1</th> <th style="padding: 2px 5px;">2^0</th> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>A</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>B</i></td> <td style="padding: 2px 5px;">1</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>C</i></td> <td style="padding: 2px 5px;">0</td> <td style="padding: 2px 5px;">0</td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px 5px;"><i>D</i></td> <td style="padding: 2px 5px;">1</td> <td style="padding: 2px 5px;">0</td> </tr> </table>	<i>OFF</i>	2^1	2^0	<i>A</i>	0	0	<i>B</i>	1	0	<i>C</i>	0	0	<i>D</i>	1	0
<i>ON</i>	2^1	2^0																																
<i>A</i>	0	0																																
<i>B</i>	1	0																																
<i>C</i>	0	0																																
<i>D</i>	0	1																																
<i>OFF</i>	2^1	2^0																																
<i>A</i>	0	0																																
<i>B</i>	1	0																																
<i>C</i>	0	0																																
<i>D</i>	1	0																																

Notice all ones have been paired into groups of two. Stage 1 then proceeds much like Stage 0 did, except that now we increment/decrement starting with the left column of bits, and the number of active prisoners has been halved from four to two. It is easy to see where this binary pattern is going; at the start of Stage k , we should have combined all the 2^{k-1} tokens into 2^k tokens, and there should be only $\frac{N}{2^k}$ active prisoners left.

When does the protocol fail? Notice that if D is never chosen in Stage 0, he will never have another chance to deposit his 1-point token into the room since the value of the bulb only goes up in future stages. Thus, the only way the protocol could succeed in this cycle (going through all stages once) is if D is the victory-declaring prisoner which collects all n points in the end. In general though, if there are ever even just two prisoners who are not chosen in a stage, this entire cycle is destined to fail. So, we can draw the following conclusion:

Up to a negligible fencepost error, the binary tokens protocol succeeds if and only if in each stage, every active prisoner is chosen at least once, where the number of active prisoners in Stage k is $\frac{n}{2^k}$.

Thus each stage reduces to a coupon collection problem. In the k^{th} stage, we collect $\frac{n}{2^k}$ coupons, and we have $n \ln n + n \ln \ln n$ days to do it. Mimicking the proof of Lemma 1, if $\mathbf{P} [F_j^{(k)}]$ is the probability of failing to collect the j^{th} coupon at the k^{th} stage, where $j \in \{1, \dots, \frac{n}{2^k}\}$, then

$$\begin{aligned} \mathbf{P} [F_j^{(k)}] &= \left(1 - \frac{1}{\frac{n}{2^k}}\right)^{n \ln n + n \ln \ln n} \\ &= \left(e^{-2^k}\right)^{\ln n + \ln \ln n} \quad \text{as } n \rightarrow \infty \\ &= \left(e^{\ln n + \ln \ln n}\right)^{-2^k} \\ &= (n \ln n)^{-2^k} \\ &\leq \frac{1}{n \ln n}. \end{aligned}$$

Then, by invoking the union bound, $\mathbf{P} [F^{(k)}]$, the probability of the k^{th} stage failing, is

$$\mathbf{P} [F^{(k)}] = \mathbf{P} \left[\bigcup_{j=1}^{\frac{n}{2^k}} F_j^{(k)} \right] \leq \sum_{j=1}^{\frac{n}{2^k}} \mathbf{P} [F_j^{(k)}] \leq \frac{1}{2^k} \frac{1}{\ln n}.$$

Let F denote the event that one cycle of the protocol fails. Since the protocol fails if and only if at least one of the $\log_2 n$ stages fails, we can again invoke the union bound:

$$\mathbf{P} [F] = \mathbf{P} \left[\bigcup_{k=0}^{(\log_2 n)-1} F^{(k)} \right] \leq \sum_{k=0}^{(\log_2 n)-1} \mathbf{P} [F^{(k)}] \leq \sum_{k=0}^{(\log_2 n)-1} \frac{1}{2^k} \frac{1}{\ln n} \leq \frac{2}{\ln n}.$$

Let S denote the event that the first cycle of the protocol succeeds. Then

$$\mathbf{P} [S] = 1 - \mathbf{P} [F] \geq 1 - \frac{2}{\ln n}.$$

If the first cycle fails, then we can upper bound the probability that the second pass fails by the probability that the first cycle fails. This is true because the likelihood of successfully collecting all coupons at a given stage increases if some of these coupons were already collected in a previous cycle, thereby allowing for more opportunities for the uncollected

coupons to be chosen. Thus, we can upper bound the expected number of cycles for the protocol by

$$\frac{1}{\mathbf{P}[S]} \leq \frac{1}{1 - \frac{2}{\ln n}} \longrightarrow 1 \quad \text{as } n \rightarrow \infty.$$

Hence, since each cycle consists of $\log_2 n$ stages, each of length $n \ln n + n \ln \ln n$, the total expected number of days till the prisoners get out is upper bounded by

$$\left(\frac{1}{1 - \frac{2}{\ln n}} \right) (\log_2 n)(n \ln n + n \ln \ln n) \longrightarrow O(n(\ln n)^2).$$

APPENDIX A. THE COUPON COLLECTOR PROBLEM

Suppose that there are n different possible coupons in the world, and each day we receive one of them uniformly at random in mail. We are then naturally interested in the following questions:

- (1) What is the average number of days till we collect all n distinct coupons?
- (2) Can we give a bound on the probability of collecting all coupons after m days?

To answer these questions, we first decompose the process into epochs. Let X be a random variable representing the total number of days until we see all N coupons. Let X_i be the number of days between first having seen $i - 1$ distinct coupons and first having i coupons. Then X_i is a geometric random variable with parameter $\frac{n-i+1}{n}$, and $X = \sum_{i=1}^n X_i$. The expectation of X is then

$$\mathbf{E}[X] = \sum_{i=1}^n \mathbf{E}[X_i] = \sum_{i=1}^n \frac{n}{n-i+1} = n \sum_{i=1}^n \frac{1}{i} = nH_n \sim n \ln n.$$

One may notice that this analysis is identical to that of the one-counter protocol, which is really a coupon collection problem.

For the second question, we will use the union bound. Suppose A is the event that not all coupons have been seen after m days have elapsed. Let A_i be the event that the i^{th} coupon has not been seen after m days. Then

$$\mathbf{P}[A_i] = \left(1 - \frac{1}{n}\right)^m$$

and applying the union bound,

$$\mathbf{P}[A] = \mathbf{P}[\cup_{i=1}^n A_i] \leq \sum_{i=1}^n \mathbf{P}[A_i] = \sum_{i=1}^n \left(1 - \frac{1}{n}\right)^m.$$

Suppose we set $m = n \ln n + cn$ for some c . Then

$$\begin{aligned} \mathbf{P}[A] &\leq \sum_{i=1}^n \left(1 - \frac{1}{n}\right)^{n(\ln n + c)} \\ &= \sum_{i=1}^n (e^{-1})^{\ln n + c} \quad \text{as } n \rightarrow \infty \\ &= \sum_{i=1}^n \frac{1}{ne^c} = \frac{1}{e^c}. \end{aligned}$$

Thus the probability of failure after $n \ln n + cn$ days is less than $\frac{1}{e^c}$.

APPENDIX B. ORIGINS OF THE RIDDLE

The origins of this riddle are unclear, so I can only discuss my personal experiences. I first heard about it in 2001, through members of the UC Berkeley Chapter of Eta Kappa Nu (HKN), an Electrical Engineering Honor Society. Puzzles are often circulated in HKN because new students seeking to be inducted into the society are required to complete a series of challenges, and often these challenges come in the form of logical or mathematical puzzles. Later, in 2002, I found the problem also listed on one of the challenges webpage of IBM Research [1], wherein it was mentioned that “this puzzle has been making the rounds of Hungarian mathematicians’ parties”.

REFERENCES

1. IBM Research, *100 prisoners and a lightbulb challenge*, http://domino.watson.ibm.com/Comm/wwwr_ponder.nsf/challenges/July2002.html, 2002.

INFORMATION SYSTEMS LAB, STANFORD UNIVERSITY

E-mail address: `willywu@stanford.edu`