The Prisoners' Perfect Plan

Brian Rothstein

July 6, 2020

1 Problem

1.1 Statement

A new warden has been appointed to a maximum security math prison that has 100 prisoners, all with life sentences*. As you know, the warden of a math prison has it within her discretion to release the prisoners if she so chooses. This new warden decides to give the prisoners an opportunity to win their freedom. Her bargain is as follows:

Each day the warden will select one prisoner at random and invite him into a special math interrogation room. The room is pretty drab. Basically it just has a switch that controls an overhead light bulb. The prisoner is allowed to turn the light on or off. The warden never messes with the light switch. The light is initially off.

If during his visit to the room, a prisoner tells the warden that he thinks all of the prisoners have visited the interrogation room and he's right, then he wins the bargain and they can all go free! But if he's wrong, then they've lost their opportunity at freedom†.

The prisoners are given one day to plan their strategy. After that, the daily interviews begin, and the prisoners are no longer allowed to communicate with each other... except by using the light switch in the math interrogation room. Please note that neither the interrogation room nor the light are visible or audible‡ to the prisoners when they're in their cells.

1.2 Questions

Question 1: Can you devise a strategy for the prisoners such that they can (with 100% certainty) figure out when they've all been in the interrogation

*for dividing by 0

†and their pencil sharpener privileges

‡or tasteable, etc... etc...

room?

Question 2: About how long do you think your strategy will take to execute?

Question 3: After how many days would there be a 99.9999% chance that all of the prisoners have visited the interrogation room?

This last question is the aspect of the problem that occurred to me while working on it and that I was the most interested in solving. Are the prisoners being foolish in their quest for a perfect plan?

2 Initial Analysis

What the prisoners want to do is keep track of a count of how many of them have been in the room. If there were a chalkboard in the room, they could each put a tally on the board the first time they went in the room, and the prisoner who put the 100th tally could announce that they'd all been there.

If there were 7 light bulbs, the prisoners could use them to count in binary up to 100 in order to figure out exactly when they had all been in the room. An on/off state is referred to as a *bit* of information. 7 bits can represent 128 different numbers. Why? Since the bits are independent, you have $2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 2^7 = 128$ possibilities.

Unfortunately, the prisoners only have one bit (the light bulb) to communicate. The light bulb itself can only count to 1*, so the actual count will somehow have to reside in one or more of the prisoners' minds.

3 First Solution

3.1 Strategy

What should the prisoners communicate with the light bulb? What will it mean when a prisoner sees it on or off?

A simple idea is that when a prisoner turns the light on, he is communicating that *this is the first time that I've turned the light on.* So if a prisoner enters the room and the light is off and he's never turned it on before, then he turns the light on.

The first prisoner to enter the room takes on the role of the counter. Every time that he enters the room and the light is on, he knows that another prisoner has been able to turn the light on. He'll increment his count by 1 and flip the light

*off = 0, on = 1

off. When his headcount reaches 100*, he can announce with 100% certainty that all of the prisoners have been in the interrogation room.

3.2 Analysis

How long will this strategy take to complete? Obviously it will take a different amount of time depending on the order that the prisoners are selected. There's not even a guarantee that all of the prisoners will ever be picked, though this becomes less likely as time goes on. Even more tragically, there's the possibility that all of the prisoners enter the room, but the counter never realizes it. For instance, maybe the counter is only picked 50 times and then never picked again. Again, though, this becomes more and more unlikely as time goes on.

What we're interested in is the *average* amount of time that the strategy takes. That is, if you were to run the experiment many times, what would be the average number of days that it would take for the counter to figure out that they'd all been in the interrogation room?

Let's start with a simpler question. How many days on average will it take for the counter to visit the interrogation room 100 times? We know that the counter has to enter the room *at least* 100 times for his count to reach 100, since he only increments it by 1 each time he enters the room. This is the minimum amount of time it will take for the counter's internal head count to reach 100 and thus for him to announce that everybody has been in the room.

On any given day, the chance of a prisoner being picked is 1/100. An interesting result in probability theory is that if you run a random trial with probability p of success over and over again until it succeeds, the average time it will take to succeed is $1/p$ trials. In the case of a prisoner being picked, $p = \frac{1}{100}$, so each prisoner will be picked, on average, once every $\frac{1}{\frac{1}{100}} = 100$ days.

The head counter visits the room on the first day. He then has to visit the room at least 99 more times to reach a head count of 100. Since it takes on average 100 days for him to be picked each time, it will take an average of $99 \cdot 100 = 9900$ days for him to visit the room the remaining 99 times. Therefore, the average time it would take this strategy to succeed (assuming the light bulb was always on when the counter visited the room) is 9901 days - or *over 27 years!*

3.3 Proof of Random Trial Time

Here's a proof that the average number of trials it takes before an event with probability p occurs is $\frac{1}{p}$ trials.

*The head counter increments his count to 1 on the first day since he now knows he's been in the room.

The chance that the event will happen on the first trial is p. The chance that the event will happen on the second trial is $(1-p)p$. The chance that it will happen on the third trial is $(1-p)^2p$. In general, the chance that the event will happen on the n^{th} trial is $(1-p)^{n-1}p$. This is because there are $n-1$ failures with chance $1-p$ before finally having a success with chance p.

To calculate the expected number of trials before success, we simply multiply each probability by the number of trials it represents and then add up all of the possibilities. The result is the expected number of trials it would take to achieve success. You can think of the expected number as the average value you'd get if you ran the experiment over and over again and counted how many trials it took to succeed, and then averaged all of those results.

$$E(p) = \sum_{n=1}^{\infty} n(1-p)^{n-1}p \tag{1}$$

Factor out the p

$$E(p) = p\sum_{n=1}^{\infty} n(1-p)^{n-1} \tag{2}$$

Recognize that $n(1-p)^{n-1}$ is the derivative* with respect to p of $-(1-p)^n$

$$E(p) = p\sum_{n=1}^{\infty} \frac{d}{dp}\left(-(1-p)^n\right) \tag{3}$$

Factor out the derivative (using the addition rule of derivatives) and the minus (using the constant multiplication rule of derivatives)

$$E(p) = -p \cdot \frac{d}{dp}\sum_{n=1}^{\infty}(1-p)^n \tag{4}$$

Recognize the infinite geometric series†: $\sum_{n=0}^{\infty} x^n = \frac{1}{1-x}$ (when $|x| < 1$).

In this case $x = 1-p$ and we need to account for the missing $n=0$ term.

$$1 + \sum_{n=1}^{\infty}(1-p)^n = \frac{1}{1-(1-p)} \tag{5}$$

Subtract 1 from both sides

$$\sum_{n=1}^{\infty}(1-p)^n = \frac{1}{1-(1-p)} - 1 \tag{6}$$

*The derivative of a function is another function which represents the slope of the original function at each point. Unfortunately, deriving calculus results is beyond the scope of this document.

†Proof: $(1+x+x^2+x^3+...)(1-x) = 1+x+x^2+x^3+...-x-x^2-x^3-... = 1$

Substitute back into equation (4)

$$E(p) = -p \cdot \frac{d}{dp}\left(\frac{1}{1-(1-p)} - 1\right) \tag{7}$$

Simplify

$$E(p) = -p \cdot \frac{d}{dp}\left(\frac{1}{p} - 1\right) \tag{8}$$

Compute the derivative of $\frac{1}{p} - 1$

$$E(p) = -p \cdot \left(-\frac{1}{p^2}\right) \tag{9}$$

Simplify

$$E(p) = \frac{1}{p} \tag{10}$$

3.4 Further Complications

What if the light is not on when the head counter visits the room? This can happen when no new prisoner* has entered the room since the head counter last entered. This further delays the head counter from figuring out when everybody has been in the room.

What's the expected time for a new prisoner to be picked? This changes over time. If there are n prisoners who haven't flipped on the light, then the chances of picking one of them are $\frac{n}{100}$. Therefore we know† that the expected number of days for a new prisoner to be picked is $E(\frac{n}{100}) = \frac{100}{n}$ when there are n new prisoners left.

Since a new prisoner has to enter before the head counter enters in order for the head counter to increment his count, we see that the total expected time for Strategy 1 is

$$\begin{aligned} S_1 = 1 + E(99/100) + E(1/100) + E(98/100) + E(1/100) + ... \\ ... + E(2/100) + E(1/100) + E(1/100) + E(1/100) \end{aligned} \tag{11}$$

In this equation we see that the head counter visits on day 1. Then we need a different prisoner to be chosen to flip on the light, which takes on average $E(\frac{99}{100})$ days. Then we need the counter to visit, which takes $E(\frac{1}{100})$ days. Then we need another new prisoner who hasn't flipped on the light, which takes $E(\frac{98}{100})$ days. And so on.

*i.e. a prisoner who has never switched on the light

†See above for proof of expected time until success

Using summation notation and rearranging the terms we can write S_1 like this

$$S_1 = 1 + \sum_{n=1}^{99} \left(E(\frac{n}{100}) + E(\frac{1}{100}) \right) \tag{12}$$

Using the formula for expected time

$$S_1 = 1 + \sum_{n=1}^{99} \left(\frac{100}{n} + \frac{100}{1} \right) \tag{13}$$

Pulling the $\frac{100}{1}$ out of the sum

$$S_1 = 1 + 9900 + \sum_{n=1}^{99} \frac{100}{n} \tag{14}$$

Factoring 100 out of the sum and simplifying

$$S_1 = 9901 + 100 \cdot \sum_{n=1}^{99} \frac{1}{n} \tag{15}$$

Now we just have to compute $\sum_{n=1}^{99} \frac{1}{n}$ in order to arrive at the answer. So how do we do it? We need to use a calculus result again.

$$\int_1^{100} \frac{1}{x} dx = \ln(100) - \ln(1) = \ln(100) \tag{16}$$

The above expression is an integral. It's very similar to a sum except instead of summing $\frac{1}{x}$ specifically for $x = 1, 2, 3, ..., 100$, you can think of it as summing $\frac{1}{x}$ for *every* real value of x between 1 and 100 multiplied by the width of the infinitely thin sliver where that value of x resides. Essentially, it's computing the area under the curve represented by $\frac{1}{x}$ for $1 \leq x \leq 100$.

This, of course, isn't the same as summing 99 rectangles with a width of 1 for each rectangle*. In fact, it's a little smaller. You can show with a simple graph that $\ln(100) < \sum_{n=1}^{99} \frac{1}{n}$ and $\ln(100) > \sum_{n=2}^{100} \frac{1}{n}$. The idea is that starting the sum at $n = 1$ overestimates the value of the integral and starting it at $n = 2$ underestimates the value of the integral. With these two inequalities you can demonstrate that $\ln(100) < \sum_{n=1}^{99} \frac{1}{n} < \ln(100) + \frac{99}{100}$.

So we plug $\ln(100)^\dagger$ into a calculator which gives $\ln(100) \approx 4.605$. Therefore

$$4.605 < \sum_{n=1}^{99} \frac{1}{n} < 5.595 \tag{17}$$

*i.e. the sum we want to compute

†ln() is the natural log function

Let's just average the upper and lowerbound to come up with*

$$\sum_{n=1}^{99} \frac{1}{n} \approx 5.1 \tag{18}$$

Plugging this estimate back into eq(15) we get

$$S_1 \approx 9901 + 100 \cdot 5.1 \tag{19}$$

Therefore we see that†

$$\boxed{S_1 \approx 10411} \tag{20}$$

Strategy 1 takes around 10411 days on average to complete or around *28.5 years*!

*I also wrote a program to compute the sum and it gave 5.177, so the simple math answer is not too far off. There are also better mathematical ways to calculate this value. Look into the harmonic numbers for more details.

†$S_1 \approx 10419$ using the more accurate value of 5.177 for the sum

3.5 Computer Simulation

Here's a Python program I wrote to simulate Strategy 1. Running the program simulates the strategy 1000 times.

```
import random

def RunTrials(theMethod,theNumTrials):
    aTotal = 0
    aMin = 100000000
    aMax = 0
    for i in range(theNumTrials):
        aNumDays = theMethod()
        aMin = min(aNumDays,aMin)
        aMax = max(aNumDays,aMax)
        aTotal += aNumDays

    anAvg = aTotal//theNumTrials
    print(f"avg={anAvg} days  (min={aMin}, max={aMax})")


def Strategy1():
    lightOn = False
    flippedSwitch = [False]*101
    day = 0
    count = 0

    while count<100:
        day += 1

        prisonerId = random.randint(1,100)
        if day==1:
            counterId = prisonerId

        if not lightOn and not flippedSwitch[prisonerId]:
            lightOn = True
            flippedSwitch[prisonerId] = True

        if prisonerId==counterId and lightOn:
            lightOn = False
            count += 1

    return day

RunTrials(Strategy1,1000)
```

Running the program 5 times produced the following results:

```
avg=10469 days  (min=7605, max=13850)
avg=10418 days  (min=7712, max=13722)
avg=10429 days  (min=7157, max=14519)
avg=10414 days  (min=7417, max=13918)
avg=10387 days  (min=7664, max=14870)
```

4 Probabilistic Approach

Twenty-eight and a half years seems like an awfully long time for the prisoners to wait on average in order to win their freedom. What if instead of waiting 28.5 years for absolute certainty, they waited some shorter amount of time such that there was a 99.9999% probability that they'd all been in the room by that day. How long would they have to wait before saying they'd probably all been in the room?

Let $A_{i,n}$ = the condition that prisoner i has never been in the room during the first n days of the warden's challenge. The probability of $A_{i,n}$ being true is

$$P(A_{i,n}) = \left(\frac{99}{100}\right)^n \tag{21}$$

This is because there's a $\frac{99}{100}$ chance of not being picked on any given day.

Let B_n = the condition that 1 or more of the prisoners have not been in the room during the first n days of the warden's challenge.

$$B_n = \bigcup_{i=1}^{100} A_{i,n} \tag{22}$$

B_n simply means that $A_{1,n}$ happened or $A_{2,n}$ happened or $A_{3,n}$ etc... That is, it means that prisoner 1 hasn't been in the room after n days, or prisoner 2 hasn't been in the room after n days, or prisoner 3, etc...

Now we wish to calculate the probability of B_n happening. What is $P(B_n)$?

$$P(B_n) = P\left(\bigcup_{i=1}^{100} A_{i,n}\right) \tag{23}$$

Using the inclusion-exclusion principle we find*

$$P(B_n) = \sum_{i=1}^{100} P(A_{i,n}) - \sum_{\forall_{i\neq j}} P(A_{i,n} \cap A_{j,n}) + \sum_{\forall_{i\neq j\neq k}} P(A_{i,n} \cap A_{j,n} \cap A_{k,n}) - ... \tag{24}$$

Basically, you add up the individual probabilities that any of them have not been in the room. This double counts cases, though. For instance A_1, n includes outcomes where prisoner 2 also didn't visit the interrogation room. And A_2, n includes outcomes where prisoner 1 didn't visit the interrogation room. Because of this, you need to subtract these overlapping cases. However, this subtraction is not exactly correct either because it erases cases where groups of 3 prisoners have not been in the room, so those have to be added back. This refinement continues for groups of 4,5,6,...,all the way to 100 prisoners.†

Let's calculate equation (24). We already saw that $P(A_{i,n}) = \left(\frac{99}{100}\right)^n$. What about the chances that two specific prisoners are not picked after n days?

$$P(A_{i,n} \cap A_{j,n}) = \left(\frac{98}{100}\right)^n \tag{25}$$

This is because only 98 of the 100 prisoners must be picked on all n days (avoiding prisoner i and prisoner j). Similarly

$$\begin{aligned} P(A_{i,n} \cap A_{j,n} \cap A_{k,n}) &= \left(\frac{97}{100}\right)^n \\ P(A_{i,n} \cap A_{j,n} \cap A_{k,n} \cap A_{l,n}) &= \left(\frac{96}{100}\right)^n \\ &\text{etc...} \end{aligned} \tag{26}$$

Another important thing we need to know in order to calculate equation (24) is how many values are in the sums such as $\sum_{\forall_{i\neq j}} P(A_{i,n} \cap A_{j,n})$? All of the values inside the sum are the same, so we simply need to know how many of them there are and then we can replace the sum with a simple multiplication by a constant.

We use the *choose function*‡ to figure out how many ways there are to pick n

*See Appendix A in https://ilikerice.org/math/clownlottery.pdf for an explanation and proof of the inclusion-exclusion principle.

†You can derive the full result starting with the simple formula $P(A \cup B) = P(A) + P(B) - P(A \cap B)$.

‡The choose function $\binom{n}{k}$ specifies how many different sets of k elements can made from a bigger set of n elements. $\binom{n}{k} = \frac{n(n-1)(n-2)...(n-k+1)}{k!} = \frac{n!}{k!(n-k)!}$
To see why this is true, consider picking the first element. There are n choices. For the second element there are $n-1$ choices. Keep picking until you've picked k elements. Since you're constructing a set, the order in which you picked the elements doesn't matter. There are $k!$ ways to represent a list of k elements. Therefore divide by $k!$ to remove the duplicates.

prisioners from a set of 100. There are $\binom{100}{2}$ ways to pick 2 prisoners, $\binom{100}{3}$ ways to pick 3 prisoners, etc...

We can now rewrite equation (24) as follows

$$P(B_n) = \binom{100}{1}\left(\frac{99}{100}\right)^n - \binom{100}{2}\left(\frac{98}{100}\right)^n + \binom{100}{3}\left(\frac{97}{100}\right)^n - ... \quad (27)$$

Or using summation notation

$$P(B_n) = \sum_{k=1}^{100}\binom{100}{k}\left(\frac{100-k}{100}\right)^n (-1)^{k-1} \quad (28)$$

Remember that $P(B_n)$ is the probability that one or more of the prisoners haven't been in the interrogation room by day n. So $1 - P(B_n)$ is the probability that all of the prisoners have been in the interogation room by day n.

We're interested in calculating n in the following equation

$$1 - P(B_n) = 0.999999 \quad (29)$$

That is, on what day will there be a 99.9999% chance that all of the prisoners have been in the room? Rearranging terms, we see that we want to solve for n in this equation

$$P(B_n) = 0.000001 \quad (30)$$

One way to calculate this would be to write a computer program to try plugging all values of n into $P(B_n)$ until it arrived at an n that gives the desired value of 0.000001. However, let's see if we can find n simply by estimating, using fewer terms of $P(B_n)$. Starting with 1 term we have

$$100\left(\frac{99}{100}\right)^n = 0.000001 = 10^{-6} \quad (31)$$

Divide both sides by 100

$$\left(\frac{99}{100}\right)^n = 10^{-8} \quad (32)$$

Take the log of both sides and use exponent rule of logs

$$n\log\left(\frac{99}{100}\right) = -8\log(10) = -8 \quad (33)$$

Divide both sides by $\log\left(\frac{99}{100}\right)$

$$n = \frac{-8}{\log(\frac{99}{100})} \quad (34)$$

Use a calculator to compute

$$n \approx 1833 \tag{35}$$

Let's try plugging that value of n into the first two terms of $1 - P(B_n)$.

$$1 - \left(100 \left(\frac{99}{100} \right)^{1833} - \binom{100}{2} \left(\frac{98}{100} \right)^{1833} \right) \approx 0.999999001 \tag{36}$$

Now let's try with 3 terms

$$1 - \left(100 \left(\frac{99}{100} \right)^{1833} - \binom{100}{2} \left(\frac{98}{100} \right)^{1833} + \binom{100}{3} \left(\frac{97}{100} \right)^{1833} \right) \approx 0.999999001 \tag{37}$$

It turns out that each subsequent term of $P(B_n)$ is too small to affect the first 6 digits calculated only using the first term.

So we have our answer. After only 1833 days (or around 5 years), there's a 99.9999% chance that the prisoners have all been in the interrogation room. I think they're better off guessing after 5 years than waiting 28 years to be 100% sure.

5 Other Strategies

Are there other strategies that might improve the average time to complete the prisoners' perfect plan? As it turns out, there are, though nobody to my knowledge has discovered a strategy anywhere close to the 1833 day 99.9999% guessing strategy. Of course, this wasn't the point of the problem, but it did strike me, philosophically, as completely ridiculous to wait for years in search of a perfect plan as opposed to simply taking a miniscule risk in order to win many more years of freedom.

5.1 Improving Strategy 1

The simplest way to improve Strategy 1 is to divide the strategy into 2 phases. The first phase lasts a short time, say 20 days. Phase 1 is dedicated to picking the head counter as well as quickly getting a quick head start on the count.

During phase 1, a prisoner only turns on the light bulb if it's not already on and it's his *second* time in the room. The prisoner that turns on the light becomes the counter and starts with a count equal to the day number minus one since he knows that no other prisoner has been in the room twice except for him.

The prisoner that enters the room on the last day of phase 1 turns off the light. If the light wasn't on, then this means that there were no duplicate prisoners

for all of phase 1. In this case the prisoner in the room on the last day of phase 1 becomes the counter and initializes his count to the day number.

Phase 1 relies on the fact that it's initially unlikely that duplicate prisoners are being chosen. It's much more probable that the first few prisoners are all different. Remember that it takes the head counter an average of 100 days between room visits to count each prisoner. So for each prisoner counted in phase 1, they're saving an average of 100 days from the total time.

As for phase 2, it operates just like the original Strategy 1 except that any prisoner that visited the room during phase 1 when the light was off has already been counted and so doesn't flip the switch.

5.2 Strategy 1B Simulation

Below, you'll find Python code for the modified Strategy 1. It uses the RunTrials() method from the first program. Running the program 5 times produced the following results:

```
avg=9335 days  (min=6308, max=12551)
avg=9352 days  (min=6560, max=13446)
avg=9325 days  (min=6314, max=12682)
avg=9303 days  (min=5877, max=13551)
avg=9343 days  (min=6147, max=13785)
```

So it looks like we saved around 1100 days with this strategy. This indicates that the expected time before getting a repeat prisoner in the room is 11 days. I'll leave that as an exercise for the reader to prove.

```
def Strategy1B():
    lightOn = False
    counted = [False]*101
    day = 0
    count = 0
    phase1Duration = 20

    # Phase 1
    while day<phase1Duration:
        day += 1
        prisonerId = random.randint(1,100)

        if not lightOn:
            if counted[prisonerId]:
                lightOn = True
                counterId = prisonerId
                count = day-1
            else:
                counted[prisonerId] = True

        if day==phase1Duration:
            if not lightOn:
                counterId = prisonerId
                count = day
                counted[prisonerId] = True

            lightOn = False

    # Phase 2
    while count<100:
        day += 1

        prisonerId = random.randint(1,100)
        if not lightOn and not counted[prisonerId]:
            lightOn = True
            counted[prisonerId] = True

        if prisonerId==counterId and lightOn:
            lightOn = False
            count += 1

    return day

RunTrials(Strategy1B,1000)
```

5.3 Strategy 2

You can take the idea of multiple phases even further. The genius behind having phases is that you can make the light bulb mean different things in different phases. The prisoners just have to agree how long each phase is and then they can each count the days so they know which phase they're in.*

Strategy 2 relies on having multiple counters. When there's only one counter, he's waiting on average 100 days between visits to the room. This makes the count paintfully slow! With multiple counters, you can try to leverage the fact that other prisoners are entering the room during these 100 days. Perhaps other counters could maintain their own counts and then *somehow communicate* these counts to the head counter.

The general idea is that you divide the process into two phases. You choose 9 assistant counters and 1 head counter. During phase 1, the assistant counters maintain counts just like the head counter. That is, they flip the switch off and increment their head count when they find the light on. They don't turn the light on in this phase. Also, importantly, the counters don't count past 10 during this phase. That is, if they've already reached their quota, they don't mess with the light switch.

In phase 2, the normal prisoners don't touch the light switch. This phase is for the assistant counters. If an assistant counter has reached an internal count of 10 and hasn't flipped the switch on before, then he turns the light on. When the head counter sees the light on during phase 2, he adds 10 to his count and turns it off. If the prisoners reach the end of phase 2 without finishing, then they go back to phase 1 and repeat.

This is the general idea, though you can play around with various parameters and details.† In my version, each phase lasts 7 years. There are 9 assistant counters and 1 head counter. The head counter also counts during phase 1 but only up to 10 total prisoners during phase 1. Essentially, the head counter acts like an assistant counter during phase 1.

One extermely important part of this strategy is what happens on the last day of a phase if the light is still on. When the light is on at the end of a phase, it means that one of the counters didn't receive this piece of information. In order to prevent the information from being lost, the prisoner that sees the light on at the end of the phase turns if off and then remembers to turn it back on the next time he's in the room during that phase. In this way, the information is

*Of course, the more complicated the strategy gets, the more likely one of the prisoners would mess it up in real life. This is yet another reason I believe that the *wait 5 years and guess* strategy is the best!

†God help you if you want to analyze this strategy mathematically as opposed to using computer simulations.

never lost.

5.4 Strategy 2 Simulation

Here is a link to my Python program for Strategy 2:

`https://ilikerice.org/math/pb_strategy2.py`

Running the program 5 times produced the following results:

```
avg=3814 days  (min=2996, max=8213)
avg=3776 days  (min=2974, max=8204)
avg=3805 days  (min=2947, max=8402)
avg=3827 days  (min=2982, max=8481)
avg=3798 days  (min=2940, max=8082)
```

As you can see, Strategy 2 is a significant improvement on Strategy 1. It takes around 10.5 years as opposed to the 28.5 years of Strategy 1 or 25.5 years of Strategy 1B. But it still is twice as slow as the guessing after 5 years strategy.

And good luck to the prisoners (who are not robots) in trying to implement Strategy 2 perfectly. You'll notice that my Strategy 2 program is much more complicated than Strategy 1 and includes various debug statements that helped me figure out various little details that I was missing while implementing it!

6 Conclusion

What have we learned? I think the lesson here is the old cliche that perfect is the enemy of good. In trying to perfectly figure out when all of the prisoners have been in the interrogation room, they throw away an extra 23 years of freedom. That's if they implement the simple plan. If they try one of the more complicated plans, it's easily possible that they mess something up and *never* realize that they've all been in the room.

Perhaps there still is a really good plan that takes less time. Maybe it's even simple to implement. If so, nobody has found it yet. Maybe you can improve the result. Make sure to check google to see what other people have done before you. Good luck!