Ukrainian Catholic University
Applied Sciences Faculty

Courses: Probability Theory and Mathematical Statistics, Algorithms and Data Structures

“Ad fontes” project

Decision Making using The Golden rule

Authors: ***Halonko Yaroslav***, ***Hevalo Andrian***
Github link: Ad Fontes

Lviv, November 2018

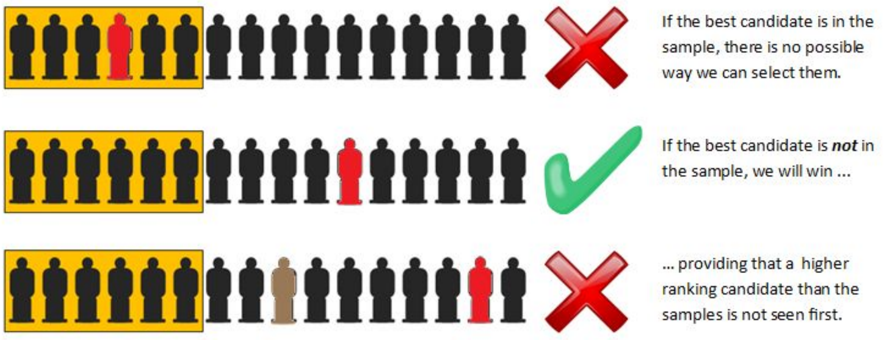
Contents

Problem setting and possible strategies

Have you ever wondered how to choose the best item which are coming, but you can’t return to previous ones?

This is one of the most popular problems in decision making theory. Sometimes when it comes to decision making it is hard to make the best decision, so **Dimitris A. Sardelis and Theodoros M. Valahas** described this problem and solved it mathematically

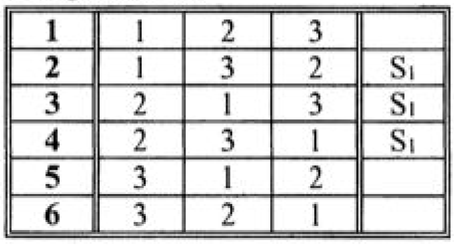
Let us describe the problem:
We have known number N of items is to be presented to an observer one by one in random order, all possible orderings being equally likely. The observer is able at any time to rank without ties the items that have so far been presented in order of desirability. As each item is presented he must accept it, in which case the process stops, or reject, in that case the next item in the sequence is presented and the observer faces the same dilemma as before. If the last item is reached it must be accepted. The observer aim is to find the best of the N items available by employing a strategy with as high a probability of success as possible. This problem is very familiar to the **Secretary Problem**. This problem demonstrates a scenario involving optimal stopping strategy.


If the best candidate is in the sample, there is no possible way we can select them.
If the best candidate is *not* in the sample, we will win ...
... providing that a higher ranking candidate than the samples is not seen first.

The observer must either accept or reject current item that is presented, he can’t go back and choose an already-presented item that turns out to be the best. The problem is to find the best item among N presented. **An observer has to balance the risk of stopping too soon and accepting an apparently desirable item, when even better item than a current one is to be presented in future**. So all possible strategies range between two equally likely extremes that continue the worst choices an observer can make. An observer is able to pick the first item, or, eventually, pick the last one. The probability of picking an item with maximum value is 1 / N which is very small as N is large. But there is some chances for an observer to increase the probability of selecting the best item. **But what is that probability?**

Derivation of the formula for success probability of Sn

TABLE 1. Orderings and Winning Strategies ($N = 3$)



1	1	2	3	
2	1	3	2	S_1
3	2	1	3	S_1
4	2	3	1	S_1
5	3	1	2	
6	3	2	1	

Let us define the following N - 2 non trivial Sn strategies:

The observer lets n items pass, $1 \leq n \leq N-2$, ranks them in order of desirability, and then among the next items selects the first one found with a higher rank.

Among all possible strategies Sn, the observer wants to select and employ one with the maximum probability of success.

We can apply brute-force method of finding the optimal strategy, but for large N we have too many different permutation (when N = 10, we have over 1 million orderings!).

The S_n **is a winning strategy if:**

a) the best item is a candidate for selection, and
b) the ranks of the items, if any, preceding the best, do not exceed those of the first n items

This two conditions give us a general expression for the probability of success of Sn which is **Pn(Sn)**
Ek - the event that the best item is at some position k. Since all ordering are equally likely, then by condition a.

$$P(E_k) = \frac{1}{N}$$

Fk - event describing the condition b. which means that the highest rank of first k - 1 terms appears in the first n terms, so

$$P(F_k) = \frac{n}{k-1}$$

Then Sn is a winning strategy
27
of both conditions are satisfied. By independence and exclusivity, we have:

$$P_n(S_n) = \sum_{k=n+1}^{N} P(E_k \cap F_k) = \sum_{k=n+1}^{N} P(F_k)P(E_k) = \frac{n}{N} * \sum_{k=n+1}^{N} \frac{1}{k-1}$$

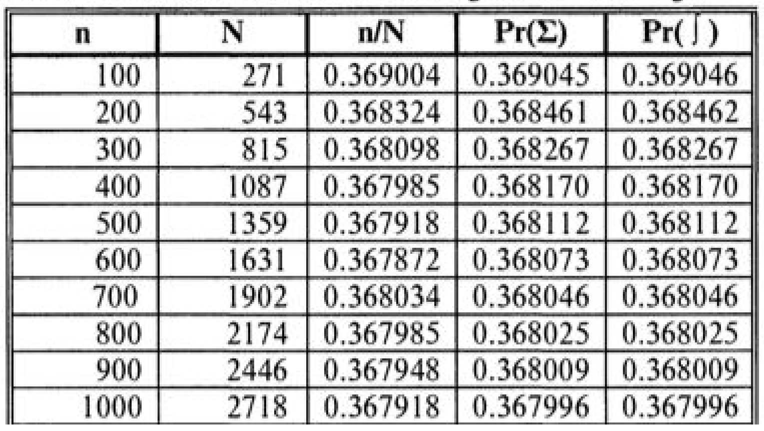
Another formula for Pn(Sn) is:
The Pn(Sn) has an upper bound which is:

$$\frac{n}{N} \ln \frac{2N-1}{2n-1}$$

Asymptotic dependence between optimal n and size N

We have 2 different formulas for Pn(Sn) - sums and integrals. To conduct asymptotic dependence analysis we compute probabilities for large N and compare the results.

TABLE 6. Probabilities of Success for Strategies when N is Large



n	N	n/N	Pr(Σ)	Pr(∫)
100	271	0.369004	0.369045	0.369046
200	543	0.368324	0.368461	0.368462
300	815	0.368098	0.368267	0.368267
400	1087	0.367985	0.368170	0.368170
500	1359	0.367918	0.368112	0.368112
600	1631	0.367872	0.368073	0.368073
700	1902	0.368034	0.368046	0.368046
800	2174	0.367985	0.368025	0.368025
900	2446	0.367948	0.368009	0.368009
1000	2718	0.367918	0.367996	0.367996

We see that as n increases, our probabilities **eventually decrease**. as N goes to infinity, the ratio of n / N converge to a very small number 0.367879 which is exp(-1). So two quantities converge to the same number as N goes to infinity.

We have:

$$\frac{1}{e}(1-\frac{1}{N}) \leq \frac{n}{N} - \frac{1}{2eN}(\frac{1}{e}-1) < \frac{1}{e}$$

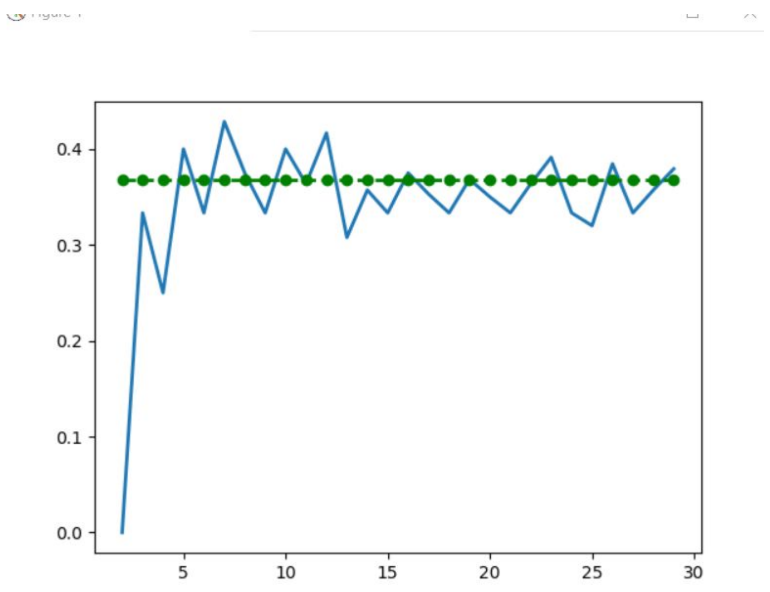
It follows that the limit as N goes to infinity of n / N is 1 / e

So we state the following Golden rule for decisions:

The optimum strategy is to wait until 1 \ e of items pass and then select the next relatively best one. The probability of success for this strategy is the highest.

We used Python and matplotlib library to visualize this problem and to prove that Golden rule makes sense.

This plot shows us that function y = n / N where N is our sample space **converges to y = 1 / e** which is green pointed straight line. Also we see that as n becomes larger the function is getting closer to the 1 / e.



Numerical analysis: calculating(A), simulating(B), comparing(C)

A: Calculating the probabilities of success. Pseudocode:

```
Golden_rule(N):
probabilities = [0] * (N - 1) #Creating list of numbers of all outcomes
if N == 1:
   raise IndexError
else:
   lst_sets = [10000 permutations of given sample]
for n in range(1, N - 1):
   for lst in lst_sets:
       local_max = strategy(n, lst)#Getting the maximum of skipped observed items
       curr_max = max(lst) # Testing whether we took maximum value or not
       if local_max == curr_max:
           probabilities[n] += 1
res = list(map(lambda x: x / factorial(N), probabilities))
a = max(res)
return res.index(a) / N - The probability of this
```

B: For simulating the strategies and taking one with max probability.

Pseudocode:

```
def strategy(n, lst):
   """
   :return: The value of current number, whis is taken by current strategy
   """
   skipped = max(lst[:n])
   for j in lst[n:]:
       if j >= skipped:
           return j
   return lst[-1]
```

C: For comparing frequencies and probabilities we did some visualization: Pseudocode:

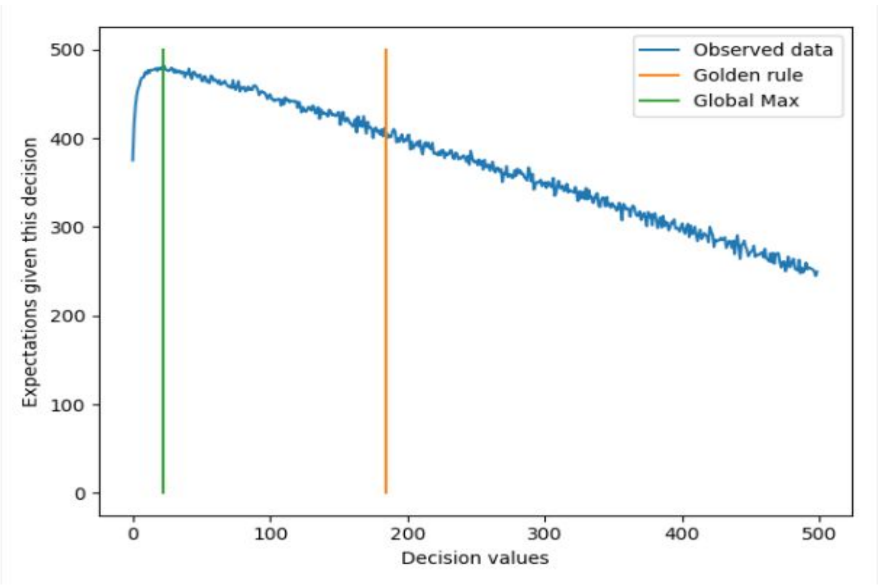
```
def main():
   y = []
   x = []
   for N in range(2, 30):
       y.append(golden_rule(N))
       x.append(N)
   y1 = [exp(-1) for i in range(2, 30)]
   x1 = x
   return x, y, x1, y1

def create_plot():
      x, y, x1, y1 = main()
   plot(x, y, x1, y1, 'go--', linewidth=2)
   show()
```

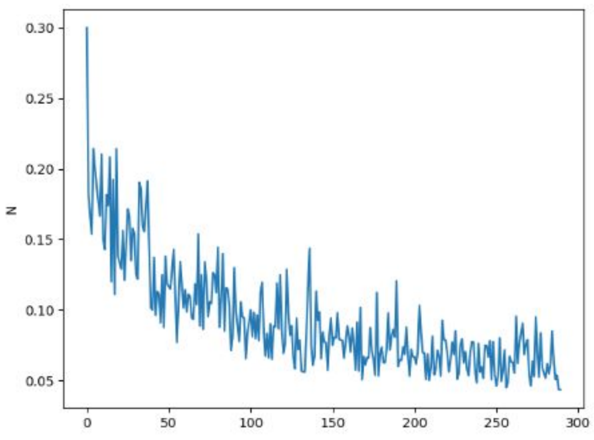
Taking real values: Expectation of success

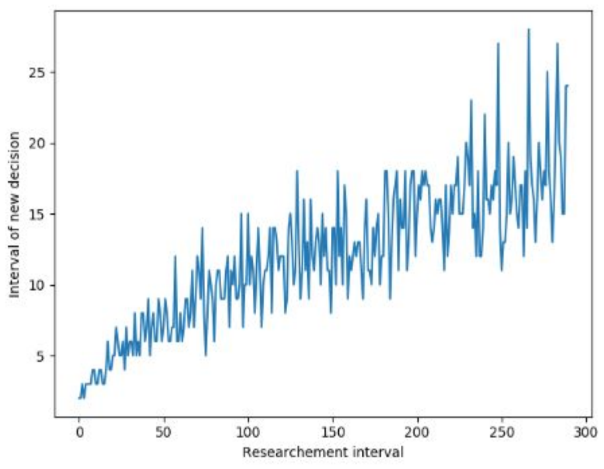
The algorithm, which was given by authors of the article gives you the maximized chance to get top-ranked item from the list. **But does this mean that this decision is best for you?**
Consider the problem of getting maximum expected value from the items we took after n rounds. Item is the element from the list with some value V > 0; round is the collection of elements, and algorithm must take one of them as in **Secretary problem**. We showed that the golden rule of decision making helps us to get maximum value most frequently. Now find the expectation of this algorithm. To do this we need to perform simulations.
Important: top rank is the maximum value!

We simulated once again, but this time we found Rn which is the rank produced by Sn and estimated E(Rn). This plot shows us the **Golden Rule** (wait until ~37% items pass and then select the next better), **Global Max** which is the maximum expected rank and our **Observed data**. It is clear that there is a big difference between **Golden Rule** and **Global Max**, so we can conclude that Golden Rule approach doesn't give accurate results. (code)



*In practise, Golden Rule does give us item with highest rank, meanwhile the best expectation of the item will be in already observed items In other words, if you are choosing a team, after the 37% skipped will **NOT** give you the best squad.*
Now we need to find the point which gives us the maximum possible expected value, the top rank
After performing simulations we determined that this is not so easy. As n increases, percentage decreases. First plot visualizes it. As you can see, the values after Sn increased go to 0. Second represents the accurate numbers which are our decisions. We see that these numbers are increasing as N goes to infinity but asymptotically they are growing slower. The speed of this growth is incomparably slower. This means that we can not consider the number certainly.





Problem when two choices must be made

There are the other problem, which can occur, when we are going to choose next best option. Consider a problem: You are interviewing people, when one goes after the other. Given the condition that in a certain moment you can hire a currently interviewed worker. The difference of this task from the standard Secretary problem is that you need to hire one more job seeker. This little change adds some uncertainty. What to do now?
Firstly, we can consider it as a Secretary problem and do the same thing again: interview the other people, to be more precise, ~37% of them and then calculate the decision. But is it optimal solution? Secondly, the above text says, that you need to consider the purposes. If you are trying to get the interviewer with the best value, the golden rule is the needed one. Otherwize, if you want to maximize the expected value, you need observe approximately 15% of the interviewers. But what to do with the second? Why waiting till 37% of the interviewers is not optimal? **Because we already have the data!** Due to this fact we do not need to skip more.

Ad Fontes Conclusions

Golden Rule is also good for decision making in Business and Economics. It helps to make such decisions that will minimize costs and maximize profits.

Golden Rule is a powerful tool of decision making which gives us the possibility of choosing the highest-ranked item. But sometimes in real life Golden Rule is not suitable for picking the best item, meanwhile best has already been observed (we proved that in our report). We can get our formula of finding the probability of success in different ways (sums, integrals). And we proved that asymptotically we have to let ~37% pass and then select the better one then already observed ones. We created Python code (GitHub) and simulated it. However, The Golden Rule is not good for the maximizing the selection of items as we proved that it passed maximum expectation point.

The project code, reports and visualization were uploaded to our Ad Fontes repository (link)