LVI SBPO SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL
DE 04 ATÉ 07 NOVEMBRO 2024
FORTALEZA | CEARÁ

MODEL FORMULATION AND SOLUTION FRAMEWORK FOR THE OPTIMAL BREEDING PROBLEM IN THE AXIE INFINITY GAME

Henrique José dos Santos Ferreira Junior
Federal University of Rio de Janeiro (UFRJ)
Rio de Janeiro, Brazil
henriquejsfj@cos.ufrj.br

Daniel Ratton Figueiredo
Federal University of Rio de Janeiro (UFRJ)
Rio de Janeiro, Brazil
daniel@cos.ufrj.br

ABSTRACT

Axie Infinity (AI) is a popular game that introduced the concept of "play-to-earn", allowing players to profit by playing and breeding Axies (creatures). This work presents a mathematical formulation for the AI Breeding Problem (AIBP) where the breeder buys Axies from the AI Marketplace, breeds pairs by paying a fee and generating offspring, and sells all the Axies. A key challenge is selecting breeding pairs, since thousands of different offspring and selling prices can be generated due to breeding randomness. Moreover, the number of breeding pairs grows quadratic with the number of Axies available. Thus, a two-phase framework is proposed to solve the AIBP. First, a customized Adaptive Large Neighborhood Search quickly finds profitable breeding pairs, then a solver determines the optimal breeding strategy using the profitable pairs. Validation using data from the AI Marketplace demonstrates the efficiency of the proposed approach in achieving near-optimal profits within reduced computational time.

KEYWORDS. Animal Husbandry. Optimal Breeding. Axie Infinity. Adaptive Large Neighborhood Search.

Combinatorial Optimization. Metaheuristics. Mathematical Programming.

DOI: 10.59254/sbpo-2024-193862
https://proceedings.science/p/193862?lang=pt-br

1. Introduction

Axie Infinity (AI) is an NFT-based online game[1] that focuses on digital pets (Axies) that can battle and breed. The game introduced the concept of "play-to-earn", where players can make a profit by playing and breeding Axies [Vidal-Tomás, 2022]. Moreover, the Axie Infinity Marketplace (AIM) provides a platform where Axies are sold and purchased according to the market prices[2].

However, CryptoKitties was the first online game to have a breeding mechanism using NFT creatures, but its goal was for a player to collect (or generate) rare kitties [Lu et al., 2022]. In general, the NFT-based games that have a breeding mechanism are similar to animal husbandry: NFT creatures can be selectively bred in order to improve some traits of the offspring. And as with animal husbandry, procreation in NFT-based games is also random in the sense that the offspring is not determined a priori, but its traits are chosen randomly (according to some rules) from its parents. Nevertheless, the offspring can be sold to generate a profit but its price depends on its traits.

In the context of animal husbandry, recent works use mathematical models and optimization to find the optimal breeding strategies to achieve an objective [Bernstein et al., 2021; Fleming et al., 2018]. This objective usually maximizes the occurrence of some specific traits chosen by the farmer according to the market demand in order to have larger profits when selling the offspring. However, to the best of our knowledge, no prior work has proposed mathematical models to represent and derive optimal breeding strategies for NFT breeding with game-dependent particularities.

The main objective of this work is to obtain an optimal breeding strategy for Axies with respect to the expected profit given a budget. A specific formulation for this problem is proposed, the Axie Infinity Breeding Problem (AIBP), considering a given Marketplace and breeding rules and costs imposed by the game. In particular, a breeder must buy creatures from the Marketplace, determine the pairs to breed to generate the offspring, and then sell all creatures (purchased and offspring) in the AIM.

A key problem is estimating the expected breeding selling price (EBSP) with data available from the AIM, analogous to the estimated breeding value (EBV) from animal husbandry. However, calculating the EBSP for all the pairs of creatures available in the AIM would take weeks, prompting faster approaches to determine the creatures to buy and pairs to breed. Thus, in order to solve the AIBP, a hybrid two-phase ALNS-MP framework is proposed: First, the Adaptive Large Neighborhood Search (ALNS) designed in this work generates a set of profitable pairs by computing the EBSP within a limited time; Then, a numerical solver for the AIBP uses as input the most profitable pairs obtained by the ALNS and generates an optimal solution (for the input).

The proposed approach is evaluated using a dataset obtained from the AIM. A ground truth dataset with the EBSP for all pairs is also generated and used to compare with the proposed approach. Experiments indicate the potential of the ALNS-MP framework since results for different scenarios are similar to results using the ground truth but require a much smaller running time.

The remainder of the paper is organized as follows. Section 2 presents the mathematical model for the AIBP. Section 3 presents the ANLS-MP framework detailing the ALNS designed with special operators created for this problem. Section 4 presents the evaluation scenario and a discussion of the results. Finally, parting considerations are drawn in Section 5.

2. Model for the Axie Infinity Breeding Problem

Axie Infinity (AI) created a diverse economic environment where players may choose between many strategies to achieve profit. Also, the Axies in the AI Marketplace (AIM) are sold at prices that may vary according to their genetics and meta-attributes since this influences their

[1] Non-Fungible Token (NFT) based games have artifacts (such as creatures) that are unique in the game and cannot be forged.

[2] Website for the AI Marketplace: `https://app.axieinfinity.com/marketplace/`

DOI: 10.59254/sbpo-2024-193862
https://proceedings.science/p/193862?lang=pt-br

capabilities performance in the game. In this context, our goal is to find the best strategy in terms of generating a profit for buying Axies from AIM constrained to some budget, breed them pairwise to generate offspring, and sell the offspring and the purchased Axies back to the AIM.

Not all Axies available in the marketplace are suited for making a breeding pair as there are limitations imposed by the rules of the game, for example, the game establishes a hard limit (of seven) for the number of times an Axie can be used for breeding. The following definition determines when a pair of Axies can breed.

Definition 1 *A pair p is **breedable** under the following two conditions:*

1. *The number of times each Axie in the pair was used for breeding is less than seven.*
2. *The Axies in p do not have a common parent and are not parent of one another.*

All Axies have an attribute that indicates the number of times the Axie has bred, denoted here η_c. The cost to a pair of Axies depends on the η_c of each Axie η plus a fixed cost that includes the transaction cost. This fixed breeding cost of a pair of Axies is equally divided between the elements of the pair, in order to simplify the problem formulation. Note that the selling price of an Axie decreases with the number of times it has bred. However, even Axies that have bred seven times (and cannot breed anymore) have a value since they can be used within the game.

For every breeding pair p, the set of possible offspring and the probability associated with each offspring can be explicitly determined using the genetic rules of the game. Moreover, for each possible offspring, its selling price can be estimated using the marketplace (how to estimate the selling price of an offspring is discussed in Subsection 2.1). Thus, the expected selling price μ_p of the offspring of the pair p can be computed. Note that a profit is present when the selling price of the offspring is larger than the cost of buying the pair p plus its breeding cost. Let $C_t(p)$ denote the total cost of buying and breeding once the pair $p = (\eta_1, \eta_2)$, given by

$$C_t(p) = \psi_{\eta_1} + \psi_{\eta_2} + C_b(\eta_1, 1) + C_b(\eta_2, 1) \tag{1}$$

where ψ_η is the selling price of Axie η and $C_b(\eta, 1)$ is the cost of breeding Axie η exactly once. Note that $C_t(p)$ is not random and depends only on the marketplace prices and breeding costs. Interestingly, the rules of the game are such that breeding an Axie twice has a higher breeding cost than twice the cost of a single breed, namely $C_b(\eta, 2) > 2C_b(\eta, 1)$.

While the the cost $C_t(p)$ is not random, the selling price of the offspring is random (since the offspring is random), and thus the profit is also random. However, the expected profit can be computed and is simply given by $\mu_p - C_t(p)$. Thus, the following definition determines which pairs are profitable.

Definition 2 *A breeding pair p is **profitable** under the following two conditions:*

1. *The expected profit is positive, namely $\mu_p - C_t(p) > 0$.*
2. *The probability of a positive profit is at least one half, namely $P_p[S_p > C_t(p)] \geq \frac{1}{2}$.*

Where P_p is the probability distribution over the offspring of pair p and S_p is a random variable denoting the selling price of an offspring of p. In the classic portfolio selection problem, the model objective maximizes profit, minimizes risk, or both [Rather et al., 2017]. In our scenario, the goal is to maximize profit. However, the second condition in the definition of a profitable pair controls the risk of experiencing a loss, with the fraction 1/2 chosen arbitrarily. A higher risk aversion can be represented by using a larger value.

DOI: 10.59254/sbpo-2024-193862
https://proceedings.science/p/193862?lang=pt-br

Consider a budget of value L_c available for covering the costs (both for buying Axies and for breeding pairs). Under this constraint, the goal is to determine the subset of breedable and profitable pairs of Axies available in the AIM that maximizes the return on the investment. Let y_η be a binary decision variable that indicates that Axie η was bought from the AIM. Let x_p denote a decision variable indicating the number of times that the pair p is bred. The following formulation captures the Axie Infinity Breeding Problem (AIBP) under consideration:

$$\max_{x\in\mathbb{N}^{|P|},y\in\mathbb{B}^{|H|},q_\eta\in\mathbb{N}^{|H|}} \sum_{p\in P} x_p\,\mu_p \; + \sum_{\eta\in H} y_\eta(s_{\eta,q_\eta} \; - \psi_\eta) - \sum_{\eta\in H} C_b(\eta,q_\eta) \tag{2}$$

$$\text{s.t.} \qquad q_\eta = \sum_{p\in P|\eta\in p} x_p \le L_\eta \;\forall\, \eta\in H \tag{3}$$

$$\sum_{\eta\in H} y_\eta\psi_\eta + \sum_{\eta\in H} C_b(\eta,q_\eta) \le L_c \tag{4}$$

$$\operatorname{sgn}(x_p) \le \frac{y_{\eta_1}+y_{\eta_2}}{2} \;\forall\, p=(\eta_1,\eta_2)\in P \tag{5}$$

Note that Eq. 2 has three terms. The first is the expected profit for the offspring of the breeding pairs. Note that if a pair p is bred x_p times, the expected profit is simply $x_p\mu_p$. The second term are the selling and buying prices of the Axies purchased from the AIM. Note that the selling price of a purchased Axie η will be smaller than its buying price, since the Axie was used for breeding and this increased its breeding counter by q_η. Thus, $s_{\eta,q_\eta} < \psi_\eta$ for all η and $q_\eta > 0$. The third term in the objective function represents the cost of breeding an Axie η a total of q_η times, respectively. Note that $C_b(\eta,0)=0$, as there is no breeding cost when an Axie is not bred.

Symbol	Description
H	Set of Axies available in the Marketplace.
η	An Axie from the set H.
P	Subset of breedable and profitable pairs from the set $\binom{H}{2}$.
L_c	Budget available for investment.
ψ_η	Buying price of Axie η.
L_η	Number of times the Axie η can still breed.
μ_p	Expected selling price of an offspring from the pair p.
y_η	Binary decision variable indicating that Axie η was purchased.
x_p	Decision variable indicating the number of times that the pair p is bred.
q_η	Decision variable indicating the number of times the Axie η is bred.
s_{η,q_η}	Estimated selling price of Axie η after breeding q times.
$C_b(\eta,q_\eta)$	Cost of breeding an Axie η a total of q_η times.
$C_t(p)$	Total cost of buying the pair p and breeding them exactly once (see Eq. 1).

Table 1: Table of symbols used in the problem formulation.

The first constraint, given by Eq. 3, establishes that the number of times that an Axie is bred, q_η, has to be smaller than the number of available breedings for this Axie, given by $L_\eta = 7-\eta_c$. Note that an Axie η can be used in different pairs p and different number of times for each pair, given by x_p.

The second constraint, given by Eq. 4, establishes that the cost of breeding and purchasing Axies has to be smaller than the budget L_c. Note that the first term represents the cost of purchasing the Axie η while the second term represents the cost of breeding an Axie a total of q_η, using the

DOI: 10.59254/sbpo-2024-193862
https://proceedings.science/p/193862?lang=pt-br

function $C_b(\cdot)$. Note that if an Axie is not purchased, $y_\eta = 0$, then $q_\eta = 0$ and the cost of breeding is zero. The third constraint, given by Eq. 5, ensures that when a pair $p = (\eta_1, \eta_2)$ is used for breeding, $\text{sgn}(x_p) = 1$, then both Axies η_1 and η_2 must be purchased from the AIM, namely $y_{\eta_1} = y_{\eta_2} = 1$. Note that $\text{sgn}(\cdot)$ is the sign function.

Table 1 summarizes the notation used in the Axie Infinity Breeding Problem (AIBP) formulation. Note that while the proposed problem formulation has many parameters, these parameters are determined by the rules of the game and by the available Axies and their selling prices in the marketplace. Thus, given a set H and the rules of the game, these parameters are explicitly calculated. These are μ_p, s_{η,q_η}, $C(\eta, q_\eta)$, and L_η. The only free parameter is the budget L_c which does not depend on the AIM or rules of the game. Thus, in the numerical evaluation to follow, all parameters will be determined using real data from the AIM with the exception of L_c.

A major difficulty in solving the AIBP is actually calculating the parameter μ_p from the AIM for every pair p. An Axie pair may generate up to 93,312 different offspring (according to the rules of the game), and for each offspring, it is necessary to calculate its probability and estimate the selling price. Thus, estimating μ_p can significantly influence the running time and also the quality of the solution obtained from AIBP.

2.1. Estimating the expected breeding selling price (EBSP)

The AI Whitepaper provided by [Sky Mavis] specifies the genetic mechanisms of the Axies, including the randomized breeding procedure. Briefly, each Axie belongs to a class and is composed of six parts. Each part has a dominant gene D and two recessive genes R_1 and R_2. When a pair of Axies breeds, and each gene type (D, R_1, R_2) of each part of the offspring is chosen randomly from one of its parents. Thus, the offspring can have any possible genetic combination of its parents, but different genetic combinations can have different probabilities.

The EBSP can be computed by an expected value which simply iterates over all the possible offspring combinations, calculating their probability, and estimating their selling price in the marketplace through a price estimation function (PEF). While the number of possible combinations is often very large (up to 93,312 different offspring), the major difficulty of EBSP is an accurate estimation of the PEF for a single offspring possibility within a short time. For example, the marketplace can have Axies with very similar (and even identical) genetic codes being sold for very different prices. Moreover, an Axie with the same genetic code of an offspring is often not available in the marketplace.

In order to address this problem, this work considered a simple PEF. Consider just the dominant gene in each part of an offspring, and all Axies in the Marketplace that have this same genetic code (ignoring the recessive genes). The price estimation for this offspring is the cheapest among all such Axies in the Marketplace. If this set is empty (no such Axies are available in the Marketplace), then the price estimation for this offspring is the cheapest Axie in the Marketplace. Note that this is a very conservative (pessimistic) PEF while relatively computationally efficient. The study of more accurate PEF is beyond the scope of this work. In particular, the PEF has no influence on the proposed framework but will influence the numerical results and analysis.

A fundamental problem is determining the EBSP for all breedable pairs available in the Marketplace since this is a parameter to the AIBP, namely μ_p. Our optimized implementation to compute the expected breeding selling price takes about 4.5ms for each pair. If the Marketplace has $100,000$ Axies, the pairwise combination leads to almost 5 billion pairs, and the expected breeding selling price for all pairs would take about 8 months of computation time. Moreover, most of such pairs are not profitable and thus not suited as input to the AIBP. Thus, a more computationally efficient approach, required to determine a set of profitable pairs, is explored in the next section.

DOI: 10.59254/sbpo-2024-193862
https://proceedings.science/p/193862?lang=pt-br

3. Proposed ALNS-MP Framework

This section proposes a framework to solve the AIBP given the Axies available in the Marketplace. The proposed ALNS-MP framework is composed of two stages:

1. **Find the profitable pairs:** First, the ALNS meta-heuristic tailored to this context executes to find the most profitable pair available in the Marketplace. While searching for the most profitable pair, the ALNS generates a set of profitable pairs which is the goal of this phase.

2. **Select pairs to maximize profit:** Second, the n most profitable pairs (usually, $n = 100$) found by the first phase along with the corresponding Axies of these pairs are given as input to the AIBP. An INLP solver executes to the optimal solution considering only these pairs and individuals (and not the entire Marketplace).

An initial analysis of the Marketplace indicates that pairs of Axies with different classes are very unlikely to be profitable. Thus, in the search for profitable pairs, only Axies of the same class are considered when forming pairs. In particular, Axies available in the Marketplace are grouped by class and the ANLS is executed independently and in parallel for each class. Thus, for each class, a set of profitable pairs is generated.

Figure 1 represents the proposed ALNS-MP framework in a flow chart: The set of Axies are separated by classes and fed into the ALNS; each ALNS runs independently and generates a set of profitable pairs; the profitable pairs for each class are aggregated and ordered by profit; the top n most profitable pairs along with the corresponding Axies are used as the input to the AIBP; an INLP solver analyses these profitable pairs and returns the best solution within the budget constraint. Note that the solution may contain pairs from different classes.

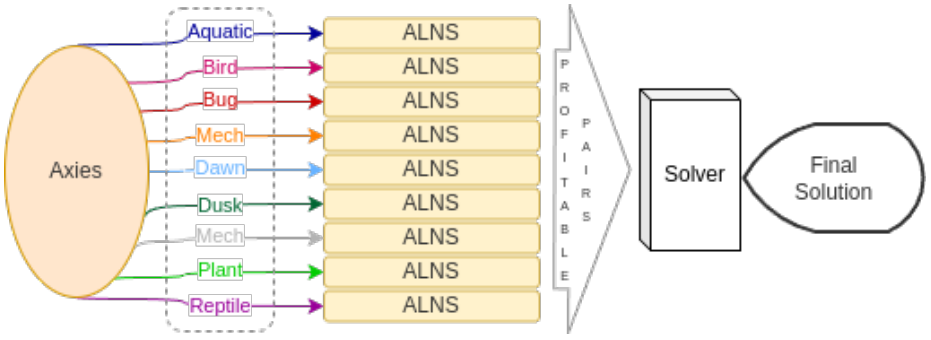

Figure 1: Flow chart of the proposed ALNS-MP framework.

Note that the proposed framework computes the expected breeding selling price inside the ANLS and only for a very small subset of all possible pairs of Axies (to be shown later). Moreover, the numerical INLP solver receives as input only n pairs (the most profitable), an even smaller subset of the profitable pairs. Thus, the ANLS-MP significantly reduces the computation time required to solve the AIBP given a real Marketplace.

3.1. ALNS Implementation

The ALNS metaheuristic introduced by [Ropke and Pisinger, 2006] extends the large neighborhood search of [Shaw, 1998] by allowing the use of multiple destroy and repair operators within the same search process. The algorithm explores the neighborhood by iterating over the Destroy Operators (DO) that remove part of the solution and the Repair Operators (RO) that complete the solution. As previously stated, the objective of the ALNS here proposed is to find the most profitable pair. So, naturally, the solution (current state) for the ALNS is a pair of Axies with the DO removing one of the Axies from the pair and then the RO adding another Axie to the solution.

DOI: 10.59254/sbpo-2024-193862
https://proceedings.science/p/193862?lang=pt-br

Our ALNS implementation is aligned and follows the guidelines found in the literature [Windras Mara et al., 2022]. In each iteration, after applying DO and RO, the new candidate solution may be accepted by using a Simulated Annealing (SA) criteria (accept if the candidate solution is superior, accept with some probability if it is inferior) from [Schrimpf et al., 2000]. The initial temperature was set as $T_0 = 5000000$ with a cooling rate $\alpha = 0.995$. The operator is selected according to the roulette-wheel mechanism with the operators' weights w updated every $\phi = 100$ iterations. When an operator is selected, a score (reward) ρ is assigned according to its performance: 5 points if the candidate solution is a new global best, 2 if it is better than the current solution, 1 if it is accepted by SA criteria, or 0.5 if it was rejected. The score of the operators are accumulated until iteration ϕ where it is updated using a decay parameter $\lambda = 0.2$ as follows, $w(k) = \lambda w(k-1) + (1-\lambda)\rho$, where $w(k)$ is a vector with the weights of all operators at the k-th update.

Although most of our implementation is classic, some modifications were necessary to achieve better results. First, we removed from the current neighborhood the last 50 visited pairs, in order to avoid local maxima. In addition, a minimum operator selection probability of 5% was enforced to avoid operator exclusion, allowing operator performance analysis. Finally, the ALNS is reset if the number of profitable pairs found does not increase for 1500 consecutive iterations. The reset sets the temperature and the operators' weights to their initial values and clears the list with the last 50 visited pairs (but keeps the current solution). Finally, the initial state of the ANLS is a randomly chosen profitable pair and the stop criterion was a time limit of 1 hour.

The upcoming subsections outline the operators that have been designed for the specific task of finding the most profitable pairs.

3.1.1. Destroy Operators

The proposed ANLS uses four different destroy operators to capture different insights concerning finding profitable pairs.

Random: Removes one Axie from the pair uniformly random. It is the destroy operator that provides the most diversification since it uses no criteria to remove a pair.

Most Expensive Breeding (MEB): Removes the Axie that costs the most to breed. Its objective is to reduce the cost and later allow the breeding of more pairs (under a fixed budget).

Lowest Partial Profit (LPP): Removes the Axie that has the lowest partial profit, where the partial profit for a given Axie η is $s_\eta - C(\eta, 1)$. The insight is that the offspring's selling price can be close to its parents' selling price and thus, a pair with a higher partial profit will likely have a higher expected profit.

Lowest Purity (LP): Removes the Axie with the lowest purity, where the purity is calculated by evaluating how many recessive genes are equal to the dominant one (D) (for each part) weighted by the probability of the recessive 1 ($R1$) and recessive 2 ($R2$). The insight is to avoid breeding Axies with large genetic variations that are likely to generate an offspring that have a low selling price.

3.1.2. Repair Operators

The repair operators can be divided into two major groups: based on gene similarity, and based on individual characteristics.

Gene-based similarity: Four different gene-based similarity repair operators are considered: Gene Random Keep (GRK), Gene Random Change (GRC), Gene Greedy Keep (GGK), Gene Greedy Change (GGC).

All gene-based operators use the same similarity function. The similarity function is used to determine a subset of Axies that are similar to an Axie in the current solution. In particular,

DOI: 10.59254/sbpo-2024-193862
https://proceedings.science/p/193862?lang=pt-br

an Axie is similar if they have three specific parts in common, where these three parts are chosen uniformly at random from the set of six parts. Thus, the similarity function is random, as similarity depends on the chosen parts. Intuitively, this allows for more diversification.

The *Keep* and *Change* operators differ with respect to the Axie used in the similarity function. *Keep* uses the Axie that was not removed after using the destroy operator, while *Change* uses the Axie that was removed by the destroy operator. Thus, the *Change* operator intuitively makes pairs that are similar to the previous pair. In contrast, the *Keep* operator intuitively makes a pair where the two Axies are more similar. Having similar Axies in the pair decreases the number of possible offspring and possibly increases the expected profit.

Individual Random (IR): Selects the replacement Axie uniformly at random from the set of profitable pairs (if there exists one). This repair operator provides the most diversification.

Individual Top Partial Profit (ITPP): Selects the replacement Axie uniformly at random from the set of the 100 most profitable pairs (if there exists one) according to their partial profit. The partial profit calculation is analogous to the LPP operator.

Individual Top Expensive Breeding (ITEB): Selects the replacement Axie uniformly at random from the set of 100 profitable pairs (if there exists one) that have the lowest breeding cost. This operator is analogous to the MEB operator.

3.2. Obtaining the Final Solution

The second stage of the proposed framework is to solve the AIBP using a numerical solver, obtaining the set of Axie pairs to buy and breed from the Marketplace. Thus, an off-the-shelf numerical INLP solver can be used for this phase.

An important consideration is the size of the input to the solver. Clearly, giving the solver all profitable pairs would allow the solver to find the optimal solution, but it may lead to unfeasible running time. On the other hand, the solver can quickly return a solution when the input is a few profitable pairs, but this solution (although optimal for this set of profitable pairs) may not be optimal for the full problem. The ALNS phase finds thousands of profitable pairs, but using all of them leads to unfeasible running time for the solver (as will be soon discussed).

Although a numerical solver can tackle the AIBP as formulated in Eq. 2, it was necessary to simplify the formulation for computational feasibility. In particular, the restriction that each pair can breed at most once was added. Thus, the decision $x_p \in \mathbb{B}$ became binary. In the original formulation, the decision variable $x_p \in \mathbb{N}$ was making the combinatorial explosion too large for the solver to finish in a reasonable time (even for small input sizes). Note that this simplification does not make the problem linear, as q_η can still be greater than one, and thus the function $C_b(\eta, q_\eta)$ still plays a non-linear role in the formulation. This modification forces the solution to include more pairs (since a pair can only be used once), and consequently, more Axies must be purchased. However, an Axie can still be present in more than one pair, breeding with different Axies.

4. Experiments and Results

Recall that the proposed ANLS-MP framework has two phases which will be evaluated separately in what follows. The proposed ANLS was fully implemented for this project. The programming language adopted was Python using SCIP from [Gamrath and et. al, 2020] as the numerical solver interfaced using the Pyomo library from [Hart et al., 2011]. The expected breeding selling price was implemented using Cython allowing the compilation and fast execution of that function, since this calculation must be performed many times (for every candidate pair, during ANLS) and takes most of the total execution time. The hardware setup had an i7 6700K (4 cores, 8M Cache, 4.20 GHz) CPU with 32GB (DDR4, 2133MHz) RAM.

DOI: 10.59254/sbpo-2024-193862
https://proceedings.science/p/193862?lang=pt-br

4.1. Dataset and ground truth

The dataset consists of data acquired from the AI Marketplace (AIM) on 2022-03-07. Although there was no official documentation concerning the Marketplace API at that time, this was circumvented and data was still retrieved. The dataset is composed of active Axie auctions available in the AIM at the time of collection. Information on $65,091$ Axies was collected from a total of $300,000$ Axies (rough estimation) present in the AIM at that time. Table 2 shows the number of Axies in the dataset per class with the number of possible pairs within the class. the Bird and Beast classes were removed because they had too few Axies.

	Aquatic	**Bug**	**Dawn**	**Dusk**	**Mech**	**Plant**	**Reptile**	**Total**
Number of Axies	25653	8910	683	7566	7300	12991	1948	65091
Possible pairs	329M	40M	233K	28M	27M	84M	2M	510M

Table 2: Number of Axies per class in the dataset (without Bird and Beast) along with the number of possible pairs within the class.

In the dataset, many Axies have very high selling prices, reaching up to millions of dollars. Although it is very unlikely that such Axies will be purchased at such a high price, they still were kept in the dataset. Selling prices is related to market volatility (how long until you sell for such a price) and is a topic beyond the scope of this work.

A Ground Truth dataset (GT) was built using the collected dataset in order to evaluate the performance of the proposed framework. The GT contains the expected breeding selling price of all breedable pairs within each class of the dataset (running time was around two weeks). The GT dataset allows for a direct comparison with the profitable pairs identified by the ANLS.

4.2. ALNS Evaluation

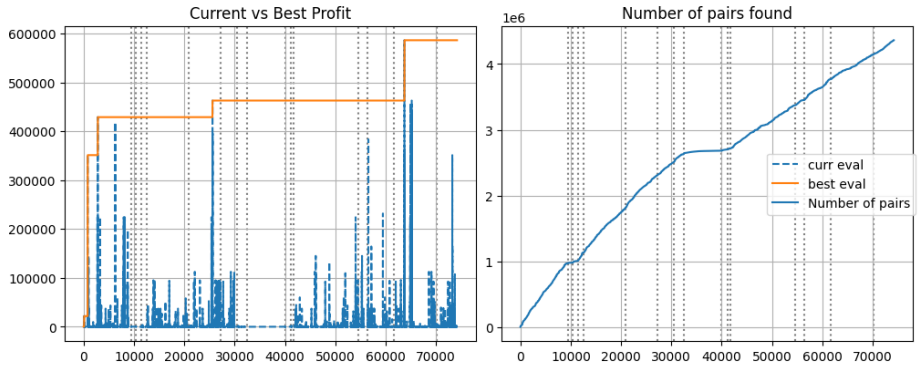

Figure 2: ALNS execution for the *Aquatic* class. Profit of most profitable (best) pair and current pair over the iterations (left); the total number of unique pairs discovered (visited) including non-profitable over the iterations (right). Vertical dashed lines indicate a reset of the ANLS.

Figure 2 shows a single execution of the ANLS for the *Aquatic* class until its termination criterion. The left plot shows the most profitable (best) pair found and the profit of the current pair over the iterations. Note that the profits found during the search vary significantly with most pairs having a small profit when compared to the most profitable pair. Interestingly, while ALNS quickly improves the profit of the most profitable pair, this continues to improve slowly over the iterations.

The right plot of Figure 2 shows the total number of unique pairs discovered (visited) across all neighborhoods explored over the iterations, including non-profitable pairs (but no duplicates). Note that the number of pairs visited grows almost linearly with the number of iterations

DOI: 10.59254/sbpo-2024-193862
https://proceedings.science/p/193862?lang=pt-br

achieving more than 4.5 million pairs in $70,000$ iterations. The vertical dashed lines indicate a reset of ALNS and this occurred a few times (due to repeated rejections), but this allowed the search to continue finding novel pairs and improving the best solution.

Due to space limitations, the ALNS performance is not shown for the other classes. However, most of them had similar performance, and details can be found in an extended report on the corresponding thesis [Ferreira Júnior, 2024].

Table 3 shows the total number of pairs visited and profitable found by ALNS within the stopping criterion (i.e., 1h of execution) for each class. The table also shows the number of pairs and profitable pairs in the GT dataset. Considering the *Aquatic* class, ALNS found $12,914$ profitable pairs from the $25,011$ present in the GT, however, ALNS visited around 4.5 million pairs while GT considered all the 329 million pairs. Thus, considering only 1.4% of the pairs in this class, ALNS found 52% of the profitable pairs. Moreover, ALNS found 8 of the top 10 most profitable pairs for this class.

		GT		ALNS			
Class	**No. Axies**	**Visited pairs**	**Profitable pairs**	**Visited pairs**	**Frac.**	**Profitable pairs**	**Frac.**
Aquatic	25653	329M	25011	4.5M	1.4%	12914	52.0%
Bug	8910	40M	679	1.4M	3.5%	582	85.7%
Dawn	683	233K	462	80K	34.3%	457	98.9%
Dusk	7566	28M	30	23K	0.1%	0	0%
Mech	7300	27M	1097	1.8M	6.7%	994	90.6%
Plant	12991	84M	31394	3.5M	4.2%	19004	60.5%
Reptile	1948	1.9M	5728	900K	47.4%	5685	99.3%
Total	65091	510M	64401	19.4M	3.8%	39636	61.5%

Table 3: Number of pairs and profitable pairs discovered by ALNS and GT.

Remarkably, this trend observed for the *Aquatic* is also present in all other classes, as shown in Table 3. In essence, ALNS can find a large fraction of the profitable pairs by visiting a very small fraction of the total number of pairs of a class. Moreover, when considering the top 10 most profitable pairs, ANLS found at least 6 of them for all classes (not shown in the table). Last, recall that ANLS had just one hour of execution while the GT took around two weeks to compute, indicating its advantage in generating sets of profitable pairs.

4.3. Optimal Strategy Evaluation

Recall that the set of profitable pairs for each class is merged into a single set and the top-N pairs from this set are used as input to the AIBP, as presented in Section 2. The solution of the AIBP is the set of pairs to breed along with the obtained profit. Note that the solution depends on the value of N and also on the budget L_c. The value of N directly influences the running time of the solver while small values of N could lead to lower profits within the budget.

Moreover, the set of profitable pairs in the top-N also influences the optimal profit. Thus, a direct comparison between the GT and ANLS is also presented, comparing the optimal profit when considering the top-N pairs from the GT with the optimal profit when considering the top-N pairs identified by the ALNS.

Results are shown in Table 4 for different top-N values and different budgets. Note that for budget $L_c = 500$ the optimal profit is identical for GT and ANLS, for all values for N. As the budget increases, the gap between the optimal for the GT and ALNS increases. However, N does not influence the profit results until the L_c becomes 25000. This occurs because $N = 100$ pairs are not enough to spend the high total L_c value.

DOI: 10.59254/sbpo-2024-193862
https://proceedings.science/p/193862?lang=pt-br

Budget ($)	Top-N (pair)	GT ($)	Time (min)	ALNS ($)	Time (min)	DIF
500	100	878299	0.01	878299	0.01	0.00%
	1000	878299	0.39	878299	0.28	0.00%
	10000	878299	7.48	878299	6.08	0.00%
1000	100	1858473	0.05	1768412	0.04	4.85%
	1000	1858473	1.17	1768412	1.03	4.85%
	10000	1858473	27.35	1768412	23.55	4.85%
2500	100	4628870	0.19	4302032	0.10	7.06%
	1000	4628870	4.97	4302032	2.75	7.06%
	10000	4516948	>240	4302032	168.96	4.76%
5000	100	8474520	0.62	7922102	0.15	6.52%
	1000	8474520	16.37	7922102	3.57	6.52%
	10000	8474520	>240	7922102	>240	6.52%
7500	100	11736000	0.48	10668540	0.11	9.10%
	1000	11736000	14.54	10668540	2.66	9.10%
	10000	11736000	>240	Timed out	>240	
10000	100	14438661	0.37	12777580	0.12	11.50%
	1000	14438661	13.02	12777580	3.40	11.50%
	10000	14438661	>240	Timed out	>240	
25000	100	23685934	0.01	19022066	0.02	19.69%
	1000	24299523	35.81	19022066	23.74	21.72%
	10000	24232097	>240	19022066	>240	21.50%
50000	100	24554616	0.00	19977287	0.00	18.64%
	1000	29528647	>60	23422033	>60	20.68%
	10000	Timed out	>240	Timed out	>240	
100000	100	24554616	0.00	19977287	0.00	18.64%
	1000	34885870	>60	26287393	>60	24.65%
	10000	Timed out	>240	Timed out	>240	

Table 4: Optimal profit values and running time when using top-N pairs from the GT and from ANLS for different budgets and values for N.

Table 4 also shows that the running time increases with N for all budgets. Moreover, when the budget is large, the solver does return a solution for larger values for N. However, it always found a solution for $N = 100$. Interestingly, for the same N value, the running time when using ALNS pairs is smaller than using GT pairs (the solver received the same number of pairs). This can be due to the lack of compatible pairs that reuse Axies in different pairs, leading to fewer feasible solutions in the ALNS set than in the GT set. This also suggests why the optimal profit for the ALNS is smaller than for the GT.

5. Conclusion

This work proposed the Axie Infinity Breeding Problem (AIBP), a mathematical formulation to breed Axies in order to maximize expected profit given a budget and the Marketplace. In order to solve the AIBP in a feasible time, a hybrid two-phase framework is proposed ALNS-MP. The tailored-designed ANLS finds a set of profitable pairs (first phase) and the numerical solver decides which pairs to purchase and breed given as input the most profitable pairs (second phase).

The ALNS-MP framework is evaluated using real data from the Marketplace and compared with a ground truth dataset with the profit of all pairs. Interestingly, the ANLS finds 61.5% of the total profitable pairs exploring only 3.8% of the total number of pairs. Note that even separating

DOI: 10.59254/sbpo-2024-193862
https://proceedings.science/p/193862?lang=pt-br

Axies by classes, only 0.01% of the total number of pairs are profitable. Moreover, the optimal expected profit using top ANLS pairs is similar to using top pairs from the ground truth dataset.

Last, while the solution of the AIBP indicates large expected profits for the breeder, the price estimation function (PEF) should be designed to more accurately predict offspring price and consequently the expected profit.

References

Bernstein, R., Du, M., Hoppe, A., and Bienefeld, K. (2021). Simulation studies to optimize genomic selection in honey bees. Genetics Selection Evolution, 53(1):1–17.

Ferreira Júnior, H. J. d. S. (2024). Profit model and optimal breeding strategy applied to the axie infinity game. Master Thesis, Universidade Federal do Rio de Janeiro (UFRJ).

Fleming, A., Abdalla, E. A., Maltecca, C., and Baes, C. F. (2018). Reproductive and genomic technologies to optimize breeding strategies for genetic progress in dairy cattle. Archives Animal Breeding, 61(1):43–57.

Gamrath, G. and et. al (2020). The SCIP Optimization Suite 7.0. Technical report, Optimization Online.

Hart, W. E., Watson, J.-P., and Woodruff, D. L. (2011). Pyomo: modeling and solving mathematical programs in python. Mathematical Programming Computation, 3(3):219–260.

Lu, C., Lauritano, G., and Peltonen, J. (2022). Cryptokitties vs. axie infinity: Computational analysis of nft game reddit discussions. In Intern. Conference on ArtsIT, Interactivity and Game Creation, p. 105–120.

Rather, A. M., Sastry, V., and Agarwal, A. (2017). Stock market prediction and portfolio selection models: a survey. Opsearch, 54:558–579.

Ropke, S. and Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transportation Science, 40:455–472.

Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., and Dueck, G. (2000). Record breaking optimization results using the ruin and recreate principle. Journal of Computational Physics, 159(2): 139–171.

Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. In Maher, M. and Puget, J.-F., editors, Principles and Practice of Constraint Programming — CP98, p. 417–431.

Sky Mavis. Axie infinity whitepaper. `https://whitepaper.axieinfinity.com/`. Accessed:14/01/2024.

Vidal-Tomás, D. (2022). The new crypto niche: Nfts, play-to-earn, and metaverse tokens. Finance research letters, 47:102742.

Windras Mara, S. T., Norcahyo, R., Jodiawan, P., Lusiantoro, L., and Rifai, A. P. (2022). A survey of adaptive large neighborhood search algorithms and applications. Computers & Operations Research, 146:105903.

DOI: 10.59254/sbpo-2024-193862
https://proceedings.science/p/193862?lang=pt-br