VGC AI Competition - A New Model of Meta-Game Balance AI Competition

Simão Reis
Artificial Intelligence and Computer Science Laboratory
University of Porto
Porto, Portugal
simao.reis@outlook.pt

Luís Paulo Reis
Artificial Intelligence and Computer Science Laboratory
University of Porto
Porto, Portugal
lpreis@fe.up.pt

Nuno Lau
Institute of Electronics and Informatics Engineering of Aveiro
University of Aveiro
Aveiro, Portugal
nunolau@ua.pt

***Abstract*—This work presents a framework for a new type of meta-game balance AI Competition based on Pokémon. Pokémon battles can be viewed as adversarial games played by AIs. Around these games, there is also a meta-game: which Pokémon to include in a team for battles, which moves to pick for every Pokémon in the team, etc. This meta-game is itself a game with a set of rules that govern which Pokémon and which moves are available in the roster that can be selected from, or which attributes (health points, damage, etc.) a Pokémon or moves should have. The aim of the framework is to facilitate competitions in creating the most balanced meta-game possible; one where there is a large variety of Pokémon and moves to choose from, and many possible combinations that are effective. AI agents could assist human designers in achieving strategically expressive meta-games, and this type of benchmark could incentivize game designers and researchers alike to advance knowledge on this type of domain.**

***Index Terms*—Competitive Games, AI Competition, Automatic Game Design, Multi-Agent Systems, Multi-Task Learning**

I. Introduction

There have been changes on how the online multiplayer games media is consumed, notably over e-sports where video games are used as a medium for competitive scenes. This universe of games must engage players on a deep level of strategy and dexterity to compete for supremacy. One of the central components of this universe of games is that players can control one or several agents/units acting in the game, and they have the option of choosing what unit(s) they intend to use before starting a match.

Although multiplayer games have distinct mechanics and gameplay styles, they share similarities in terms of pre-game preparations: (i) in collectible cards games, players must assemble a deck, choosing strategically the cards from the available pool to compete against other players, with notorious examples Magic: The Gathering (Wizards of the Coast, 1994) both in tabletop and online variants, and Hearthstone (Blizzard Entertainment, 2014); (ii) in fighting games, players must choose from an array of available fighters and confront the opponent to compare their fighting skill, like in Street Fighter series (Capcom, 1987) and Tekken (Bandai Namco, 1994); (iii) in real-time strategic games, players or teams control one or multiple units to defeat the opponent using a vast set of abilities, some unique to certain champions, like Dota 2 (Valve, 2013) and StarCraft (Blizzard Entertainment 1998).

Strategic interest of the above games results from the fact that some units are good at some things while others at other things; or they are good at different times in the games; which allows for greater player choice and gameplay variety. If all units were identical it would make a boring game. However, competitive games are critically evaluated on how balanced they are and on how much diversity of gameplay they can offer, and it is a decisive factor for players' retention.

During the game balance phase of game development, which is the practice of tuning game mechanics to produce a more immersive and fair game, game designers must perform exhaustive testing and gameplay analysis as to properly tune the mechanics of each type of unit (cards, fighters, champions) allowing for a greater variety of playable mechanics and strategies in the meta-game, which is the process of predicting and responding to the opponents' strategies by knowing the current usage rates or most used strategies.

Game balance is easily prone to human error which frequently causes flawed designs, for example overpowered units that have a high win rate independent of the opponent unit(s), hence removing diversity from the meta-game. Artificial Intelligence (AI) agents could assist developers by play testing thousands or millions of games, detecting the cause of eventual overpowered units and game mechanics to propose and enforce fixes. It is however not obvious how to train AI agents to value units or composition of units in this type of scenario. This is due to game fixes resulting in changes on the game dynamics, combined with the high combinatorial of possible unit selections some multiplayer games present, even more so in a system governed by interdependence task model: knowing how to play is needed for good unit selection, as knowing what units are the most probable to be used affects the game itself when there is hidden information about the opponent's units.

This work was supported by: National funding by FCT, Foundation for Science and Technology, and European funding by ESF, European Social Fund, through the individual research grant SFRH/BD/129445/2017; Artificial Intelligence and Computer Science Laboratory – LIACC (UIDB/00027/2020); and Institute of Electronics and Informatics Engineering of Aveiro – IEETA (UIDB/00127/2020).

978-1-6654-3886-5/21/$31.00 ©2021 IEEE

Game AI research had many recent milestones. Dota 2 with large scale Reinforcement Learning [1] showed that AI agents are currently at best only able to defeat human experts by exploiting their dexterity. Super-human performance achievements in classic tabletop games like Chess, Go, and Shogi [2], [3] were possible because they are perfect information games. But work in two player imperfect information games like Poker [4], [5] also shows progress. We believe similar milestones could be achieved for tasks involving team building, gameplay and game balance tasks, which would enhance knowledge for automatic game design through a suitable competition model and further motivate the collaboration between game industry practitioners and Game AI researchers and gain new insights on this domain of knowledge. This meets with the recent identified problem of a game master (designer agent) gaining points for good rules [6].

In this work, we propose a new kind of AI competition model where AI agents must learn how to perform meta-game balancing. We assume to have an arbitrary level of freedom in manipulating game units, albeit restricted by a set of rules defined by a game designer, in a formal language that allows him or her to translate their intentions for the meta-game (i.e., how many units, what mechanics are available, what metrics the AI agent should prioritize, etc.). To be able to run this new type of competition, we developed the Video Game Championships (VGC) [7] AI Framework, which allows to run an ecosystem that simulates an interdependent task model over the domain of Pokémon and allows agents to solve the VGC challenge based on the method used by competitive human players who try to win by predicting the meta-game.

The remainder of this paper is structured as follows. In Section II, it is discussed related work in regards with Game AI competitions, automatic game design techniques, the review of the current results that contributed to the analysis and development of our VGC AI Competition proposal. In Section III, we present some background relatively to the rules, challenges of Pokémon battling, VGCs and meta-game balancing. Our proposed model of competition, framework and rules are detailed in Section IV. Finally, in Section V we take our final notes and present our next goals.

II. Related Work

A. Game Design AI Competitions

We can observe the recent proposals for the annual IEEE Conference on Games (CoG) where most entries are based on real-time video games with perfect information like the StarCraft AI Competition [8], or the Fighting Game AI Competition [9]. However some competitions started giving the first steps towards game design and balance tasks, but with major limitations and we discuss some below.

The Hearthstone-AI Competition [10] was proposed for the IEEE CoG. Not only the card game has challenging properties like partially observable states, as the players do not know which cards they are going to draw each turn or which cards are in the opponent's hand and deck, of which there are over 2000 cards each with different attributes and game changing effects that drastically increases the complexity of the game. Currently the competitions proposed a game track and a deck-building track, where humans may design the team prior to the competition and optimize their agents to the designer team. The work suggests an automatic deck build track, which is hard since the cards' power depends on synergies and combinations between themselves during gameplay; and a game-balance track where card rate usage is crucial to choose what are the best deck building options. The balance could be done at the card pool level or at rules level (game mechanics). Our target domain coincides with the first, as Pokémons have the same role as units, similar to cards in card games. The Strategy Card Game AI competition [11] serves the purpose of a more lightweight version of the Hearthstone-AI Competition.

The General Video Game AI (GVGAI) competition [12], [13] poses the challenge of creating AI solutions that can play a wide range of games. It promotes the devising of algorithms that are able to play any game they are given, even if the game is unknown *a priori*. This highly contrasts with all other competitions which focus on a single game domain. It is built upon the Video Game Description Language (VGDL) [14], accommodating two-player games, and, more interestingly provides challenge for procedural content generation (PCG) in terms of level [15] and rule generation [16], which is one of the few domains being proposed in automatic game design competitions. A next benchmark for the Pokémon domain would be to also be able to extend game rules (or mechanics) in a balanced fashion, conserving the core gameplay.

Ludii [17] is a framework for general board-games, where new games can be synthesized. Ludii was proposed as a platform for agent-based competitions but also for Procedural Content Generation competitions [18]. By using the Ludii framework it is possible to run competitions for game generation, or certain aspect of games (rules, puzzles, or tutorials). It can enforce multiple generational requirements and proposes evaluation by human opinion or agent validation. This contrasts with our game balance competition, where we restrict ourselves to an original Pokémon roster (rule set) and try to convert it into a more high-quality roster.

In conclusion, exist studies for automatic design of games or game balance, but these are still limited, since they lack a formal execution model and formal evaluation model, which is crucial to further research in these fields through competitions.

B. Pokémon AI Environments and Competitions

Multiple fan-made Pokémon simulators have been developed over the years like the Pokémon Showdown Battle Simulator [19] and the Pokémon Online [20], which focuses on human user-experience. Pokémon Battle Engine (PBE) [21] complies to the *de facto* standard Reinforcement Learning API Gym from OpenAI [22] and enables generalized training through random team configurations. To a meta-game balance competition, where a roster is tuned, balance (or designer) agents must be able to propose fixes, but the roster must be unpredictable before the competition, as balance fixes may be hard coded by human competitors on the designer agents. The

Pokémon Showdown and the Pokémon Online work with the official Pokémon roster, not complying with our requirements, motivating us to extend PBE into the VGC AI Framework.

A Pokémon battling AI competition was recently proposed, known as Showdown AI Competition [23]. The Showdown AI Competition API is a modified version of the Showdown battle simulator, providing methods for agents to act over the full roster and mechanics, but does not propose a game balance track. However, the study identifies eight main properties that contrast Pokémon Battling with other games worth discussing: (i) Branching Factor - with four move actions and up to five switch actions; (ii) Infinite Looping - when both agents choose to switch endlessly without causing any damage to the opposing Pokémon. (iii) Turn Atomicity - In a clean state, HP and statistics are sufficient to determine the actions' immediate value, but over-time conditions like *burned* or field effects like *sandstorm* become hard to quantify, resulting in delayed rewards; (iv) Stochasticity - damage calculation is dependent of random parameters; (v) Hidden Information - this environment is partially-observable (opponent party and their Pokémon moves); (vi) Deception - caused by a single ability, where a Pokémon appears in battle disguised as another Pokémon from the opponent's party until is hit by a damaging move; (vii) Lightweight Simulation - with low graphical requirements, the environment outperforms many other video-game simulators. Therefore, Pokémon battles features several properties that make it a challenging AI environment.

C. AI Agents for Pokémon Battles

Pokémon Battling agents were developed using multiple AI approaches. In [24], hard-coded battle agents are used, however they are sub-optimal as they do not perform tactical long-term decisions, such as switching when a type disadvantage is presented. [25] focus on a Q-learning [26] agent achieving a win-rate of 65% against random opponents, or 90% when using a Minimax-Q agent. Supervised learning can be used to predict the likely outcome of Pokémon match-ups [27].

The applicability of deep learning paradigm to Pokémon was previously demonstrated through a pair of algorithms [21], GIGAθ and WPLθ [28]. Both are based on Asynchronous Q-Learning [29], which is a faster-distributed extension of Deep Q-Learning (DQN) [30]. The base GIGA-WoLF [31] using the *Win or Learn Fast* (WoLF) principle, has different learning rates depending if the agent is winning or losing. WPL [32] instead makes use of a variable learning rate. Both GIGAθ and WPLθ were trained in self-play for two million episodes and were tested in a scenario where a trained agent in a disadvantageous position played against a random opponent. The learning results were compared against a hard-coded baseline that followed the guidelines given by two human experts. In a 7-type scenario both GIGAθ and WPLθ were able to converge, but with the full 18 types, only GIGAθ achieved human performance. This is probably due to GIGA-WoLF better converge to deterministic policies. Since the base PBE only supports clean states, deterministic strategies are sufficient to achieve victory.

Another deep reinforcement learning based approach with an actor-critic neural network was recently proposed [33]. The methodology showed to have a low cost training and resulting with a good performance against random, greedy agent and tree-search agent and human players alike.

III. Background

A. Pokémon Battles and Video Game Championships Rules

Pokémon (Nintendo, Creatures, Game Freak, 1996) battling is a large state turn-based stochastic simultaneous game with hidden information from the opponent battle team. In Pokémon VGCs competitions go beyond battling, a limited but extensive roster of Pokémon units are available, and it is largely up to the players to choose and customize their units. After their selection they must commit with their team for the entirety of a competition. We selected Pokémon VGCs as the domain of test for this work because it complies with our requirements of an interdependent task model, and Pokémon requires not only unit selection, but unit configuration as well.

In a Pokémon battle, a player competes with a team of Pokémons, and each Pokémon possesses a set of modifiers such as hit points, attack, defense and speed, and a set of four moves. Each turn, the player may select one of four moves from his active Pokémon, to attack the opponent's active Pokémon, or may switch the active Pokémon with one in the bench. When a Pokémon's HP depletes, it becomes unusable for the remainder of the battle and the player is forced to switch them for another Pokémon. The first player to knock out the opponent's entire team wins.

A core mechanic is type advantage, Pokémons and moves possess a type, and types have different effectiveness against different types. A FIRE move is super effective against GRASS Pokémon, and WATER Pokémon resist FIRE moves. New mechanics were added to the Pokémon battles over the years, but the core mechanics and goals remained the same. Some of the most notable changes of more recent Pokémon generations are the addition of passive abilities, triggered abilities activated under a certain game condition, and various equipable items with a wide range of effects.

In VGCs, matches are done as best of three. At the beginning of each battle a player has partial information about the opponent's team. In a competition, there is a limited but extensive roster of allowed Pokémon units (whitelist) where players may select from to build their team.

B. Meta-Game Design

Meta-gaming is the process of using knowledge outside of a game to gain an advantage. In competitive multiplayer video games, meta-game information helps players gain a vision from the most successful units they can select to play with and the most probable choices from their opponents. Human players use their knowledge about the game rules, common units or composition of units and through theory crafting, collaboration and competition with other players anticipate the meta-game to find the most optimal strategies, and finally engaging in tournaments using a fine-tuned team. The issue

of meta-gaming emerges, in the perspective of players, if available strategies are too vast becoming too unpredictable, or the opposite case where the meta-game is centralized over one dominant strategy, diminishing gameplay depth.

Human game designers performs similar tasks, they must analyze the meta-game, and if flawed, understand the reasons and employ fixes (add, remove, or configure units). Therefore, designers, much like players, must anticipate the meta-game to tailor its faults and arrange for a good compromise for a reasonable number of good strategies, so they can offer a variety of gameplay experiences. However, if every possible combination of Pokémon becomes equally viable, there is no more skill expression in the meta-game as every possible team would be equally strong and that is not desirable either.

For Pokémon, Pykalitics [34] is a free online analytic platform that aggregates information from VGCs to assist human players in building their teams. It provides Pokémon usage statistics and moves, items, abilities most used for a given Pokémon, and most common team compositions, etc.

IV. Pokémon VGC AI Competition

The VGC AI Competition aims for the development of player agents and meta-game balance agents and compare them in each task. Player agents compete in an ecosystem based of human VGC competitions and balance agents must manipulate the existing roster so that the strategy level becomes more expressive. First we define what is an Interdependent Task Model, which the competition framework uses as the base model, describe the framework implementation, then each track and their purpose.

A. Interdependent Task Model

In an Interdependent Task Model, skills must be acquired in conjunction, as there is a dependency cycle. In informal terms, one can think like the chicken and egg problem, which one came first? In traditional hierarchical learning, low level behaviors are learned first in isolation and then a high-level controller learns to coordinate the multiple low-level behaviors [35]. In an interdependent model, skills must be obtained in an iterative fashion. If tasks A and B are interdependent, one can only master A by mastering B and vice versa. In our case study, as stated above, a game can only be mastered by knowing how to predict the meta-game, but to predict the meta-game is also required to master the game.

B. VGC Framework

The VGC Framework purpose is twofold. The first is to run AI competitions, where contestants submit their solutions. The second is to allow players to design and train their models in the same environment where the competition itself is run.

The VGC Framework manages an ecosystem ruled by an Interdependent Task Model, emulating the main tasks of competitive Pokémon: (i) team selection; (ii) battling; (iii) team building; (iv) meta-game balance. Since each task differs in nature, there may be the need to coordinate different techniques. The framework abstracts all the logic behind managing the ecosystem, allowing the user to focus on development.

The framework is versatile, it allows use of any of its parts so contestants can develop and test specific challenges in isolation, each one associated with a different competition track. The motivation behind this decision is that isolated contributions to each task can guide us to the overall comprehension of the challenge.

We segmented the framework description in four parts. First, we introduce the core data objects of the framework, which are manipulated by the system and software agents. Next, we describe the architecture, the main modules of the framework, how they interconnect and how they can be run in isolation. Third, we detail the new version of the PBE. Lastly, we describe the framework API, so competitors know how they can develop and test player and balance agents.

1) Data Model: The core concepts of the framework and their relationship are illustrated in Fig. 1. A Pokémon Template defines the possible moves a Pokémon specimen may have, what are the range of values their attributes can have, and so on. A Pokémon is an instance of a template with concrete moves and attributes configured. A Pokémon Team is composed of six different Pokémon (a Pokémon Team may not contain two or more instances of the same template). The Move Roster defines which moves are legal as the Pokémon Roster defines which Pokémon Templates are legal for for the duration of a competition (the roster is a set of rules for player agents that balance agents can change). A Meta-Data objects does not have a singular definition, each contestant may use its own defined structure, i.e., each competitor may use distinct statistics as strategy to analyze the meta-game. Meta-Data should contains statistics about gameplay like win-rates, team/individual Pokémon compositions and other useful data for decision making of player and balance agents.

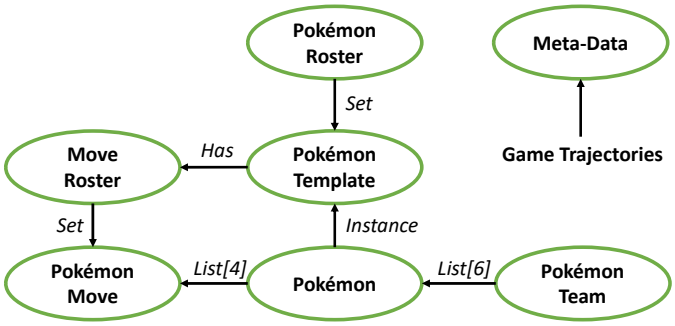

Fig. 1. VGC Framework Data Model. Game Trajectories provided are not only from a players' played games but also from other played games, i.e., games historical are public and broadcast to every agent.

2) Architecture: The main requirement that guided our architecture design was modularity, to be able to run isolated parts of the whole VGC ecosystem, as we are dealing with interdependent tasks. We name each major part of the ecosystem a module, which runs sequentially and/or concurrently to others modules, depending on the case. Each module is incorporated with logic to manipulate the core data objects through processes, whose outcome is dependent on behaviors' decisions. Behaviors and processes are connected through

interactions, where the process gives the AI behavior an observation and the AI module replies with an action. Processes are connected by a logic flow and by I/O channels between modules. The four major modules from the framework are: (i) the Selection Phase; (ii) the Battle Phase; (iii) the Team Building Phase; and (iv) the Meta-Game Balance phase. The four are described below.

In the Selection Phase, given the full player team, partial information of the opponent team (common knowledge derived from the roster like type, possible moves and attributes for each opponent team Pokémon) and meta-data the aim is to choose the most optimal sub-team selection. This is done in two steps. First, by using meta-data and partial information of the opponent's team we predict the opponent units. Given the predicted team we choose our team that best suits against the opponent one. The Selection Phase is illustrated in Fig. 2.

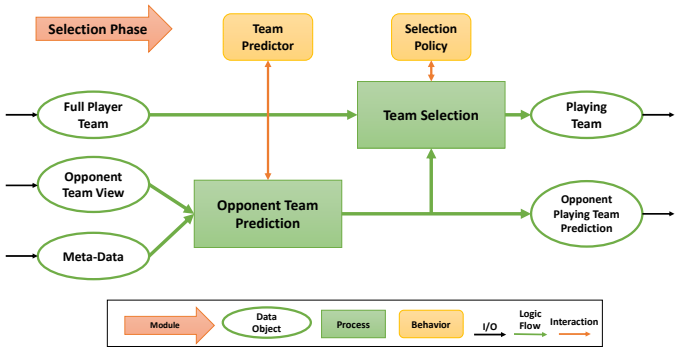

Fig. 2. A Selection Phase is constituted of two processes, each interacting with a respective behavior. The Team Predictor behavior predicts the opponent team structure. The prediction is provided to the Selection Policy to choose strategically the initial active Pokémon and party (non-active Pokémon team members) against the opponent. This process is an one-shot run where there are no loops in the logic flow.

In the Battle Phase, in which a player agent competitor, entering with their selected team, must defeat the opponent. Each player observes the current game state and choose their action for this turn. Turns snapshots are stores in the form of trajectories that used both to update knowledge we have about the opponent's team and update the ecosystem's meta-game data. The Battle Phase is illustrated in Fig. 3.

Both previous modules make use of a Team Predictor behaviour, which allows us to reduce the workload of the Battle Policy, where it can assume it has perfect information about the game, while we leave the burden of filling the hidden information to the Team Predictor, which can be reused in the Selection Phase as well.

The next is Team Building, where between battles each player is given the opportunity to analyze the meta-game and restructure their team. First team performance is evaluated giving their current team and meta-data. The current team value together with the available roster is used to adapt the team if needed. When ready the player can engage against in the competition ecosystem, where is paired against other players to battle. This process can be repeated until the player finds a satisfying team within a time limit. Team building

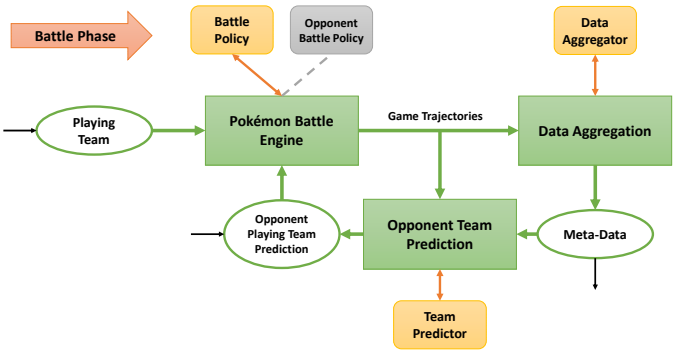

Fig. 3. A Battle Phase is composed of three processes, the PBE which interacts with a pair of battle policies, a data aggregation process and team predictor. Game states and joint actions are propagated in trajectories and delivered to Team Predictors to update their opponents team prediction. At the end of battle, game trajectories are broadcast to every agent's Data Aggregation process which will update the meta-data (usage rates of Pokémons, moves, teams, etc). There is a loop in the information flow, the PBE runs until the game reaches a terminal condition.

modules run concurrently with the competition ecosystem, which consists of sequential pairs of selection phase and a battle phase. Competitors are put in a match queue in the competition ecosystem, and after paired, they battle, and meta-game information is updated after the fact. When allowed, a competitor may leave the battle queue and proceed to a team building phase. The Team Building is illustrated in Fig. 4.

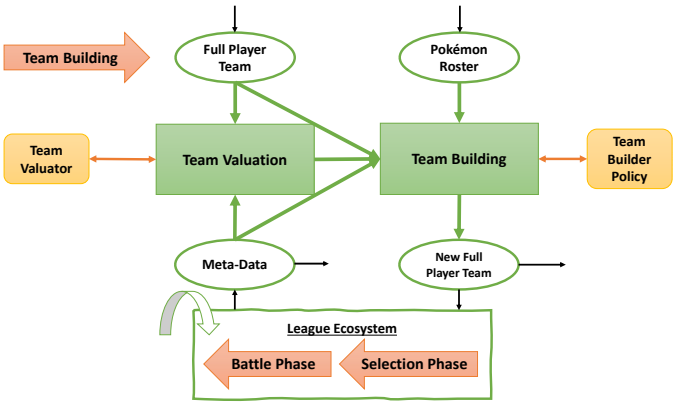

Fig. 4. Team Building Phase with a Team Valuation and Team building processes. The form of valuation is defined by the developer of the Team Valuator and Team Builder Policy modules. The team value could range from an overall team performance to individual units' performance or their synergies and efficacy. The team building policy has access to the roster, as it needs to know what changes can be made to their team. It finally outputs a new (possibly changed) team. The framework provides mechanisms to validate the output team, which may be rejected and changed back to its previous configuration in the case an illegal team is outputted. Is a one-shot run where there are no loops in the logic flow.

Lastly, in the Meta-Game Balance module, we aim to tune the Pokémon roster by analysing an ever evolving VGC Ecosystem where player agents build teams and battle against each other. Using that information, a balance agent makes changes to the Roster in an online fashion aiming to maximize some balance metrics while subject to design constraints,

both are set by the organizers before the competition. When the balance agent changes the roster, every player agent is interrupted and moved to the Team Building Phase with their teams reflecting the changes of the roster, i.e., changed moves or even banned Pokémon. The Meta-Game Balance is illustrated in Fig. 5.

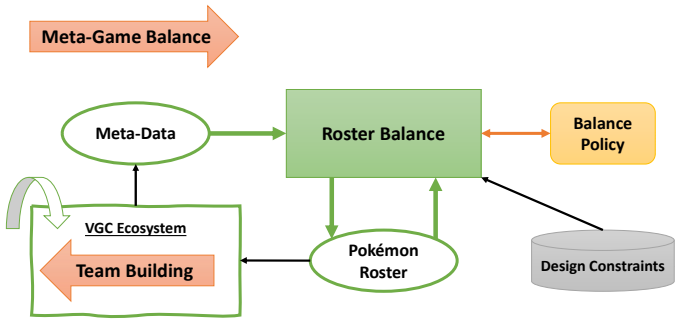

Fig. 5. Meta-Game Balance Phase. The roster balance runs concurrently to a VGC Ecosystem. Organizers can also impose a set of design restrictions to the agents, i.e., cannot fully change the roster to a completely new one.

In sum, the VGC framework allows to run traditional AI battle competitions, extend it with team selection, team building, and run a separated challenge on meta-game balance.

3) Pokémon Battle Engine 2.0: PBE 2.0 inspires from the primary rules of Pokémon 1v1 battles and now supports game mechanics to contemplate with all Showdown properties except for deception. First, all agents that selected to switch do so. Move order is decided by the greatest *speed* stage level or uniformly random if equal for both players. The speed stage is softly increased by one if the move to be used has priority. After switching into the new active Pokémon, all entry hazard damage is applied. Then pre battle effects are processed. These include if status conditions should be removed from active Pokémons. Pokémon status conditions are checked and if they can move, they do so in order. If their move PP is zero a default move is done instead. When a move is used its PP is decreased. If the first moving Pokémon causes the second one to faint, the latter is not allowed to move. Same logic is applied if their fainting was caused by Entry Hazard damage. Then post battle effects are processed. These include clearing the weather after finishing the countdown and applying status damage. Then fainted Pokémon are recursively switched, applying Entry Hazard damage and switching again until all Pokémons of one player are fainted or both active Pokémon are still not fainted. On reset, all special conditions are cleared, and stats reset to maximum.

A main learning goal in PBE is the optimization of a long-term strategy in favor of a worse strategy with immediate rewards. As such, for reinforcement learning based algorithms, the environment provides a default reward function given by $R = R_d + R_f + R_v - R_t$, where R_d represents the damage dealt as a fraction of the opponent's maximum HP, $R_f, R_v \in \{0, 1\}$ is a bonus if the opponent fainted or victory over the opponent team was achieved, and R_t represents the damage taken as a fraction of Pokémon's maximum possible HP. With unclean states (effects are on place), recovery moves, and recoil moves, R_t is the sum of all damage taken, by moves, recoil, status condition, weather condition or entry hazards.

4) Application Programming Interface: To compete, participants must submit an agent that abides by the VGC Framework Competitor API containing the multiple categories of behaviours previously discussed. As illustrated in Fig. 6, depending of the role of the agent (player or balance/designer), it must contain specific behaviours and will act at different levels of abstraction.

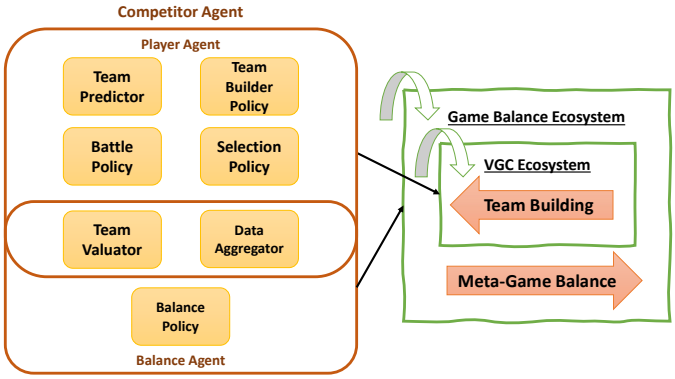

Fig. 6. The VGC Framework Competitor API. A player agent acts in the VGC Ecosystem environment, while a balance agent acts in the Game Balance Ecosystem environment. Depending on the role, only a selection of behaviours are needed to be integrated in the agent.

C. Competition Tracks and Rules

Each proposed track tackles different challenges on the VGC AI Competition. In the Battle Track player agents only compete with battle policies. In the VGC Track, competitor agents must master the skills of battling, team selection and team building. In the Balance Track, balance agents must manage the roster better than their opponents. Participants only need to submit a competitor agent with the minimum required behaviours for the track their participating. Same competitor agent can be reused for multiple tracks.

1) Battle Track: This is the simplest track, where player agents compete in a double elimination competition or tree tournament, where a winner is determined by the outcome of isolated battles. The agent possesses only a Battle Policy to compete. At each round, three sequences of matches are performed between two agents. At the beginning of each sequence a random team is generated for each player. Players compete over five battle, and then have more five battles with teams switched, so a player isn't affected by teams' unbalance.

2) VGC Track: In this track, player agents must be able to compete in a full VGC, where they must build their teams, do meta game analysis and battle. The model of competition works as follows: First a Pokémon roster is generated and player agents may assemble their teams. We then run a meta-game evolution phase, where agents are paired with multiple opponents during epochs where they perform at least a minimum stipulated number of battles, after which players are allowed to adjust their team, entering a new epoch. By

the end, all players are registered to a final tournament phase where agents compete like in the Battle Track to determine the winner, but using their current team configuration instead of randomly generated ones. This results in a meta-evolutionary ecosystem where agents can prepare for the final decisive tournament. However, agents may only choose one of the team configurations they used in the preliminary stage. These incentives agents to perform well during the preparation phase and enrich the meta-data, if they would perform badly on purpose to affect the meta-game, they will only be able to compete with weaker used teams.

3) Balance Track: In this track, a balance agent is assigned to design over time the roster for a full VGC Ecosystem using data from previous matches. Balance agents compete in parallel, i.e., each is set in an isolated VGC Ecosystem instance, but provided with the same initial conditions, which are a triplet of population of player agents, design restrictions united with balance metrics and a Pokémon Roster. Organizers must: (i) provide the player agents, which should be as close as possible to theoretically-optimal, to run inside the VGC Ecosystems; (ii) generate an initial roster; and (iii) define a set of design constraints. These policies make so no trivial balance agents can be developed and submitted *a priori* (roster and restrictions must be unknown to the competitors). Restrictions (further discussed in Section IV-D) have the role of game semantics, i.e., the changes should not make a Pokémon unrecognizable. Balance agents are evaluated by score accumulated over multiple measurements made in epochs of the VGC Ecosystems. Statistics of the VGC competition (like overall usages rates of Pokémons and moves) are evaluated by the balance metric and a percentile of the maximum evaluation is accumulated until the end of the balance competition.

D. Balance Metrics and Design Constraints

Human players perception of balance is usually correlated with the meta-game's diversity, of which there are many types of: specific team configurations, Pokémon usage, moves usage, type predominance, types of moves, abilities, what archetypes are most relevant, etc. A balance agent must perform global optimization over many parameters, eventually some with priority over others. Therefore, we need formal metrics to evaluate the balance task performance.

Typical design changes are done by adding new elements, removing existing ones, or tuning parameters of existing ones. In the case of the VGC AI Competition, moves' attributes can be tuned, moves can be added or removed from a Pokémon template, or attributes from the template could be re-adjusted. But, fixes to the roster could result in uninteresting results, we could want Pokémons to maintain their main characteristics, so only moves could be changed but not its type (due to some aesthetic characteristic).

For the purposes of enabling organizers to define the design restrictions and balance metrics, we implemented the Design Constraint Language. Restriction rules can be applied to every Pokémon or only to specific templates. These rules can, for example, disable changes on move sets, types and other attributes, or limit how much a Pokémon can be modified (the degree of change can be measured by an information distance metric). We can also define global parameters like the Pokémon roster size and move roster size.

We propose the four main evaluation dimensions for the VGC AI Competition: (i) Specimens; (ii) Moves; (iii) Teams; (iv) Move Effects; (v) Archetypes of the first three dimensions. The evaluation metric must have some considerations. First, we cannot evaluate in isolation the various dimensions, because they may have different impacts in the game outcome. Therefore, more weight must be given to instances of each dimension (specimen, move or team), and variety must not be only measured in quantity of instances but the usage percentage as well. We also want to evaluate win-rates of each instance in the same fashion. We reduce this to three criteria: instance usage rate distribution, and instance win-rate distribution. See Fig 7 for an example.

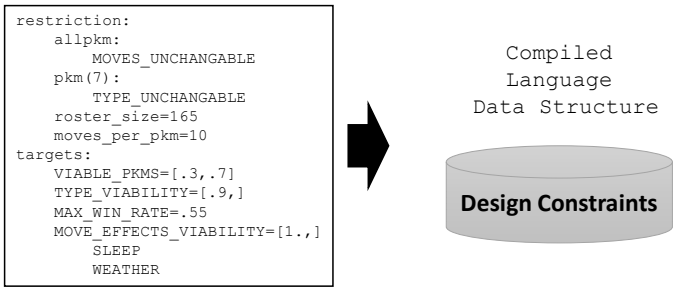

Fig. 7. Design Constraint Language Mock Example: Moves cannot be changed for any Pokémon, template 7 cannot have its type changed. The target of balance is to have a viable population between 30% and 70%, type viability of 90%, max win-rate for Pokémons and/or teams is 55% and Sleep and Weather effects must be present at least in 10% of teams used. The configuration language is compiled for a data structure for the VGC framework.

The size of the Pokémon Roster and Move Roster will have a large impact on the challenge difficulty. Since the original official Pokémon Roster had 151 species, this should be an initial benchmark for the challenge. In the same fashion the size of the move roster should be at least 165. We propose an average of 10 possible moves for each Pokémon template.

V. Final Discussion

In this work, we formalize the model and evaluation method of an AI meta-game balance competition where a balancing agent must generate or re-adjust the present roster of available Pokémons to players to make the viable team-building strategies diverse enough to satisfy human players. We foresee an agent that achieves a balance by generating several strategies compatible with team building assessment, while, at the same time, ensures there is no single dominant strategy. Not merely are the multiple proposed tracks complex by themselves, but the challenge of being able to solve them requires task coordination since they are interdependent. This challenge may be interesting for other domains, with special interest to the automatic game design research field. Our engine and competition proposal may bring a new benchmark and new

solutions for this type of challenge and inspire others to develop new benchmarks for this type of competition.

Exploring the precise equilibrium of factors that makes humans being engaged in a competitive scenario, allowing to model a more accurate balance metric function for the game balance agent would be a future interesting line of research.

We plan to run the competition at an academic conference like the future CoG 2022 edition. Baseline solutions are necessary for participants to measure the quality of their agents with respect to baseline results, and we plan on communicating such developments in the near future. We believe fields like imperfect information games, multi-objective optimisation and evolutionary game theory will guide us in the right research direction. The developed open-source implementation can be found at https://gitlab.com/DracoStriker/pokemon-vgc-engine.

REFERENCES

[1] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. W. Pachocki, M. Petrov, H. P. de Oliveira Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, and S. Zhang, "Dota 2 with large scale deep reinforcement learning," *ArXiv*, vol. abs/1912.06680, 2019.

[2] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, "Mastering the game of go without human knowledge," *nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[3] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel *et al.*, "Mastering atari, go, chess and shogi by planning with a learned model," *arXiv preprint arXiv:1911.08265*, 2019.

[4] M. Moravčík, M. Schmid, N. Burch, V. Lisỳ, D. Morrill, N. Bard, T. Davis, K. Waugh, M. Johanson, and M. Bowling, "Deepstack: Expert-level artificial intelligence in heads-up no-limit poker," *Science*, vol. 356, no. 6337, pp. 508–513, 2017.

[5] N. Brown and T. Sandholm, "Superhuman ai for heads-up no-limit poker: Libratus beats top professionals," *Science*, vol. 359, no. 6374, pp. 418–424, 2018.

[6] A. K. Hoover, J. Togelius, F. Ri-choux, J.-H. Seok, S. Temsiririrkkul, , and A. Zook, "Which games should we (ai) explore next?" *Artificial General Intelligence in Games: Where Play Meets Design and User Experience*, vol. 130, no. 6, pp. 17–19, 2019. [Online]. Available: https://shonan.nii.ac.jp/docs/1230d7e76d0ab68ce19904c614748871f26759db.pdf

[7] T. P. C. TPC, "2020 pokémon video game championships (vgc) format rules," Web: https://www.pokemon.com/us/pokemon-news/2020-pokemon-video-game-championships-vgc-format-rules/, 2020, accessed: 2021-03-18.

[8] M. Čertický, D. Churchill, K. Kim, M. Čertický, and R. Kelly, "Starcraft ai competitions, bots, and tournament manager software," *IEEE Transactions on Games*, vol. 11, no. 3, pp. 227–237, 2019.

[9] F. Lu, K. Yamamoto, L. H. Nomura, S. Mizuno, Y. Lee, and R. Thawonmas, "Fighting game artificial intelligence competition platform," in *2013 IEEE 2nd Global Conference on Consumer Electronics (GCCE)*, 2013, pp. 320–323.

[10] A. Dockhorn and S. Mostaghim, "Introducing the hearthstone-ai competition," *arXiv preprint arXiv:1906.04238*, 2019.

[11] R. M. Jakub Kowalski, "Strategy card game ai competition cog 2019," Web: https://jakubkowalski.tech/Projects/LOCM/COG19/, 2019.

[12] D. Perez-Liebana, S. Samothrakis, J. Togelius, S. M. Lucas, and T. Schaul, "General video game ai: Competition, challenges, and opportunities," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, ser. AAAI'16. AAAI Press, 2016, p. 4335–4337.

[13] D. Perez-Liebana, J. Liu, A. Khalifa, R. D. Gaina, J. Togelius, and S. M. Lucas, "General video game ai: A multitrack framework for evaluating agents, games, and content generation algorithms," *IEEE Transactions on Games*, vol. 11, no. 3, pp. 195–214, 2019.

[14] T. Schaul, "A video game description language for model-based or interactive learning," in *2013 IEEE Conference on Computational Inteligence in Games (CIG)*, 2013, pp. 1–8.

[15] A. Khalifa, D. Perez-Liebana, S. M. Lucas, and J. Togelius, "General video game level generation," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, ser. GECCO '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 253–259. [Online]. Available: https://doi.org/10.1145/2908812.2908920

[16] A. Khalifa, M. C. Green, D. Perez-Liebana, and J. Togelius, "General video game rule generation," *2017 IEEE Conference on Computational Intelligence and Games (CIG)*, Aug 2017. [Online]. Available: http://dx.doi.org/10.1109/CIG.2017.8080431

[17] E. Piette, D. Soemers, M. Stephenson, C. Sironi, M. Winands, and C. Browne, "Ludii - the ludemic general game system," in *ECAI 2020 : 24th European Conference on Artificial Intelligence*, ser. Frontiers in Artificial Intelligence and Applications, G. De Giacomo, A. Catala, B. Dilkina, M. Milano, S. Barro, A. Bugarín, and J. Lang, Eds., vol. 325. Netherlands: IOS Press, 2020, pp. 411–418, 24th European Conference on Artificial Intelligence, ECAI 2020 ; Conference date: 29-08-2020 Through 05-09-2020. [Online]. Available: http://www.ecai2020.eu

[18] M. Stephenson, Éric Piette, D. J. N. J. Soemers, and C. Browne, "Ludii as a competition platform," 2019.

[19] Zarel, "Pokémon showdown," Web: https://pokemonshowdown.com/, 2019.

[20] coyotte508, "Pokémon online," Web: http://pokemon-online.eu/, 2019.

[21] D. Simões, S. Reis, N. Lau, and L. P. Reis, "Competitive deep reinforcement learning over a pokémon battling simulator," in *2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 2020, pp. 40–45.

[22] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[23] S. Lee and J. Togelius, "Showdown ai competition," in *2017 IEEE Conference on Computational Intelligence and Games (CIG)*, Aug 2017, pp. 191–198.

[24] A. Alfonso, "Pokémon battle," Web: https://pokemon-battle.herokuapp.com/, 2019.

[25] A. Kalose, K. Kaya, and A. Kim, "Optimal battle strategy in pokemon using reinforcement learning," Web: https://web.stanford.edu/class/aa228/reports/2018/final151.pdf, 2018.

[26] R. S. Sutton, A. G. Barto *et al.*, *Introduction to reinforcement learning*. MIT press Cambridge, 1998, vol. 135.

[27] S. Charde, "Predicting pokémon battle winner using machine learning," Submitted to conference. Web: shorturl.at/kGRS3, 2019.

[28] D. Simões, N. Lau, and L. P. Reis, "Mixed-policy asynchronous deep q-learning," in *ROBOT 2017: Third Iberian Robotics Conference*, A. Ollero, A. Sanfeliu, L. Montano, N. Lau, and C. Cardeira, Eds. Springer International Publishing, 2018, pp. 129–140.

[29] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1928–1937. [Online]. Available: http://proceedings.mlr.press/v48/mniha16.html

[30] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, "Playing atari with deep reinforcement learning," *CoRR*, vol. abs/1312.5602, 2013. [Online]. Available: http://arxiv.org/abs/1312.5602

[31] M. Bowling, "Convergence and no-regret in multiagent learning," in *Proceedings of the 17th International Conference on Neural Information Processing Systems*, ser. NIPS'04. Cambridge, MA, USA: MIT Press, 2004, pp. 209–216.

[32] S. Abdallah and V. Lesser, "A multiagent reinforcement learning algorithm with non-linear dynamics," *Journal of Artificial Intelligence Research*, vol. 33, pp. 521–549, 2008.

[33] D. Huang and S. Lee, "A self-play policy optimization approach to battling pokémon," in *2019 IEEE Conference on Games (CoG)*. IEEE, 2019, pp. 1–4.

[34] Pykalitics, "Pykalitics," Web: https://www.pikalytics.com/, 2017.

[35] O. Nachum, S. Gu, H. Lee, and S. Levine, "Data-efficient hierarchical reinforcement learning," 2018.