BACHELOR

Strategizing Single Player Video Games
Using mixed integer linear programming techniques to maximize income in Stradew Valley

de Weerd, Roel A.R.

Award date:
2023

Awarding institution:
Tilburg University

STRATEGIZING SINGLE PLAYER VIDEO GAMES

USING MIXED INTEGER LINEAR PROGRAMMING TECHNIQUES TO MAXIMIZE INCOME IN STRADEW VALLEY

Bachelor Thesis
Bacholor End Project

R.A.R. de Weerd
1258257(TU/E), u535252 (UvT)

Supervisor:
M. Delorme

Final Version

Tilburg, June 2023

Eindhoven University of Technology
Department of Mathematics and Computer Science

Tilburg University
Tilburg Law School

Contents

Abstract

In this thesis, a possibility is researched how to make strategies for single player video games. More precisely, a strategy is defined and optimized, how to maximize income in the game Stardew Valley by ConcernedApe [24] using mixed integer linear optimization (MILP). It is described how different aspects of the game can be defined as variables for a (M)ILP model, as well as how other aspects can be represented by correctly defining constraints. Moreover, it is shown how to deal with non-linear features, and how auxiliary variables can help to correctly detail these features, while keeping the model linear. The use of expected values is described as a way to deal with stochasticity, and the effect of luck in these situations is briefly shown. Big resources are described, and conditionally constrained, to maximize the accuracy of the strategy and thus stay as close to the game's boundaries and limitations. In the end, the strategy made gets compared to different man-made strategies. Added on that, the limitations are stated. In this case it means that more possible aspects of the game are discussed, which are a considerable part of the game, and for which the model could possibly alter its strategy.

1 Introduction

1.1 Game Theory

Strategizing video games should already be a familiar concept. For most games, players need to have or develop a general idea of what they are doing to get a better grip of the game. These players , also commonly called gamers, tend to become better at games by continuously evaluating their options and acting reactively. This approach to gaming is closely related to game theory. The type of analysis of correctly identifying options, and choosing the 'best' way, has been around for over a century. As described by D. Ross [21], game theory is the study of the ways in which interacting choices of economic agents produce outcomes with respect to the preferences (or utilities) of those agents, where the outcomes in question might have been intended by none of the agents.

Starting in mid 20th century, more and more research and development towards game theory was conducted. It was concluded that game theory related perspectives had several important contributions towards sociology and economics. Game theory has expanded the repertoire of social theory as well as its analytical potential, and it has added to the growth of mathematical sociology [26]. Game theory has also become a staple tool in economics and economist regularly combine game theory with work in other fields [22]. The spread of game theory towards other areas of science can be quickly seen in multiple studies, for example in biology [15], in electrical engineering [13], in operations research [23], and more.

1.2 Game Theory in Video Games

Video games are in general games, which gets played on some kind of audiovisual apparatus [7]. To add on that, the definition of a game gets noted by using the definition (Zimmerman, 2004 [30]) has given it:

"A game is a voluntary interactive activity, in which one or more players follow rules that constrain their behavior, enacting an artificial conflict that ends in a quantifiable outcome."

Most video games include some type of goal, with resources to help players get to that goal, limited by boundaries and consequences in the form of penalties and rewards [17]. Therefore, game theory can often be used to predict outcomes of games, or the goal players work towards, depending on the choices they make. The boundaries of video games are closely related to the environment for which games are defined. Following from that, players can often derive such set of choices, for which the outcome come closest to the goal they want to achieve. These set of choices will be the optimal strategy. For smaller games, defined by less options of choices, optimal strategy can be derived quickly and seem may obvious, but the larger the game the more complex it can get. Moreover, it is also possible for the game to have multiple optimal solutions, or outcomes so similar, that it is even more difficult to derive the best strategy. This complexity is what attracts most gamers to video games. If every game was quick and simple to optimize, the fun of games would be very limited.

1.3 Theorycrafting

The search for better strategies is nothing new. Paul, C.A. [16] describes the increasing practices of theory crafting and how it defines how games get played. In his study revolving around *World of Warcraft (WoW)*, he describes theorycrafting as:

"Theorycraft, a strategy designed around the mathematical analysis of WoW, is a discursive construct predicated on advising players how to optimally 'play' WoW, suggesting what equipment to wear, what talents to choose, and an order in which to cast spells."

Theory crafting is mainly used to make gamers better players, and to make them have a better understanding about the game. Even though video games are sold and played with the main focus of having fun, and being a leisure activity, a lot of players tend to feel the need to become better at the game. This winning behaviour was more seen by men than women [29]. However, theory crafting is also seem as enjoyable and fun [16]. The engagement into something players enjoy is entertaining. Additional, some players link winning with fun, which makes the thought of constructing better winning ideas also fun. (Wenz, K. 2013 [28]) adds on this by stating that theorycrafting tools (for example trackers of proffesional builds, strategies, streams etc.) are used by players for surveillance and sousveillance, to both observe and control from those in power to lesser powerful players, and vice versa. Players look or listen to high level players to see what they do, which decisions they make compared to their own. But it works also the other way, high level players are expected to play 'good' and act alike due to them knowing they are being watched seen or heard.

1.4 Objective

Arriving at the goal of this thesis. Using known optimization techniques, strategies will be crafted for (single player) video games, and compare them to regular, men-made strategies, to see if these optimizations techniques can increase the potential outcome.

1.5 Stardew Valley

This research will be conducted in the environment Stardew Valley by ConcernedApe [24]. It is a mainly peaceful farming RPG (Role Playing Game) simulator where the player controls one character. This character, as can be seen in the introduction of a new game, gets sick and tired working for a mega cooperation and decides after a letter from its grandpa, showing that he has inherited a farmland on a small island far away, to pursue a life there. The player starts in spring 1st year 1, and can play trough out the 4 seasons. Each season consist of 28 in-game days or 4 weeks. From there, the player can farm crops fig. 1, fish fig. 2, mine fig. 3, make money, complete quests, upgrade its farm or farm buildings, and more. The possibilities are endless, and there is no clear win-condition or goal, and promotes creativity in doing whatever the player wants. Even though Stardew Valley is meant to be a refreshing way of escapism towards the countryside, to simulate a simpler and minimalistic live, players are quickly incentivized towards capitalistic and neo-liberalistic values. The game is oriented towards perpetual economic growth and optimizing activities, leading towards maximizing income [6]. That is also where the scope of this thesis will be set towards, maximizing income. Led by the players decisions on crop choices and time spend fishing, this thesis will dive deeper towards the possibilities of different strategies, and try to find optimal solutions.

Figure 1: Farming

Figure 2: Fishing

Figure 3: Mining

1.6 Real Life Cases

As can be seen in real-live cases: (Alotaibi and Nadeem [1] 2021) and (Filippi, Mansini and Stevanato [9] 2017), linear programming methods are already used for crop choices and rotations. The first paper by Alotaibi and Nadeem describes a way to use linear programming to maximize production and thus income of farmers. It emphasizes on correctly allocate resources and decision making, to obtain more efficiency. Some resources discussed are labor, fertilizers water supply, etc. The second paper by Filippi, Mansini and Stevanato describes 2 MILP models, to obtain optimal crop selection for maximizing income. It is based on a real case of a farmer in Northern Italy. It looks for an optimal crop mix, while dealing with expected profits, due to prices and yield being stochastic. It is something which will be later discussed in this thesis as well. It takes into account the different actions which has to be done, like ploughing, sowing, hoeing, watering, etc. And the main resources they require, time and money. It's second model looks for maximizing the expected profit while using a conditional value-at-risk, a predefined percentile of worst realizations.

These techniques can also be used for planning crop choices and more in Stardew Valley, due to the same basis, but different hurdles and limitations. The in-game boundaries resemble real life boundaries to a certain degree, and the objective is in both cases the same, maximizing income/efficiency. The research question will then be denoted as:

How can we maximize in-game income in Stardew Valley, using a mixed integer linear programming optimization approach.

1.7 (M)ILP modeling

Modern-day linear programming (LP) started with the work of Dantzig in 1947, at least it it perceived that way for most people. His use of LP for not only economic problems, but also real world problems was in contrast to economist of that time. He also developed the simplex algorithm for solving LP problems, an algorithm which is still a primary tool in linear and mixed integer linear programming (MILP) [3]. LP modeling is the art of translating problems to a combination of linear variables and constraints, together with a linear objective function. It then becomes integer linear programming when all the variables have the condition to have integer values. This makes optimizing for the objective function a lot harder. One can think of scheduling problems, like making school timetables[14]. Due to the fact that both teachers and classes can not be at two places at the same time, thus only rounded numbers (= integers) can be used for this scheduling problem. Other common problems solved by ILP are for example resource allocation [20] and traveling salesman [31]. These ILP models turn into mixed models, when some variables are integer bounded, and some are not. Some examples of problems solved by MILP are more scheduling problems [12], planning operations for energy systems [2] and the previously mentioned crop selection problems.

2 Methodology

2.1 General Approach

To design an advanced strategy for maximizing income, is not as straight forward as one may think. In theory, there are endless possibilities of things to do, as it is somewhat of a open world where one can walk and act in whatever way one can imagine, within the limit of the game of course. Therefore, to make a advanced strategy, one has to simplify first. Once one basic ILP model is working and correctly resembling the game, one can expand the model with more and more aspects of the game. In this thesis, the structure and the result of such basic model will be shown and elaborated further upon, to give better insight in how strategies can be extracted from these ILP models. Then, the final model will be shown, and which strategies arise from it. Thus, the end product should be multiple MILP models, differentiating themselves by the fishing and farming luck, which maximizes the in-game income at the end of day 28 spring year 1 (one in-game season). This is chosen because of all the crops perish after day 28 and a new rotation of different crops emerge for the new season. The model is constrained by resources representing in-game time and energy usage, as well as no negative balance at any point in game, and limited by niche in-game limitations, like in-game events.

2.2 Data Collection

Most of the data used can be found on the extensive wiki of the game [25]. These contain prices of all crops and fishes, the prices of their higher quality versions and how long it takes to grow. One could also look into the decompiled code of the game. It is not the full source code, but a approximation of it. This decompiled code is a readable version of the machine code in a C$^\sharp$ style. If these sources can not provide the data needed, then the game can also be played of course, to obtain the necessary data.

2.3 Programs Used

For this research, Python [19] was used as the programming language. Its advanced packages for data analysis like pandas [27] are very usefull and quick. The data obtained can be munually written over to pandas' Dataframes. Due to the smaller scale of this data, this is very doable in a reasonable amount of time. The ILP models are constructed using the Gurobi solver [10], which is an advanced linear solver for all kinds of linear programming complications.

2.4 Basic Model

To show how daily plans can be derived, a basic model will be defined first. This model will only have the simple, one time harvest subset of all available crops in spring year 1. It will show which combination of seeds, to be bought and planted on certain days, will lead to the largest amount of income at the end of the season. To arrive at such model, first we will define some information vectors and matrices. Then the variables gets defined. And lastly the objective function and the constraints of the linear model get defined.

Firstly, denote D as a list of days, ranging from 1 to (& including) 28.
$D = [1, 2, .., 27, 28]$

Then, denote C as a list of all possible crops, thus all 1 time harvest crops available in year 1 spring.
$C = [BlueJazz, Cauliflower, Kale, Parsnip, Potato, Tulip]$

Denote CI (Crop Information) as a matrix of the buy, sell prices of crops and the days required for the crop to grow. These days to grow is actually the amount of nights which have to pass, as farming is a from day to day activity. It does not matter at what in-game time the seeds gets planted during the day, they will always be ready after the right amount of passed nights. See table 1.

CI	Buy Price	Sell Price	Days to Grow
Blue Jazz	30	50	7
Cauliflower	80	175	12
Kale	70	110	6
Parsnip	20	35	4
Potato	50	100	6
Tulip	20	30	6

Table 1: Crops prices & Harvest times

Variables:
Now, a Plan matrix P gets created. It denotes how many seeds need to be bought on which day. This will be defined as integer, as players cannot buy or plant halve seeds. This will be the main tool for which the solver can tweak its values, to search for better objective values (which will be defined later).

$P_{dc} \in \mathbb{Z}$ with $d \in D$ & $c \in C$

The second variable matrix is a money matrix M. This matrix is one which keeps track of how much money will be earned each day, how much it has started the day with and how much it ends the day with. This matrix will be heavily constrained, so it cannot magically make more money appear.

M_{dm} with $d \in D$ & $m \in [StartDay, Earnings, EndDay]$

Objective function:
The goal is to end up with as much money as possible at the end of day 28. Thus if the model adds up the amount the player ends the day on with the amount of what they have earned, it ends up with the desired outcome.

Maximize $M_{28,Earnings} + M_{28,EndDay}$

Constraints:
Constraint 1: Non negativity in both P and M. Players can't have negative balance in the game, can't have negative earnings and can't buy a negative amount of seeds.

$$(1.1) \quad P_{dc} \geq 0 \qquad \forall_{d,c}[d \in D, c \in C]$$
$$(1.2) \quad M_{dm} \geq 0 \qquad \forall_{d,m}[d \in D, m \in [StartDay, Earnings, EndDay]]$$

Constraint 2: Players end up the day with the amount they have started with, minus the amount they have spent that day on seeds.

$$(2) \quad M_{d,EndDay} = M_{d,StartDay} - \sum_{c \in C}(P_{d,c} * CI_{c,BuyPrice}) \qquad \forall_d[d \in D]$$

Constraint 3: Players earnings are what they harvest that day. Thus what they have planted the exact amount of days it takes for each crop to grow, multiplied by the price they sell for. This can only occur if d minus the days to grow are larger than 0 (Players can't have planted crops earlier than day 1). Therefore, $harvest_{dc}$ gets defined as the set of crops which are ready to be harvested.

$harvest_{dc} = P_{(d-CI_{c,DaysToGrow}),c}$ if $d - CI_{c,DaysToGrow} \geq 1$

Also note that if there aren't any crops it could have harvested already, then those days automatically end up with 0 earnings.

$$(3) \qquad M_{d,Earnings} = \sum_{c \in C} (harvest_{dc} * CI_{c,SellPrice}) \qquad \forall_d [d \in D]$$

Constraint 4: Players start the next day with the money they ended with the day before, and what they earned the day before. Day 29 does not get calculated because that is the objective function.

$$(4) \qquad M_{d+1,StartDay} = M_{d,EndDay} + M_{d,Earnings} \qquad \forall_d [d \in D \setminus \{28\}]$$

Constraint 5: Players start day 1 with 500 gold and 15 parsnip seeds. This can be translated to obligation of buying at least 15 parsnip seeds, and starting with 500 + 15*20 = 800 money.

$$(5.1) \qquad M_{1,StartDay} = 800$$
$$(5.2) \qquad P_{1,Parsnip} \geq 15$$

All this combined gives a basic model, which chooses when to buy which seeds, for which income is maximized after day 28.

2.5 Full Model

To clarify the full model, it will be explained by dividing it into different parts. For each part the relevant variables and constraints get elaborated upon. Note that the constraints numbering are respectively to the constraint numbering in the program, and hence not ordered. The objective function stays the same for the full model:

$$\textbf{Maximize } M_{28,Earnings} + M_{28,EndDay}$$

2.5.1 Assumptions

Before the main model gets defined, some assumptions are listed, for what the model is based on.

- The model is only a strategy planner for day to day activities. It resembles what can be done each day to maximize the player's income at the end of the month. Therefore it is of course not a simulation of any kind of the game.
- One other main assumption is that the player only is interested in farming and fishing. Even though the game has more aspects like mining or foraging, the lack of (direct) income following those is what narrows this strategy down.
- The in-game time it takes to plant, water and harvest are taken from a average stand point, as in the player knows what to do, but it may not be the fastest as it can be. These values are estimated by playing and gathering data myself, and thus may be slightly off to what others experience. Also there exist animation cancelling tricks to speed up things, but this does not get seen as a aspect of the game. This is to strategize for a general player and not a speed-runner.
- Income is at all times affected by RNG. RNG, or random number generators, is the most common method for games to approach random events and introduce different probabilities for different outcomes. It is not fully random, as it is beyond of computer capabilities. In Stardew Valley, randomness gets determined by various factors, like the game seed, current day, coordinates of the action, time of the action, daily luck (which gets determent by more actions the day before), amount of actions already done, and much more. Considering all of this, it decided in the main model, to use the expected value of payout gets used. It wont differ

much for income from farming, but fishing has a high variance, where both the exp gain and the potential income gets affected by RNG. This results in more uncertaincy.

- Players sell their crops and fish via the shipping bin, which is a big crate on the players farm. This gets emptied at the end of every day when the player goes to sleep, thus when the day ends. in between days, the income gets then added to the players balance. It is possible to sell some items to shopkeepers directly, to earn income trough out the day instead of afterwards, but this does not get considered as an option.
- There exist rainy days, where the crops do not have to be watered. This of course saves some time and energy, but is from day 5 and onwards fully random (with day 3 being a rainy day and day 1,2,4 being a regular sunny day). The chance of a rainy day is about 24%. This would complex this model heavily, and hence the model does not consider any rainy days. This is, of course, less favourable for the amount of crops it will consider.
- The most profitable place to fish in spring year 1 is, by a considerable amount, the mountain lake. Thus all the fish will be considered to get caught there. It is also considered that none of the the fish break out (failed catch), and no fish will be caught with a perfect catch (this is in regards to potential more exp).
- A day can be played with 20 in game hours, 06:00 AM to 02:00 AM. However, sleeping after 24:00 leaves the player with less energy the day after, which could invalidate the result. therefore the player has at most 18 hours of in-game time.
- Pierres shop is closed on Wednesdays, which is the place most people gather their seeds. Seeds can still be bought at the so called Jojo-Market, for a higher price. However, this also is not taken as a option, and thus the model will not have any plant days on Wednesday.

2.5.2 Farming

Fisrt of, some of the information matrices/lists gets adjusted. While D stays the same, C gets 2 new crops: Green Bean and Strawberries. Then CI gets adjusted accordingly. Here both new crops gets added, as well as that the input of Days to Grow becomes a list. The first value of the list is the first harvest, each value afterwards are the multiple harvests. Note that Strawberries can only be bought on day 13, so players can only harvest them twice before day 28, and thus don't need more values. A column with exp gained upon harvest also gets added. See table 2.

CI	Buy Price	Sell Price	Days to Grow	Exp
Blue Jazz	30	50	[7]	10
Cauliflower	80	175	[12]	23
Kale	70	110	[6]	17
Parsnip	20	35	[4]	8
Potato	50	100	[6]	14
Tulip	20	30	[6]	7
Green Bean	60	40	[10, 13, 16, 19, 22, 25]	9
Strawberries	100	120	[8, 12]	18

Table 2: Crops prices & Harvest times updated

The different seeds which gets bought on different days still get tracked by P_{dc}, with integer values.

Constraints:
First of, P_dc still cannot be negative.

(1.1) $P_{dc} \geq 0$ $\forall_{d,c}[d \in D, c \in C]$

Following, Player still start with 15 parsnip seeds. This can be translated to a obligation of having to buy at least 15 parsnip seeds on day 1.

(5.2) $P_{1,Parsnip} \geq 15$

Next up, two constraint as a consequence of the in-game event, the flower dance. This makes it so that players can't buy any other seed on day 13 and players can't buy strawberries on any other day then day 13.

(6) $\sum_{c \in C \setminus \{Strawberries\}} P_{13,c} = 0$

(7) $\sum_{d \in D \setminus \{13\}} P_{d,Strawberries} = 0$

Players cannot buy seeds on Wednesdays:

(9) $\sum_{d \in [3,10,17,24]} (\sum_{c \in C} P_{d,c}) = 0$

2.5.3 Farming Experience

Farming is considered to be a skill in Stardew Valley. Each time crops get harvested, they yield experience, or exp in short, for this skill. The exp gained never gets lost, and thus generally increases as time goes on. Once certain thresholds of total exp is gained, levels get earned. This ranges from level 0 up and to level 10. Each level gives more benefits to the player, like more income and lower recource costs. However, to limit the model from becoming increasingly complex and time-consuming, the levels get split up into 4 different classes, one for the lower levels (0 to 5), one for the mid levels(5-7), one for the high levels (8-9) and finally one for if max level is reached. These level proportions are chosen like this, to indicate significant differences. Especially in the lower levels, there is not much difference in payout between those.

First of, the experience gets tracked by E_{de} with $d \in D$ and $e \in [FarmTot, FarmEarn]$. This variable contains both the total exp and the exp earned every day. There are also three binary milestone variables, which turn 1 once the right amount of exp is gathered, and 0 otherwise. $FarmMid_d$, $FarmHigh_d$, $FarmMax_d \in \{0, 1\}$ with $d \in D$. If all are 0, then the player is still considered to be in the lowest class.

Constraints:
Exp, total and earned, cannot be negative.

(1.4.1) $E_{de} \geq 0$ $\forall_{d,e}[d \in D, e \in [FarmTot, FarmEarn]]$

Total farming exp the next day is total farming exp from the previous day plus exp earned the previous day.

(14.1) $E_{(d+1),FarmTot} = E_{d,FarmTot} + E_{d,FarmEarn}$ $\forall_d[d \in D \setminus \{28\}]$

Players start with 0 exp.

(15.1) $E_{1,FarmTot} = 0$

And most importantly, players gain exp for what they harvest. Quality of corps does not increase the amount of exp yields upon harvest, and hence the player farming level has no impact on it. Here almost the same definition for $harvest_{dc}$ is used, only difference is that it has to be adjusted for multiple harvests:

$harvest_{dc} = P_{(d-CI_{c,DaysToGrow}),c}$ $\forall[CI_{c,DaysToGrow}]$ if $d - CI_{c,DaysToGrow} \geq 1$

$$(16) \quad E_{d,FarmEarn} = \sum_{c \in C} (harvest_{dc} * CI_{c,Exp}) \qquad \forall_d[d \in D]$$

Finally, some logical constraints are defined to correctly identify whether the required amount of exp has been reached. Here each class will have a constraint which translates to 'if exp $\geq$ threshold, then binary var $=$ 1'. This can be obtained by use of $\mathbf{M}$. It represent a reasonable large number. The multiplication with a binary value and this $\mathbf{M}$ can make a constraint binding or non-binding. It will also have a counter part (an else part), to ensure no false classifications are given. This counterpart will translate to 'if exp $<$ threshold, then binary var $=$ 0'. In linear optimization, there is not such thing as strict inequality. To bypass this, ϵ gets used, as a reasonable small number.

$$(18.1) \quad E_{d,FarmTot} \geq 2150 - \mathbf{M} * (1 - FarmMid_d)$$

$$(18.2) \quad E_{d,FarmTot} \leq 2150 - \epsilon + \mathbf{M} * FarmMid_d$$

$$(18.3) \quad E_{d,FarmTot} \geq 6900 - \mathbf{M} * (1 - FarmHigh_d)$$

$$(18.4) \quad E_{d,FarmTot} \leq 6900 - \epsilon + \mathbf{M} * FarmHigh_d$$

$$(18.5) \quad E_{d,FarmTot} \geq 15000 - \mathbf{M} * (1 - FarmMaxd_d)$$

$$(18.6) \quad E_{d,FarmTot} \leq 15000 - \epsilon + \mathbf{M} * FarmMax_d \qquad \forall[d \in D]$$

2.5.4 Fishing

The fishing in Stardew Valley is a bit more complex. It is a stochastic, and near random process, where each time the player cast their rod towards the water, it either, after some time passes, gives immediately a algae or trash, or it starts a mini-game where it will result in catching a fish. This process is though to reproduce, due to the extreme randomness. Once the player cast their rod, it shuffles all possible outcomes, except trash, in a random order. From the top of the list, it checks with RNG and some hidden values of the fish, if that fish takes the bait. Else it goes to the next one in line. If all RNG rolls on the possible fish turns out to be false, it returns trash. Moreover, on different times during the day, different fish can be caught. As well as that the players fishing level and casting distance (which also increases with casting distance) change the possible outcome/odds of the fish which can be caught. This is why the occurrence odds are used, made by Youtuber BlaDe [4], published on Stardew Valley forums. He made a collection of google sheets, which simulated all possible outcomes of a single fish cast, taken into account the time of the day, fishing level and depth, location and sunny or rainy day appendix A.2. The latter two will not change for our values. With these, the expected payout per cast and expected exp gain per cast will be calculated, and adjusted once fish luck gets adjusted. This can be seen in FI, table 3. The same level classes will be used as at farming. More on this later.

For now, $F_d \in \mathbb{Z}$ with $d \in D$ gets defined as the amount of casts, each day.

Constraints: Once again, there is a non-negative constraint.

$$(1.3) \quad F_d \geq 0 \qquad \forall_d[d \in D]$$

Fishing starts on day 2, and players can not fish on the 13th.

Fishing Class	Fishing level	Depth	Expected payout	Expected exp gain
Low	0	1	fm_{low}	fe_{low}
Mid	5	3	fm_{mid}	fe_{mid}
High	8	5	fm_{high}	fe_{high}
Max	10	5	fm_{Max}	fe_{Max}

Table 3: FI, Fish Information

$$(10) \qquad \sum_{d\in[1,13]} F_d = 0$$

2.5.5 Fishing Experience

Fishing is also a skill in Stardew Valley, with its own exp tracker. Each time a fish or something else gets caught, it gives a certain amount of exp, depending on the type of fish and the base quality of it. The base quality increases when the player level increases and when the depth of the cast increases. This is a little bit extra randomness, which is also considered when calculating the expected exp. For the model, fishing also gets tracked by $E_{d,e}$ with $d \in D$ and $e \in [FishTot, FishEarn]$. As said before, Fishing gets the same classes as farming, thus similar binary variables, denoted as $FishMid_d$, $FishHigh_d$, $FishMax_d \in \{0,1\}$ with $d \in D$.

Constraints:
Once again, non-negativity.

$$(1.4.2) \qquad E_{de} \geq 0 \qquad \forall_{d,e}[d \in D, e \in [FishTot, FishEarn]]$$

The total fishing exp is the total fishing exp from the day before, plus what is earned the day before.

$$(14.2) \qquad E_{(d+1),FishTot} = E_{d,FishTot} + E_{d,FishEarn} \qquad \forall_d[d \in D \setminus \{28\}]$$

Players start with 0 fishing exp.

$$(15.2) \qquad E_{1,FishTot} = 0$$

Due to the fact that fishing exp scales with level, the constraints get divided into 4 parts, one for each class. Then, the theory of on/off constraints as described by (H.L. Hijazi, P. Bonami and A. Ouorou) [11], get applied, to make constraints obsolete when the player is not in the right class. This divides each of the 4 parts of this constraint in 2, due to exp gain being a equality constraint. For example, (17.3) and (17.4) are only binding when they do not get multiplied by $\mathbf{M}$. This happens when $(1 - FishMid_d + FishHigh_d) = 0$, and thus only exactly when both $FishMid_d = 1 \;\; \& \;\; FishHigh_d = 0$. Thus they are only binding when the fishing class is exactly medium.

$$(17.1) \qquad E_{d,FishEarn} \leq fe_{low} * F_d + \mathbf{M} * FishMid_d$$

$$(17.2) \qquad E_{d,FishEarn} \geq fe_{low} * F_d - \mathbf{M} * FishMid_d$$

$$(17.3) \qquad E_{d,FishEarn} \leq fe_{mid} * F_d + \mathbf{M} * (1 - FishMid_d + FishHigh_d)$$

$$(17.4) \qquad E_{d,FishEarn} \geq fe_{mid} * F_d - \mathbf{M} * (1 - FishMid_d + FishHigh_d)$$

$$(17.5) \qquad E_{d,FishEarn} \leq fe_{high} * F_d + \mathbf{M} * (1 - FishHigh_d + FishMax_d)$$

$$(17.6) \qquad E_{d,FishEarn} \geq fe_{high} * F_d - \mathbf{M} * (1 - FishHigh_d + FishMax_d)$$

(17.7) $E_{d,FishEarn} \leq fe_{max} * F_d + \mathbf{M} * (1 - FishMax_d)$

(17.8) $E_{d,FishEarn} \geq fe_{max} * F_d - \mathbf{M} * (1 - FishMax_d)$ $\qquad \forall_d[d \in D]$

Finally, the binary variables get correctly defined. This is once again by the same type of 'if-then' statements, and will look similar.

(19.1) $E_{d,FishTot} \geq 2150 - \mathbf{M} * (1 - FishMid_d)$

(19.2) $E_{d,FishTot} \leq 2150 - \epsilon + \mathbf{M} * FishMid_d$

(19.3) $E_{d,FishTot} \geq 6900 - \mathbf{M} * (1 - FishHigh_d)$

(19.4) $E_{d,FishTot} \leq 6900 - \epsilon + \mathbf{M} * FishHigh_d$

(19.5) $E_{d,FishTot} \geq 15000 - \mathbf{M} * (1 - FishMax_d)$

(19.6) $E_{d,FishTot} \leq 15000 - \epsilon + \mathbf{M} * FishMax_d$ $\qquad \forall[d \in D]$

2.5.6 Income

For everything regarding income, the same M_{dm} with $d \in D$ and $m \in [StartDay, Earnings, EndDay]$ is used. The values are in-game currency or money, with no specific name. Most constraints remain the same and straight forward.

Constraints:
Another non-negativity.

(1.2) $M_{dz} \geq 0$ $\qquad \forall_{d,z}[d \in D, c \in [StartDay, Earnings, EndDay]]$

Player, in this model, still only can spend money seeds. Therefore, player ends the day with the money it started the day with, minus the amount it spends on seeds.

(2) $M_{d,EndDay} = M_{d,StartDay} - \sum_{c \in C}(P_{d,c} * CI_{c,BuyPrice})$ $\qquad \forall_d[d \in D]$

Money the next day is what the player had left the day before, plus what he earned that day.

(4) $M_{d+1,StartDay} = M_{d,EndDay} + M_{d,Earnings}$ $\qquad \forall_d[d \in D \setminus \{28\}]$

Players start day 1 with 500 gold and 15 parsnip seeds. This can be translated to obligation of buying at least 15 parsnip seeds, and starting with 500 + 15*20 = 800 money.

(5.1) $M_{1,StartDay} = 800$

And finally, the more demanding constraint. Earnings took a big change, due to it being subjective to both farming and fishing level. The same type of on/off constraints get used, but now the constraints will only be active with exactly the right combination of both classes of farming and fishing, and thus not only one. This means that it will and up being 4 x 4 pairs (=32) of constraints for each day. They are pairs due to the fact that earnings is an equality constraint. $fa_{low}, fa_{mid}, fa_{high}, fa_{max}$ are multipliers of the payout, in regards to the respective player levels. These originate from the chance of increased crop quality. An example of how these constraints look like in gurobi, day 10, can be seen in appendix B.1.

(3.1) $M_{d,Earnings} \leq \sum_{c \in C}(harvest_{d,c} * CI_{c,SellPrice} * fa_{low}) + fm_{low} * F_d$
$+\mathbf{M} * FarmMid_d + \mathbf{M} * FishMid_d$

$$(3.2) \quad M_{d,Earnings} \geq \sum_{c \in C}(harvest_{d,c} * CI_{c,SellPrice} * fa_{low}) + fm_{low} * F_d$$
$$-\mathbf{M} * FarmMid_d - \mathbf{M} * FishMid_d$$

$$(3.3) \quad M_{d,Earnings} \leq \sum_{c \in C}(harvest_{d,c} * CI_{c,SellPrice} * fa_{mid}) + fm_{low} * F_d$$
$$+\mathbf{M} * (1 - FarmMid_d + FarmHigh_d) + \mathbf{M} * FishMid_d$$

$$(3.4) \quad M_{d,Earnings} \geq \sum_{c \in C}(harvest_{d,c} * CI_{c,SellPrice} * fa_{mid}) + fm_{low} * F_d$$
$$-\mathbf{M} * (1 - FarmMid_d + FarmHigh_d) - \mathbf{M} * FishMid_d$$

...

$$(3.31) \quad M_{d,Earnings} \leq \sum_{c \in C}(harvest_{d,c} * CI_{c,SellPrice} * fa_{max}) + fm_{max} * F_d$$
$$+\mathbf{M} * (1 - FarmMax_d) + \mathbf{M} * (1 - FishMax_d)$$

$$(3.32) \quad M_{d,Earnings} \geq \sum_{c \in C}(harvest_{d,c} * CI_{c,SellPrice} * fa_{max}) + fm_{max} * F_d$$
$$-\mathbf{M} * (1 - FarmMax_d) - \mathbf{M} * (1 - FishMax_d) \qquad \forall_d[d \in D]$$

2.5.7 Time Investment

One of the main resource this model relies on is time. All player actions cost (in-game) time, and therefore limits the possibility's of daily activities. For this resource, two auxiliary binary variables get introduced. these are GS_d and GF_d, and represent player Goes Shopping and Goes Fishing respectively. If the player buys any seeds, it has to walk to the shop, buy them, and then walk back, which costs a considerable amount of time. The same goes for fishing, if the player goes fishing, it has to walk to the mountain lake, and walk back in time. A day is considered to contain 18 hours of 6 in-game blocks of 10 minutes, thus a total of 108 time blocks. Subtracting from that is the time waking up and walking to the farm, going to sleep (together 10 min), walking to the shipping bin and back (10 min) and refilling the water bucket (10 min), all once a day. So the resource is at most 105. One exception is that the player has to grab his fishing rod from the beach docks on day 2, which cost more walking time. The times it cost to do daily actions can be seen in table 4. The time to bait for fishing is once again random, and gets very slightly reduced by fishing level. However this is negligible and thus an average time to bait is taken.

	GS	GF	Harvest	Watering	Hoeing	Planting = Water+Hoeing	Fishing
Actions	/	/	20	8	10	/	0.33
Time per action	6	10 (16)	0.05	0.125	0.100	0.225	3

Table 4: Actions per 10 in-game minutes, and the inverse (how much time one action cost)

This leaves the constraint for time usage. Here two new sets get used, $water_{dc}$ is the set of crops which has to be watered on day d, and $plant_{dc}$ the set of crops which get planted on day d (=P_{dc}). We then define:

$$water_{dc} = \sum_{i=1}^{max(CI_{c,DaysToGrow})} P_{(d-i),c} \text{ for } CI_{c,DaysToGrow} \text{ s.t. } d - CI_{c,DaysToGrow} \geq 1$$

Thus all the days in between planting crops, and harvesting crops (if planted from day 1 onwards). Max() is used for if it is a harvest day, but not the final one. Then the crops still need to be watered, due to wanting the next harvest as well. Then the time constraint gets defined as:

(11) $$\sum_{c \in C} (0.125 * water_{dc} + 0.225 * plant_{dc} + 0.05 * harvest_{dc}) + 3 * F_d$$
$$+10 * GF_d + 10 * GS_d \leq 105 \qquad \forall_d [d \in D \setminus \{2\}]$$

(11.1) $$\sum_{c \in C} (0.125 * water_{2,c} + 0.225 * plant_{2,c} + 0.05 * harvest_{2,c}) + 3 * F_2$$
$$+16 * GF_2 + 10 * GS_2 \leq 105$$

An example, day 10, can be seen at appendix B.2. The correct if then statements for the auxiliary variables has to be added as well: if any seeds are bought, then $GS = 1$. If there is at least one cast of a fishing rod, then $GF = 1$.

(12) $$\sum_{c \in C} P_{dc} \leq \mathbf{M} * GS_d$$

(13) $$F_d \leq \mathbf{M} * GF_d \qquad \forall_d [d \in D]$$

2.5.8 Energy Usage

Final part of the main model is the resource energy. A lot of in-game actions cost energy, and thus also limits the player. Each day the energy meter gets reset to 270, and can not be below 0. Energy usage does get affected by player level, for both fishing and farming. Therefore, once again, the constraint for each day gets split up to 4 x 4 on/off constraints, one for each correct combination of level classes of fishing and farming. This time, it is a lower-equal constraint and hence does not have to be split up in pairs. The amount of energy usage for each action can be seen in table 5. Anything not in the table does not use any energy, most noticeable is that both harvesting and walking do not use anything.

Skill level	Watering	Hoeing	Planting = Water+Hoeing	Fishing
General	2-0.1*lvl	2-0.1*lvl	4-0.2*lvl	8-0.1*lvl
Low	2	2	4	8
Mid	1.5	1.5	3	7.5
High	1.2	1.2	2.4	7.2
Max	1	1	2	7

Table 5: Energy usage per action

The constraints can then be defined accordingly. Note that not all constraint require the binairy check, due to the constraint's left hand side of a higher level class of farming is always lower than the left hand side of a lower class farming (when the fishing class stays equal). An example of this constraint, day 10, can be seen at appendix B.3.

(8.1) $$\sum_{c \in C} (2 * water_{dc} + 4 * plant_{dc}) + 8 * F_j \leq 270 * \mathbf{M} * FarmMid_d + \mathbf{M} * FishMid_d$$

(8.2) $\sum_{c \in C}(1.5{*}water_{dc}{+}3{*}plant_{dc}){+}8{*}F_j \leq 270{*}\mathbf{M}{*}FarmHigh_d{+}\mathbf{M}{*}FishMid_d$

...

(8.15) $\sum_{c \in C}(1.2 * water_{dc} + 2.4 * plant_{dc}) + 7 * F_j \leq 270 * \mathbf{M} * FarmMax_d$

(8.16) $\sum_{c \in C}(water_{dc} + 2 * plant_{dc}) + 7 * F_j \leq 270 \qquad \forall_d[d \in D]$

2.6 Luck

As already discussed, luck plays a considerable roll. A players luck can affect the payouts and the fishing exp gains. Therefore the constants $fa_{low}, fa_{mid}, fa_{high}, fa_{max}$ were already shown as multipliers for the farming payout. As well as $fm_{low}, fm_{mid}, fm_{high}, fm_{max}$, which are expected payouts per cast of the fishing rod, and $fe_{low}, fe_{mid}, fe_{high}, fe_{max}$ as expected fishing experience gains, per cast of the fishing rod. When calculating these 12 expected values, the variance of those can also be calculated with them, by the formula: $V[X] = E[X^2] - E[X]^2$. The Variance of fishing gets divided by 30, and the farming by 20, about the average amounts of action for a day. This is so that the daily luck is about what is expected. It should be calculated over the total amount of crops harvested and fish caught, but that will disregard the linearity, which will mess up the model. Then, assuming normality of each one of those, the luckiest x percentage of people can be calculated, where x is a earlier determined percentage, by first calculating Z-scores from x. The Z-scores tell how many standard deviations it lays from the mean, and thus with a single calculation: $X^* = E[X]{+}Z * \sqrt{V[X]}$, the value for the unluckiest x percentage of people gets obtained. This has to be done for $X =$ all of the 12 constants, using $x =$ farming luck for the four farming multipliers, and $x =$ fishing luck for the adjusted payout and exp gains values.

3 Results

3.1 Basic Model

In table 6, the strategy of the basic model can be seen. It shows which days to buy what seeds. It is fairly interesting, as it mainly goes for a full parsnip strategy. It prefers the quick payout parsnips give over the higher money-per-day, as can be obtained with other seeds. Only exception is a big batch of potato seeds in between, and some small additions of either parsnips or potato. It does conclude with a big harvest of 411 parsnips, ready on the final day. The near perfect overflow of the 3 batches parsnips, 1 batch of potatoes and then 1 more batch of parsnips is what makes this strategy optimal in these circumstances. However, anyone who has played the game, will see it seems impossible to plant & harvest this many crops in one go, this early in the game. The limitations of the full model will bring change to that.

	Blue Jazz	Cauliflower	Kale	Parsnip	Potato	Tulip
1	0	0	0	25	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	0	0	0	70	0	0
7	0	0	0	0	0	0
8	0	0	0	0	0	0
9	0	0	0	0	0	0
10	0	0	0	0	0	0
11	0	0	0	120	1	0
12	0	0	0	0	0	0
13	0	0	0	0	0	0
14	0	0	0	0	0	0
15	0	0	0	0	0	0
16	0	0	0	0	0	0
17	0	0	0	5	82	0
18	0	0	0	5	0	0
19	0	0	0	0	0	0
20	0	0	0	0	0	0
21	0	0	0	0	0	0
22	0	0	0	0	3	0
23	0	0	0	9	0	0
24	0	0	0	411	0	0
25	0	0	0	0	0	0
26	0	0	0	0	0	0
27	0	0	0	0	0	0
28	0	0	0	0	0	0

Table 6: Basic model plan, which day to buy and plant what amount of seeds

3.2 Full Model

In table table 7, the full model can be seen. The strategy has a clear direction: fishing. It tries to fish as much as possible, while only adding minimal amounts of crops to it. The crops seem to be selected to only lose 1 to 3 of the potential fish caught most days, while still earning some extra amount from the crops. It quickly gains the medium level of fishing on day 9, while the farming level reaches the medium class on day 24. Fishing also end in the high classes, which can be seen in the profits per day, which gradually rise. After a small portion of strawberries, it does not bother to plant any crops anymore, with the exception of 5 potatoes, as it regards fishing to be more profitable.

	Blue Jazz	Cauliflower	Kale	Parsnip	Potato	Tulip	Green Bean	Strawberries	Fish
1	0	0	0	15	0	0	8	0	0
2	0	0	0	0	0	0	0	0	29
3	0	0	0	0	0	0	0	0	31
4	0	30	0	0	1	0	0	0	18
5	0	7	0	0	4	0	0	0	28
6	0	8	0	0	0	0	0	0	29
7	0	0	0	0	0	0	0	0	31
8	0	0	5	0	6	0	0	0	27
9	0	0	0	0	0	0	0	0	31
10	0	0	0	0	0	0	0	0	31
11	0	0	0	0	0	0	0	0	31
12	0	0	0	0	0	0	0	0	31
13	0	0	0	0	0	0	0	22	0
14	0	0	0	0	0	0	0	0	18
15	0	0	0	0	0	0	0	0	18
16	0	0	0	0	0	0	0	0	26
17	0	0	0	0	0	0	0	0	28
18	0	0	0	0	0	0	0	0	30
19	0	0	0	0	0	0	0	0	30
20	0	0	0	0	0	0	0	0	30
21	0	0	0	0	0	0	0	0	30
22	0	0	0	0	5	0	0	0	28
23	0	0	0	0	0	0	0	0	30
24	0	0	0	0	0	0	0	0	30
25	0	0	0	0	0	0	0	0	31
26	0	0	0	0	0	0	0	0	31
27	0	0	0	0	0	0	0	0	31
28	0	0	0	0	0	0	0	0	31

Table 7: Main strategy, which day buying and planting what amount of seeds and/or fishing for some amount of fish.

3.2.1 Luck Affected

Because the strategy relies on expected values and thus the RNG, there was some more models made, to see if different luck would affect the strategies made. Even though luck is not something a player can have a impact on (while regularly playing the game), it is still something they can be affected by. The different outcomes can be seen in table 8. As can be seen, different farming luck minimal impact, while fishing luck is heavily affecting outcomes. This is mostly due to the big RNG aspect of fishing, and the fact that both the income and the exp gets affected by player luck. The main strategy for now still stays, most of the time, on a lot of fishing, with small crops batches added. The only big exception seem to be atrocious fishing luck, while having amazing farming luck. This does make a reasonable jump in amount of crops planted & harvested, at the expense of the amount of fishing. The lesser luck in fishing can also be related to a lack of skill in the fishing mini-game. If the player loses some fishes after they took the bait, it results in the same, less potential earnings and exp. However, in most cases, this lack in skill should not affect the strategy, and the player should still focus on mostly fishing.

Fish Luck Farm Luck	Atrocious	Bad	Below Average	Average	Above Average	Great	Amazing
Atrocious	138 707 **46630**	91 756 **51462**	74 776 **59085**	111 739 **64483**	76 774 **69677**	84 771 **76962**	121 734 **82004**
Bad	138 707 **46940**	96 748 **51610**	74 776 **59191**	111 739 **64712**	76 774 **69786**	122 733 **77225**	121 734 **82272**
Below Average	138 707 **47376**	96 748 **51832**	74 776 **59342**	111 739 **65037**	76 774 **69944**	122 733 **77603**	121 734 **82651**
Average	173 642 **47728**	144 688 **52180**	148 686 **59580**	111 739 **65265**	76 774 **70053**	122 733 **77871**	121 734 **82919**
Above Average	173 642 **48189**	144 688 **52530**	148 684 **59944**	111 739 **65491**	141 707 **70383**	122 733 **78135**	121 734 **83182**
Great	173 642 **48850**	144 688 **53033**	148 684 **60466**	111 739 **65815**	141 707 **70866**	122 733 **78513**	121 734 **83561**
Amazing	243 544 **49390**	170 657 **53466**	148 686 **60838**	111 739 **66044**	142 707 **71246**	122 733 **78782**	121 734 **83829**

Table 8: Different Incomes affected by player luck
In green to total crops planted, blue total fish attempted to catch

4 Conclusion

In this thesis is shown how a strategy can be defined, designed, and optimized using ILP methods. Starting from a simple ILP model, the environment of Stardew Valley is represented as a day to day planning of which crops to buy and plan. The money gets tracked by the model and correctly adjusted, for the seeds bought and the income of harvested crops. Gurobi had ease, optimizing the planning for the in-game month and got an optimal value for income after 28 days. Then was shown how to add more and more aspects of the game, even though not all have linear roots. The use of logical constraints, assisted by auxiliary variables were used to correctly identify conditions. Moreover, classes of player levels were introduced, which had to be conditionally constraint, but on their part affected other constraints. With those classes, on/off constraint were used to correctly apply the benefits of reaching higher classes, without disregarding the linearity of the model. In the end, there was also looked at how impactful luck can be on these models, and how much it changes the strategy. However, this model is only the beginning of possibilities, as there are still a lot of aspect of the game, which can be added to either more accurately describe the game or to add more options for daily activities to choose from.

5 Discussion

5.1 Man-made Strategies

There are some man-made strategies to be found online, with the same goal, maximizing income at the end of spring. First off, youtuber brandiganBTW made a skull cavern orientated strategy in his min/max run [5]. He beelined to a supposedly late-game cave, where he made a lot of money from mining ores and minerals, which resulted in about 500000 money worth of materials, which is a lot higher than the main model, which focused on farming and fishing only. One strategy closer to this thesis' is the min/max strategy by youtuber Foxy Fern [8]. She made a more 'human' strategy, where farming and fishing were the main focus. Early game, a lot of fishing was done for the quick payout, while doing some farming and mining on the side. The latter two were done to prepare for the making of many sprinklers, for which both mining resources and farming levels are needed for. Then, from about day 13, all the focus was on farming, and increasing the amount of sprinklers. This resulted in the free time not spend on any farming aspects, were spend gathering more ores in the mines, to eventually make more sprinklers. This resulted in a income of 120158 at the end of spring, also reasonable higher than the main model. One final strategy to look at is by youtuber Poxial [18]. He, even though the video is about a general playtrough of 100 days, tried to end spring with as much money as possible. This seems to be to start the summer as wealthy and unhurdled as possible. His strategy can be summarized as a 'buy as much strawberries as possible' strategy. The strategy once again starts with early fishing and mining, while slowly increasing farming exp. The focus was to have as much money as possible on day 13, to buy as much strawberry seeds as possible. Then, instead of immediately planting the seeds, starts gathering more ores the next day from the mines, to make more sprinklers. At day 16, he planted all the seeds, accompanied by sprinklers, and then does not bother farming anything else for the rest of spring, but spend it mostly mining instead. Even though this may seem as a lazy approach, it does result in a total income of 87238. The sprinklers are once again vital, and result in better an outcome compared to the main model.

5.2 Limitations & Future Work

5.2.1 Rain

There are some apparent limitations to the strategy generated from the approach this thesis took. The first thing to notice is that farming got very limited, while fishing had all the aspects implemented. This 'nerfing' of farming lead to a heavy fishing orientated strategy. One thing which was assumed, was that every day was a sunny day. While sunny days may sound nice and cheerful, rainy days is what is actually wanted, as it leads to not having to water the crops, without any drawback. The problem with that, is that they are fully random whether they occur (at a probability around 0.24), with exception of the first four days. And even if rainy days are generated, the model will then know which days they are going to occur and anticipate on those. Nevertheless, one potential strategy with rainy days generated is shown in table 9. There it can be seen that is does desire crops more, but it is not enough to drag it away from a fishing orientated strategy. It also goes for a wider variety of crops, something what has not yet occurred.

5.2.2 Sprinklers

One other main thing already mentioned sometimes, which happens to benefit farming, are sprinklers. These, once set up, automatically water all surrounding crops. This will then results in a lot less watering. As may be expected, these are a great investment and will reduce a lot of resources once set up. Most of the man-made strategies rely on these sprinklers, for example the earlier mentioned

	Blue Jazz	Cauliflower	Kale	Parsnip	Potato	Tulip	Green Bean	Strawberries	Fish
1	0	0	0	16	0	0	8	0	0
2	0	0	0	0	0	0	0	0	29
3	0	0	0	0	0	0	0	0	31
4	0	29	0	0	0	0	0	0	19
5	0	11	0	0	0	0	0	0	28
6	0	15	0	0	0	0	0	0	26
7	0	11	2	0	6	0	0	0	29
8	0	0	0	0	0	0	0	0	31
9	0	4	0	0	7	0	0	0	28
10	0	0	0	0	0	0	0	0	31
11	0	0	0	0	0	0	0	0	31
12	0	0	0	0	0	0	0	0	31
13	0	0	0	0	0	0	0	29	0
14	0	0	0	0	0	0	0	0	31
15	0	1	0	0	0	0	0	0	9
16	0	0	0	0	0	0	0	0	17
17	0	0	0	0	0	0	0	0	20
18	0	0	0	0	0	0	0	0	24
19	0	0	0	0	30	0	0	0	28
20	0	0	0	0	30	0	0	0	28
21	0	0	0	0	0	0	0	0	18
22	0	0	0	0	40	0	0	0	28
23	0	0	0	0	0	0	0	0	10
24	0	0	0	0	0	0	0	0	10
25	0	0	0	0	0	0	0	0	30
26	0	0	0	0	0	0	0	0	31
27	0	0	0	0	0	0	0	0	29
28	0	0	0	0	0	0	0	0	31

Table 9: Potential strategy, with generated rain days
Total crops placed: **239**, Total fish attempted to catch: **658**, Income: **69093**

strategies by Foxy Fern [8] and Poxial [18]. They are used for their convenience and mostly the time saved, to then be spend on other activities. One potential strategy can be seen in table 10. Here, the model assumes that the player has set up his sprinklers from week 3 or day 15 and onwards. This does have a big difference in the decision making of the model. First of, it decides to spend some resources in more early green beans, which are regrowable and thus stay around longer, so in the end the profit as well for the lack of needing to be watered. The same idea is used for the early cauliflowers, although they are an one time harvest crop, they do take a reasonable amount of time to grow, and thus also benefit from the sprinklers. The strawberries also increase slightly. But most importantly, after day 15, it drastically changes it's strategy. It starts pumping out big numbers in all kinds of crops, to take the profit to new heights. A big batch of cauliflowers planted in day 16, two batches of parsnips, and a constant supply of potatoes. However, this model is not fully feasible. The sprinklers cost either some money to buy the items to make them, or cost time and energy to mine those resources. And if the player spend those resources early on, there will not be enough money left to buy these amounts of seeds. Nevertheless, this is definitely something to look further into in future research, as it is one of the main aspects in other made strategies.

	Blue Jazz	Cauliflower	Kale	Parsnip	Potato	Tulip	Green Bean	Strawberries	Fish
1	0	0	0	16	0	0	8	0	0
2	0	0	0	0	0	0	0	0	29
3	0	0	0	0	0	0	0	0	31
4	0	0	0	0	0	0	21	0	23
5	0	9	0	0	1	0	1	0	28
6	0	11	0	0	0	0	0	0	28
7	0	11	0	0	0	0	0	0	28
8	0	15	0	0	0	0	0	0	26
9	0	0	0	0	0	0	0	0	31
10	0	0	0	0	0	0	0	0	31
11	0	0	0	0	0	0	0	0	30
12	0	0	0	0	0	0	0	0	30
13	0	0	0	0	0	0	0	33	0
14	0	0	0	0	17	0	0	0	0
15	0	0	0	86	34	0	0	0	4
16	0	75	0	4	22	0	0	0	9
17	0	0	0	0	0	0	0	0	31
18	0	0	0	0	71	0	0	0	17
19	0	0	0	0	45	0	0	0	24
20	0	0	0	0	165	0	0	0	3
21	0	0	0	0	180	0	0	0	0
22	0	0	0	0	180	0	0	0	0
23	0	0	0	130	0	0	0	0	10
24	0	0	0	0	0	0	0	0	30
25	0	0	0	0	0	0	0	0	30
26	0	0	0	0	0	0	0	0	28
27	0	0	0	0	0	0	0	0	26
28	0	0	0	0	0	0	0	0	27

Table 10: Potential strategy, if player has set up sprinklers form week 3 and onwards
Total crops placed: **1135**, Total fish attempted to catch: **554**, Income: **123252**

5.2.3 Eating

One final noteworthy disregarded aspect is that energy can be replenished by eating food. This can make bigger harvests possible, as well as longer fishing sessions, if there is time for it. In most playthroughs online, it is done as early as day 2, eating for example the green algae, because the little energy it replenishes is worth more than the 15 gold it provides when sold. This balances the days more on both energy and time, as it makes it possible to end a day with both resources on close to 0, if a day was originally only bounded by energy. There are also possibilities to buy food for a small amount, which replenishes a lot of energy, for only a small proportion of money. These can reduce the hefty toll on energy usage by farming, and thus possibly make it more attractive.

References

[1] Alanoud Alotaibi and Farrukh Nadeem. “A Review of Applications of Linear Programming to Optimize Agricultural Solutions.” In: *International Journal of Information Engineering & Electronic Business* 13.2 (2021).

[2] Aldo Bischi et al. “A detailed MILP optimization model for combined cooling, heat and power system operation planning”. In: *Energy* 74 (2014), pp. 12–26. ISSN: 0360-5442. URL: https://www.sciencedirect.com/science/article/pii/S0360544214001765.

[3] Robert E Bixby. “A brief history of linear and mixed-integer programming computation”. In: *Documenta Mathematica* 2012 (2012), pp. 107–121.

[4] BlaDe. *Fish chances based on all possible permutations, by youtuber BlaDe*. URL: https://www.youtube.com/@BlaDeSDV.

[5] brandiganBTW. *Min/Max run, skull cavern orientated*. URL: https://www.youtube.com/watch?v=zZr9SPQy7ac&ab_channel=brandiganBTW.

[6] Nick Butler and Sverre Spoelstra. *Media Review: Experimental Games: Critique, Play, and Design in the Age of Gamification*. 2023.

[7] Nicolas Esposito. “A Short and Simple Definition of What a Videogame Is.” In: *DiGRA Conference*. 2005.

[8] Foxy Fern. *Min/Max run, Farming orientated*. URL: https://www.youtube.com/watch?v=yaNVpz2-AZs&ab_channel=FoxyFern.

[9] Carlo Filippi, Renata Mansini, and Elisa Stevanato. “Mixed integer linear programming models for optimal crop selection”. In: *Computers & Operations Research* 81 (2017), pp. 26–39.

[10] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*. 2023. URL: https://www.gurobi.com.

[11] Hassan L. Hijazi, Pierre Bonami, and Adam Ouorou. “A Note on Linear On/Off Constraints”. In: *NICTA Technical Report* (2014).

[12] E. Kondili, C.C. Pantelides, and R.W.H. Sargent. “A general algorithm for short-term scheduling of batch operations—I. MILP formulation”. In: *Computers & Chemical Engineering* 17.2 (1993). An International Journal of Computer Applications in Chemical Enginnering, pp. 211–227. ISSN: 0098-1354. URL: https://www.sciencedirect.com/science/article/pii/009813549380015F.

[13] Samson Lasaulce and Hamidou Tembine. *Game theory and learning for wireless networks: fundamentals and applications*. Academic Press, 2011.

[14] N. L. Lawrie. “An integer linear programming model of a school timetabling problem”. In: *The Computer Journal* 12.4 (Jan. 1969), pp. 307–316. ISSN: 0010-4620. eprint: https://academic.oup.com/comjnl/article-pdf/12/4/307/1025317/120307.pdf. URL: https://doi.org/10.1093/comjnl/12.4.307.

[15] Olof Leimar and John M. McNamara. “Game theory in biology: 50 years and onwards”. In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 378.1876 (2023), p. 20210509. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rstb.2021.0509.

[16] Christopher A Paul. “Optimizing play: How theorycraft changes gameplay and design”. In: *Game Studies* 11.2 (2011), p. 100.

[17] Celia Pearce. “Towards a game theory of game”. In: *First person: New media as story, performance, and game* 1 (2004), pp. 143–153.

[18] Poxial. *Strawberries orientated strategy*. URL: https://www.youtube.com/watch?v=TYXNvC49Cs4&ab_channel=Poxial.

[19] *Python*. Version 3.9.13. URL: https://Python.org.

[20] Mostafa Rezvani, Mohammad Kazem Akbari, and Bahman Javadi. "Resource Allocation in Cloud Computing Environments Based on Integer Linear Programming". In: *The Computer Journal* 58.2 (Apr. 2014), pp. 300–314. ISSN: 0010-4620. eprint: https://academic.oup.com/comjnl/article-pdf/58/2/300/1037963/bxu024.pdf. URL: https://doi.org/10.1093/comjnl/bxu024.

[21] Don Ross. "Game Theory". In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta. Fall 2021. Metaphysics Research Lab, Stanford University, 2021.

[22] Larry Samuelson. "Game Theory in Economics and Beyond". In: *The Journal of Economic Perspectives* 30.4 (2016), pp. 107–130. ISSN: 08953309. URL: http://www.jstor.org/stable/44028260 (visited on 04/14/2023).

[23] Martin Shubik. "Game theory and operations research: some musings 50 years later". In: *Operations Research* 50.1 (2002), pp. 192–196.

[24] *Stardew Valley*. 2016. URL: https://www.stardewvalley.net/.

[25] *Stardew Valley Wiki*. URL: https://stardewvalleywiki.com/Stardew_Valley_Wiki.

[26] Richard Swedberg. "Sociology and Game Theory: Contemporary and Historical Perspectives". In: *Theory and Society* 30.3 (2001), pp. 301–335. ISSN: 03042421, 15737853. URL: http://www.jstor.org/stable/657964 (visited on 04/14/2023).

[27] The pandas development team. *pandas-dev/pandas: Pandas*. Version latest. Feb. 2020. URL: https://doi.org/10.5281/zenodo.3509134.

[28] Karin Wenz. "THEORYCRAFTING: Knowledge production and surveillance". In: *Information, Communication & Society* 16.2 (2013), pp. 178–193.

[29] Narnia C Worth and Angela S Book. "Dimensions of video game behavior and their relationships with personality". In: *Computers in Human Behavior* 50 (2015), pp. 132–140.

[30] Eric Zimmerman. "Narrative, interactivity, play, and games: Four naughty concepts in need of discipline". In: *First person: New media as story, performance, and game*. Vol. 154. MIT Press Cambridge, MA, 2004.

[31] Andris A. Zoltners. "Integer Programming Models for Sales Territory Alignment to Maximize Profit". In: *Journal of Marketing Research* 13.4 (1976), pp. 426–430. DOI: 10.1177/002224377601300415. URL: https://doi.org/10.1177/002224377601300415.

A Urls

A.1 GitHub

https://github.com/roeldeweerd/BEP

A.2 Fish Chances

https://docs.google.com/spreadsheets/d/1ybIMFw3f9UcLMjX-rE_D5-cxrMFC3J_y7yYVZ0arkfc/edit?usp=sharing

B Example Constraints

B.1 Earnings (3.1)

money[10,earnings] + -121.2 plan[2,Strawberries] + -50.5 plan[3,BlueJazz] + -111.1 plan[4,Kale] + -101.0 plan[4,Potato] + -30.3 plan[4,Tulip] + -35.35 plan[6,Parsnip] + -41.126 fish[10] + -100000.0 midlevelfarming[10] + -100000.0 midlevelfishing[10] < 0.0

B.2 Time (11)

0.05 plan[2,Strawberries] + 0.05 plan[3,BlueJazz] + 0.125 plan[3,Strawberries] + 0.125 plan[4,BlueJazz] + 0.05 plan[4,Kale] + 0.05 plan[4,Potato] + 0.05 plan[4,Tulip] + 0.125 plan[4,Strawberries] + 0.125 plan[5,BlueJazz] + 0.125 plan[5,Kale] + 0.125 plan[5,Potato] + 0.125 plan[5,Tulip] + 0.125 plan[5,Strawberries] + 0.125 plan[6,BlueJazz] + 0.125 plan[6,Kale] + 0.05 plan[6,Parsnip] + 0.125 plan[6,Potato] + 0.125 plan[6,Tulip] + 0.125 plan[6,Strawberries] + 0.125 plan[7,BlueJazz] + 0.125 plan[7,Kale] + 0.125 plan[7,Parsnip] + 0.125 plan[7,Potato] + 0.125 plan[7,Tulip] + 0.125 plan[7,Strawberries] + 0.125 plan[8,BlueJazz] + 0.125 plan[8,Kale] + 0.125 plan[8,Parsnip] + 0.125 plan[8,Potato] + 0.125 plan[8,Tulip] + 0.125 plan[8,Strawberries] + 0.125 plan[9,BlueJazz] + 0.125 plan[9,Kale] + 0.125 plan[9,Parsnip] + 0.125 plan[9,Potato] + 0.125 plan[9,Tulip] + 0.125 plan[9,Strawberries] + 0.225 plan[10,BlueJazz] + 0.225 plan[10,Cauliflower] + 0.225 plan[10,Kale] + 0.225 plan[10,Parsnip] + 0.225 plan[10,Potato] + 0.225 plan[10,Tulip] + 0.225 plan[10,GreenBean] + 0.225 plan[10,Strawberries] + 3.0 fish[10] + 10.0 Goesfishing[10] + 6.0 Goesshopping[10] < 105.0

B.3 Energy (8.1)

2.0 plan[3,Strawberries] + 2.0 plan[4,BlueJazz] + 2.0 plan[4,Strawberries] + 2.0 plan[5,BlueJazz] + 2.0 plan[5,Kale] + 2.0 plan[5,Potato] + 2.0 plan[5,Tulip] + 2.0 plan[5,Strawberries] + 2.0 plan[6,BlueJazz] + 2.0 plan[6,Kale] + 2.0 plan[6,Potato] + 2.0 plan[6,Tulip] + 2.0 plan[6,Strawberries] + 2.0 plan[7,BlueJazz] + 2.0 plan[7,Kale] + 2.0 plan[7,Parsnip] + 2.0 plan[7,Potato] + 2.0 plan[7,Tulip] + 2.0 plan[7,Strawberries] + 2.0 plan[8,BlueJazz] + 2.0 plan[8,Kale] + 2.0 plan[8,Parsnip] + 2.0 plan[8,Potato] + 2.0 plan[8,Tulip] + 2.0 plan[8,Strawberries] + 2.0 plan[9,BlueJazz] + 2.0 plan[9,Kale] + 2.0 plan[9,Parsnip] + 2.0 plan[9,Potato] + 2.0 plan[9,Tulip] + 2.0 plan[9,Strawberries] + 4.0 plan[10,BlueJazz] + 4.0 plan[10,Cauliflower] + 4.0 plan[10,Kale] + 4.0 plan[10,Parsnip] + 4.0 plan[10,Potato] + 4.0 plan[10,Tulip] + 4.0 plan[10,GreenBean] + 4.0 plan[10,Strawberries] + 8.0 fish[10] + -100000.0 midlevelfarming[10] + -100000.0 midlevelfishing[10] < 270.0