Application of the Greedy Algorithm to Lower Infection in the Game “Pathologic”

Hayya Zuhailii Kinasih - 13522102
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13522102@std.stei.itb.ac.id

***Abstract*—Pathologic is a survival game in which the player must survive for twelve days in a pandemic. The main mechanic of the game is to manage resources in order to survive. One of the main threats to survival is getting infected with the plague, which can be managed using the correct items. This problem can be conceptualized as a 0/1 knapsack problem, which can be solved using the greedy algorithm. The algorithm provides an effective way of using the available infection-lowering items while considering the cost of using each item. It weighs the benefits of using an item against its cost, giving a solution that has the most value for lowering and maintaining a low infection. It does not always give the most optimal solution, but it can still serve its purpose, as choosing the right item can be difficult due to the vague descriptions of each item.**

***Keywords*—greedy algorithm; pathologic; pathologic classic hd; knapsack problem**

I. Introduction

Pathologic, also known as Мор. Утопия, is a first-person survival game originally released in 2005, developed by Russian studio Ice-Pick Lodge. Later, in 2015, a remaster of the game with improved graphics and a more understandable English translation was released digitally under the name Pathologic Classic HD. At the beginning of the game, the player is given a choice between three characters to play as: Bachelor, Haruspex, or Changeling. Each character offers unique perspectives and abilities. Regardless of which character is chosen, the player enters a strange and unfamiliar town in pursuit of some goal or other. However, the player is almost immediately thrust into the role of a “healer”, entrusted with finding a way to stop the spread of a deadly disease, commonly referred to as the plague.

Fig. 1. Pathologic Classic HD (source:https://store.steampowered.com/app/384110/Pathologic_Classic_HD/)

The player must complete quests and survive until day 12, in which the game will end. One of the largest threats to player survival is getting infected with the plague, because the plague will eat away at the players’ health over time, all the while the level of infection itself rises, eating away larger and larger chunks of health. Cures do exist in the game; however, they are scarce and better saved for when a quest demands it. Therefore, it is important to maintain a low level of infection using drugs that can lower it, at the cost of other player stats, such as health. There is still the issue that the player is often pressed for time and looking through the inventory does not pause the game. That, alongside with the fact that most drugs have vague descriptions, can make it difficult to know which ones will effectively reduce the player’s infection without sending the player to death’s door.

This poses a knapsack problem. Which drugs should the player take in order to maximize the benefits from taking it without dying? One method of solving this problem is to use the Greedy algorithm, which will assign values to certain drugs based on its properties and choose the most valuable one to use, then the next most valuable one, and so on until none of the drugs can be taken without killing the player.

II. Theory

A. Greedy Algorithm

The greedy algorithm is a simple algorithm designed to quickly solve optimization problems. It produces steps to achieve the optimal goal, with each step being the result of considering the current situation and taking the best action at that time. Due to this focus on the best course of action only at each step, the algorithm may come up with a suboptimal solution. It does not go back to earlier steps even though doing so will produce a better solution further ahead.

The greedy algorithm usually has the following elements:

- Candidates set: A set of choice candidates at every step.

- Solution set: A set of choices that were already taken.
- Solution function: A function that decides whether the solution set is already made up of the full solution to the problem.
- Selection function: A function that selects the best candidate within the candidate set.
- Feasibility function: A function that decides if the selected candidate can feasibly be placed in the solution set.
- Objective function: The type of optimization, between maximizing or minimizing the value of the solution.

B. *Infection Management*

Infection is a property of the player, describing the level of infection the player currently possesses, represented by a bar that fills up as infection levels go up. Without using a cure, there is no way to completely eradicate infection. Fig. 2 shows a player who is not infected. Other properties that are relevant in this situation are Health and Immunity.

Health, quite simply, describes the players' health. If it reaches zero, the player dies. Fig. 2 shows a player who has very little health. Every hour that goes by while the player is infected, the players' Health is reduced by an amount proportional to the level of infection.

Immunity describes the players' capability of fighting back an infection. The default value of Immunity is 0.5, and if the players' Immunity is more or less than that, it will gradually increase or decrease to return to that value. The role that Immunity plays here is in slowing the rate in which Infection grows, buying more time for the player. Fig. 2 shows a player with roughly 0.8 Immunity.

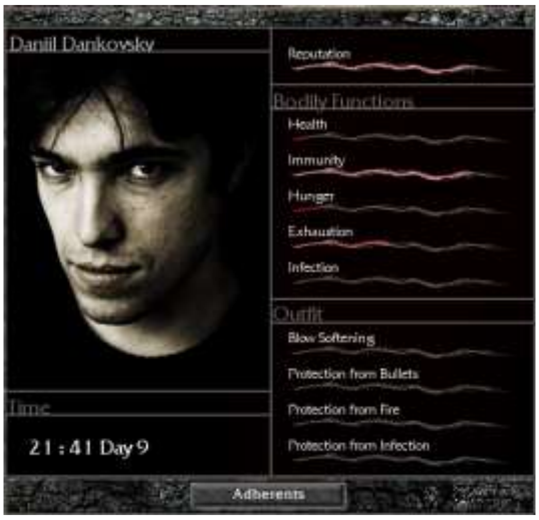


Fig. 2. Player Properties

It should be noted that the game's interface itself does not provide the exact values for each property. One way to know it is to estimate based on the bar. Another way to know it is by using console commands, accessed by pressing ` on the keyboard. The command to input follows this format: `prop <id_player> <stat>`.

To get the players' ID number, input `id_player` into the console. The stats are `health` for Health, `immunity` for Immunity, and `disease` for Infection.

In Pathologic, drugs are sorted into several categories, such as antibiotics, cures, immunity boosters, painkillers, and vaccines. They are stored in the tab labeled "Drugs" in the players' inventory. A drug is immediately consumed when the player clicks on it.

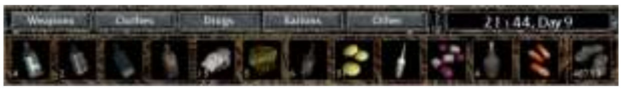


Fig. 3. Drugs Tab in Inventory

Drugs can be acquired by purchasing them in pharmacies, bartering knick-knacks with children, or receiving them through quests. All three of those sources are either unpredictable or unrepeatable. Purchasing drugs from pharmacies requires having enough money to buy them (price hikes are exorbitant in a pandemic) and the availability of the drug in the first place. Bartering with children require coming across the children who barter drugs and having the specific trinkets the children want, not to mention that they are inaccessible at night. Quests are not very reliable sources of drugs, as they are not repeatable.

1) *Antibiotics*

Fig. 4. Feromycinum, Monomycinum, Neomycinum (In Order)

Antibiotics are a type of drug that can lower- but not cure- infection, at the cost of Health and Exhaustion. They will reduce Health by a set amount, increase Exhaustion by a set amount, and multiply the players' infection by a set amount. The properties of each antibiotic are listed below. The data is taken from the Pathologic Wiki.

TABLE I. ANTIBIOTICS

	Infection	**Health**	**Exhaustion**
Feromycinum	0.2	30	30
Monomycinum	0.4	25	25
Neomycinum	0.6	15	15

There is another type of antibiotic that is quite different from the rest, called Dead Gruel. It does not affect the players' Exhaustion. It can be obtained through the brewing mechanic available only to the Haruspex and its' effects change based on the ingredients used to make it.

Fig. 5. The Haruspex Brewing Up Dead Gruel

2) *Immunity Booster*

Fig. 6. Alpha-Tablets, Beta-Tablets, Gamma-Tablets, Delta-Tablets (In Order)

Immunity Booster is a type of drug that can increase Immunity at the cost of Health. It raises and reduces Immunity and Health respectively, by an amount depending on the immunity booster. The properties of each immunity booster are listed below. The data is also taken from the Pathologic Wiki.

TABLE II. IMMUNITY BOOSTERS

	Immunity	Health
Alpha-Tablets	10	1.5
Beta-Tablets	15	1.5
Gamma-Tablets	30	1.5
Delta-Tablets	60	2

Like the antibiotics, there is a unique type of immunity booster that is different from the ones listed. It is called Twyrine Extract, and its' method of acquisition is through the brewing mechanic only available to the Haruspex. Other than that, both the Haruspex and Changeling can trade herbs for it. The Bachelor can only acquire them through a quest on day 7. Like Dead Gruel, several different Twyrine Extracts can be made with different effects on Health and Immunity.

Fig. 7. The Bachelor Acquiring Twyrine Extract on Day 7

3) *Other*

The last item that is relevant for this piece is the painkiller called Twyrine which increases both Immunity and Exhaustion by 5. There are several drugs that are excluded from this application due to reasons listed below.

- Vaccines are a type of drug that will set the players' Immunity to a certain value and maintain it at that value for a certain length of time. They are not included because they are very rare and they are much better used in specific situations, such as when the player is about to spend some time in a plague infected area.
- Cures can eradicate Infection entirely. They are excluded because cures are sometimes required to complete quests. It should be up to the player if they want to cure themselves or not.
- All other drugs are excluded because they do not affect Infection nor Immunity.
- There is one method of lowering Infection that is excluded, which is approaching a flamethrower wielder while infected, available after day 9. The flames will burn away (but not cure) the infection, at significant cost of health. The player is free to try this at their own risk.

III. IMPLEMENTATION

The problem at hand here resembles the integer knapsack problem. The problem goes like this: given n objects, each with weight w and profit p, as well as a knapsack with a weight capacity of K, which objects should be placed in the knapsack to maximize profit while staying within the knapsack's capacity?

In this scenario, the objects are the drugs currently available within the players' inventory, the profit p is the amount of Infection it reduces or Immunity it adds, the weight w is the cost of Health, and the knapsack's capacity K is the player's current Health. However, the player may want to input a limit on how much Health they are willing to lose, as it would be quite unfortunate to take some drugs, only to then die of getting stabbed by muggers. In that case, the capacity K is the player's current Health subtracted by the limit the player would like to impose.

The problem is mapped to each element of the greedy algorithm like so:

- Candidate set: A set of every drug currently in the players' inventory.
- Solution set: A set of drugs already chosen by the algorithm.
- Solution function: A function that checks if there are still any drugs within the candidate set that will not kill the player upon consumption.

- Selection function: A function that selects a drug from the candidate set, measured by some metric.
- Feasibility function: A function that checks that the selected drug will not kill the player upon consumption.
- Objective function: The overall Infection reduced, and Immunity raised is maximized.

The program has several classes to represent the drugs. Those classes are Antibiotic, ImmunityBooster, Gruel that extends Antibiotic, and Extract that extends ImmunityBooster. Twyrine will be lumped in with ImmunityBooster, with a check for when the player consumes it. Each of those classes have a function that returns the value of the drug. Other than that, there is also a Player class, with methods that consume a drug and inflict the effects on the player.

The candidate set will be divided into a list of antibiotics and a list of immunity boosters. The reason for this is that lowering infection is the main priority, with immunity coming in second. The program will first go through the antibiotics before going through the immunity boosters.

The program itself will follow this structure:

```
antibiotics: array of Antibiotic
boosters: array of ImmunityBooster
p: Player
next, cap: integer
antibiotics, boosters <- readInventory()
cap <- input()
next <- select index of best antibiotic
while next != -1 and antibiotics[next] is feasible do
  p.consume(antibiotics[next])
  antibiotics.pop(next)
  next <- select index of next best antibiotic

next <- select index of best immunity booster
while next != -1 and boosters[next] is feasible do
  p.consume(boosters[next])
  boosters.pop(next)
  next <- select index of next best immunity booster
```

The selected candidate is feasible if the health cost of the drug is less than or equal to the current health of the player subtracted by the health cap imposed by the player. In the case of immunity boosters, it will also stop if the player's immunity is at 1.

The metric used for selecting a candidate drug will be chosen between greedy by profit (Infection or Immunity), greedy by cost (Health cost, added with Exhaustion if it is present), and greedy by density (profit over cost). There was consideration to also take into account the monetary cost of items in pharmacies, perhaps even giving the suggestion to purchase a drug. However, after further consideration, it has been determined that factoring that into the cost will needlessly overcomplicate the problem. First, determining that sort of cost for Dead Gruel and Twyrine Extract will be quite difficult, as they are not exactly available for purchase through traditional means. Second, not every drug is available at any day. In fact, all but two of them are not available until day 3. Third, not every drug is available at any pharmacy. The drug may be available that day, but a pharmacy might not have it. A suggestion to buy something at a pharmacy will be rather useless when it turns out that the closest pharmacy that has the specific drug is halfway across the town.

The algorithm for selecting a candidate goes like the following, with the calculated value depending on the type of greedy algorithm used:

```
function selectBest(list: array) -> integer
  idx <- -1
  val <- 0
  i interval [1..len(list)]
    v <- calculated value of list[i]
    if v > val then
      idx <- i
      val <- v
  -> idx
```

The calculated value of each type of drug for each algorithm is as follows:

TABLE III. CALCULATED VALUES

	Profit	Cost	Density
Antibiotic	1 – Infection	Health + Exhaustion	(1 – Infection) / (Health + Exhaustion)
Immunity Booster	Immunity	Health	Immunity / Health

IV. TESTING

A. Test 1

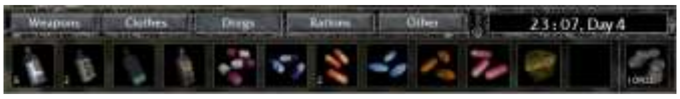


Fig. 8. Inventory for Test 1

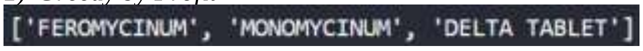
Drugs: Feromycinum(1), Neomycinum(1), Monomycinum(2), Beta-Tablets(1), Gamma-Tablets(1), Delta-Tablets(1)

Health: 0.597

Immunity: 0.874

Infection: 0.102

1) Greedy by Profit



```
['FEROMYCINUM', 'MONOMYCINUM', 'DELTA TABLET']
```

Fig. 9. Solution for Test 1 Using Greedy by Profit

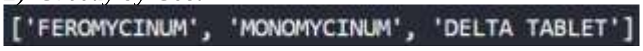
Solution: Feromycinum(1), Monomycinum(1), Delta-Tablets(1)

Health Result: 0.027

Immunity Result: 0.999

Infection Result: 0.008

2) Greedy by Cost



```
['FEROMYCINUM', 'MONOMYCINUM', 'DELTA TABLET']
```

Fig. 10. Solution for Test 1 Using Greedy by Cost

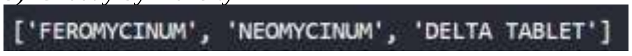
Solution: Feromycinum(1), Monomycinum(1), Delta-Tablets(1)

Health Result: 0.027

Immunity Result: 0.999

Infection Result: 0.008

3) Greedy by Density



```
['FEROMYCINUM', 'NEOMYCINUM', 'DELTA TABLET']
```

Fig. 11. Solution for Test 1 Using Greedy by Density

Solution: Feromycinum(1), Neomycinum(1), Delta-Tablets(1)

Health Result: 0.128

Immunity Result: 0.999

Infection Result: 0.012

B. Test 2

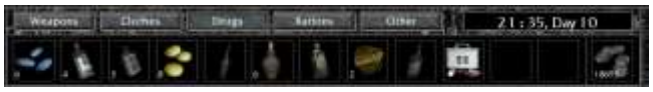


Fig. 12. Inventory for Test 2

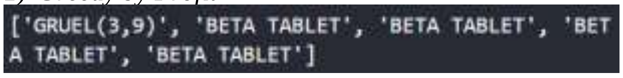
Drugs: Beta-Tablets(6), Alpha-Tablets(8), Dead Gruel 3/9 (1), Twyrine (1)

Health: 0.426

Immunity: 0.5

Infection: 0.151

1) Greedy by Profit



```
['GRUEL(3,9)', 'BETA TABLET', 'BETA TABLET', 'BET
A TABLET', 'BETA TABLET']
```

Fig. 13. Solution for Test 2 Using Greedy by Profit

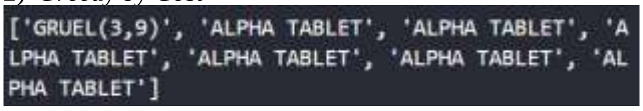
Solution: Dead Gruel 3/9 (1), Beta-Tablets(4)

Health Result: 0.275

Immunity Result: 0.999

Infection Result: 0.004

2) Greedy by Cost



```
['GRUEL(3,9)', 'ALPHA TABLET', 'ALPHA TABLET', 'A
LPHA TABLET', 'ALPHA TABLET', 'ALPHA TABLET', 'AL
PHA TABLET']
```

Fig. 14. Solution for Test 2 Using Greedy by Cost

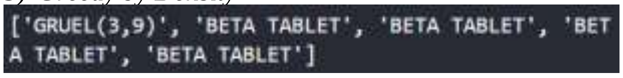
Solution: Dead Gruel 3/9 (1), Alpha-Tablets(6)

Health Result: 0.245

Immunity Result: 0.999

Infection Result: 0.004

3) Greedy by Density



```
['GRUEL(3,9)', 'BETA TABLET', 'BETA TABLET', 'BET
A TABLET', 'BETA TABLET']
```

Fig. 15. Solution for Test 2 Using Greedy by Density

Solution: Dead Gruel 3/9 (1), Beta-Tablets(4)

Health Result: 0.275

Immunity Result: 0.999

Infection Result: 0.004

C. Test 3

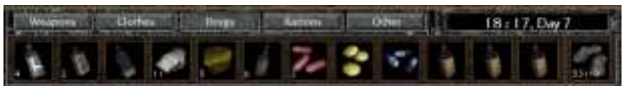


Fig. 16. Inventory for Test 3

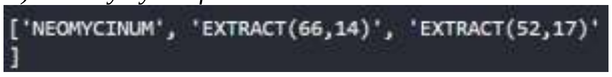
Drugs: Twyrine(6), Gamma-Tablets(2), Alpha-Tablets(1), Neomycinum(1), Twyrine Extract 44/19 (1), Twyrine Extract 66/14 (1), Twyrine Extract 52/17 (1)

Health: 0.739

Immunity: 0.188

Infection: 0.250

1) Greedy by Profit



```
['NEOMYCINUM', 'EXTRACT(66,14)', 'EXTRACT(52,17)'
]
```

Fig. 17. Solution for Test 3 Using Greedy by Profit

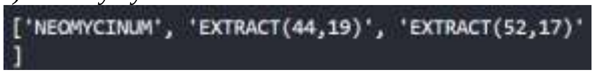
Solution: Neomycinum(1), Twyrine Extract 66/14 (1), Twyrine Extract 52/17 (1)

Health Result: 0.279

Immunity Result: 0.999

Infection Result: 0.150

2) Greedy by Cost



```
['NEOMYCINUM', 'EXTRACT(44,19)', 'EXTRACT(52,17)'
]
```

Fig. 18. Solution for Test 3 Using Greedy by Cost

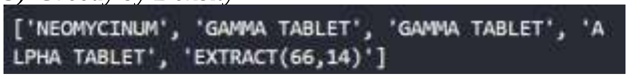
Solution: Neomycinum(1), Twyrine Extract 44/19 (1), Twyrine Extract 52/17 (1)

Health Result: 0.229

Immunity Result: 0.999

Infection Result: 0.150

3) Greedy by Density



```
['NEOMYCINUM', 'GAMMA TABLET', 'GAMMA TABLET', 'A
LPHA TABLET', 'EXTRACT(66,14)']
```

Fig. 19. Solution for Test 3 Using Greedy by Density

Solution: Neomycinum(1), Gamma-Tablets(2), Alpha-Tablets(1), Twyrine Extract 66/14 (1)

Health Result: 0.402

Immunity Result: 0.999

Infection Result: 0.150

D. Test Analysis

The program can give a solution that can lower the player's Infection by quite a significant amount, and given the availability of immunity boosters, gives a solution that will max out the player's Immunity. However, each given method does not always give the most optimal solution. In the first test, greedy by profit and cost managed to lower infection more than greedy by density. In the second test, greedy by profit and density managed to cost less health than greedy by cost, while still reducing the same amount of infection and raising the same amount of immunity. In the third test, greedy by profit and cost gave a solution that costs more health than greedy by density. To recap, greedy by density failed to give the optimal solution in the first test, greedy by profit failed to give the optimal solution in the third test, and greedy by cost failed to give the optimal solution in both the second and third test.

V. CONCLUSION

The greedy algorithm is an algorithm built to solve optimization problems simply and quickly. It evaluates the situation at each step and produces the best action to take at that step, without turning back or looking ahead for more optimal overall solutions. The algorithm is used to solve a problem present in the game Pathologic, in which the player needs to choose which drugs to consume in order to lower and maintain a low infection level while staying alive. The best course of action at each step is evaluated through three methods, which are according to the drugs' profit, cost, and density. Each of these three methods are able to give the optimal solution, though not always. There are scenarios in which one of these methods do not give the optimal solution, but other methods do. Even so, this algorithm provides an easy and accurate way of doing what needs to be done. It succeeds in lowering infection while keeping the player alive. This is useful due to the fact that within the game itself, the descriptions of most drugs are vague and does not communicate to the player what the exact effects of each drug are. Coupled with the inability to get specific numbers for player statistics without rooting through the console, choosing the correct drug to use can be a daunting task, something that the application of this algorithm can ease.

ACKNOWLEDGMENT

The author would like to thank Dr. Nur Ulfa Maulidevi, S.T, M.Sc. as the lecturer of college course Algorithm Strategies for class number two, who taught the class the subject of the greedy algorithm as well as all of the other subjects in the course. The author would also like to thank Dr. Ir. Rinaldi, M.T for the material of this course that is accessible on his website. Lastly, the author would like to thank the people who contributed to the Pathologic Wiki for the comprehensive overview of the game's items and console commands. This paper would not be possible without the support of any of the aforementioned people.

REFERENCES

[1] R. Munir, "Algoritma Greedy (Bagian I)," 2021. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag1.pdf

[2] "Drugs," *Pathologic Wiki*. https://pathologic.fandom.com/wiki/Drugs (accessed Jun. 11, 2024).

[3] "Console Commands," *Pathologic Wiki*. https://pathologic.fandom.com/wiki/Console_Commands (accessed Jun. 11, 2024).

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 12 Juni 2024

[illegible]

Hayya Zuhailii Kinasih 13522102