Graph-Based Modeling for Optimal Draft Pick Recommendation in Mobile Legends Using Professional Tournament Data

Audric Yusuf Maynard Simatupang - 13524010
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: cubapetuking@gmail.com , 13524010@std.stei.itb.ac.id

***Abstract*—This paper presents the design and implementation of an optimal draft pick recommendation system for Mobile Legends : Bang Bang, utilizing both directed and undirected graph representations derived from professional tournament data. Draft pick refers to the strategic phase preceding a match where teams alternately select and ban heroes, heavily influencing team dynamics and match outcomes. Hero synergy is modeled through undirected graphs, capturing co-pick frequency and team performance, while counter relationships are represented using directed graphs that reflect advantage interactions between opposing heroes. Edge weights are calculated based on match outcomes, pick-win statistics, and inter-hero relationships to quantify strategic value. By evaluating team compositions and opponent drafts in real-time, the system delivers data-driven hero recommendations to support competitive decision-making.**

***Keywords*—Mobile Legends; draft pick; graph-based modelling; node strength centrality; maximum spanning tree**

I. Introduction

Mobile Legends: Bang Bang (MLBB) is one of the many MOBA games available to play on the mobile phone. Just like any other MOBA games, Mobile Legends: Bang Bang's gameplay consists of two main phases, the draft pick phase and the match. In the draft pick phase, each team has the chance to pick the heroes they need and ban the heroes that are determined to be obstacles to their team composition. This phase is where the strategies of each team are determined. The chosen strategies and heroes must align with the players' individual strengths. The outcome of the draft pick phase may very well determine the outcome of the whole match, that is why the draft pick phase is important.

MLBB has a player base worldwide. With more than 1.5 billion downloads and over 110 million monthly active users, MLBB naturally supports a global competitive scene. One of the most popular global tournaments that MLBB has held is the M series and one of the most powerful competitors in this tournament are teams that originate from Indonesia. Indonesia won the first M series trophy and is still dominant until now, being the runner-up of the latest M series tournament.

In addition to global tournaments, MLBB regularly hosts regional competitions. One of the latest regional tournaments held in Indonesia is Mobile Legends Professional League Indonesia Season 15 (MPL ID S15). Since Indonesia is one of the most dominant regions worldwide and the recent conclusion of MPL ID Season 15, this paper uses data from MPL ID S15 as the basis for analysis.

Because draft pick dynamics in MLBB are complex and difficult for many players to master, this paper develops and constructs a design that helps players create team compositions like how a professional team would.

II. Theoretical Framework

A. Graph

1) Definition of a Graph

A graph is a mathematical structure used to model pairwise relationships between objects. Formally, a graph G is defined as an ordered pair:

$$G = (V,E)$$

where:

- V is a non-empty set of vertices (also called nodes)
- $E \subseteq V \times V$ is a set of edges (also called links) that connect pairs of vertices

2) Simple Graph

A simple graph refers to the simplest form of a graph. The basic graph has its own characteristics which are unweighted, undirected, no self-loops, and no parallel edges.

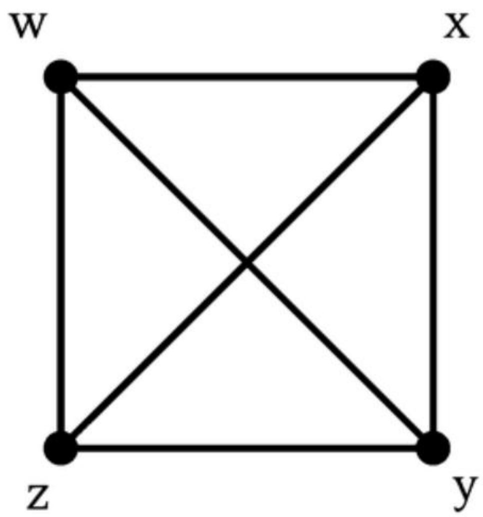

Figure 1. Simple Graph Example
Source: https://www.researchgate.net/figure/Simple-graph-drawn-in-two-different-ways-examples-of-walks-are-g-a-f-h-b-c-f_fig1_273062204

3) Undirected Graph and Directed graph

An undirected graph is a graph where the edges don't have a direction. This shows that if two nodes are connected with an edge, the connection goes both ways — a sign that shows that the relationship is mutual or bidirectional.

A directed graph is a graph where the edges have a direction, indicating a one-way relationship between a node with another node. Unlike undirected graphs, a directed graph's connections are not mutual and are usually used to model asymmetric relationships between nodes.

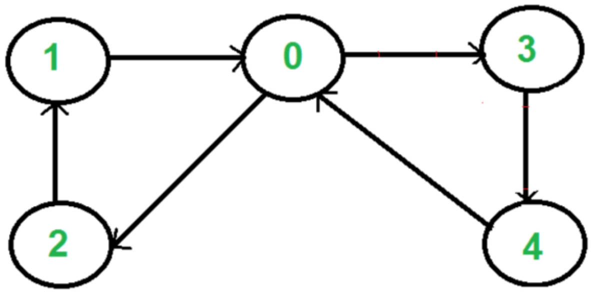

Figure 2. Directed Graph Example
Source: https://www.geeksforgeeks.org/dsa/what-is-directed-graph-directed-graph-meaning/

4) Unweighted Graph and Weighted Graph

An unweighted graph is a graph where each edge is not given a numerical value. This shows that each edge is considered to have an equal value. The presence of an edge shows that the related nodes have a connection, without any indication of strength or importance.

A weighted graph is a graph where each edge is given a numerical value, called a weight, that shows the strength, cost, or importance of the relationship between two nodes. The weight can be positive, negative, or zero. The graph can also be directed or undirected.

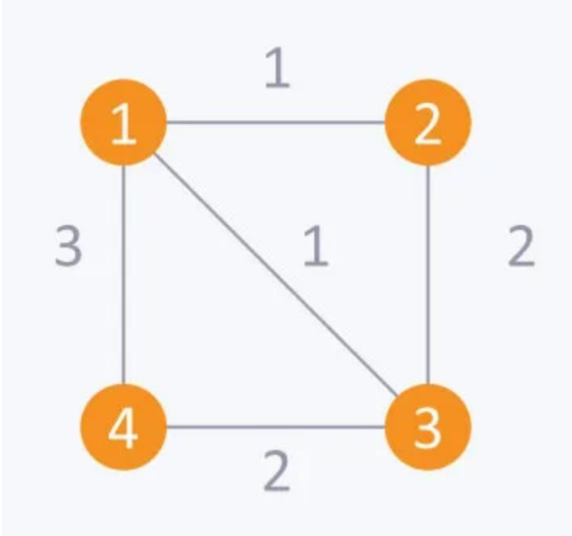

Figure 3. Undirected Weighted Graph Example
Source: https://sethuram52001.medium.com/data-structures-weighted-graphs-3cd86b1b5aa9

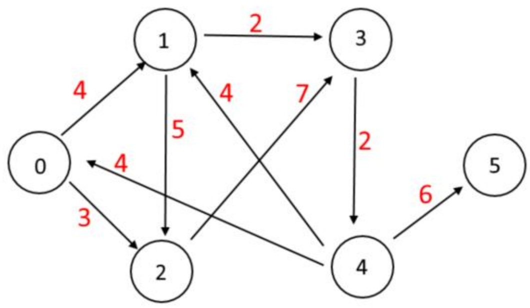

Figure 4. Directed Weighted Graph Example
Source: https://tutorialhorizon.com/algorithms/weighted-graph-implementation-java/

B. *Tree*

1) Definition of a Tree

A tree is a type of graph that is connected, undirected, and acyclic. It means that all the nodes of a tree are connected, the edges of a tree don't have a direction, and in the whole tree, there isn't a single circuit.

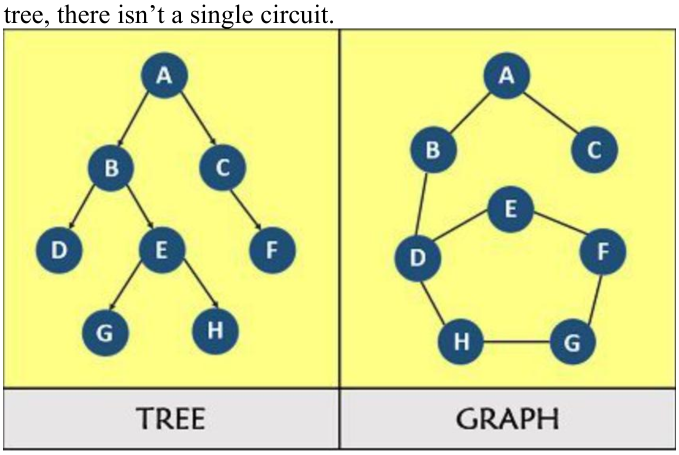

Figure 5. Example of a Tree
Source: https://techdifferences.com/difference-between-tree-and-graph.html

2) Spanning Tree

A spanning tree T is a subgraph of a graph G where T includes all the nodes of G, is a tree, and has exactly V – 1 edge, with V is the number of nodes in graph G. Each node has a unique path connecting it with another node. There are multiple applications of the spanning tree, one of which is the minimum spanning tree. There are two main algorithms used to determine the minimum spanning tree of a graph G. The algorithms are the Prim and Kruskal algorithm.

- Prim algorithm

  These are the steps in Prim algorithm:
  - Find the edge with the least weight then insert the node and edge into T
  - Choose an edge with the least weight adjacent to the node already in T, but does not form a cycle in T
  - Repeat step 2 for n-2 times

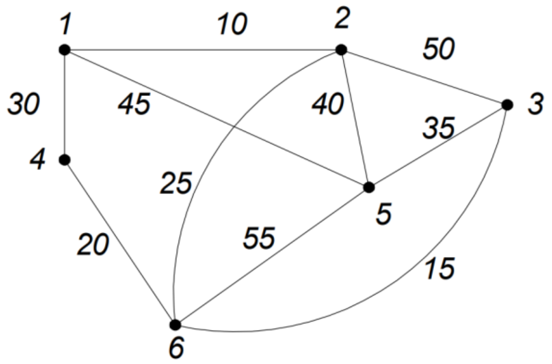

Figure 6. Example of a Graph
Source: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/23-Pohon-Bag1-2024.pdf

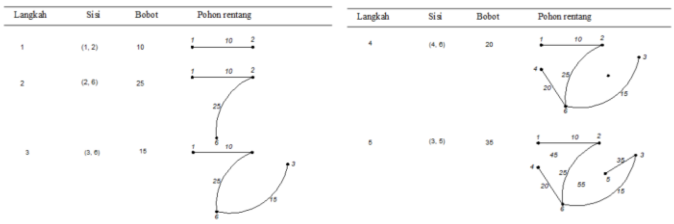

Figure 7. Prim algorithm
Source: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/23-Pohon-Bag1-2024.pdf

- Kruskal algorithm

  These are the steps in Kruskal algorithm:
  - Let T be initially empty
  - Choose an edge with the least weight and does not form a cycle in T then add that edge into T
  - Repeat step 2 for n – 1 times

Sisi	(1,2)	(3,6)	(4,6)	(2,6)	(1,4)	(3,5)	(2,5)	(1,5)	(2,3)	(5,6)
Bobot	10	15	20	25	30	35	40	45	50	55

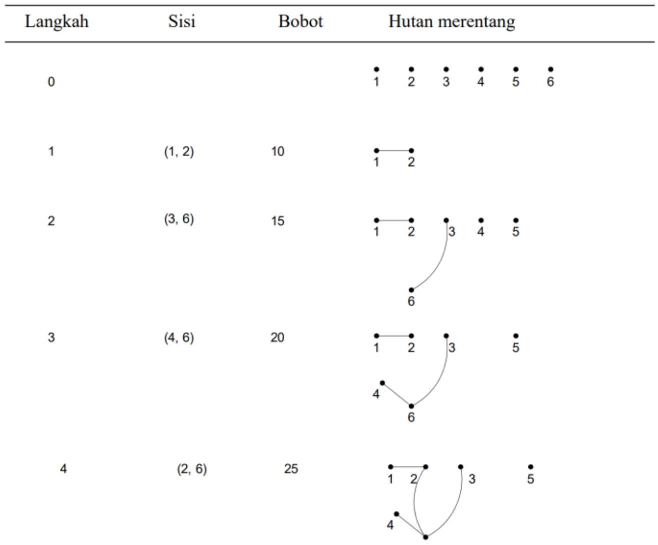

Figure 8.1. Kruskal algorithm for Figure 6’s graph
Source: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/23-Pohon-Bag1-2024.pdf

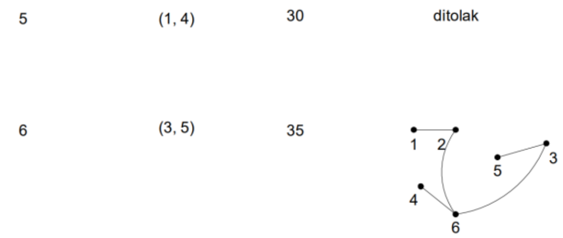

Figure 8.2. Kruskal algorithm for Figure 6’s graph
Source: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/23-Pohon-Bag1-2024.pdf

C. *More Theorems Regarding Graph and Tree*

1) Maximum Spanning Tree

The maximum spanning tree is almost the same as the minimum spanning tree, the only difference being the goal of the tree. The minimum spanning tree aims to get the least amount of weight possible while the maximum spanning tree aims to get the most amount of weight possible. The theorems used in minimum spanning tree, with a little change in the algorithm, can also be used to find the maximum spanning tree.

2) Subtree

A subtree is a portion of a tree that is a tree itself.

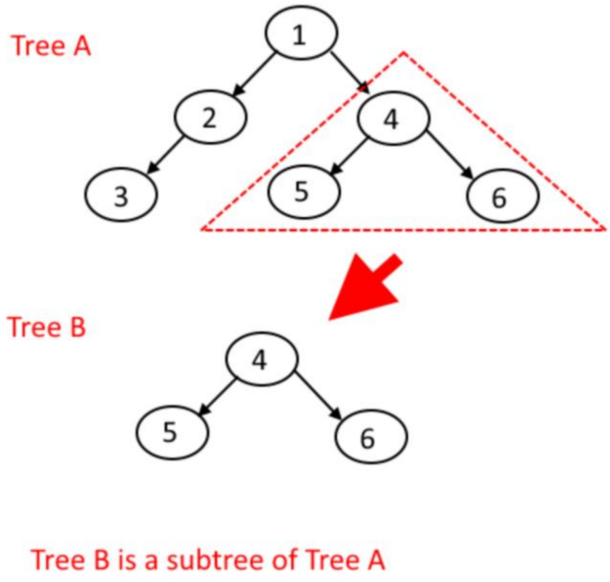

Figure 9. Example of a Subtree
Source: https://tutorialhorizon.com/algorithms/check-if-one-binary-tree-is-a-subtree-of-another/

3) Node Strength Centrality

Node strength centrality is a measure used in weighted graphs to determine how 'central' or 'influential' a node is. The value of the node strength centrality is based on the sum of all the weights of all edges connected to the node.

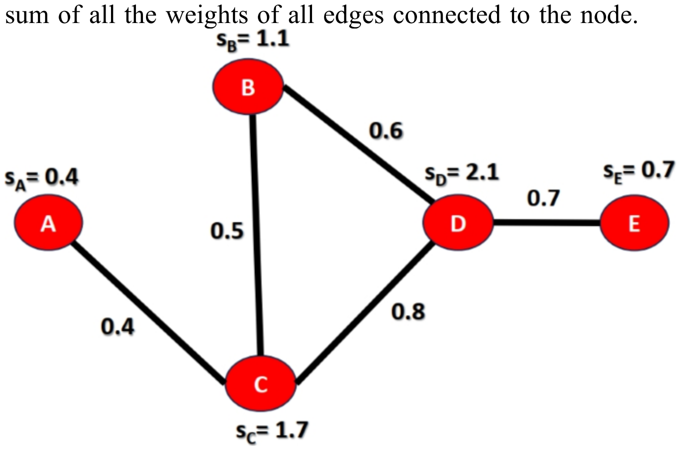

Figure 10. Example of Node Strength Centrality
Source: https://link.springer.com/article/10.1007/s13278-024-01388-6

III. Implementation

The implementation of the theories to make the system can be divided into multiple parts. These are the steps to make the system:

A. Designing the Graph

The system uses three main graphs with two types of graphs. These three graphs are called the synergy graph, team synergy graph, and the counter graph. The synergy graph and the team synergy graph shows the compatibility of the heroes, the higher the weight of the edge, the more compatible the heroes. The synergy graph shows the compatibility of the heroes in a more general way while the team synergy graph shows the compatibility of the heroes in in a more team specific way. On the other hand, the counter graph shows how much a hero counters another hero.

To fully represent the synergy between heroes, both the synergy graph and the team synergy graph uses an undirected weighted graph. On the other hand, to fully show the counter relationship between the heroes, the counter graph uses a directed weighted graph where if node A points to node B, it means that A counters B.

B. Designing the Weight of the Graph

There are multiple aspects and parameters that can be considered to count the weight of the graph. The parameters that are used to calculate the weight of the graph in this paper are mainly the frequency of a hero played with another hero, frequency of a hero played against another hero, frequency of a hero played by a specific team, pick rate, ban rate, and win rate. There are different formulas used for each of the graphs, these are the formulas for each of the graphs:

1) Synergy Graph

The formula used to count the weight of the edge connecting hero A and B is:

$$\text{Weight} = 0.6 * X + 0.3 * Y$$

X = frequency of A played with B that results in a win
Y = frequency of A played with B that results in a loss

After counting the base weight of the graph, the weight is further added by this rule:

If (Weight = 0) then:

$$\text{Final Weight} = 0.1 * Z$$

Z = win rate of the least picked hero between A and B

Else:

$$\text{Final Weight} = \text{Weight}$$

The final weight is the weight assigned to the edges of the synergy graph.

2) Team Synergy Graph

The weight assigned to the team synergy graph is the edited version of the weight originally assigned to the synergy graph. The steps to make a team synergy graph for team U are:

- Count the power of hero A in team U, the formula used is:

$$\text{Power} = 0.5 * X + 0.3 * Y + 0.2 * Z$$

X = total bans of hero A by the opponent when facing team U
Y = total picks of hero A by team U that results in a win
Z = total picks of hero A by team U that results in a loss

- After assigning the power value for hero A in team U, all the edges connected to hero A in the synergy graph are added by the power value.

$$\text{Final Weight} = \text{Weight} + \text{Power}$$

- Repeat these steps for all the heroes. After that, the result is the team synergy graph.

3) Counter Graph

The weight assigned to the counter graph for hero A's relation with hero B is

$$\text{Weight} = X – Y$$

X = total frequency of A going against B and winning
Y = total frequency of A going against B and losing

The direction of the edge is based on the result of the weight, if it's positive, then the edge points to B and if it's negative, the edge points to A.

C. Designing the Draft Pick System Rules

Only using the graph is not enough to make a truly effective system, there are still some rules that must be added as common sense for the system. These rules are:

1) Banned heroes cannot be picked

If a hero has been banned, it cannot be picked. With this rule in place, the system won't recommend heroes that have already been banned.

2) No repicking and rebanning heroes

One of the rules in draft pick in MLBB is that each hero must be picked or banned once. That way, each pick and ban are unique, there will be no two teams using the same hero nor will there be any two teams banning the same hero. With this rule, the system won't recommend heroes that have already been picked or banned.

3) Don't pick heroes that has the same role

This rule isn't a game rule; it is more game sense. If a lane has been filled with a hero, the system shouldn't pick another hero that has to fill the same lane, it should pick a hero that has to fill a different lane. With this rule in place, the system won't recommend heroes that already have their lanes filled.

4) Don't ban heroes that has their lanes filled

This rule is similar to rule (3). If the enemy has already picked a hero that fills a lane, there is no need to ban another hero that also fills that lane because the enemy wouldn't want to choose another hero that fills that lane. With this rule in place, the system won't ban the heroes that already has a lane in the enemy team.

D. Designing the Hero Ban and Selection System

The pick and ban recommendation design is based on the node strength centrality and the maximum spanning tree theory. Here is how the recommendation for pick and ban is designed:

1) Pick Recommendation

First, the system will build the synergy graph, the allied team synergy graph, and the counter graph. Then, the system will iterate through all the nodes in the allied team synergy graph to make a list of nodes that is sorted from the most node strength centrality to the least. Then, this list will be screened through the draft pick system rules to find out the valid choices. After that, the system will check each of the heroes in the counter graph, arrows from the enemy heroes pointing to the hero will decrease the hero score by the weight of the edge and arrows from the hero pointing to the enemy heroes will increase the score of the current hero by the weight of the edge. In conclusion, this is the mathematical formula for the hero score:

$$\text{Score} = X + Y$$

X = total of all the edges around the hero in the team synergy graph
Y = total of all the edges around the hero in the counter synergy graph

The list will then be sorted again from the highest score to the lowest score, then the top five heroes with the highest score will be recommended.

2) Ban Recommendation

The system will build the enemy synergy graph and the counter graph. The system will then use the same method to count the scores of the heroes in the enemy team. After screening the list of heroes through the draft pick system rules, the top five heroes with the highest score from the enemy team will be recommended to be banned by the system.

E. Result Scoring

The scoring of the final team will be by finding the maximum spanning tree of the subgraph containing only the hero nodes and totaling the amount of weight. The higher the score, the better the final team.

F. Implementation in Python

The system is implemented in python, these are the important steps and the parts of the python code related to the graph and draft pick:

1) Building Synergy Graph

This is the code used to make the synergy graph based on a CSV file :

```
def build_sinergi_dasar_graph(pair_wins, hero_stats):
    G = nx.Graph()
    for (h1, h2), stats in pair_wins.items():
        win = stats['win']
        lose = stats['lose']
        total = win + lose
        base_weight = 0.6 * win + 0.3 * lose

        if base_weight == 0:
            # Use hero with fewer total picks
            pick_i = hero_stats[h1]['pick_win'] + hero_stats[h1]['pick_lose']
            pick_j = hero_stats[h2]['pick_win'] + hero_stats[h2]['pick_lose']
            if pick_i == 0 and pick_j == 0:
                continue
            elif pick_i == 0:
                base_weight = 0.1 * (hero_stats[h1]['pick_win'] / 1)
            elif pick_j == 0:
                base_weight = 0.1 * (hero_stats[h2]['pick_win'] / 1)
            else:
                hero = h1 if pick_i < pick_j else h2
                total_picks = hero_stats[hero]['pick_win'] + hero_stats[hero]['pick_lose']
                if total_picks > 0:
                    winrate = hero_stats[hero]['pick_win'] / total_picks
                    base_weight = 0.1 * winrate

        if base_weight > 0:
            G.add_edge(h1, h2, weight=base_weight)

    return G
```

2) Building Team Synergy Graph

This is the code used to build team synergy graph based on a CSV file:

```
def build_sinergi_tim_graph(G_dasar, hero_stats, team_name, matches):
hero_team_weights = defaultdict(float)

for match in matches:
if team_name not in match['teams']:
continue
team_data = match['teams'][team_name]
enemy_team = [t for t in match['teams'] if t != team_name]
if not enemy_team:
continue
enemy_data = match['teams'][enemy_team[0]]

for hero in set(team_data['pick']):
if team_data['is_winner']:
        hero_team_weights[hero] += 0.3
else:
        hero_team_weights[hero] += 0.2

for hero in set(enemy_data['ban']):
hero_team_weights[hero] += 0.5

G_tim = G_dasar.copy()

for node in G_tim.nodes:
if node in hero_team_weights:
for neighbor in G_tim.neighbors(node):
        if G_tim.has_edge(node, neighbor):
        G_tim[node][neighbor]['weight'] +=
        hero_team_weights[node]

return G_tim
```

3) Building Counter Graph

This is the code used to build counter graph based on a CSV file:

```
def build_counter_graph_simple(versus):
    G = nx.DiGraph()
    for (h1, h2), stats in versus.items():
        win = stats['win']
        lose = stats['lose']
        score = win - lose
        if score > 0:
            G.add_edge(h1, h2, weight=score)
        elif score < 0:
            G.add_edge(h2, h1, weight=-score)
    return G
```

IV. RESULT

The result will be divided into two parts, the visualization of the graphs and the result of the system.

1) *Visualization of the Graph*

Here are the visualizations of the graph made by those codes :

- Synergy Graph

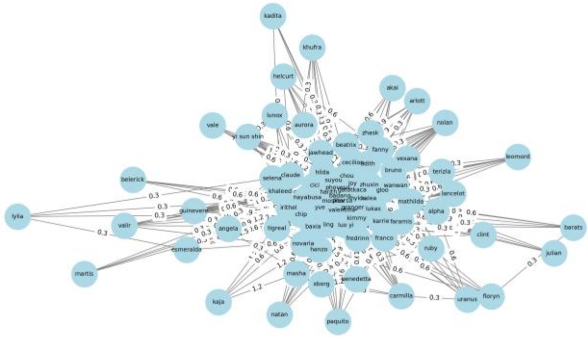

Figure 11. Synergy Graph - Complete
Source: From author’s implementation

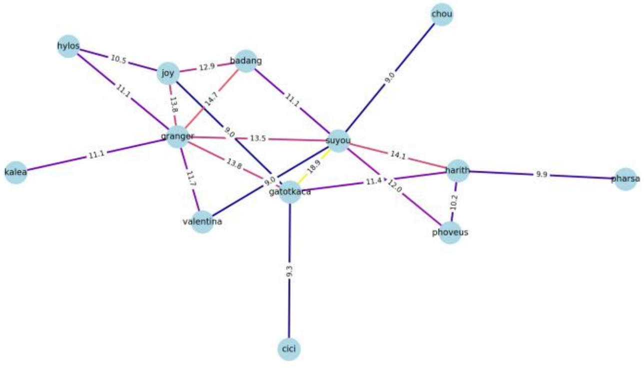

Figure 12. Synergy Graph – Top 20 Edges
Source: From author’s implementation

- Team Synergy Graph – ONIC

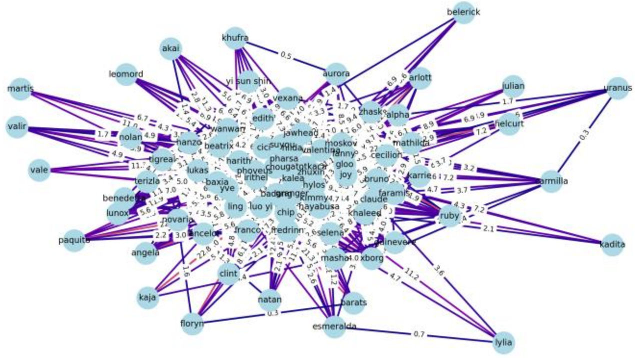

Figure 13. Team Synergy Graph – Complete
Source: From author’s implementation

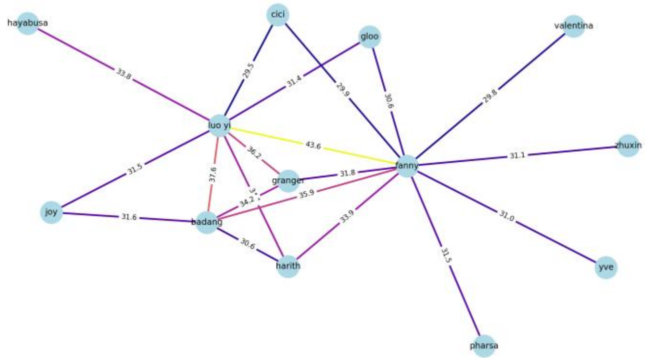

Figure 14. Team Synergy Graph – Top 20 Edges
Source: From author’s implementation

- Counter Graph

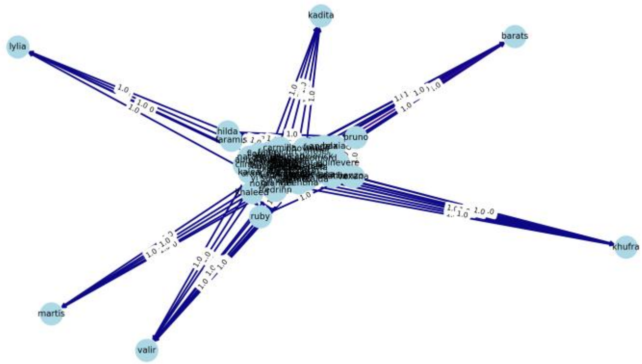

Figure 15. Counter Graph – Complete
Source: From author’s implementation

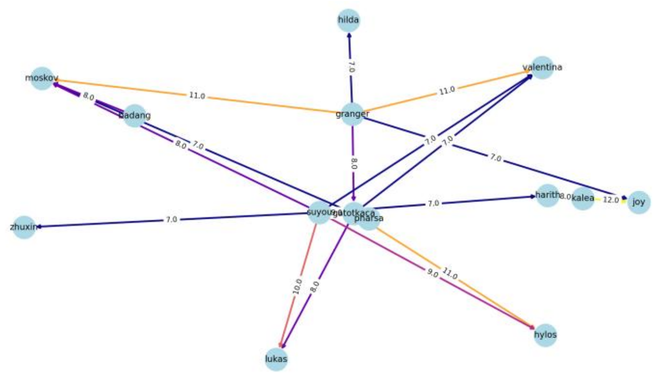

Figure 16. Counter Graph – Top 20 Edges
Source: From author's implementation

2) System Results

To test the system and score its results, a scoring system has been developed. The system will be tested based on its capability in guessing the draft picks of professional teams in a professional scene. There are two parameters that will score the performance of the system, which are hit rate and hit score.

- Hit rate

  Hit rate is the rate at which the system can guess the heroes within its top five recommendations. For example, in the game, the player picks hero A, if the system manages to recommend hero A in its top five recommendations, that's one point for the system.

$$\text{Hit rate} = (\text{System hits}/\text{Total hits}) * 100\%$$

- Hit score

  For each hits, there are scores based on the placement of the hero in the recommendation. If hero A is placed in the first place in the top 5, the system gets five points, if hero A is placed in the second place, the system gets four points, and so on.

$$\text{Hit score} = (\text{System score}/\text{Total score}) * 100\%$$

To make sure there isn't a bias, the system will be scored on two conditions, which are the general perspective and the winner's perspective.

The match used to test the system is the grand final match of MPL ID S15, ONIC vs RRQ. This is the result of the system:

Game	Pers	Hit/Miss	Amount	Total	Amount
Game 1	Gen	Hit	zhuxin-5, chip-5, fanny-5, lukas-5, hayabusa-5, harith-4, joy-3, lancelot-2	Total Hit Score	34
		Miss	wanwan, pharsa, yve, luo yi, kalea, baxia, moskov, gloo, yss, ling, vexana, terizla	Total Miss	12
	Win	Hit	zhuxin-5, chip-5, lukas-5, harith-4, lancelot-2	Total Hit Score	21
		Miss	wanwan, kalea, gloo, yss, vexana	Total Miss	5
Game 2	Gen	Hit	chip-5, lukas-5, zhuxin-5, luo yi-5, fanny-5, pharsa-1, badang-5, joy-2, harith-4, yve-1, irithel-1, franco-3	Total Hit Score	42
		Miss	kalea, hayabusa, hylos, gloo, cici, tigreal, selena, baxia	Total Miss	8
	Win	Hit	chip-5, lukas-5, zhuxin-5, pharsa-1, badang-5, joy-2, irithel-1	Total Hit Score	24
		Miss	tigreal, selena, baxia	Total Miss	3
Game 3	Gen	Hit	chip-5, lukas-5, badang-5, luo yi-5, fanny-5, pharsa-2, kalea-1, joy-2, harith-3,	Total Hit Score	50

			gloo-3, cici-1, valentina-3, gatotkaca-5		
		Miss	wanwan, baxia, irithel, masha, tigreal, khaleed, phoveus	Total Miss	7
	Win	Hit	badang-5, luo yi-5, fanny-5, kalea-1, harith-3, gloo-3, cici-1, valentina-3	Total Hit Score	26
		Miss	baxia, phoveus	Total Miss	2
Game 4	Gen	Hit	chip-5, luo yi-5, badang-5, fanny-5, gatotkaca-1, granger-5, hayabusa-4, hilda-1, harith-5, irithel-4	Total Hit Score	40
		Miss	kalea, baxia, pharsa, valentina, fredrinn, masha, cici, gloo, benedetta, tigreal	Total Miss	10
	Win	Hit	luo yi-5, badang-5, fanny-5, granger-5, harith-5	Total Hit Score	25
		Miss	pharsa, fredrinn, cici, benedetta, tigreal	Total Miss	5
Game 5	Gen	Hit	luo yi-5, badang-5, fanny-5, lukas-5, chip-5, kalea-1, pharsa-2,	Total Hit Score	55

			granger-4, joy-3, harith-5, zhuxin-5, hayabusa-5, yve-1, ling-4		
		Miss	yss, baxia, gloo, cici, tigreal, phoveus	Total Miss	6
	Win	Hit	luo yi-5, badang-5, fanny-5, kalea-1, granger-4, joy-3, hayabusa-5, yve-1	Total Hit Score	29
		Miss	cici, phoveus	Total Miss	2
Game 6	Gen	Hit	chip-5, luo yi-5, badang-5, fanny-5, pharsa-2, granger-4, valentina-2	Total Hit Score	28
		Miss	joy, kalea, baxia, chou, tigreal, wanwan, ling, yss, cici, masha, gloo, benedetta, alpha	Total Miss	13
	Win	Hit	chip-5, pharsa-2	Total Hit Score	7
		Miss	joy, kalea, chou, wanwan, ling, yss, gloo, alpha	Total Miss	8
Game 7	Gen	Hit	fanny-5, luo yi-5, badang-5, chip-5, pharsa-2, granger-4, lukas-5, gatotkaca-4, hayabusa-3, suyou-5	Total Hit Score	43

		Miss	kalea, joy, baxia, chou, yve, cici, alpha, ling, yss, claude	Total Miss	10
	Win	Hit	chip-5, pharsa-2, lukas-5, suyou-5	Total Hit Score	17
		Miss	kalea, joy, chou, cici, yss, claude	Total Miss	6
Hit Rate General					52.86%
Hit Score General					41.71%
Hit Rate Winner					55.71%
Hit Score Winner					33.11%

Figure 17. System Result Table
Source: From author's implementation

V. Conclusion

This system is the first prototype of its series, therefore, there are mistakes that can be fixed to further improve the accuracy of the system. Such mistakes are:

- No parameters to take surprise picks and deny picks into account
- Lack of data to train the system
- No countermeasure for the fact that MPL ID S15 had a patch update in the middle of the tournament league
- The simplicity of the formulas used to count the weight, it would be better to have better formulas to count the weight that takes more parameter into account.

Despite the errors and possible improvements, the system does function well to help players think of possible picks in their games. With more generalization, players of all ranks can use this system to help them do draft picks like a professional.

Moving forward, the next iterations of the system will aim to address these limitations with more sophisticated data analytics, incorporating machine learning to make the system more dynamic, and expand the dataset to include more tournaments. Real-time adaptation to meta shifts is also in consideration to make the system more sustainable. These improvements are expected to significantly increase the system's accuracy and relevance, making it valuable for not only players on pro scenes, but also general players.

VI. Appendix

Video Link at Youtube

https://youtu.be/xD7igfehECA
Duration : 28 minutes and 3 seconds

Video Link at Google Drive for Better Quality

https://drive.google.com/drive/folders/1z1tn_MynUtG1aarx3eqE5iDNL-9NiV0R?usp=sharing

Code Implementation

https://github.com/alivovue/Tugas-Matdis-Graf-Draft-Pick.git

Acknowledgment

The author expresses heartfelt gratitude to the Almighty God for granting the strength and opportunity to complete this paper. Deep appreciation is also extended to Mr. Arrival Dwi Sentosa, S.Kom., M.T., lecturer of K02 IF1220 Discrete Mathematics, for his invaluable guidance and support throughout the development of this work. The author would also like to thank the organizers of MPL Indonesia for providing publicly available match data, which served as the foundation for this research. Congratulations are in order for the winning team, whose outstanding achievement reflects dedication, preparation, and high-level execution throughout the tournament.

References

[1] Liquipedia, "Mobile Legends: Bang Bang Professional League Indonesia Season 15," Liquipedia.net, Accessed: June 20, 2025. [Online]. Available: https://liquipedia.net/mobilelegends/MPL/Indonesia/Season_15 [Accessed : June 20, 2025]

[2] MLBB Indonesia, "MPL Indonesia Season 15 – Grand Finals," YouTube, July 16, 2025. [Online]. Available: https://www.youtube.com/watch?v=8wOi4vseeKM&t=14118s [Accessed : June 20, 2025]

[3] C. Chapple, "Mobile Legends revenue passes $500 million as Southeast Asia powers explosive growth," Sensor Tower, Jan. 2020. [Online]. Available: https://sensortower.com/blog/mobile-legends-revenue-500-million [Accessed : June 20, 2025]

[4] Moonton Games, "About Us," Moonton, accessed Jun. 20, 2025. [Online]. Available: https://en.moonton.com/about/index.html#:~:text=Mobile%20Legends%3A%20Bang%20Bang%20is [Accessed : June 20, 2025]

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, 20 Juni 2025

Audric Yusuf Maynard Simatupang - 13524010