Minimizing Risk in Baldur's Gate 3 Honor Mode Run using State Dependent Open Traveling Salesperson Problem

Nathaniel Marvelo - 13525107
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: marvelnathaniel07@gmail.com , 13525107@std.stei.itb.ac.id

Abstract*—Baldur's Gate 3 features a challenging difficulty setting called Honor Mode, which limits players to a single save file and ends the campaign upon a party wipe. These restrictions make route planning crucial for successful progression. This study applies graph theory to model Act I of the game, where combat encounters are represented as vertices and possible transitions between encounters as edges in a weighted graph. A risk-based optimization approach inspired by the Traveling Salesman Problem (TSP) is used to determine an encounter order that minimizes overall risk while maximizing experience gain. The algorithm evaluates encounter difficulty relative to the player's current level and prioritizes safer encounters to improve character progression. The results show that graph-based optimization can produce a safer route through Act I and reduce the likelihood of Honor Mode run failure. This research demonstrates the practical application of graph theory and optimization techniques in video game route planning.

Keywords: Graph Theory, Traveling Salesman Problem, Optimization, Baldur's Gate 3, Honor Mode, Weighted Graph

I. INTRODUCTION

Baldur's Gate 3 is a role-playing game (RPG) developed by Larian Studios. It's a critically acclaimed video game that won The Game Awards 2023 Game of The Year award. It is a really beloved game by the community because of its DND-esque mechanic and many branching paths.

Baldur's Gate 3 has many game modes, including the highly challenging Honor Mode. In this mode, players are forced to plan their run meticulously because they are restricted to a single save slot and this mode also has a permanent death mechanic, meaning that the entire playthrough will end if the player's party is defeated. In addition to these, the mode also offers tougher bosses and stat adjustments that makes the game a lot harder.

Fig1. Baldur's Gate 3 Game Icon

Source: Game

Due to the high difficulty and restrictions, many players experience failures in this mode. According to the achievement statistics on Steam, only a small percentage (3,1%) of total player is able to successfully complete this mode. This suggests that finding a safe progression route can significantly improve the probability of finishing a run on Honor Mode.

This paper presents a way to reduce the risk of failing a honor mode run using a graph-based approach. The objective is to search for a route that provides the safest method of completing the first act of the game. The author chooses to only analyze the first act of the game because this is where most players found the game to be the most difficult. This is because during this stage, characters have limited abilities, resources, and equipment.

II. THEORY

A. Graph

In discrete math, a graph is used to represent discrete objects and their relationship with each other. A graph is formally defined as an ordered pair:

$$G = (V, E)$$

Where:

- V represents a set of vertices
- E represents a set of edges connecting pairs of vertices

1. Types of Graphs

Graph can be categorized into several types based on their properties. Below is some of the graphs used in this paper.

- Undirected Graph: Its edges have no direction.
- Directed Graph: Each of its edges have a specified direction.
- Complete Graph: Each of its vertex is connected to every other vertex. A complete graph with n number of edges is denoted by K_n.

- Weighted Graph: A weighted graph assigns a numerical value to each of its edge.

$$w : E \rightarrow R$$

where w(e) represents the weight associated with edge e.

2. Subgraph

A subgraph is a graph formed from a subset of vertices and edges of a larger graph. For example, if $G = (V, E)$. $G_1 = (V_1, E_1)$ is a subgraph of G if $V_1 \subseteq V$ and $E_1 \subseteq E$.

3. Traveling Salesman Problem

The Traveling Salesman Problem is a well-known combinatorial optimization problem. Given a set of locations and travel costs, the objective is to determine the shortest possible route that visits each location exactly once. Mathematically, TSP tries to find a Hamiltonian cycle with a minimum total cost:

$$min \sum_{i=1}^{n} c_{i\,j}$$

Where $c_{i\,j}$ represents the cost of traveling from location i to location j.

B. *Baldur's Gate 3 Mechanics*

1. Game Progression

This game is divided into a prologue and 3 main acts. The prologue takes places strictly on the Nautiloid, a mind flayer vessel that serves as the game's introductory area. After escaping the Nautiloid, the player will start the first act. The first act is divided into three main regions, such as the Wilderness, Rosymorn Monastery, as well as the Underdark. The progression of the game is largely non-linear. Players may complete encounters and quest in deferring orders, making route selection an important strategic decision.

2. Honor Mode

Honor mode is a game mode that is added on the patch 5 of Baldur's Gate 3. This mode is designed to be the ultimate challenge for players playing this game. This mode includes:

- Single-save file restriction.
- Permanent campaign failure upon party wipe.
- Enhanced enemy statistics.
- Additional Legendary Actions for bosses.

These mechanics significantly increase the importance of planning and risk management.

3. Combat Encounters

Combat encounters are an integral part of Baldur's Gate 3 progression. Baldur's Gate 3 has fixed encounters spread around its world. Each encounter is unique and gives a certain amount of XP and items as a reward for its completion.

4. Level and Experience Gain

Baldur's Gate 3 leveling system has many similarities to DND 5e's rule set. In this game, experience (XP) will be gained for each combat encounters and other actions such as completing quests, gaining inspiration, and unlocking new region. Below is the needed amount of XP for each level.

Level	Cumulative XP	XP for next level
1	0	300
2	300	600
3	900	1800
4	2700	3800
5	6500	6500
6	13000	8000
7	21000	9000
8	30000	12000
9	42000	14000
10	56000	20000
11	76000	24000
12	100000	N/A

Fig2. XP needed per level up

Source: https://bg3.wiki/wiki/Experience

III. METHODS

A. *Constraint*

This paper will only analyze the combat encounters found in the first act of the game. In honor mode, most players will face difficulty in the first act of the game (especially before level 5). This is because the players still possess limited resources and equipment. Therefore, the author thought that optimizing encounter order in the first act of the game provides the greatest relevance. The other reason for this restriction is by restricting the data to act 1 will reduce the model complexity and keep it more manageable.

Furthermore, this paper will only add combat encounters in the dataset. Non-combat activities such as dialogue-based quest, inspiration gain, and quest completion will be excluded as they do not contribute to the difficulty assessment of the game. This is also to make it more aligned to the objective of the paper, that is to minimize the risk of a honor mode run in Baldur's Gate 3.

B. *Data Collection*

The data for the graph is collected from various sources, including information on the game with some help from an interactive Baldur's Gate 3 Map site, https://mapgenie.io/baldurs-gate-3. From this process, there are

69 combat encounters found that will be used as the vertices of the graph.

After listing all the combat encounters, the author then listed the estimated needed level to complete each encounter. This is mostly found by using the highest enemy level for each of the encounter. There are of course some exceptions to make the data more accurate.

The author decided to add 0.5 of level for each encounter that has an enemy with legendary action. This is because legendary action adds much to the difficulty of the encounter.

Other than that, the author also subtracts 0.5 of level for each encounter where there are non-playable characters that helps the player on the encounter. The author also subtracts 0.5 level for encounters that only has one normal mob as the enemy.

Moreover, the author also decided to give some exception for a few encounters that its difficulty can't be depicted by the level of the enemies. These encounters include the DuergarBeach, Mimic, Gremishka, and Grym. The level of the encounter is adjusted based on the difficulty of the encounters on the vicinity of each of those encounter and the appropriate level the party will reach during the encounter.

As a semi open world game, most of the encounters can be done anytime the player wants to. This makes most of the encounters can be done whenever the player wanted to without any constraint. The author also has already tested this extensively and found that all encounters that seems unskippable or may seem to be a prerequisite to another encounter can be skipped by using available in game mechanics. However, there is an exception, that is for the encounters found on the prologue of the game. Because of this, the connection of most of the combat encounter found will create a complete graph, with the exception of three encounters: Imp1, Tadpoled, and Imp2.

C. *Graph Modeling*

The encounter network is represented as an undirected graph:

$$G = (V, E)$$

where:

- V = encounter nodes.
- E = connections between encounters

Each node represents a combat encounter and contains two attributes:

$$v_i = (L_i, XP_i)$$

where:

- Li = Recommended level
- XPi = XP reward obtained from encounter i

This graph is implemented using the NetworkX library in Python.

```
65  G_complete = nx.Graph()
66
67  for node, data in encounters.items():
68      G_complete.add_node(node, level=data["level"])
```

Fig3. Complete graph implementation code

Source: Author

Connections between graph are created based on the encounter matrix.

```
72  for i in range(len(nodes_list)):
73      for j in range(i + 1, len(nodes_list)):
74          node1, node2 = nodes_list[i], nodes_list[j]
75
76          connection_value = encounter_matrix.loc[node1, node2]
77
78          if connection_value > 0:
79              G_complete.add_edge(node1, node2, weight=connection_value)
```

Fig4. Graph connection code

Source: Author

The node positions are generated using Kamada-Kawai force-directed layout algorithm that is available in NetworkX.

```
81  pos = nx.kamada_kawai_layout(G_complete)
```

Fig5. Node positions code

Source: Author

D. *Route Optimization*

A risk function is used to quantify the difficulty of attempting a combat encounter. The risk is calculated based on the difference between the encounter recommended level and the player's current level.

$$d = Le\text{-}Lp$$

where:

- d = difference
- Le = encounter recommended level
- Lp = player's current level

The risk value is then defined as:

$$Risk(L_e, L_p) = \begin{cases} 0, & d \leq 0 \\ (d+1)^2, & d > 0 \end{cases}$$

```
28  def risk(encounter_level, player_level):
29      diff = encounter_level - player_level
30      return 0 if diff <= 0 else (diff + 1) ** 2
```

Fig6. Risk value code

Source: Author

This formula assigns zero risk when the player level is the same or exceeds the encounter's recommended level. Otherwise, the risk will quadratically increase as the level difference widen. This quadratic formula is chosen to reflect the non-linear increase in combat difficulty that is commonly found in role-playing games. Small difference in level is usually more manageable, while larger level gaps will become increasingly dangerous.

The player level is determined by the cumulative experience points gained throughout the route taken.

$$XP_{total} = \sum_{i=i}^{n} XP_i$$

The player's level is calculated according to the predefined XP threshold in the theory section that is taken from Baldur's Gate 3 mechanic.

```
LEVEL_THRESHOLDS = {1: 0,
                    2: 300,
                    3: 900,
                    4: 2700,
                    5: 6500,
                    6: 13000,
                    7: 21000,
                    8: 30000,
                    9: 42000,
                    10: 56000,
                    11: 76000,
                    12: 100000}
```

Fig7. Level thresholds

Source: Author

Implementation:

```
def get_level(xp):
    level = 1
    for lvl, threshold in LEVEL_THRESHOLDS.items():
        if xp >= threshold:
            level = lvl
    return level
```

Fig8. Level up code

Source: Author

Because the XP gain is only calculated based on the one gained from battle encounters, the author decides to add a multiplier to the XP gained from each of the encounter. The multiplier chosen is multiplied by 2.25. This number is chosen based on the accounts of author own and others experience that will reach between level 7-8 when finishing act 1 optimally. Using this multiplier, the XP amount reached will be 23643 between level 7 and 8. By using this multiplier will also make it possible to gain 0 risk accumulation at the end.

The route generation process is heavily inspired by the Traveling Salesperson Problem (TSP). However, unlike the classical TSP there are some differences. This study minimizes encounter risk, unlike TSP that minimizes travel distance.

For every candidate encounter vi, a cost value is computed as:

$$Cost(v_i) = Risk(L_i, L_p)$$

The next encounter is then selected by choosing the unvisited node with the minimum cost:

$$v_{next} = \arg\min Cost(v_i)$$

where:

- U = unvisited encounters

Implementation:

```
    for node in unvisited:
        connection_value = encounter_matrix.loc[current_node, node]

        if connection_value == 0 and current_node != node:
            continue

        cost = risk(encounters[node]["level"], level)

        if cost < best_cost:
            best_cost = cost
            best_node = node
```

Fig9. Route generation code

Source: Author

The algorithm starts from the encounter "Imp", which serves as the initial node. At each of the iterations, the player's XP and level are updated according to the XP obtained from the selected encounter.

```
xp = encounters[start]["xp"]
level = get_level(xp)
```

Fig10. Xp and level update

Source: Author

This process will continue until there is no valid unvisited encounters remaining.

This solution employs a greedy nearest-risk heuristic because an exact TSP solution would be too complicated for a larger graph, such as this one. This approach will provide a practical approximation of the best route that will be consistent with the intended difficulty of the game.

IV. Results

Based on the method told, this paper will only model the route for the combat encounters on the first act of the game. Below is the complete graph made of the available combat encounters in the game.

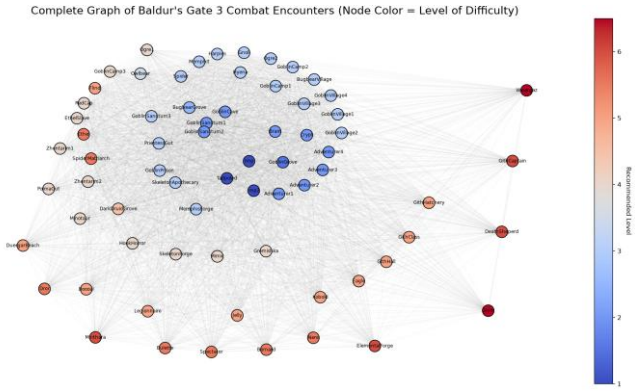

Fig11. Complete graph

Source: Author

After modelling the complete graph, the program will also search for the best route available using the method determined in the method section of this paper. Below is the route made with the program.

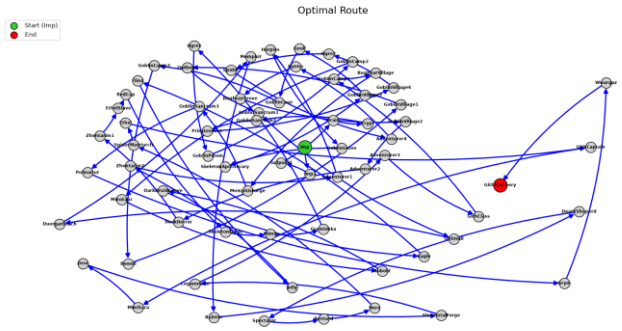

Fig12. Optimized Route

Source: Author

The author also decided to make an output of all of the steps taken including the level of difficulty and gained XP and the state of player at the end of the route. The generated route produced a total experience gain of 23643, allowing the player to reach Level **7** by the end of the analyzed encounters. The cumulative risk value obtained from the route was 0, indicating that the selected encounter order successfully minimized exposure to high-risk battles during the early stages of the game.

```
Result of Combat Encounter Optimization in Baldur's Gate 3 Honor Mode
============================================================
Total EXP Gained : 23643 XP
Player Level  : Level 7
Total Risk Accumulation: 0
------------------------------------------------------------
Optimized Route:
01. Imp                  | Enemy Level: 1.0 | XP Gain: 112
02. Imp2                 | Enemy Level: 1.0 | XP Gain: 112
03. Tadpoled             | Enemy Level: 1.0 | XP Gain: 112
04. Crypt                | Enemy Level: 2.0 | XP Gain: 225
05. GoblinCave           | Enemy Level: 2.0 | XP Gain: 180
06. GoblinSanctum2       | Enemy Level: 2.0 | XP Gain: 270
07. SkeletonApothecary   | Enemy Level: 3.0 | XP Gain: 270
08. GoblinVillage2       | Enemy Level: 3.0 | XP Gain: 225
09. PriestessGut         | Enemy Level: 3.0 | XP Gain: 270
10. GoblinCamp1          | Enemy Level: 3.0 | XP Gain: 270
11. MemphisForge         | Enemy Level: 3.0 | XP Gain: 337
12. BugbearVillage       | Enemy Level: 3.0 | XP Gain: 213
13. Gnoll                | Enemy Level: 3.0 | XP Gain: 315
14. EthelSlave           | Enemy Level: 4.0 | XP Gain: 315
15. RedCap               | Enemy Level: 4.0 | XP Gain: 405
16. Zhentarim1           | Enemy Level: 4.0 | XP Gain: 337
17. Mimic                | Enemy Level: 4.0 | XP Gain: 337
18. HookHorror           | Enemy Level: 4.0 | XP Gain: 450
19. Minotaur             | Enemy Level: 4.0 | XP Gain: 337
20. SkeletonForge        | Enemy Level: 4.0 | XP Gain: 270
21. GoblinCamp3          | Enemy Level: 4.0 | XP Gain: 337
22. Memphit              | Enemy Level: 3.0 | XP Gain: 360
23. Adventurer4          | Enemy Level: 2.0 | XP Gain: 202
24. Owlbear              | Enemy Level: 3.5 | XP Gain: 337
25. Eagle                | Enemy Level: 5.0 | XP Gain: 225
26. Flind                | Enemy Level: 5.0 | XP Gain: 450
27. DarkDruidGrove       | Enemy Level: 4.5 | XP Gain: 405
28. GithHatchery         | Enemy Level: 5.0 | XP Gain: 337
29. Boooal               | Enemy Level: 5.0 | XP Gain: 337
30. GoblinSanctum1       | Enemy Level: 2.0 | XP Gain: 270
31. Jelly                | Enemy Level: 5.0 | XP Gain: 202
32. GoblinVillage1       | Enemy Level: 3.0 | XP Gain: 225
33. GoblinSanctum3       | Enemy Level: 3.0 | XP Gain: 270
34. Adventurer1          | Enemy Level: 2.0 | XP Gain: 202
35. GoblinPrison         | Enemy Level: 3.0 | XP Gain: 247
36. GithClass            | Enemy Level: 5.0 | XP Gain: 337
37. GoblinVillage3       | Enemy Level: 3.0 | XP Gain: 225
38. Spider               | Enemy Level: 3.0 | XP Gain: 315
39. Legionnaire          | Enemy Level: 5.0 | XP Gain: 337
40. Ogre2                | Enemy Level: 3.0 | XP Gain: 168
41. GithHall             | Enemy Level: 5.0 | XP Gain: 450
42. GoblinVillage4       | Enemy Level: 3.0 | XP Gain: 270
43. PolmaGut             | Enemy Level: 4.0 | XP Gain: 168
44. Brain                | Enemy Level: 2.0 | XP Gain: 101
45. Kobold               | Enemy Level: 5.0 | XP Gain: 360
46. Adventurer2          | Enemy Level: 2.0 | XP Gain: 202
47. GoblinCamp2          | Enemy Level: 3.0 | XP Gain: 270
48. BugbearGrove         | Enemy Level: 2.5 | XP Gain: 112
49. ElementalForge       | Enemy Level: 6.0 | XP Gain: 270
50. Dror                 | Enemy Level: 5.5 | XP Gain: 540
51. Nere                 | Enemy Level: 5.5 | XP Gain: 675
52. Minthara             | Enemy Level: 6.0 | XP Gain: 360
53. Ethel                | Enemy Level: 5.5 | XP Gain: 562
54. DeathSheperd         | Enemy Level: 6.0 | XP Gain: 675
55. GithCaptain          | Enemy Level: 6.0 | XP Gain: 562
56. Bulette              | Enemy Level: 5.5 | XP Gain: 562
57. SpiderMatriarch      | Enemy Level: 5.5 | XP Gain: 562
58. Bernard              | Enemy Level: 5.5 | XP Gain: 495
59. Harpies              | Enemy Level: 3.0 | XP Gain: 315
60. Adventurer3          | Enemy Level: 2.0 | XP Gain: 202
61. Hyena                | Enemy Level: 3.0 | XP Gain: 135
62. DuergarBeach         | Enemy Level: 5.0 | XP Gain: 562
63. Gremishka            | Enemy Level: 4.0 | XP Gain: 270
64. GoblinGrove          | Enemy Level: 1.5 | XP Gain: 270
65. Zhentarim2           | Enemy Level: 4.0 | XP Gain: 337
66. Spectator            | Enemy Level: 5.5 | XP Gain: 450
67. Ogre1                | Enemy Level: 4.0 | XP Gain: 506
68. Grym                 | Enemy Level: 6.5 | XP Gain: 1125
69. Wwargaz              | Enemy Level: 6.5 | XP Gain: 1125
------------------------------------------------------------
```

Fig13. Program output

Source: Author

V. CONCLUSION

This research demonstrates that graph theory can be effectively applied to optimize progression routes in Baldur's Gate 3 Honor Mode. By representing combat encounters as vertices and the connections between encounters as edges in a weighted graph, the progression through Act I can be modeled

as an optimization problem. The implementation of a risk-based route selection algorithm inspired by the Traveling Salesman Problem allows encounters to be ordered according to their relative difficulty and the player's current progression level.

The results show that completing lower risk encounters first enables the player to gain experience points efficiently, increase character levels earlier, and reduce the probability of facing encounters that exceed the party's capabilities. Consequently, the cumulative risk of the run can be minimized while maintaining steady character progression throughout Act I. This approach provides a systematic alternative to relying solely on player intuition or trial-and-error when planning an Honor Mode playthrough.

Furthermore, this study highlights the practical application of graph theory and combinatorial optimization in video game analysis. Concepts such as weighted graphs, pathfinding, and route optimization can be used to model complex decision-making processes within games and assist players in developing safer and more efficient strategies.

Future research may extend this work by incorporating additional factors such as equipment acquisition, companion recruitment, quest rewards, travel distance, and probabilistic combat outcomes. More advanced optimization techniques, including genetic algorithms, dynamic programming, or heuristic TSP solvers, may also be explored to obtain routes that further minimize risk and maximize overall efficiency. Additionally, the analysis can be expanded beyond Act I to include the entirety of Baldur's Gate 3 Honor Mode progression.

ACKNOWLEDGMENT *(Heading 5)*

The author would like to give a sincere gratitude to Dr.Ir. Rinaldi, M.T. for his guidance and lectures of the IF1220 Discrete Mathematics Course.

ATTACHMENTS

- Miscellaneous https://drive.google.com/drive/folders/187P_t0Sio5EUsZcwkhzCo0MyUB1TjLM1?usp=sharing
- Source Code https://drive.google.com/drive/folders/187P_t0Sio5EUsZcwkhzCo0MyUB1TjLM1?usp=sharing

REFERENCES

[1] R. Munir, "Graf (Bagian 1)", [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2025-2026/20-Graf-Bagian1-2025.pdf.[Accessed: 17-Juni-2026].

[2] Steam Community, "Baldur's Gate 3 Achievements", [Online]. Available: https://steamcommunity.com/stats/1086940/achievements. [Accessed: 18-Juni-2026].

[3] BG3 Wiki, "Experience", [Online]. Available: https://bg3.wiki/wiki/Experience. [Accessed: 17-Juni-2026].

[4] BG3 Wiki, "Area Level", [Online]. Available: https://bg3.wiki/wiki/Area_level. [Accessed: 18-Juni-2026].

[5] Fextralife Wiki, "Enemies | Baldur's Gate 3 Wiki", [Online]. Available: https://baldursgate3.wiki.fextralife.com/Enemies. [Accessed: 18-Juni-2026].

[6] M. Hahsler and K. Hornik, "Two Types of Open-Loop Travelling Salesman Problem", [Online]. Available: https://www.researchgate.net/figure/Two-types-of-Open-Loop-Travelling-Salesman-Problem-a-Classic-Open-Loop-Travelling_fig2_285808716. [Accessed: 16-Juni-2026].

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, 19 Juni 2026

Nathaniel Marvelo dan 13525107