A Time-Efficient Competitive Pokémon Team-Building Algorithm

Hugo Reijm, 4272692

TU Delft

June 30, 2016

Abstract

The Pokémon video game franchise is based on players, also known as trainers, capturing and battling with fictitious creatures called Pokémon. competitive Pokémon battling is a complicated business. Before even stepping into the arena, a trainer must design a competitively viable team of six Pokémon to battle with. This task is a difficult one because there are currently 721 species of Pokémon to construct a team from, and new Pokémon are revealed every year. The author therefore developed the Score-Based Pokémon Analysis algorithm, also known as SBPA, to aid users in easily and quickly designing new Pokémon teams.

The SBPA algorithm relies heavily on user interaction, and therefore requires the user to have already chosen at least one Pokémon to build his/her team around. The algorithm then analyzes the Pokémon the user has already chosen and returns a selection of Pokémon that would best further the development of the team. The user then chooses another team member, and the process is repeated until the user has developed his/her team. The SBPA algorithm analyzes the inputted team using three factors: Base Statistics, Typing, and the Popularity Factor. These three factors are then combined to determine the best potential team members per iteration.

The algorithm focuses not only on user interactivity, but also on time efficiency in order to provide valuable information to the user in a time frame that would be impossible to a technologically unaided trainer. As a cost for its speed and user-friendliness, SBPA is less accurate in determining the absolutely optimal team than certain other algorithms. However, when considering the target market of the algorithm, the author found these compromises worthwhile.

Contents

1 Introduction

Welcome to the Pokémon franchise! The Pokémon franchise is one that has existed since 1996, starting with the release of their first video games in Japan. Since then, it has become one of the world's most successful "children's entertainment properties" in the world [1]. Among other merchandise, the Pokémon franchise specializes in the making of trading cards and related video games [16]. This thesis will focus on the latter.

The author attributes the astounding success the Pokémon video games have had to the incredible amount of variety and complexity present within the games' mechanics. So much potential for variation is present, that it has become difficult for competitive players to decide how to play the game. Therefore, the author developed a computer algorithm to mathematically analyze almost any situation a competitive player finds himself/herself in, and give advice in how to proceed. This algorithm has been named the Scored-Based Pokémon Analysis, or SBPA, algorithm.

This paper will give an in-depth look at the theory behind SBPA and its workings. However, before the algorithm can be fully comprehended, one must first understand the Pokémon games and more importantly the mechanics behind competitive Pokémon gameplay.

2 An In-depth Look at Pokémon Mechanics

The Pokémon video games are set in a world a little different from reality; this world is filled with creatures called Pokémon, short for "Pocket Monster" [10,16]. What does a Pokémon look like? Almost anything. Just like there are many various animal species in the real world, so there are many different species of Pokémon in the Pokémon world. In the case of Pokémon, there are currently 721 different usable species of Pokémon, and more are revealed every year [10]. What makes things even more complicated is that several Pokémon species have multiple forms that they can take on in and out of battle. An example of such a Pokémon species is Species #386, also known as Deoxys. An individual Deoxys can take on one of four forms, shown below:

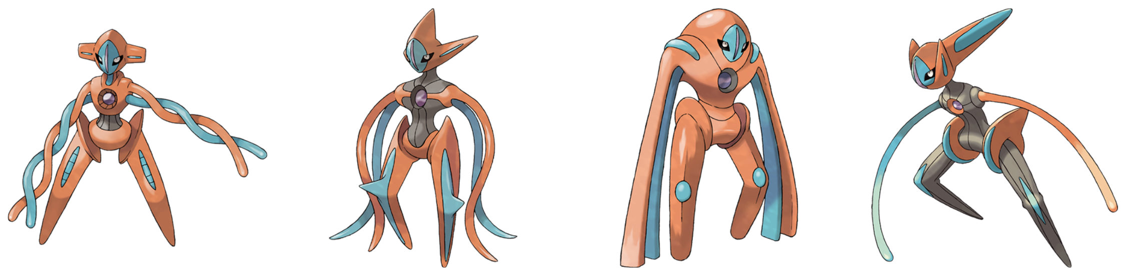


Figure 1: Image credits goes to the Pokémon Company International [6]

Although each form can provide certain benefits to Deoxys that the other forms can not, the forms are still considered to be of the same species [6]. For the sake of simplicity, the author has for the most part ignored these forms in this thesis; however he has taken these into account in SBPA.

The Pokémon shown in Figure 2 will be used as an example from now on:

Figure 2: Image credit goes to the Pokémon Company International [27]

This is a Pokémon nicknamed "Fluffy". He is an individual of Pokémon species #263, named Zigzagoon. Zigzagoons are about the size of a small dog [27].

Players of the Pokémon games can catch and train Pokémon like Fluffy in order to battle other players and their Pokémon. Therefore, players are also referred to as trainers. This concept is what the Pokémon video games center around: the player becoming stronger by catching and battling other Pokémon [16].

Over the years, players from all over the world have started to battle each other in this way. They have strategized and trained their Pokémon to peak performance in order to become the very best. This is called competitive Pokémon battling. This competition has its climax every year during the Pokémon VG World Championships, which was held last year in Boston [20]. The workings of one of these competitive matches will be briefly discussed in the following section.

2.1 A Typical Pokémon Battle

Before a trainer can battle another trainer, he/she must construct a team.

Definition 1. *A **Team** is a collection of six individual Pokémon.*

Teams contain the Pokémon necessary for a trainer to play out his/her battle strategy. Once a trainer has engaged in battle, he/she may only battle with the Pokémon in the one team he/she chose to take into battle [14,17].

Constructing a team requires a considerable amount of thought. First off, the one constructing a team must decide which tiers he/she wishes to select his/her Pokémon team members from.

Definition 2. *Not all Pokémon are made equal; they are ranked according to their power and potential into sets called **Tiers**. In general, a tier is ranked based on the average power and potential of the Pokémon within this tier: the higher the tier's rank, the more powerful the Pokémon within the tier generally are* [7].

Some formats of battle only allow for Pokémon from certain tiers to be used in a team. This is done in order to establish not only fairness in battle, but also allowing for diversity in teams [9]. If a trainer wishes to participate in a certain battle format, he/she must know which Pokémon are allowed to be used in a team and which are not. Generally, a battle format's name is the same as the highest ranked tier from which members are still allowed to participate in battle. All Pokémon from lower-ranked tiers may always participate in these particular battle formats. For the battle formats that the SBPA algorithm focuses on, this is always the case.

SBPA focuses on seven different battle formats in order to give the user a significant amount of choice. The seven different formats are:

- **Anything Goes**, abbreviated here as AG, which allows all 721 Pokémon species to battle, a total of 817 different Pokémon if including Pokémon forms [2]
- **Uber**, which also allows all 721 Pokémon species to battle. However, it forbids a certain form of a certain Pokémon species to battle, resulting in a total of 816 different Pokémon if including Pokémon forms [25]
- **Battle Spot Singles**, abbreviated here as BS, which is the official battle format in the Pokémon games and allows 690 Pokémon species to battle, a total of 753 different Pokémon if including Pokémon forms [4]
- **Over Used**, abbreviated OU, which allows 701 Pokémon species to battle, a total of 760 different Pokémon if including Pokémon forms [13]
- **Under Used**, abbreviated UU, which allows 656 Pokémon species to battle, a total of 687 different Pokémon if including Pokémon forms [26]
- **Rarely Used**, abbreviated RU, which allows 584 Pokémon species to battle, a total of 602 different Pokémon if including Pokémon forms [21]
- **Never Used**, abbreviated NU, which allows 527 Pokémon species to battle, a total of 539 different Pokémon if including Pokémon forms [12]

The author chose to focus on these particular formats due to experience that told him that these were the most common. The graph below visualizes the number of Pokémon species and forms allowed in the seven formats described above:

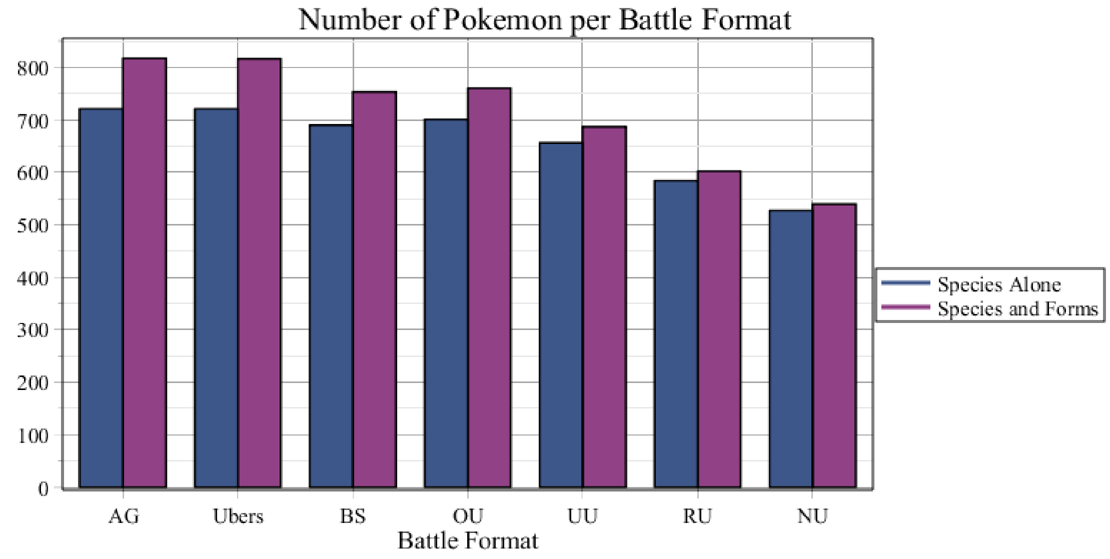


Figure 3:

Obviously, should a trainer want to construct a team for a battle format that allows a large set of Pokémon, the SBPA algorithm will require more time to run to completion than for a battle format that allows a smaller set of Pokémon; more data to analyze requires more time to do so. This fact becomes apparent in later analysis.

Furthermore, most battle formats enact certain clauses. A clause that is very pertinent to Pokémon team building itself is the Species Clause.

Definition 3. *The Species Clause requires any competitive trainer to use six Pokémon of different species in his/her team [9].*

In other words, a trainer may only use our example Pokémon Fluffy once in a team, and no other individual of Fluffy's species may be present in that same team. Out of all the battle formats that SBPA focuses on, only the Anything Goes format does not enact this clause [2]. However, the algorithm still assumes that all of the formats do enact it in order to generate teams with various team members and to preserve a certain uniformity in the coding.

The Species Clause provides a way of estimating the total number of Pokémon teams that could be built. Assuming that any of the 721 species could be used in the construction of a team while ignoring different Pokémon forms, a quick calculation reveals the maximum number of groupings of six different Pokémon species one could put together:

$$\binom{721}{6} = 1.91082439565304 \text{ x } 10^{14}$$

Roughly 200 trillion different groups of six different Pokémon species can be made. Note that this is not the total number of Pokémon teams that can be made; the maximum number of actual teams is far greater. The Pokémon Company designed individual Pokémon to be moderately modifiable. A player can to a certain extent modify every Pokémon in his/her team in order to maximize the team's win-lose ratio. This can be done in several ways, which this thesis will not get into. However, it is clear that individuals from a Pokémon species can be widely differing from each other, and thereby significantly increasing the number of Pokémon teams that could be made [9]. This is the reason why the SBPA algorithm is more complex than just brute-force calculating which team would be best in a certain situation: the runtime to compute this would be far too impractical for such an approach.

For simplicity, SBPA does not focus on every possible Pokémon individual, but rather removes these individualities and analyzes the species as a whole. Remember that a Pokémon is moderately modifiable. This would imply that Pokémon species have a default setting, which happens to be the same for all individuals of that Pokémon species [18]. Therefore, the algorithm focuses on these default settings, thereby still providing viable team-building advice while reducing runtime significantly.

Once a trainer has constructed a team that is allowed by the battle format he/she wishes to participate in, the trainer is automatically matched with an opponent through a computer system. Both trainers start by viewing a basic overview of their opponent's team. In this Team Preview, nothing is revealed about the opponent's team except which Pokémon it contains. Based on this, each player privately decides in which order to send his/her Pokémon into battle. Once both players have made their decisions, the battle begins [11].

A Pokémon battle is a turn-based system. Each turn begins with both players privately selecting their preferred actions for that turn. Once both players have decided what they will do for that turn, the results of the players' decisions is played out and a new turn begins. This cycle is repeated until one player has no more usable Pokémon to battle with, a situation which usually results from all the player's Pokémon being "knocked out" by the opponent's Pokémon. When this happens, the player with no usable Pokémon loses the match [17]. In the rare case that both players lose their last Pokémon simutaneously, the victor is decided by who hit last [9].

This is a basic overview of a competitive Pokémon battle. Many variations in battle style exist, but all can be boiled down to this system.

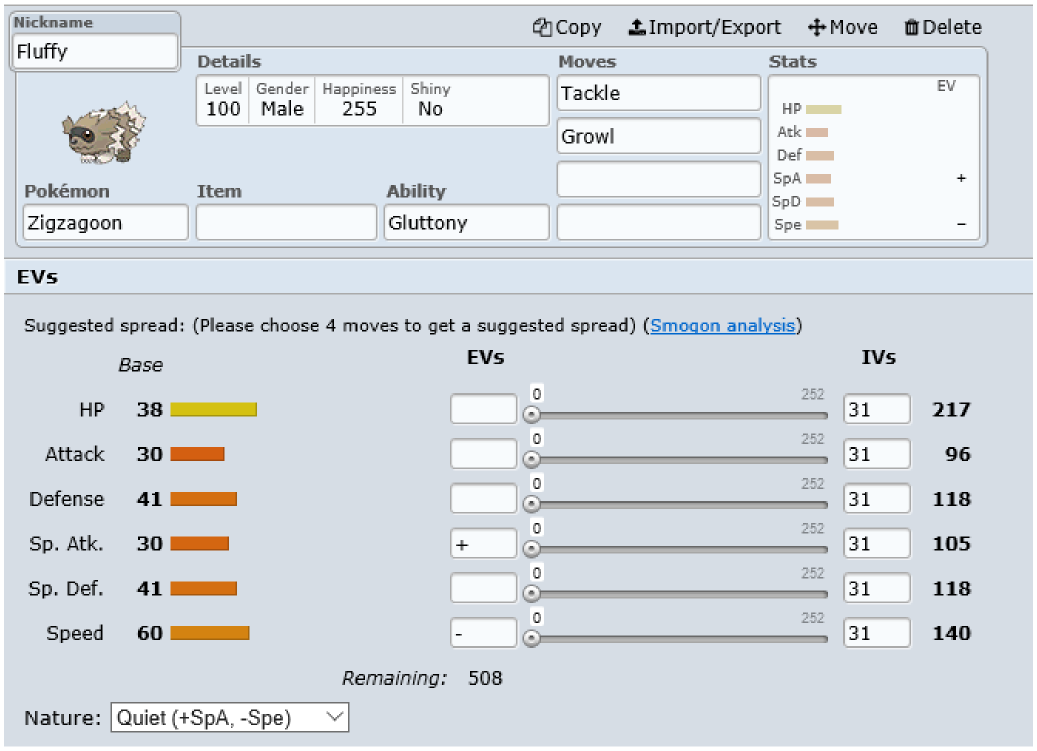
The quality of a Pokémon team could therefore be described as follows. The human element in this case is removed from the definition for simplicity's sake:

Definition 4. *Say that a trainer uses team t in N battles, from which he/she wins n times. Then the quality of team t is n/N, where $n \leq N$ and $N >> 1$*

Therefore, for a team to have a high quality, the trainer must think of all the factors involved in building a team and optimize them. This thesis will analyze a certain set of these factors in a bottom-up fashion, relying on the philosophy that one can not fully understand the whole without understanding all the parts. Therefore, the next few sections describe the mechanics of individual Pokémon, because one can not fully understand a team without understanding all the intricacies of a single Pokémon.

2.2 Individual Pokémon Mechanics

Pokémon, even at an abstract level, are complex creatures with many various characteristics. As per earlier, observe Fluffy. He may look simple, but a further analysis reveals the following:



Nickname: Fluffy

Details

Level	Gender	Happiness	Shiny
100	Male	255	No

Pokémon: Zigzagoon

Item:

Ability: Gluttony

Moves: Tackle, Growl

Stats: HP, Atk, Def, SpA +, SpD, Spe −

EVs

Suggested spread: (Please choose 4 moves to get a suggested spread) (Smogon analysis)

	Base	EVs	IVs	
HP	38		31	217
Attack	30		31	96
Defense	41		31	118
Sp. Atk.	30	+	31	105
Sp. Def.	41		31	118
Speed	60	-	31	140

Remaining: 508

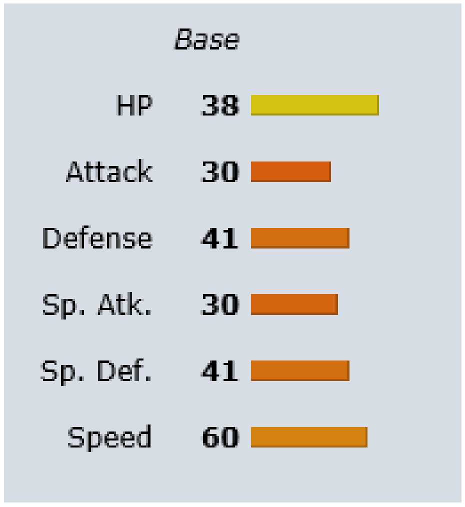
Nature: Quiet (+SpA, -Spe)

Figure 4: Screen shot from Fluffy's custom-made Pokémon Showdown! file, sponsored by Smogon University [18]

This is Fluffy expressed in numbers and figures; a rather large set of mathematical data. Due to time constraints in the developmental phase, the SBPA algorithm does not take into account many of the factors present here. Therefore, SBPA is an algorithm that merely gives advice; it can not calculate ever single situation to result in a perfect optimum. However, SBPA can still give viable advice by using two important factors found in every Pokémon: Base Statistics and Typing.

2.2.1 Base Statistics

Taking another look at Fluffy and his file, observe the following section:



	Base
HP	38
Attack	30
Defense	41
Sp. Atk.	30
Sp. Def.	41
Speed	60

Figure 5: Photo credit goes to Pokémon Showdown!, sponsored by Smogon University [18]

These are Fluffy's Base Statistics.

Definition 5. ***Base Statistics****, also know as "Base Stats", are a set of six integers that determine how well a Pokémon can fill a certain role in a team.*

For example, if a Pokémon team requires a new member that is incredibly fast, then the trainer will have to search for a Pokémon with a very high Speed Stat. Fluffy's Base Stats are not very high when compared to other Pokémon, which accounts for the fact that Fluffy's species does not see much competitive play.

The following definitions shed light on what each Base Stat stands for:

- The **HP** Base Stat stands for the Hit Points that a Pokémon has. Very basically, Hit Points determines how large of an attack the defending Pokémon can survive [23].
- The **Attack** Base Stat stands for the physical power a Pokémon possesses. If a Pokémon were to use a physically attacking move, such as throwing a punch, then the power of that move would be determined by this Base Stat [15,23].
- The **Defense** Base Stat stands for how well a Pokémon can defend itself from physical moves [23].
- The **Sp.Atk.** Base Stat, an abbreviation for "Special Attack", stands for the Special power a Pokémon possess. The word "Special" refers to any move that doesn't include direct physical contact between the attacking and defending Pokémon. Examples of such moves are "Thunderbolt", "Shadow Ball", and "Telekinesis" [22,23].
- The **Sp.Def.** Base Stat, an abbreviation for "Special Defense", stands for how well a Pokémon can defend itself from Special moves [23].

- The **Speed** Base Stat stands for how fast a Pokémon is. In a battle, the Speed Base Stat determines which Pokémon moves first in a turn [23].

A trainer that is building a Pokémon team must keep track of how the Base Stats of the team members interact with each other. Perhaps the trainer wants to make a team that is more aggressive than most teams. That would entail that he/she needs to put more focus on the overall Attack, Special Attack, and Speed of his/her team. Perhaps he/she wishes to make a slower, defensive team. Then the overall HP, Defense, and Special Defense of the team are of more importance.

For simplicity's sake, assume that Base Stats are species-dependent characteristics; that is to say: every individual of a Pokémon species has the same Base Stats. Technically, Base Stats can change from individual to individual due to modifiers, but analyzing these would be too complicated and time-consuming [3]. The SBPA algorithm assumes that Base Stats are species-dependent, thus resulting in the algorithm not looking at every possible individual of every possible Pokémon species, but merely looking a species as a whole. As discussed before, this makes the algorithm much more time-efficient.

2.2.2 Typing

Another species-dependent characteristic is Typing. Every species can have up to two Types, which classify what that Pokémon species is. For example, recall that Fluffy is an individual of Pokémon species #263. All individuals of species #263 have only one Type: the "Normal" Type [27]. This attribute is not shown in Figure 4.

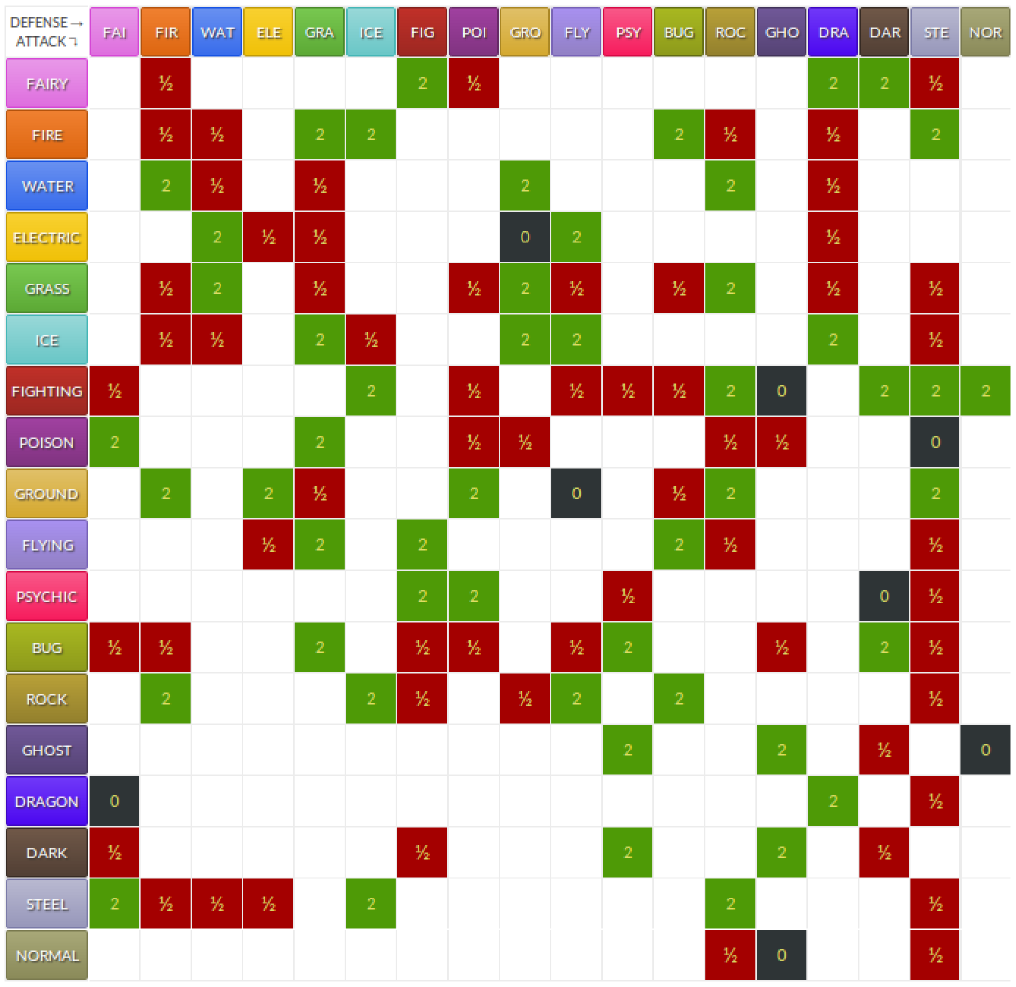
There are 18 Types in total: Bug, Dark, Dragon, Electric, Fairy, Fighting, Fire, Flying, Ghost, Grass, Ground, Ice, Normal, Poison, Psychic, Rock, Steel, and Water. These Types interact with each other in a complex game similar to Rock-Paper-Scissors [24]. The situation is more complicated, however, because of Weaknesses, Resistances, and Immunities. The basic definitions are given below:

Definition. *A Type's* ***Weaknesses*** *are a set of Types. If a Pokémon were to be attacked by an opposing Pokémon, where the Typing of the attacking Pokémon happened to be in the set of Weaknesses of the defending Pokémon, then the attacking Pokémon would do m-times more damage than normal. Usually, m is equal to 2. This would also be known as a Super Effective attack* [5, 24].

Definition. *A Type's* ***Resistances*** *are a set of Types. If a Pokémon were to be attacked by an opposing Pokémon, where the Typing of the attacking Pokémon happened to be in the set of Resistances of the defending Pokémon, then the attacking Pokémon would do n-times less damage than normal. Usually, n is equal to 2. This would also be known as a Not Very Effective attack* [5, 24].

Definition. *A Type's* ***Immunities*** *are a set of Types. If a Pokémon were to be attacked by an opposing Pokémon, where the Typing of the attacking Pokémon happened to be in the set of Immunities of the defending Pokémon, then the attacking Pokémon would do no damage at all. This would also be known as a Not Effective attack* [5, 24].

The following chart shows the weaknesses (green squares), resistances (red squares), and immunities (dark grey squares) of each Pokémon Type, whereby a Pokémon with a Type in the column on the right is attacking a Pokémon with a Type in the row on top.



DEFENSE→ ATTACK↴	FAI	FIR	WAT	ELE	GRA	ICE	FIG	POI	GRO	FLY	PSY	BUG	ROC	GHO	DRA	DAR	STE	NOR
FAIRY		½					2	½							2	2	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	½					2		½		½	½	½	2	0		2	2	2
POISON	2				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	½			2		½	½		½	2			½		2	½	
ROCK		2				2	½		½	2		2					½	
GHOST											2			2		½		0
DRAGON	0														2		½	
DARK	½						½				2			2		½		
STEEL	2	½	½	½		2							2				½	
NORMAL													½	0			½	

Figure 6: The Type Chart found in The Pokémon Database [19]

To further complicate matters, remember that every Pokémon species can have a minimum of one, maximum of two Types. Should a Pokémon have two Types, then those Types could cancel out each other's weaknesses with their resistances and vice versa. However, these Type could also interfere constructively, resulting in a species with a double weakness or a double resistance to a certain Type [24].

Therefore, in competitive Pokémon battling, a trainer must try to form a team where the members

- balance out each other's weaknesses with their resistances.
- have enough variety amongst themselves so that they can deal with as many different Pokémon and their associated Types as possible.

The Pokémon gameplay mechanics discussed above are the predominant characteristics of individual Pokémon that the SBPA algorithm will take into account. As stated previously, this algorithm is by no means perfect; it merely gives advice for the situation at hand. It ignores many gameplay mechanics that are far too complex to program into the algorithm in the allotted time for this project. However, as one shall see below, the algorithm still is able to give valuable information to the user in a time-efficient manner.

3 The Mechanics Behind the Algorithm

Now that the necessary background information has been established and documented, the SBPA algorithm can be described in detail. SBPA is a user interactive, iteration-based computer algorithm written in Java. The author chose not to use Matlab or Maple as the main programming platform because Java is not only more convenient to share with other people, but also because writing the algorithm in Java deemed more useful to the author. This section will describe the mathematical theory behind SBPA; software details will be ignored unless they have pertinence in relation to the mathematics.

At the very beginning, the algorithm requires the user to input which tiers the user wants to use, the style of team the user wants to make (whether it be defensive, balanced, or aggressive), and at least one Pokémon that the user wants to build a team around. The program then has all the necessary information for it to start its first iteration. Every iteration follows the same procedure:

1. Gather the necessary information from the partial team that the user has already put into the algorithm. The program then uses the Species Clause (Definition 3) to remove any potential violators from the list of Pokémon species that the algorithm needs to run through.

2. For every Pokémon in the list determined in the previous step, apply a series of self-designed scoring systems and give an overall score based on these systems.

3. Sort the list from Step 1 in descending order based on the scores found in the previous step.

4. Display the first K results from the sorted list and wait until the user has chosen a Pokémon for his/her team based on the displayed suggestions or his/her own decision.

In other words, SBPA takes into account the partial team that the user already has and suggests the best addition to that team based on a self-designed scoring system. As one can see, SBPA was designed to be very input-driven; it relies heavily on the input of the user to complete an iteration. Once it has scored which Pokémon would be the best addition to the user's team, it waits for further input. This process repeats until the user has assembled a team of six Pokémon, at which point the algorithm no longer searches for more potential team members. Because the user starts by inputting the first Pokémon, the algorithm requires five iterations to help the user generate an entire team.

Attention must be paid to the fact that this methodology of constructing a team could be substituted for another. Notice that the SBPA algorithm does not guarantee the user an absolutely optimal team; other methodologies could ensure this. However, the author chose to use this methodology for a few reasons. First off, he wanted to construct a time efficient algorithm so that runtime would not become a hindrance when users construct a team using his program. But more importantly, he wanted to preserve user interactivity. Even if the author were to use optimization techniques that efficiently found the absolutely optimal team, they would by definition ignore the user's personality, style, and preferences. The author has the following philosophy when concerning himself with trainers and their teams: half the credit of a victory goes to the trainer, and the other half goes to the trainer's team. Together they form one entity. The trainer must therefore know everything about his/her team and play an active role its development. Solving for an absolutely optimal team through perhaps the Simplex algorithm would remove this interactivity from the team building process. The SBPA algorithm could be expanded to include such an optimization technique in the future, but this would only be used to provide more information to the user so that he/she can make the ultimate decision. For all of these arguments, the author chose the methodology described in this thesis.

The accuracy and efficiency of SBPA depend on what scoring systems have been implemented in its coding. However, there is no absolute scoring system for deciding whether a Pokémon is a "good" or "bad" candidate for a team. Therefore, the scoring system can not be based on an absolute score, but must be based on a relative one, comparing Pokémon to each other to find a solution. The overal scoring system takes three factors into account: a Pokémon's Base Stats, a Pokémon's Typing, and a Pokémon's Popularity Factor. These scoring factors build upon each other as depicted in the diagram below:

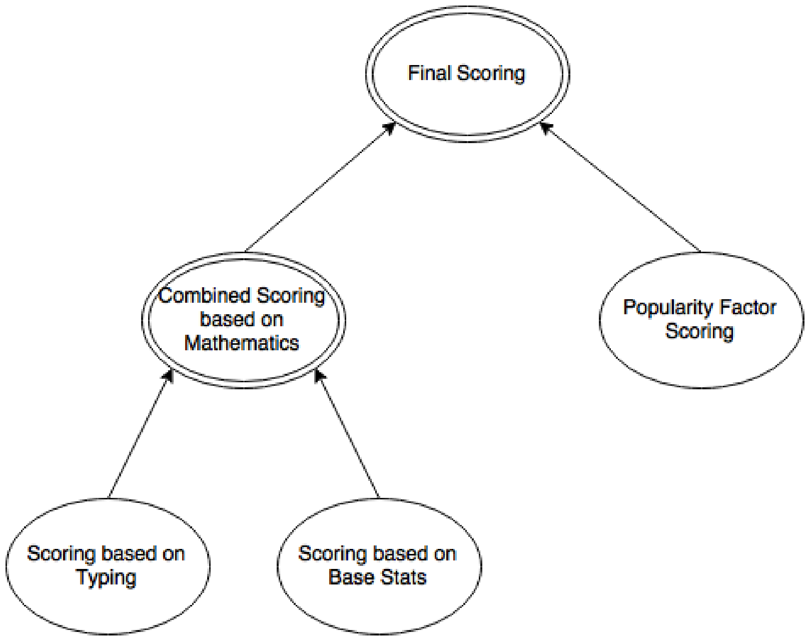

Final Scoring
Combined Scoring based on Mathematics
Popularity Factor Scoring
Scoring based on Typing
Scoring based on Base Stats

Figure 7: Diagram Showing the Composition of the Scoring System

The composition and combination of the scoring factors is explained in the sections below.

3.1 Scoring Based on Base Stats

As discussed earlier, the Base Stats of a Pokémon determine how well that Pokémon can fulfill certain roles in a team. Therefore, a logical approach to determining a score for a Pokémon is to analyze how well it fills the roles that the partial team of the user still needs to fill.

The algorithm does this by using the following equation:

$$S(t,P)_{Stats} = (\mathbf{A}(t \cup P) - \mathbf{A}(T_{max})) \bullet \mathbf{M}_{Stats} \tag{1}$$

Here, $S(t,P)_{Stats}$ stands for the score given to Pokémon P, relative to its Base Stats and the user's already partially constructed team t. The real vector $\mathbf{A}(t \cup P)$ contains the averages of the Base Stats of partial team t with the addition of Pokémon P, where:

- $\mathbf{A}(t \cup P)_1$ is the average HP Base Stat of P and all Pokémon in t
- $\mathbf{A}(t \cup P)_2$ is the average Attack Base Stat of P and all Pokémon in t
- $\mathbf{A}(t \cup P)_3$ is the average Defense Base Stat of P and all Pokémon in t
- $\mathbf{A}(t \cup P)_4$ is the average Special Attack Base Stat of P and all Pokémon in t
- $\mathbf{A}(t \cup P)_5$ is the average Special Defense Base Stat of P and all Pokémon in t
- $\mathbf{A}(t \cup P)_6$ is the average Speed Base Stat of P and all Pokémon in t

The vector $\mathbf{A}(T_{max})$ is constructed in a similar way; however it contains the average Base Stats of the Pokémon in the highest ranked tier T_{max} allowed in the battle format that the user is working with.

Vector $\mathbf{M}_{Stats}$ is also constructed in much the same way as $\mathbf{A}(t \cup P)$. However, rather than each of $\mathbf{M}_{Stats}$'s elements representing the average of a Base Stat, they instead stand for a Base Stats' bias, dependent on the user's preference of team style. Perhaps the user wants to construct a team that, for example, is more aggressive when compared to other teams. In that case, Pokémon with high Attack, Special Attack, and Speed Base Stats are of more importance than Pokémon with high HP, Defense, and Special Defense Stats. The bias vector $\mathbf{M}_{Stats}$ was therefore introduced in order to give the user the ability to choose which Base Stats on average matter more for the user's team. The algorithm gives three options in team style: Defensive, Balanced, and Aggressive. The bias vector is then chosen as follows:

- $\mathbf{M}_{Stats} = [\frac{9}{30}, \frac{1}{30}, \frac{9}{30}, \frac{1}{30}, \frac{9}{30}, \frac{1}{30}]^T$ when the user builds a Defensive team
- $\mathbf{M}_{Stats} = [\frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}]^T$ when the user builds a Balanced team
- $\mathbf{M}_{Stats} = [\frac{1}{30}, \frac{9}{30}, \frac{1}{30}, \frac{9}{30}, \frac{1}{30}, \frac{9}{30}]^T$ when the user builds an Aggressive team

The exact values of these vectors are based on the author's intuition and choice to generate a noticeable difference between team styles. However, the values have been set in such a way that the sum of the elements in $\mathbf{M}_{Stats}$ is always equal to 1. That way, $S(t,P)_{Stats}$ is the weighted sum of the elements of $(\mathbf{A}(t \cup P) - \mathbf{A}(T_{max}))$, which therefore takes into account all the given data while still awarding higher importance to certain Base Stats according to the user's team style selection.

In theory (and also of course in practice), Equation 1 first scores Pokémon P on the impact it would have when added to partial team t, relative to the average Base Stats of tier T_{max}. This score is then combined with the previously discussed bias $\mathbf{M}_{Stats}$ into a dot product to result in a final scalar score. The dot product operator was chosen for two reasons. First of all, the dot product is a simplistic operation that is easy to understand and relatively time-efficient. But more importantly, the dot product allows the scoring to take into account the influence of every Base Stat of P and not just the largest or most important for example. Thereby, SBPA can completely and throughly evaluate P based on its Base Stats, yet still do this in an efficient manner.

3.2 Scoring Based on Typing

The SBPA algorithm also scores a Pokémon based on its Typing. Keeping track of a team's overall Typing is important so that any glaring weaknesses or excessive investment in a certain resistance can be avoided. Therefore, Pokémon are scored based on the weaknesses, resistances, and immunities that the user's partial team already has.

In this case, the scoring uses the equation below, which again is based on vectors and dot products:

$$S(t,P)_{Type} = (\mathbf{T}(t \cup P) - \mathbf{T}(t)) \bullet \mathbf{M}_{Type} \tag{2}$$

In this case, $S(t,P)_{Type}$ is the score given to a Pokémon P relative to its Typing.

Vector $\mathbf{T}(t)$ is a integer vector that described how well partial team t handles opposing Pokémon of a certain Type. The vector needs to monitor how well t interacts with all Types; since there are 19 Types (18 Pokémon Types and one added NULL Type for computational simplicity), $\mathbf{T}(t)$ is 19 elements long. Every iteration, the elements of $\mathbf{T}(t)$ are reset to zero. Every element $\mathbf{T}(t)_i$ is then calculated as described below.

For every Pokémon j in partial team t:

- $\mathbf{T}(t)_i = \mathbf{T}(t)_i - 1$ if Pokémon j in t is weak to Type i
- $\mathbf{T}(t)_i = \mathbf{T}(t)_i + 1$ if Pokémon j in t is resistant to Type i
- $\mathbf{T}(t)_i = \mathbf{T}(t)_i + 2$ if Pokémon j in t is immune to Type i

The vector $\mathbf{T}(t \cup P)$ is calculated much in the same way, except for the fact that it monitors how well t would handle certain Types if Pokémon P were to be added to it.

The bias vector $\mathbf{M}_{Type}$ is present in this equation to once again dictate what is more important and what is less important. At the moment, the Types are dictated as being equivalently important for simplicity's sake, but a possible update to the program could be the implementation of a bias vector $\mathbf{M}_{Type}$ that changes as the user's team grows. More research in this field is necessary to discover how such a bias vector could be computed.

The Equations 1 and 2 are structured in a similar way, and thus the mathematical theories behind both equations are almost identical. Just as in Equation 1, $S(t,P)_{Type}$ is based on the Typing benefits that Pokémon P could add to partial team t. Once that has been computed, a dot product is taken with the bias vector $\mathbf{M}_{Type}$ in order to efficiently combine all elements of $(\mathbf{T}(t \cup P) - \mathbf{T}(t))$ and $\mathbf{M}_{Type}$ into a scalar value.

3.3 Scoring Based on Popularity Factor

As explained before, a Pokémon isn't defined merely by its Base Stats and Typing. Unfortunately, most other characteristics are time-consuming to program and could not be included in the mathematics of the SBPA algorithm in the allotted time. However, a method was devised to still take a shadow of these characteristics into account: the Popularity Factor $S(t, P)_{Pop}$.

The Popularity Factor is based on the past experiences of other players. Since the Pokémon franchise is one of the most successful of all time, thousands of people battle every day in order to improve their skills and become the very best that they can be [16]. With that many players, certain patterns start to evolve over time, centering around successful strategies. One could compare the world-wide group of competitive Pokémon trainers as an immense, learning computer program; its purpose being to seek out new battle strategies while preserving those of high quality (A team's quality is defined in Definition 4). It would be foolish to not use all the successful strategies that have been found thus far.

The Popularity Factor is not a mathematics-based construct; rather, it is based on collecting as many teams as possible and storing them in such a way that they can later be requested and used in a time-efficient manner. Therefore, the author constructed a integer-valued matrix $\mathbf{P}_{T_f}$ for battle format f's set T_f of allowed Pokémon (For a reminder of what battle formats are, read Section 2.1, starting at Definition 2). In $\mathbf{P}_{T_f}$, every Pokémon allowed by format f has its own separate column and its own separate row. Note that the matrix for the format that allows k Pokémon to battle has k columns and k rows, resulting in k^2 different elements. This is quite a large matrix and thus quite a large portion of data. However, doing so allows for data requests to be completed simply and quickly, as will be shown later on.

At first, all the elements in these matrices are zero. Populating these matrices with useful data is done as follows. Let's say the author finds a Pokémon team t, allowed by set T_f in battle format f, that he wishes to use as data for SBPA. He already has written a program to import any team he wishes into the algorithm. The program scans through t and imports only the names of the Pokémon species present in the team; all other characteristics are not included in order to decrease the algorithm's runtime as much as possible. The program then finds the rows and columns in $\mathbf{P}_{T_f}$ corresponding to each of the members of t, and groups them into sets R_t and C_t respectively. The matrix $\mathbf{P}_{T_f}$ is then updated as follows:

$$\mathbf{P}_{T_f,r,c} = \mathbf{P}_{T_f,r,c} + 1,\ r \in R_t \text{ and } c \in C_t$$

In this way, $\mathbf{P}_{T_f}$ is built to register patterns between Pokémon species in teams. The larger an element of $\mathbf{P}_{T_f}$, the more people use the two Pokémon corresponding to that element in a team, and thus the more likely that the two corresponding Pokémon work well together. What is more, this method does not depend on Pokémon Base Stats or Typing, resulting in a simple method capable of finding effective Pokémon team combinations that would otherwise go unnoticed in the analysis of Base Stats and Typing.

Understanding all this, the Popularity Factor $S(t,P)_{Pop}$ is simple to express for Pokémon P and partial team t allowed in set T_f of battle format f:

$$S(t,P)_{Pop} = \sum_{s \in t} \mathbf{P}_{T_f,s,P} \tag{3}$$

In other words, $S(t,P)_{Pop}$ is simply a measure to see how many times Pokémon P and a member of partial team t have been members of the same team in the past.

3.4 Combining Scoring factors

Combining Equations 1, 2, and 3 takes some thought. As stated earlier, there is no absolute measure for deciding whether Pokémon P is a "good" or "bad" candidate for partial team t. Rather, the final score given to P should be a relative one, comparing P's score to that of all the other candidates.

The author decided to combine the scoring systems in such a way as to meet the following criteria:

1. Results for Equation 2 should be about 1.4 times as important as results from Equation 1. This observation comes from Jenty Heijstek, a competitive trainer who came in fourth place during the Dutch National Video Game Tournament in 2013 [8]. Furthermore, Equations 1 and 2 should be combined into one scoring system based on mathematics alone, with Equation 3 added later (See Figure 7). In this way, compartmentalization occurs, which organizes the scoring systems and allows for the relative ease of implementing new scoring systems to the algorithm in possible later updates.
2. The importance of Equation 3 should decrease as more Pokémon are added to the user's team, while at the same time the importance of the scoring sytem based on Equations 1 and 2 should increase. This is because as a team nears completion, all imperfections and glaring flaws still present in the team need to be compensated for, which is much more easily done when focusing on Base Stats and Typing than when focusing on the Popularity Factor.

3.4.1 Criterion 1

The first part of Criterion 1 can easily be met by performing the following scaling transformations on $S(t,P)_{Stats}$ and $S(t,P)_{Type}$:

$$S'(t,P)_{Stats} = (S(t,P)_{Stats} - \min_P(S(t,P)_{Stats})) * \frac{10}{\max_P(S(t,P)_{Stats}) - \min_P(S(t,P)_{Stats})}$$
$$S'(t,P)_{Type} = (S(t,P)_{Type} - \min_P(S(t,P)_{Type})) * \frac{10}{\max_P(S(t,P)_{Type}) - \min_P(S(t,P)_{Type})}$$

Through this transformation, both $S(t,P)_{Stats}$ and $S(t,P)_{Type}$ are changed to always have a range equal to the interval [0,10] in $\mathbb{Z}$, thereby removing any bias that could result from the two scoring systems having different ranges. Unfortunately, these transformations needs to be performed every iteration of the algorithm since $\max_P(S(t,P)_{Stats})$, $\min_P(S(t,P)_{Stats})$, $\max_P(S(t,P)_{Type})$, and $\min_P(S(t,P)_{Type})$ take on different values for every iteration. However, a computer can easily perform such transformations, so thankfully not much time is lost during this step.

Now define a new scoring system $S(t,P)_{Math}$ that can be expressed as follows:

$$S(t,P)_{Math} = S'(t,P)_{Stats} + 1.4 * S'(t,P)_{Type}$$

The score $S(t,P)_{Math}$, also known as the Math scoring system, is defined as such for a few reasons. First of all, $S'(t,P)_{Type}$ now has 0.4 times more influence than $S'(t,P)_{Stats}$, which is exactly what Criterion 1 demanded. Equally as important is that this definition results in a more thorough SBPA. Notice that the most important operator in this definition is the addition operator. If the algorithm were to be powered by, for example, the root-mean-square operator, SBPA could have the tendency to focus more on Pokémon with extreme scores according to one measure while ignoring the possibly low scores according to the other. This phenomenon occurs because the root-mean-square operator squares both $S'(t,P)_{Stats}$ and $S'(t,P)_{Type}$ and then subsequently adds the results together, resulting in one score possibly being swallowed by the other much larger score. The addition operator does not do this, and in fact gives the Pokémon under investigation a score it well deserves by always keeping $S'(t,P)_{Stats}$'s and $S'(t,P)_{Type}$'s relational importances constant. For this reason, the author chose to use the addition operator in his algorithm.

Now define the final scoring system as follows:

$$S(t,P)_{Final} = F(S(t,P)_{Math}, S(t,P)_{Pop})$$

The function $F(S(t,P)_{Math}, S(t,P)_{Pop})$ must be defined with the help of Criterion 2.

3.4.2 Criterion 2

For much the same reasons as described in the previous section, meeting Criterion 2 can be done by defining $S(t,P)_{Final}$ as follows:

$$S(t,P)_{Final} = F(S(t,P)_{Math}, S(t,P)_{Pop}) = c_1(t) * S(t,P)_{Math} + c_2(t) * S'(t,P)_{Pop}$$

In this case, $S(t,P)_{Final}$ is defined as the weighted sum of $S(t,P)_{Math}$ and $S'(t,P)_{Pop}$, whereby $S'(t,P)_{Pop}$ is a scaling transformation much like the ones performed to $S(t,P)_{Stats}$ and $S(t,P)_{Type}$ to meet Criterion 1. In this case, the author wants to avoid any unintended bias from the possibility that the functions $S(t,P)_{Math}$ and $S(t,P)_{Pop}$ have differing ranges. Since $S(t,P)_{Math}$ is defined on the interval [0,24], $S'(t,P)_{Pop}$ can be defined as follows:

$$S'(t,P)_{Pop} = (S(t,P)_{Pop} - \min_P(S(t,P)_{Pop})) * \frac{24}{\max_P(S(t,P)_{Pop}) - \min_P(S(t,P)_{Pop})}$$

The weights $c_1(t)$ and $c_2(t)$ must then differ as t grows. They are not so much dependent on t itself, but more on $|t|$, or how many members t already contains. Also, since the author wants the importance of $S(t,P)_{Math}$ to increase as the importance of $S'(t,P)_{Pop}$ decrease, and vice versa, $c_2(t)$ was chosen to be equal to $1 - c_1(t)$, dictating that $c_1(t)$, $c_2(t) \in [0,1] \in \mathbb{R}$.

Since $|t| \in [0,6] \in \mathbb{Z}$, $c_1(t)$ and $c_2(t)$ can only assume a limited number of values. Taking into account that the SBPA algorithm can have a maximum of five iterations and that $c_1(t)$ and $c_2(t)$

only need to be defined per iteration, the author decided to define the values of $c_1(t)$ and $c_2(t)$ as follows:

$\|t\|$	1	2	3	4	5
$c_1(t)$	1/3	1/2	2/3	4/5	9/10
$c_2(t)$	2/3	1/2	1/3	1/5	1/10

Besides the author's own intuition after having participated in hundreds of Pokémon battles, several factors went into deciding these constants

- When $|t| = 1$, a trainer usually looks for Pokémon that strengthen a certain strategy he/she has in mind. Although Typing and Base Stats are of importance, using the Popularity Factor is a much better method for finding such interesting and effective Pokémon partners.
- When $|t| = 2$, the algorithm needs to start compensating for unbalanced Typings and Base Stats while still paying attention to what other trainers have made over the years. Roughly at this point, the importance of $S(t,P)_{Math}$ and $S'(t,P)_{Pop}$ are about equal.
- As t continues to grow, the importance of $S(t,P)_{Math}$ grows to approximately 1, while the importance of $S(t,P)_{Pop}$ decreases at the same rate to approximately 0. Less and less does t need to take heed of what other trainers have done in the past; what the team needs more and more is to become whole while compensating for as many weaknesses and flaws as possible.

Of course, the exact values for these constants have been chosen based on the Pokémon battle experience of the author. No hard proof exists that determines whether this selection of values yeilds the best results; further research is needed. However, the earlier mentioned competitive Pokémon battler Jenty Heijstek did look at these constants and the first impressions they gave him were positive [8]. Maybe through a collaborative effort, Mr. Heijstek and the author can fine tune the constants' values to improve the algorithm even more.

All in all, the final scoring system for analyzing a Pokémon P for partial team t is done as follows:

$$S(t,P)_{Final} = c_1(t) * (S'(t,P)_{Stats} + 1.4 * S'(t,P)_{Type}) + (1 - c_1(t)) * S'(t,P)_{Pop}$$

$$S'(t,P)_{Stats} = (S(t,P)_{Stats} - \min_P(S(t,P)_{Stats})) * \frac{10}{\max_P(S(t,P)_{Stats}) - \min_P(S(t,P)_{Stats})}$$

$$S'(t,P)_{Type} = (S(t,P)_{Type} - \min_P(S(t,P)_{Type})) * \frac{10}{\max_P(S(t,P)_{Type}) - \min_P(S(t,P)_{Type})}$$

$$S'(t,P)_{Pop} = (S(t,P)_{Pop} - \min_P(S(t,P)_{Pop})) * \frac{24}{\max_P(S(t,P)_{Pop}) - \min_P(S(t,P)_{Pop})}$$

4 Run-Time Analysis

The SBPA algorithm is a much more time-efficient method of giving suggestions for Pokémon team building than, say, brute force calculating the best teams. Brute forcing the problem has its merits of course, the most predominate being the ability to specifically calculate the absolute best teams. However, doing such a calculation would cost so much time that this methodology loses its validity rather quickly. But the question still remains: how fast is SBPA?

To answer this question, the author performed a multitude of test runs and recorded the run time of each test. The tests were designed as follows:

1. Choose a team style (Defensive, Balanced, or Aggressive. See Section 3.1). Do this merely for consistency's sake; the choice of team style should not affect the elapsed runtime of the algorithm.
2. Choose a battle format f.
3. Choose beforehand a team of six Pokémon at random and input the team into SBPA one Pokémon at a time, recording the elapsed runtime for each iteration of the algorithm. In this way, the algorithm still suggests new team members, but the suggestions are ignored as the team has already been established. This doesn't affect the algorithm at all; it still does exactly what its supposed to, exactly how its supposed to.
4. Repeat the third step 10 times, each time inputting a new random team one member at a time.
5. Repeat steps 1-5 for each battle format not equal to f. The resulting test results will give an indication of how quickly the algorithm runs to completion per battle format.

For his experimentation, the author used the Defensive team style. The following seven charts shows the resulting elapsed runtimes for all the test runs, in milliseconds, per battle format and per iteration:

AG	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
It. 1	764	565	613	648	571	577	575	640	621	663
It. 2	594	578	659	650	637	644	633	604	674	731
It. 3	312	284	288	321	274	297	306	293	319	329
It. 4	250	216	273	294	298	293	255	273	251	302
It. 5	339	237	311	370	315	307	310	341	269	315

Uber	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
It. 1	686	617	631	621	632	600	634	701	633	659
It. 2	667	655	624	610	650	659	633	577	580	609
It. 3	298	295	275	296	297	308	273	281	315	309
It. 4	286	291	244	262	281	252	273	256	265	255
It. 5	283	303	219	278	301	292	259	317	281	341

BS	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
It. 1	723	624	554	603	603	625	642	582	589	589
It. 2	309	414	329	308	289	292	341	288	345	260
It. 3	299	668	593	569	552	573	588	586	581	525
It. 4	228	253	230	206	256	206	257	277	286	304
It. 5	314	320	299	264	284	252	282	300	299	299

OU	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
It. 1	605	597	587	608	620	593	610	633	598	538
It. 2	403	289	306	369	365	293	328	295	281	355
It. 3	648	627	526	656	682	550	620	618	589	598
It. 4	279	227	184	329	236	204	213	228	251	266
It. 5	275	271	271	286	258	302	278	259	288	287

UU	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
It. 1	523	536	546	537	554	561	515	523	539	532
It. 2	327	274	304	278	364	369	331	252	375	254
It. 3	324	303	281	232	302	307	269	242	337	294
It. 4	278	210	304	209	213	285	211	215	299	253
It. 5	225	211	221	198	232	255	212	222	226	195

RU	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
It. 1	504	466	429	449	487	464	485	431	470	424
It. 2	245	279	294	316	311	296	300	293	278	304
It. 3	412	411	265	272	280	242	257	303	286	287
It. 4	168	168	185	188	216	167	198	214	180	187
It. 5	208	200	250	224	253	221	179	231	191	235

NU	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
It. 1	442	389	401	410	414	391	407	430	389	411
It. 2	325	224	241	261	241	217	221	227	255	288
It. 3	287	244	253	243	233	283	316	243	272	279
It. 4	215	188	237	200	187	158	235	179	168	190
It. 5	167	157	175	143	147	139	166	156	147	156

Please again note that the teams chosen for the testing of SBPA were chosen at random by a computer program. This greatly reduces the chance of some sort of bias appearing from constantly using the same team. However, that does mean that these results only give data about the most common scenarios the algorithm will encounter, while most runtime analyses on algorithms focus on worst case scenarios. But the worst case scenarios for SBPA will not result in data much different than what can be seen above. Looking at the equations involved in the SBPA algorithm, the same operations and functions are applied to every partial team that the user inputs. In other words, the worst case scenario is treated just like any other scenario; the only thing the worst case scenario could accomplish is slightly slowing down the computational time needed to perform all the operations of the SBPA algorithm, which would only add a relatively small number of milliseconds to the runtime of the algorithm.

The following graph visualizes the average runtimes from the data above:

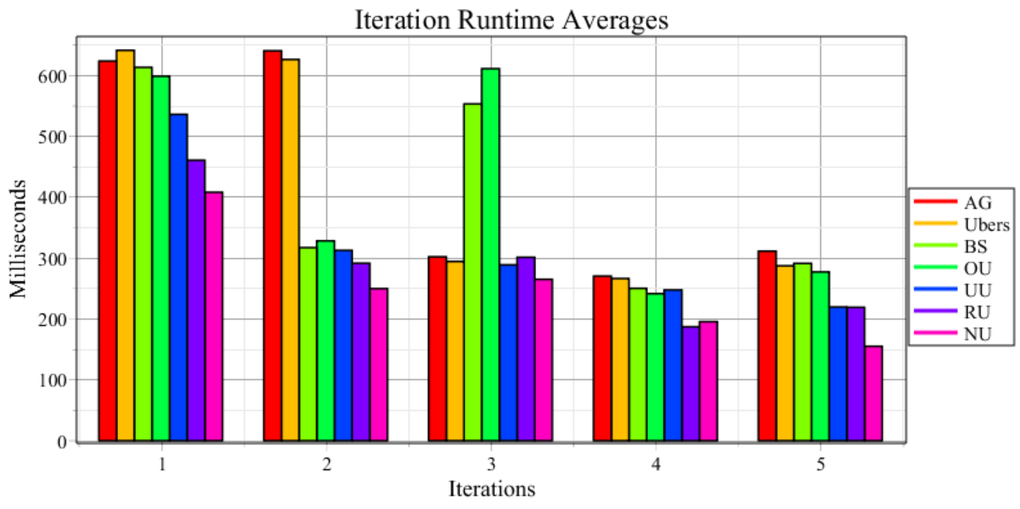


Figure 8:

The data indicates some note-worthy phenomena:

1. In general and per iteration, the more Pokémon that are allowed by a battle format, the longer it takes for SBPA to search for a solution. This was rather expected, since more data to process equates to a longer runtime. What is of interest is that there seems to be a linear correlation between the number of Pokémon that a format allows and the time required for SBPA to complete an iteration, as shown by Figure 3 in combination with Figure 8. Whether this is a true linear correlation is hard to say: the possibility exists that there isn't enough data for the algorithm to start exhibiting non-linear characteristics. Only through the making of more formats can this hypothesis be tested.

2. In general and per battle format, every iteration in the SBPA algorithm seems to take less time. This occurrence is again to be expected for much the same reasons as phenomena 1: every iteration processes less and less data by using the Species Clause and ignoring all Pokémon (and forms thereof) that have already been selected for the user's team. Phenomena 3 and 4, however, describe exceptions to this observation.

3. During the second iteration for battle format BS (Battle Spot Singles) and OU (Overused), the SBPA algorithm requires around 325 milliseconds to run to completion. However, the iteration thereafter takes almost twice as long. This phenomena is odd, since it does not occur for any other format and defies all logic. Though unlikely, perhaps this occurrence results from faulty programming somewhere in the algorithm, but more research is needed to discover the exact origin of this phenomenon.

4. During the second iteration, the algorithm requires around 300 milliseconds to run to completion for all but two battle formats. However, for the Anything Goes and Uber formats, SBPA still requires around 640 milliseconds. Both formats allow nearly identical sets of Pokémon species, which explains why the graphs for these two formats are so similar. However, it does

not explain why the algorithm requires much less time to complete Iteration 2 for all other formats. More research is required to investigate this matter.

5 Conclusion

The SBPA algorithm is an algorithm that gives advice on how to build a competitive Pokémon team. By analyzing the partial team the user already has, it uses simple mathematics to analyze Pokémon species on an individual level and determine the best potential team members based on Base Stats, Typing, and the Popularity Factor.

SBPA focuses on time efficiency and user interactivity. Through experimentation and algorithm runtime analysis, it has been shown that the algorithm can complete any iteration in less than a second, which is astronomically more time efficient than analyzing the over 200 trillion different Pokémon teams in order to find an optimum. Of course, different optimization techniques do exist that would find the optimal team in a time-efficient manner, but they would remove user-interactivity from the process. Therefore, albeit that SBPA is less accurate in determining the absolute optimum, the author finds that the algorithm's efficiency and user-friendliness is compensation enough.

The author does plan to continue developing SBPA into an even more accurate and time-efficient program. New scoring systems can be developed to give user's an even better understanding of the Pokémon they are working with. Optimization techniques can be implemented that allow users to be inspired by mathematically optimal teams. SBPA could be used as an app or website, made available for everyone. Hopefully one day, SBPA will be a huge aid for all trainers alike, united by one thing: Pokémon.

6 Acknowledgments and Bibliography

6.1 Acknowledgments

Special thanks to Dr. K. P. Hart, Mathematician of General and Set-theoretic Topology at the Technical University of Delft, for his invaluable assistance as Bachelor Supervisor and Judge.

Also special thanks to the committee members Dr. Bart van den Dries and Dr. Dion Gijswijt, both Mathematicians at the Technical University of Delft.

6.2 Bibliography

1. "About the Pokémon Company International." Pokémon. The Pokémon Company International, n.d. Web. 15 June 2016. <http://www.pokemon.com/us/about-pokemon/>.

2. "Anything Goes." Smogon University, n.d. Web. 16 June 2016. <http://www.smogon.com/dex/xy/formats/ag/>.

3. "Base stats." Bulbapedia, the Community-driven Pokémon Encyclopedia. Bulbagarden, n.d. Web. 16 June 2016. <http://bulbapedia.bulbagarden.net/wiki/Base_stats>.

4. "Battle Spot Singles." Smogon University, n.d. Web. 16 June 2016. <http://www.smogon.com/dex/xy/formats/battle_spot_singles/>.

5. "Damage." Bulbapedia, the Community-driven Pokémon Encyclopedia. Bulbagarden, n.d. Web. 17 June 2016. <http://bulbapedia.bulbagarden.net/wiki/Damage#Super_effective>.

6. "Deoxys." Pokémon. The Pokémon Company International, n.d. Web. 15 June 2016. <http://www.pokemon.com/us/pokedex/deoxys>.

7. "Gen VI Tiers." Smogon University, n.d. Web. 16 June 2016. <http://www.smogon.com/xyhub/tiers>.

8. Heijstek, Jenty."Improvements to the SBPA Algorithm." Personal interview. 10 June 2016.

9. "Introduction To Competitive Pokémon." Smogon University, 2013. Web. 15 June 2016. <http://www.smogon.com/dp/articles/intro_comp_pokemon>.

10. "List of Pokémon." Wikipedia. Wikimedia Foundation, n.d. Web. 16 June 2016. <https://en.wikipedia.org/wiki/List_of_Pok%C3%A9mon>.

11. "Making the Jump to Video Game Competitive Play." Pokémon. The Pokémon Company International, n.d. Web. 16 June 2016. <http://www.pokemon.com/us/strategy/making-the-jump-to-video-game-competitive-play/>.

12. "Neverused." Smogon University, n.d. Web. 16 June 2016. <http://www.smogon.com/dex/xy/formats/nu/>.

13. "Overused." Smogon University, n.d. Web. 16 June 2016. <http://www.smogon.com/dex/xy/formats/ou/>.

14. "Party." Bulbapedia, the Community-driven Pokémon Encyclopedia. Bulbagarden, n.d. Web. 16 June 2016. <http://bulbapedia.bulbagarden.net/wiki/Party>.

15. "Physical move." Bulbapedia, the Community-driven Pokémon Encyclopedia. Bulbagarden, n.d. Web. 17 June 2016. <http://bulbapedia.bulbagarden.net/wiki/Physical_move>.

16. "Pokémon." Wikipedia. Wikimedia Foundation, n.d. Web. 15 June 2016. <https://en.wikipedia.org/wiki/Pok%C3%A9mon>.

17. "Pokémon Battle." Bulbapedia, the Community-driven Pokémon Encyclopedia. Bulbagarden, n.d. Web. 16 June 2016. <http://bulbapedia.bulbagarden.net/wiki/Pok%C3%A9mon_battle>.

18. "Pokémon File." Pokémon Showdown! Smogon University, n.d. Web. 17 June 2016. <http://play.pokemonshowdown.com/teambuilder>.

19. "Pokémon Types & Type Chart." Pokémon Database, n.d. Web. 17 June 2016. <http://pokemondb.net/type>.

20. "Pokémon VG World Championships." Pokémon. The Pokémon Company International, n.d. Web. 15 June 2016. <http://www.pokemon.com/us/play-pokemon/pokemon-events/pokemon-tournaments/vg-world-championships/>.

21. "Rarelyused." Smogon University, n.d. Web. 16 June 2016. <http://www.smogon.com/dex/xy/formats/ru/>.

22. "Special move." Bulbapedia, the Community-driven Pokémon Encyclopedia. Bulbagarden, n.d. Web. 16 June 2016. <http://bulbapedia.bulbagarden.net/wiki/Special_move>.

23. "Statistic." Bulbapedia, the Community-driven Pokémon Encyclopedia. Bulbagarden, n.d. Web. 17 June 2016. <http://bulbapedia.bulbagarden.net/wiki/Statistic>.

24. "Type." Bulbapedia, the Community-driven Pokémon Encyclopedia. Bulbagarden, n.d. Web. 17 June 2016. <http://bulbapedia.bulbagarden.net/wiki/Type>.

25. "Uber." Smogon University, n.d. Web. 16 June 2016. <http://www.smogon.com/dex/xy/formats/uber/>.

26. "Underused." Smogon University, n.d. Web. 16 June 2016. <http://www.smogon.com/dex/xy/formats/uu/>.

27. "Zigzagoon." Pokémon. The Pokémon Company International, n.d. Web. 15 June 2016. <http://www.pokemon.com/us/pokedex/zigzagoon>.

7 Appendix

For the actual code of the program, please visit the Github repository PokeTeam by HugoReijm:

https://github.com/HugoReijm/PokeTeam