Using Backtracking Algorithm To Determine An Effective Team Composition In Pokémon White

Muhammad Galih Raihan Ramadhan / 13519017
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13519017@std.stei.itb.ac.id

***Abstract*—Video games has been a hobby that a lot of people has grown to love, one of the biggest franchise in gaming is Pokémon, which has existed since the release of Pokémon Red and Pokémon Blue in 1996. In the game, player are given the freedom to make a team out of the existing Pokémon in the game and use them to go through the game, and here we will use the backtracking algorithm to make one that is effective in the context of Pokémon White.**

***Keywords*— Pokémon, backtracking, stats**

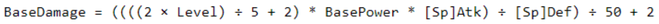
I. Introduction

Pokémon White is a Nintendo DS game from the widely known Pokémon franchise. The game itself follows the traditional gameplay of previously established cores series Pokémon games while taking place in a new region known as Unova.[1]

In the game player will start with a choice of 3 Pokémon, and from there they are set to roam the entirety of the Unova region as they progress throughout the game. Progress of the game is marked by items called “Gym Badge”, an item that can only be obtained by beating a certain NPC or Non-Playable Characters called a “Gym Leader”, each Gym Leader gives out a unique Gym Badge and each of them may only be beaten once. There are a total of 8 Gym Badges in the game, all of which must be collected if a player were to finish the game. After collecting every gym badges the player must then challenge the “Elite Four”, a set of 4 NPCs, each having a role quite similar to that of a Gym Leader, and defeat “Team Plasma”, the main antagonist of the game’s story to finish the game.

Defeating every NPC in this game will require the player to challenge them to a so called “Pokémon Battle”, where the player will pit their Pokémon against the NPC’s Pokémon and whichever has all of their Pokémon knocked out first will be the loser, and the other side will be declared as the winner. A Pokémon’s prowess in battle can be measured using their stats, their move pool, and their typing in accordance with the enemy’s Pokémon. Stats determines the raw power of a given Pokémon. Every Pokémon in Pokémon White has 6 stats, those being HP, Attack, Special Attack, Defense, Special Defense, and Speed. HP determines how much punishment a Pokémon can take before they are knocked out, when a Pokémon’s HP reaches 0 they are knocked out. Attack and Special Attack are used to measure the offensive capability of a Pokémon. They are separated into 2 categories, because every offensive move in the game is either a normal attack, where the power or in other words, how much that attack will decrease the enemy’s Pokémon HP is scaled up with the Attack stat of the Pokémon, or a special attack, where it works the same way as a normal attack, but it is scaled up with the Special Attack stat instead of the Attack stat of the Pokémon.

Defense and Special Defense works to mitigate the power of the attack that is coming from the enemy’s Pokémon. The Defense stat will scale down the power of the enemy’s attacking move if it is a normal attack, while the Special Defense stat will scale down the power of the enemy’s attacking move if it is a special attack. Below is the actual equation used to determine the power of an attack in the game, where A is either the Attack or the Special Attack stat, and D is either the Defense or Special Defense stat. It needs to be noted that the equation will not be discussed in length as it is unnecessary to understand it.



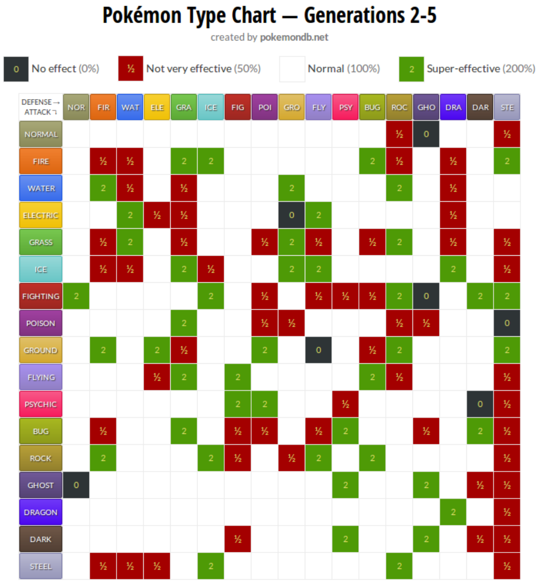
```
BaseDamage = ((((2 × Level) ÷ 5 + 2) * BasePower * [Sp]Atk) ÷ [Sp]Def) ÷ 50 + 2
```

Image 1.1. The Damage Equation [2]

The Speed stat determines which Pokémon gets to move first in any given turn. A Pokémon battle is segmented into turns, where every party involved in the battle may take exactly 1 action in any given turn. If 1 or more parties were to use a move, then the Pokémon with the higher Speed stat will execute their move first. This is a basic summarization of the battle system used in the game and will suffice as a foundation of this paper.

A move pool is a select collection of moves each Pokémon can choose during battle. Each Pokémon has a maximum capacity of knowing Pokémon 4 moves at any given times, however it is possible for them to know less than 4, but not less than 1. Every offensive move in the game has exactly 1 type, which will determine their effectiveness against the defending Pokémon.

A Pokémon type determines the type of the Pokémon and how effective they may be while fighting against another Pokémon. Below are the type effectiveness chart of every type that exist in the game.



Pokémon Type Chart — Generations 2-5

created by pokemondb.net

0 No effect (0%) ½ Not very effective (50%) Normal (100%) 2 Super-effective (200%)

DEFENSE → ATTACK ↴	NOR	FIR	WAT	ELE	GRA	ICE	FIG	POI	GRO	FLY	PSY	BUG	ROC	GHO	DRA	DAR	STE
NORMAL													½	0			½
FIRE		½	½		2	2						2	½		½		2
WATER		2	½		½				2				2		½		
ELECTRIC			2	½	½				0	2					½		
GRASS		½	2		½			½	2	½		½	2		½		½
ICE		½	½		2	½			2	2					2		½
FIGHTING	2					2		½		½	½	½	2	0		2	2
POISON					2			½	½				½	½			0
GROUND		2		2	½			2		0		½	2				2
FLYING				½	2		2					2	½				½
PSYCHIC							2	2			½					0	½
BUG		½			2		½	½		½	2			½		2	½
ROCK		2				2	½		½	2		2					½
GHOST	0										2			2		½	½
DRAGON															2		½
DARK							½				2			2		½	½
STEEL		½	½	½		2							2				½

Image 1.2. Type Effectiveness Chart [3]

The rows represent the type of the attack that is being executed, while the columns represent the type of the Pokémon that is receiving the attack. The number that is in the corresponding cell represent a factor by which the attack power will be scaled to. For example, an attack of the Fire type will have its power multiplied by 2 if used against a Grass type Pokémon, but if it's used against a Water type Pokémon, it will have its power cut in half. There's also a lot of cases of Pokémon having dual typing, in which case the damage calculation is the multiplication of every factor of the type effectiveness. For example, a Fire type attack will not have its power modified if it's used against a Water and Grass type Pokémon. A Pokémon may only have 1 or 2 types.

While playing the game, the players can capture any "Wild Pokémon", which are Pokémon that does not belong to the player nor any NPC. The player may then put the said Pokémon in their team and use them for future battles. The player's team however, has a maximum capacity of 6 Pokémon and also cannot be empty, which is why the player starts the game with a choice of 3 Pokémon which will serve them as the first member of their team. Every Pokémon that is captured while the player has a full team will be stored in a "PC" which then the player may use if they want to use the captured Pokémon instead of what are currently in their team.

II. Backtracking Algorithm

Backtracking algorithm is a type of algorithm that works quite similar to another type of algorithm called *exhaustive search*, where an algorithm will try every possible solution that exists in the *solutions' space* until it gets a solution. Backtracking improves upon this concept, where an answer that does not yield to a solution will not be checked and the algorithm will move on to the next possible solution. This algorithm was first introduced by D. H. Lehmer in 1950, which was then developed further by R. J Walker, Golomb, and Baumert. [4]

A. Solution

The solution of a backtracking algorithm is usually expressed in the form of an *n-tuple* $X = (x_1, x_2, \ldots, x_n)$, where $x_i \in S_i$. In most cases, the value of S_i doesn't change no matter what number i may be. For example, say we use this algorithm to solve a knapsack 1/0 problem, and we yield the solution that between the 3 items that the knapsack may hold, the first and last item should be carried. In this case, then X would have the value $(1, 0, 1)$ and every S_i would have the value of $\{0, 1\}$.

B. Building Function

A building function in a backtracking algorithm refers to a function that is responsible for creating the appropriate next value of an answer.

C. Bounding Function

A bounding function in a backtracking algorithm refers to a function that is responsible for evaluating an answer and dictates if further evaluation of said answer should be done or not.

D. Constructing A Solution

The most common way to construct a solution using the backtracking algorithm is by using a tree that symbolizes the solution space that is possible.

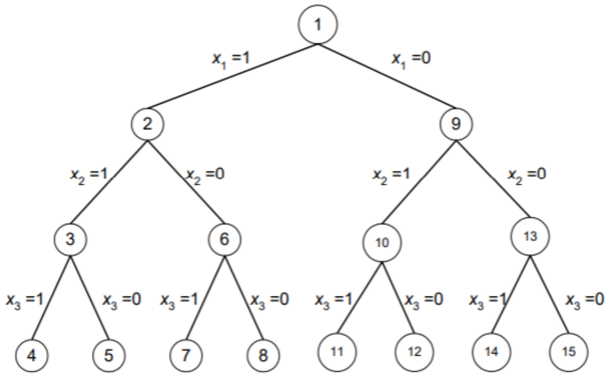


Image 2.1. An Example Of A Tree For A Knapsack 1/0 Problem [4]

The algorithm will go through the tree, evaluating every leaf until it finds a solution or solutions. Since the backtracking algorithm ignores every leaf that does not lead to a solution, then those leaves are effectively "killed" and so the effective appearance of the tree will also change, where it only shows the leaves that were checked by the algorithm.

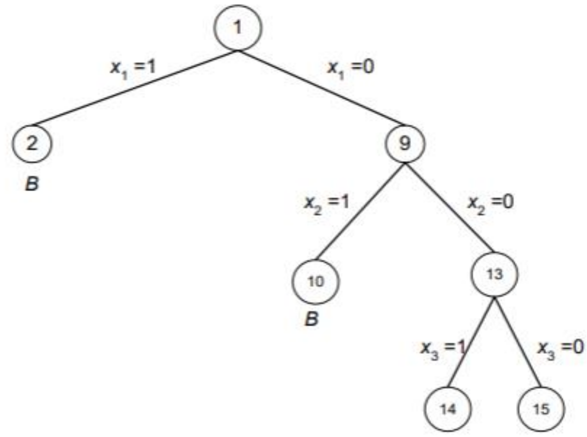


Image 2.2. One Possible Effective Appearance Of Image 2.1. Through A Backtracking Algorithm

III. Making The Team

As stated before, a team is a collection of 6 or less Pokémon (but not less than 1) that the player can utilize in a Pokémon battle. Due to the rules of the game, no one single Pokémon is the "best" Pokémon, because of the existence of the rules that governs the interaction of each types in the game, therefore an ideal team is one that has a Pokémon with an advantageous for each type that exist in the game.

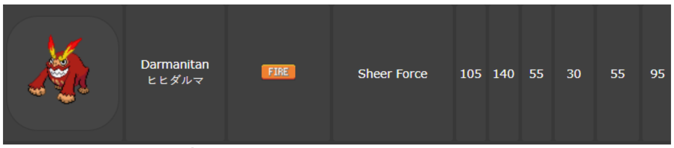
A. Constraints And Assumptions Used And Why

In order to not make the algorithm overly complex, for this paper, certain constraints and assumptions will be used while making the team.

- The algorithm does not take into account how much effort a player must do to acquire a certain Pokémon. This point exists because certain Pokémon can only be obtained after a certain amount of progress has been made in the game or a certain level has been reached by a Pokémon.

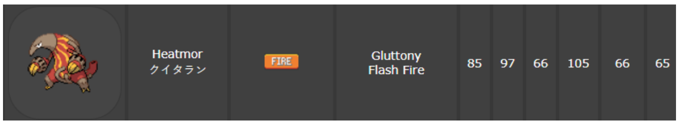
- Every Pokémon has a number of offensive moves that corresponds to their types. For example, a Fire type Pokémon will always have a Fire type offensive move and a Water and Ground type Pokémon will always have a Water type offensive move and also a Ground type offensive move. This assumption is made so that every Pokémon in the team would have an offensive advantage if their typing is advantageous towards the enemy's Pokémon.

- Every combination of types of Pokémon that exists in the game will only be represented by 1 Pokémon in the solution space. This point exists to reduce redundancy as some Pokémon have the same types while being completely different Pokémon, in which case the Pokémon with the highest base stat will be part of the solution space. Using the Pokémon Darmanitan and Heatmor for example, since both of them are of the single type Fire Pokémon variety, and Heatmor has a higher total base stat compared to Darmanitan, therefore Darmanitan will not be considered as part of the solution space, while Heatmor will.



	Darmanitan ヒヒダルマ	FIRE	Sheer Force	105	140	55	30	55	95

Image 3.1. The Pokémon Darmanitan, Its Type, And Its Base Stats [5]



	Heatmor クイタラン	FIRE	Gluttony Flash Fire	85	97	66	105	66	65

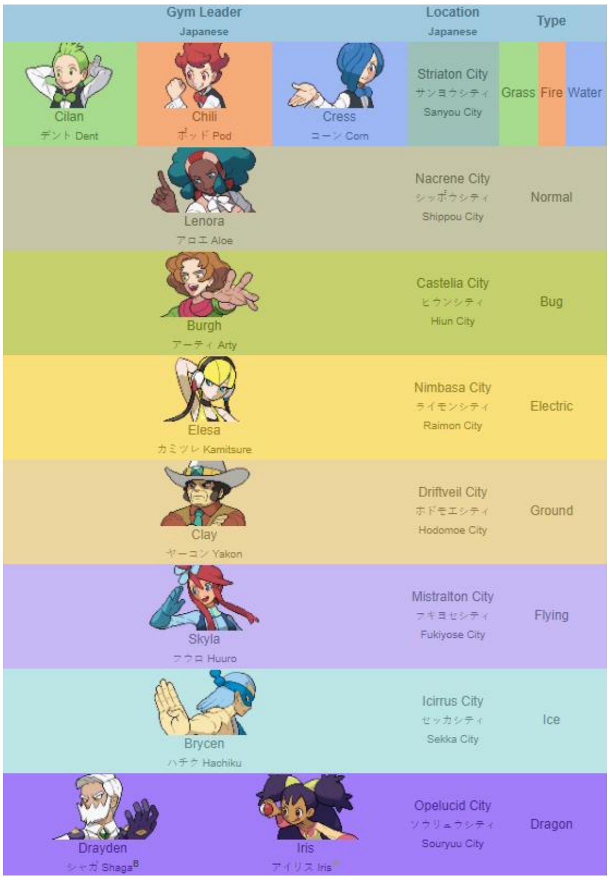
Image 3.2. The Pokémon Heatmor, Its Type, And Its Base Stats [5]

- The solution space will not include "Legendary" and "Mythical" Pokémon. Legendary Pokémon are Pokémon that can only be fought once in the game and therefore counts as the rarest Pokémon a player can get in the game, while a mythical Pokémon is a Pokémon that can only be obtained through special means, usually from outside the game. Since Pokémon White is a fairly old game by the time of the writing of this paper, there is no possible way to obtain these mythical Pokémon in the game through legal means. The reason of the exclusion of these classes of Pokémon is because they are usually much stronger than the "normal" Pokémon and are also much harder to obtain.

B. A Good Team

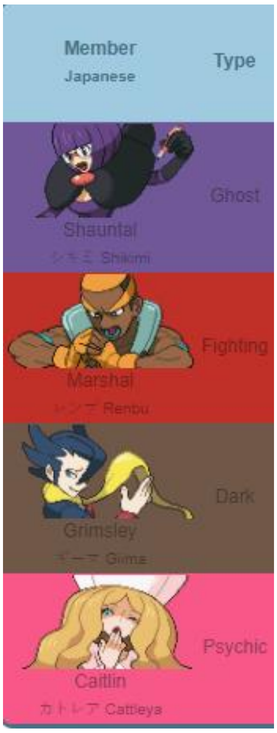
As stated before, and ideal team would be one that has a Pokémon with an advantageous for each type that exist in the game, or in short, the team "covers" for every type in the game, however since a team can only have a maximum of 6 Pokémon, this goal might not be possible. Therefore, for this algorithm, a team that covers for every Gym Leader and Elite Four that exists in the game would qualify as a solution.

Every Gym Leader and Elite Four has a type that they specializes. If a team manages to cover for all of this type, then it is accepted as a solution. The types that must be covered by the team then are the Grass, Fire, Water, Normal, Bug, Electric, Ground, Flying, Ice, Dragon, Ghost, Fighting, Dark, and Psychic types.



Gym Leader Japanese			Location Japanese	Type		
Cilan デント Dent	Chili ポッド Pod	Cress コーン Corn	Striaton City サンヨウシティ Sanyou City	Grass	Fire	Water
	Lenora アロエ Aloe		Nacrene City シッポウシティ Shippou City	Normal		
	Burgh アーティ Arty		Castelia City ヒウンシティ Hiun City	Bug		
	Elesa カミツレ Kamitsure		Nimbasa City ライモンシティ Raimon City	Electric		
	Clay ヤーコン Yakon		Driftveil City ホドモエシティ Hodomoe City	Ground		
	Skyla フウロ Huuro		Mistralton City フキヨセシティ Fukiyose City	Flying		
	Brycen ハチク Hachiku		Icirrus City セッカシティ Sekka City	Ice		
Drayden シャガ Shaga[B]		Iris アイリス Iris	Opelucid City ソウリュウシティ Souryuu City	Dragon		

Image 3.3. Gym Leaders And Their Specialized Types
Cilan, Chili, And Cress Are Counted As Their Own Individual Gym Leader [6]



Member Japanese	Type
Shauntal シキミ Shikimi	Ghost
Marshal レンブ Renbu	Fighting
Grimsley ギーマ Gima	Dark
Caitlin カトレア Cattleya	Psychic

Image 3.4. The Elite Four And Their Specialized Types [7]

C. *Solution Space*

The solution space of this algorithm consists of the Pokémon that exist in the game that follows our given constraint. The solution space is then given as,

TABLE I. SOLUTION SPACE

No	Name	Type 1	Type 2	Total Stat
1	Serperior	Grass	-	528
2	Simisear	Fire	-	498
3	Emboar	Fire	Fighting	528
4	Samurott	Water	-	528
5	Stoutland	Normal	-	490
6	Liepard	Dark	-	446
7	Reuniclus	Psychic	-	490
8	Braviary	Normal	Flying	510
9	Eelektross	Electric	-	515
10	Gigalith	Rock	-	505
11	Sigilyph	Psychic	Flying	490
12	Drillbur	Ground	-	328
13	Excadrill	Ground	Steel	508
14	Mienshao	Fighting	-	510
15	Seismitoad	Water	Ground	499
16	Leavanny	Bug	Grass	490
17	Scolipede	Bug	Poison	475
18	Krookodile	Ground	Dark	509
19	Crustle	Bug	Rock	475
20	Scrafty	Dark	Fighting	488
21	Cofagrigus	Ghost	-	483
22	Archeops	Rock	Flying	567
23	Garbodor	Poison	-	474
24	Swanna	Water	Flying	473
25	Vanilluxe	Ice	-	535
26	Sawsbuck	Normal	Grass	475
27	Emolga	Electric	Flying	428
28	Accelgor	Bug	-	495
29	Escavalier	Bug	Steel	495
30	Amoonguss	Grass	Poison	464
31	Jellicent	Water	Ghost	480
32	Galvantula	Bug	Electric	472

33	Ferrothorn	Grass	Steel	489
34	Klinklang	Steel	-	520
35	Chandelure	Ghost	Fire	520
36	Haxorus	Dragon	-	540
37	Stunfisk	Ground	Electric	471
38	Golurk	Ground	Ghost	483
39	Bisharp	Dark	Steel	490
40	Hydreigon	Dark	Dragon	600

Data regarding all of the Pokémon is taken from [8]. Since there are 40 solution space for this problem, therefore the solution of the algorithm can be written as a *6-tuple* $X = (x_1, x_2, \ldots, x_6)$ where $x_i \in S_i$, every S_i have the value of {1, 2, …, 40}, which corresponds to a Pokémon in Table I. As an example if $X = (1, 3, 7, 32, 19, 22)$, then it would entail that the team consists of Serperior, Emboar, Gothitelle, Galvantula, Crustle, and Archeops.

D. Solving The Problem

Now with the solution space of the problem defined, we can start constructing on how the algorithm will work. The building function will be a Pokémon that has the highest Total Stat, since it can be used as an objective metric for comparing a Pokémon against each other, if their typing were to be ignored. If we find multiple Pokémon with the same amount of base stat, then we will use the one that comes up first in the solution space list.

For the bounding function, we need 2 functions, one is $p(x_1, x_2, \ldots, x_n)$, which determines how many types that team covers, which is the number of types the union of what type each individual Pokémon x_1 to x_n covers. The other function is $f(x_i)$ where,

$$f(x_i) = true, if\ i = 1 \quad (1)$$

$$f(x_i) = false, if\ i > 6 \quad (2)$$

$$f(x_i) = true, if\ p(x_1, \ldots, x_i) > p(x_1, \ldots, x_{i-1}), else\ false \quad (3)$$

As an example for the *p* function, say we have *p(40, 39)*, which means that our team consists of Hydreigon and Bisharp. Hydreigon will have a Dark and Dragon type attacks, which would mean it covers for the Psychic, Ghost, and Dragon types, meanwhile the Bisharp will have a Dark and Steel type attacks, which means it would cover for the Psychic, Ghost, Ice, and Rock types. Now if we were to union the types Hydreigon and Bisharp cover, we would have Psychic, Ghost, Dragon, Ice, and Rock types, which means our team covers for 5 types, 4 of which are types either the Elite Four or the Gym Leader specializes in.

The goal of this algorithm then, can be viewed as finding a leaf that covers the Grass, Fire, Water, Normal, Bug, Electric, Ground, Flying, Ice, Dragon, Ghost, Fighting, Dark, and Psychic types, while being on the sixth or lower level of the tree.

E. Visualizing The Algorithm

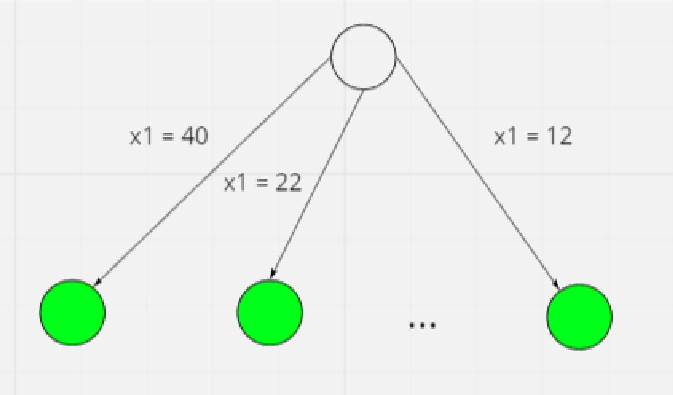


Image 3.5. First Step Of The Algorithm After The Root

For the first step of the algorithm we will simply build every next possible leaf from the root, then we continue by checking the leaf with the highest base stat. In Image 3.5., the leaves has been sorted so the left-most leaf has the highest base stat, while the right-most has the lowest base stat.

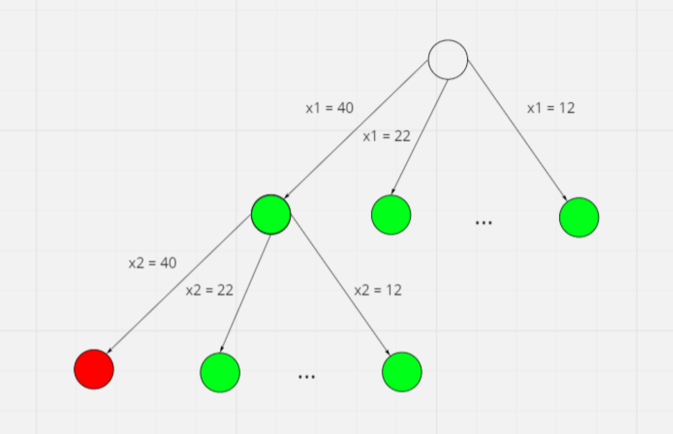


Image 3.6. Second Step Of The Algorithm After The Root

For the second step, we will build upon the leaf with the highest total stat. Now, from this leaf, we will find a rejected leaf, since having a second Hydreigon (x_i = *40*) does not help the team to cover more types, therefore we will then try to build upon the next possible solution.

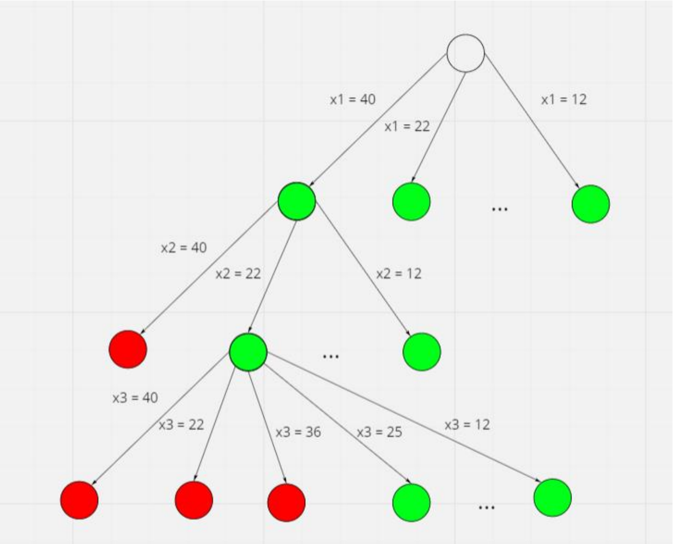


Image 3.7. Third Step Of The Algorithm After The Root

IV. Possible Improvements

This algorithm can be improved upon quite easily by examining the nature of Pokémon in the game itself. An attribute of Pokémon has been skipped over as too not make the algorithm overly complex which are called "Ability". An Ability is a special attribute that can either make a Pokémon better or worse, for example Eelektorss has the Ability Levitate that will make it immune to Ground type attacks, which would render with no weakness, while Archeops has the Ability Defeatist which would half all of its stats that are not HP when their HP reaches half or less than their maximum HP. In short, Ability can make a Pokémon better or worse, which is one way to improve this algorithm so it yields a better team on average.

Another simple way to improve this algorithm is to take a look at each Gym Leader and Elite Four Pokémon in their team and create a team that would "counter" all of them. This would acquire acute knowledge of the game and how it works, and therefore might not feasible for an algorithm to gain a good solution unless it repeatedly fight against them until it gets a good solution. Another way would be to make a team that covers for every types or even every Pokémon that exists in the game, however doing this might be impossible with the constraint of only having 6 Pokémon in a team.

The final way to improve this algorithm is to take into account what Pokémon the player can acquire at certain point in the game, and since the amount of Pokémon the player can acquire as the game progresses, the team would not be static, rather that it would change over time as better Pokémon are available for the player to get.

V. Conclusion

Backtracking algorithm is an ideal algorithm to use if there's a certain boundary that the solution has to satisfy, in this case that being a solution that covers all of the Gym Leaders' and Elite Four's specialized type and consist of only 6 Pokémon. It is however might not be as good in the time department, and it will only made worse if there's multiple boundary that a solution must adhere to.

Video Link at YouTube

None.

Acknowledgment

Thank you to God for giving me the ability to write this paper, while also having a lot of fun while doing so, given the subject at hand. I would also extent my appreciation to Mr. Rinaldi Munir and Mr. Rila Mandala, both for being exceptional teachers even in the time of the Covid-19 Pandemic, which is when this paper is written. I would also like to thank my friends for giving me ideas for this paper and for being rather supportive in their ways.

References

[1] Taken from https://bulbapedia.bulbagarden.net/wiki/Pokémon_Black_and_White_Versions, accessed on 6 May 2021.

[2] Xfr, Bond697, Kaphotics, and V4Victini, "The Complete Damage Formula for Black & White", taken from https://www.smogon.com/bw/articles/bw_complete_damage_formula#, accessed on 6 May 2021.

[3] ASAP. Great valley, "Pokémon Type Chart (Generations 2-5)", taken from http://gvpokemon101.blogspot.com/2015/02/pokemon-type-chart-generations-2-5.html, accessed on 6 May 2021.

[4] Munir, Rinaldi, "Algoritma runut-balik (bakctracking) (Bagian 1) (Versi baru 2021)", taken from https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/stima20-21.html, accessed on 6 May 2021.

[5] Taken from https://www.serebii.net/blackwhite/unovadex.shtml, accessed on 6 May 2021.

[6] Taken from https://bulbapedia.bulbagarden.net/wiki/Unova_League#Gym_Leaders, accessed on 6 May 2021.

[7] Taken from https://bulbapedia.bulbagarden.net/wiki/Elite_Four#Unova_Elite_Four, accessed on 6 May 2021.

[8] Taken from https://www.serebii.net/blackwhite/unovadex.shtml, accessed on 7 May 2021.

STATEMENT

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, 8 Mei 2021

Muhammad Galih Raihan Ramadhan
13519017