Organisms Report

Group 4:

Norris Chen(norrisch@seas.upenn.edu)

Rebecca Metzman(rmetzman@seas.upenn.edu)

Ruxue Yan(yanruxue@seas.upenn.edu)

10.24.2023

INTRODUCTION - Norris

Game Information

The project revolves around a simulated world that comprises an m by n grid housing virtual organisms, each with limited energy levels. These organisms can undertake various actions during a time cycle, including movement in horizontal or vertical directions, reproduction, or staying idle. The world is filled with the presence of food units that can spontaneously appear or duplicate based on defined probabilities. Organisms can consume food to increase their energy levels, with a maximum energy limit of M units.

An integral aspect of the organisms is their external state, an integer ranging from 0 to 255, visible to other organisms. This state can be altered by the organism itself during the simulation. Organisms can perceive their surroundings, detecting the presence of food or other organisms in neighboring cells, albeit without precise quantities. They can't identify their neighbors but can differentiate between organisms based on their external states, which is crucial for interactions in multi-species simulations.

The simulation process is systematic, with organisms acting sequentially based on their positions on the grid. The project aims to create an algorithm to serve as the "brain" of the organisms, enabling them to survive and reproduce in diverse environments. It encourages the development of strategies to ensure the organism's survival in isolation and in the presence of competing organisms, fostering adaptability and effective resource management. Furthermore, the project highlights the impact of brain complexity on the organisms' behavior and survival, offering an opportunity for exploring diverse strategies within the defined parameters and constraints.

Approach Overview

Looking at the problem, we decided to approach the problem in a more statistical manner. We believed that because of the games lasting quite long, the probability of food spawning is the most significant variable to calculate so our organisms can decide its course of action based on it. Otherwise, we also wanted to come up with our own specific algorithm that efficiently searches and hunts for food to complement our statistical methods.

INITIAL INSIGHTS AND OBSERVATIONS

Initial Strategy

For our initial strategy, we decided to focus on a straightforward approach that made use of some logical assumptions we had about movement and reproduction. We decided by implementing some basic guidelines for moving and reproducing we would have a good baseline for how the organism would perform given these assumptions. We also thought it would be good structurally to structure our code in such a way that there would be separate functions for deciding whether to move or reproduce. This modularity would make our code easier to edit and adjust for later milestones.

One basic assumption we made was to stay put if the organism doesn't have enough energy to move or reproduce. This was based on an arbitrary threshold we set just to get a baseline. We also decided to have our organism move towards food if its current cell is empty. Lastly, it would reproduce if its current cell had some food and an adjacent cell had food in order to capture more food. These assumptions didn't make use of any more sophisticated ideas about camping food in order to give it the chance to double. It was also a quite middle of the road approach in terms of moving and reproducing meaning it wasn't very conservative but also wasn't particularly aggressive when viewing it against later iterations and other players.

Observations and Adjustments

In the subsequent sections, a detailed breakdown of the final implementations and adjustments to our strategies will be provided. Below is a summary of immediate adjustments we made to our initial strategy:

Challenge	Adjustment in Response
Only performed well in the default configuration	• Focused more on testing in the desert in particular to ensure survival in tougher conditions
Not adaptive to changes in food and enemies	• Implemented P estimation to make use of enemies seen and food seen to get a sense of what P is, allowing our organism to adopt a more strategic approach

Implemented the same logic for moving and reproducing	• Decided to separate logic for moving and reproducing in order to add more nuanced thresholds for each
Didn't make use of prior moves	• Implemented a memory stack

STRATEGIES AND CONCEPTS

Movement

In our initial strategy, organisms conserved energy for prolonged survival by initiating movement only when food was detected in their vicinity. This approach, however, proved disadvantageous in desert environments where food scarcity led to a high extinction rate. Organisms often remained stationary, depleting their energy reserves without encountering sustenance.

To mitigate this, we introduced a stochastic element to our survival strategy. After an interval of 10 consecutive stationary rounds, organisms execute a random move. This adaptation is premised on the notion that immobility, in the context of scarce resources, invariably leads to depletion of energy reserves and subsequent death. Consequently, a random move provides a marginal chance of locating food, thereby potentially prolonging survival.

Furthermore, empirical observations derived from multiple simulation rounds indicated a notable increase in food availability in the post-initial phase of the game. This led to the strategic decision to limit the random movement protocol to the first 100 rounds. Post this phase, the organisms conserve energy, capitalizing on the heightened food availability rather than expending energy in exploratory movement.

This strategy demonstrates an adaptive response to environmental conditions, optimizing energy conservation and food search behavior based on real-time situational analysis. Future iterations of this project could explore algorithmic strategies that enable organisms to more effectively predict food availability or identify cues indicative of resource-rich zones, thereby further refining their survival mechanisms.

Reproduction

Based on the fact that the energy consumption of moving and reproduction are the same, we prioritize reproduction over movement. The organism's reproduction decision is a multifaceted process that considers both internal and external states. This nuanced approach ensures the organism optimally allocates its energy resources, balancing between immediate survival needs and the evolutionary imperative of gene propagation.

Internally, the organism assesses its energy reserves, comparing them to a dynamic reproduction threshold. It isn't a static figure but a dynamic value responsive to the organism's ongoing assessment of its environment. Central to this environmental evaluation is p, a calculated ratio representing the proportion of squares containing food to the total number of squares explored. This ratio serves as a barometer for food availability and, by extension, the presumed competitive landscape the organism inhabits.

When food is abundant, the organism adopts a more aggressive strategy, lowering the energy threshold required for reproduction. This approach emphasizes expansion rather than conservation without concern for energy consumption. In contrast, in food-scarce scenarios, the organism raises the energy threshold, prioritizing survival over propagation due to limited resources, thereby ensuring genetic continuity in challenging conditions.

Externally, the organism evaluates its immediate surroundings for food presence and competitor organisms. Reproduction is favored in squares with food, ensuring the offspring has an immediate energy source, essential for its survival in the organism's early life stages. If food-laden squares are occupied or absent, the organism reproduces into empty squares, balancing the need for offspring safety and resource accessibility.

P Estimation

Anticipating environmental conditions is a vital component of our organism's strategy. Rather than relying on a singular approach to govern organism movement, we devised a dynamic strategy adaptable to diverse board configurations. Naturally, this presented several challenges, notably the necessity of deciphering the board's unknown parameters. It was here that the concept of P estimation took root. By identifying parameters such as p (the probability of food spawning) and q (the probability of food doubling), we could theoretically devise a policy leveraging this information to make more astute decisions.

This concept draws from the principles of reinforcement learning, particularly the bandit problem. The problem mandates that a player allocate a fixed number of resources among competing choices without knowing the anticipated gains. This underscores the fundamental contrast between this game and parallel football. Given our awareness of the complete board state at each stage, we could theoretically determine the highest expected return. However, in Organisms, we must explore our anticipated gains and the board's parameters to enhance the organism's gameplay. Building upon the solution to the bandit problem, we prioritized exploring the most uncertain parameters (p and q) to identify the optimal policy for our organism.

Memory

The primary catalyst for the development of our novel memory strategy stems from the significant challenge organisms face due to the uncertainty of exact locations of others, compounded by potentially vast distances between them. Our strategy equips organisms with the capability to remember and revisit known food locations instead of reliance on communication for conveying information about available food.

This approach operates on a spatial memory strategy. When food sources are identified in multiple directions, the organism logs these directions for future reference. The procedure initiates with the organism navigating towards the first recorded direction. Upon reaching this destination, it employs a backtrack maneuver to reverse its most recent pathway, subsequently proceeding to the next recorded direction. This iterative process is maintained until all known food locales in the vicinity of the original cell have been explored.

A critical aspect to note is the potential occurrence of illegal movements, a term we use to describe instances when a desired spot is occupied by another organism during the exploration cycle. To mitigate this, our system incorporates a real-time verification mechanism post-move. If a movement is deemed illegal, the organism's food memory stack is immediately updated, thereby negating extensive delays and promoting operational efficiency.

This strategy, therefore, not only enhances individual survival probabilities in environments with scarce communication but also introduces an adaptive, self-correcting mechanism vital for navigating complex, dynamic ecosystems.

Stolen BH & NotSoSlow

Our final strategy for Memory made use of Team 1’s BH and NotSoSlow strategies. When beginning the final iteration of the project, our MemoryOld organism was performing well overall in most configurations. We spent a lot of time trying to tailor the memory stack and P estimation to make optimal reproduction and movement thresholds that could perform well in any environment. However, because of this we were in a sense not specialized to perform exceptional in any configuration. Our organism was competitive against some teams in certain configurations and other teams in other configurations. Since many teams thrived in a very specific environment our players could reach competitive levels with those teams in those environments.

We decided with the time we had left to use our P estimation to switch between a competitive desert player and a competitive overall player. When running trials we found BH to be very competitive in a default configuration and NotSoSlow to be competitive in a desert configuration, almost always beating the competition and surviving. We decided after running many trials with different P values as well as different combinations of our own players and other teams’ players including BH and NotSoSlow, that the combination of NotSoSlow with $p < 0.0125$ and BH for $p > 0.0125$ worked best. There is no specific reason for this threshold, we chose it based on informal experimentation trying to optimize performance in different environments.

IMPLEMENTATION

Movement

Initially, each organism is programmed to survey its adjacent cells for food. If food is detected, a move function is invoked to consume it. In the absence of food, the organism defaults to a stationary state, conserving energy. This state is maintained by not invoking the move function and keeping the organism's coordinates unchanged.

We counter the limitations of the reactive strategy by introducing stochastic movements. To implement this, each organism's state includes a counter tracking its stationary rounds, resetting after every move. If the organism remains stationary for ten consecutive rounds, indicative of a food-scarce scenario, the system triggers a forced move. This is executed via a separate function, executeRandomMove, that generates a random direction within the permissible range, ensuring the organism doesn't remain static. This function disregards food presence and focuses solely on altering the organism's position, thus increasing the chance of entering a region with undiscovered food sources.

To optimize energy usage, the stochastic movement is confined to the initial 100 rounds. We implement this by introducing a global round counter within the environment that tracks the number of elapsed rounds. Post the 100-round mark, organisms revert to a strictly reactive movement pattern. This shift is managed by a conditional statement checking the round counter before invoking executeRandomMove. If the organism is beyond this phase, the function is bypassed, even if the stationary round condition is met.

The movement functions are embedded within a larger control loop representing each organism's decision-making process. This loop integrates sensory input, energy conservation status, the organism's current health state, and the environmental round counter to decide the optimal movement strategy. Efficiency in energy usage is ensured by continually assessing the organism's energy levels before making movement decisions. If energy reserves are low, even in the early rounds, the system is designed to prioritize conservative strategies to prolong organism survival.

```
if (totalMovesMade <= 100) {
    boolean isStuck = Arrays.stream(trackingMoves)
            .allMatch(action -> action == Constants.Action.STAY_PUT);

    if (isStuck && finalMove.getAction() == Constants.Action.STAY_PUT) {
        moveDir = randomMove(neighborN, neighborE, neighborS, neighborW);
        finalMove = Move.movement(moveDir);
    }
    // Record last 10 moves
    trackingMoves[moveIndex] = finalMove.getAction();
    moveIndex = (moveIndex + 1) % 10;
}
```

Picture 1. Random move after 10 consecutive stagnations within first 100 rounds

Reproduction

Reproduction is an energy-intensive activity. To ensure an organism doesn't compromise its survival chances by reproducing at low energy levels, we've set a dynamic energy threshold for reproduction. This threshold is calculated based on the presence of food in the surrounding cells and the organism's current energy reserve. The reproduceThreshold is lower when food is abundantly available, encouraging organisms to reproduce more when sustenance is guaranteed. It is set by considering various factors, including the constant p representing the proportion of food seen versus squares visited, the maximum energy M an organism can have, and the energy cost v of moving. This adaptive threshold helps in maintaining energy balance and ensures that organisms reproduce only when they have sufficient energy to spare.

The decision to reproduce is made within the move function, where various conditions, including energy levels, food availability, and organism's state, are evaluated. If the organism's energy exceeds the reproduceThreshold and it's not in immediate need to consume available food indicated by hungry, it considers reproducing.Reproduction is favored in cells containing food, promoting the survival of the offspring. If cells with food are not available, the organism looks for empty adjacent cells. This is determined by the foodMove function, which returns a direction towards food if available, and the randomMove function, which provides an alternative empty cell if no food is available.

Once a direction is determined, the Move.reproduce function is invoked, which takes in the chosen direction and a unique childKey. This function initiates the creation of a new organism in the specified adjacent cell. The childKey ensures a unique identifier for the offspring, crucial for tracking and evolutionary purposes. Post-reproduction, the parent organism's energy is reduced to (M-v)/2, representing the biological cost of reproduction. This necessitates the organism to possibly shift its immediate focus back to foraging for food to replenish its energy reserves.

```
boolean surroundedByFood = foodN || foodE || foodS || foodW;
double reproduceThreshold = surroundedByFood ? (1-p)*((game.M() / (game.v() * 0.4))) : (1-p)*(game.M() / (game.v() * 0.3));
boolean reproduce = energyLeft > reproduceThreshold && energyLeft >= game.v();
```

Picture 2. Dynamic reproduction threshold

P Estimation

The estimation is essentially a simple equation that tries to detect food around it:

$$P = Food\ Seen / Squares\ Seen$$

There are a few caveats in determining p:

1. We do not update p every single step

This is to prevent food from being recounted by the organism so P does not blow up. There are two specific instances where food seen can be recounted for the same food: when the organism decides to stand still and when the organism decides to move and come back to the original square. We dealt with the first case by having a boolean keep track of if there is food in the cardinal directions and resets the boolean to false if the organism moved. The second problem was a lot more complicated to solve. If we just kept track of the location of the food (based on moves), we don't actually know if this food is new or has been there since we first saw it. The algorithm exacerbated this problem even more, because her algorithm will always go back for the food that has been seen before. In the end, we decided we want to counter both problems by only keeping track of the food we've seen in our memory and not counting those if we come by it again. We decided this was the best compromise for using our algorithm and computing P.

2. P is NOT passed down using DNA

We first thought about passing down DNA from the organism to organism because it gives a better global estimation for p. However, after a few trials we found out that having a greedier new organism is better. For example, one of the reasons why we think we did well in the desert is because when we reproduced and passed down p, the p calculations were heavily impacted by the food there were immediately surrounding it. This made the organisms greedy for food especially in desert situations where food is abundant (due to everyone being conservative). We noticed that not passing down p means that it can exploit the boom and bust cycle of the desert more efficiently. Where this concept fails a bit is in the rainforest configuration. Since the organism always spawns with a p prediction of 0, it will always be slightly conservative in the beginning until it starts seeing a myriad of food. These few turns, however, are crucial for a rainforest configuration because whoever reproduces the most in these turns gets to take over the board, thus choking out the other organisms. We thought this was a fair trade off since the default situation worked in the favor of the non-passing p down concept too.

3. We stopped calculating P at 150 moves of the current organism

We noticed that at the later stages of the game, updating P becomes more and more inaccurate and difficult. One of the biggest reasons for this is that organisms in rainforest configurations swamp the board, making it almost impossible to calculate P. We combat this problem by limiting the calculations of P to within 150 moves. We tested different move limit P calculations, and found that 150 was about the sweet spot where we saw more accurate predictions of p.

```
private void updateP(boolean foodN, boolean foodE,
                     boolean foodS, boolean foodW, int neighborN, int neighborE,
                     int neighborS, int neighborW) {
    if (neighborN == -1) {
        squaresSeen += 1;
        if (foodN) {
            foodSeen += 1;
        }
    }
    if (neighborS == -1){
        squaresSeen += 1;
        if (foodS) {
            foodSeen += 1;
        }
    }
    if (neighborE == -1){
        squaresSeen += 1;
        if (foodE) {
            foodSeen += 1;
        }
    }
    if (neighborW == -1) {
        squaresSeen += 1;
        if (foodW) {
            foodSeen += 1;
        }
    }

    p = ((float)foodSeen/(float)squaresSeen);

}
```

Picture 3. Estimated p based on seen food over squares

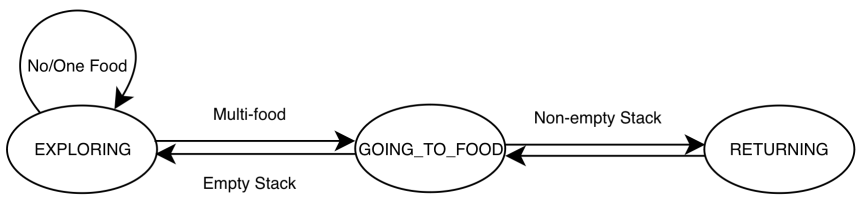
Memory

The foodMemory is essentially a Stack data structure, emblematic of the Last In, First Out (LIFO) principle. This choice is deliberate, as it empowers the organism to backtrack its steps in the reverse order they were taken, which is pivotal when navigating back to a known food source or while returning from it. When an organism encounters multiple food sources, it pushes the directions of these sources onto the foodMemory stack.

The transition between states (EXPLORING, GOING_TO_FOOD, RETURNING) is deeply intertwined with the foodMemory mechanism. Initially, organisms reside in the

EXPLORING phase, wherein foundational strategies for movement and reproduction are operationalized. Singular food presence prompts direct organism engagement without state alteration. However, when multiple nourishment sources are detected, the organism catalogs these directions within a 'foodMemory' stack, transitioning into the GOING_TO_FOOD phase. Within this phase, the organism retrieves a direction from the stack, executing a corresponding movement. Depletion of the stack cues a revert to the EXPLORING state for continued nourishment pursuit. Conversely, the RETURNING phase engages, initiating a backtrack to the prior position without stack interrogation, given its non-execution of pop operations, subsequently reverting to the GOING_TO_FOOD phase.

Inherently, the GOING_TO_FOOD and RETURNING phases are characterized by continuous motion. However, there are scenarios where food sources may deplete before the organism reaches them or situations where the organism's path is blocked, leading to them staying put. This anomaly triggers an immediate foodMemory stack update, ensuring the organism's responsiveness to its dynamic environment.



Picture 4. States flowchart of memory

Stolen BH & NotSoSlow

Our BH and NotSoSlow strategy builds upon Team 1's strategies. We use our own p estimation and as mentioned earlier we only update it for the first 150 iterations. Using the updated P value of a given organism, we run BH and return NotSoSlow if $p < 0.0125$. This is because we still want to update any variables in BH for every turn even if we don't use it. We only return the final move of NotSoSlow in this case. Once p reaches > 0.0125 we simply run and return BH's move.

ANALYSIS AND RESULTS

In this analysis, there are three key questions we want to summarize and explore:

1. **Performance:** How did our players fare in the official tournaments against other players?
2. **Strengths and Weaknesses:** What are the strengths and weaknesses of Memory and MemoryOld respectively and how are these explicitly apparent in the results of tournaments?

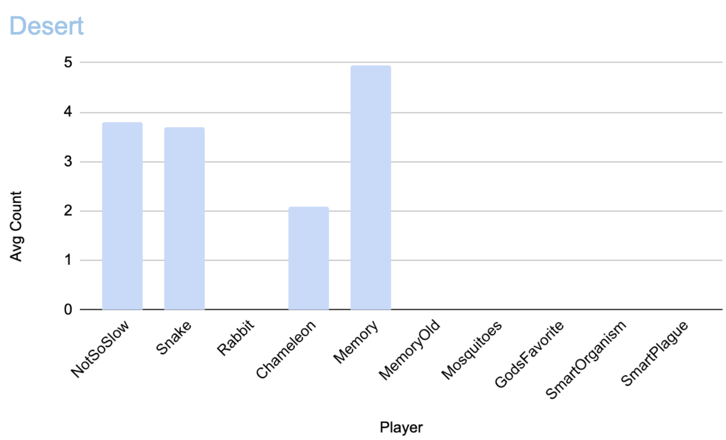
Performance

Here are some highlights from our results using our best organism denoted in square brackets for each given configuration.

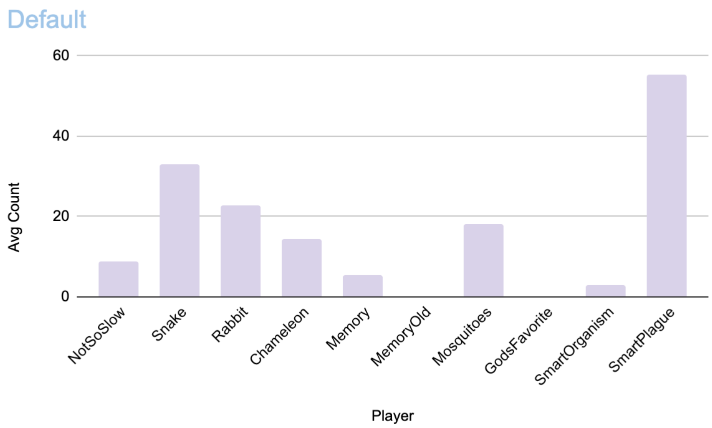
[Memory] Desert: 24% survival regular | 34% survival singleton | 44% big regular

[Memory] Default: 24% survival regular | 24% survival singleton | 66% big regular

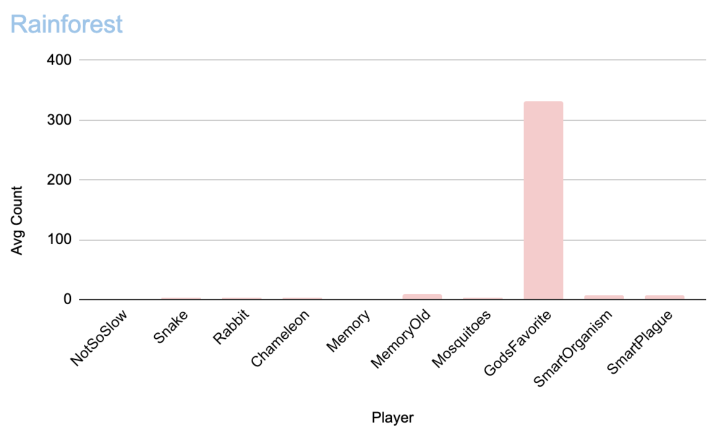
[MemoryOld] Rainforest: 24% survival regular | 23% survival singleton | 33% big regular



Picture 5. Average counts (end) in the desert



Picture 6. Average counts (end) in the default



Picture 7. Average counts (end) in the rainforest

Both Memory and MemoryOld demonstrate unique survival tactics and evolutionary traits in diverse environments. MemoryOld is characterized by a high-risk, high-reward strategy, thriving in resource-rich settings like Farmland and Rainforest due to its aggressive reproduction and high energy levels. However, this strategy falters in resource-depleted or highly competitive areas like the Ocean and Monday scenarios, leading to higher extinction rates.

In contrast, Memory shows an evolutionary advantage in resource-scarce settings, particularly the Desert, with a strategy focusing on endurance and efficient energy use. However, this approach doesn't fare as well in resource-abundant or highly competitive environments like the Default and Rainforest.

Their survival rates indicate that both lack adaptability, with MemoryOld struggling in scarcity and Memory not fully capitalizing on resource abundance. Their performances underscore the necessity of a balanced survival strategy, combining the energy efficiency of Memory and the opportunistic nature of MemoryOld, to enhance adaptability across various ecosystems.

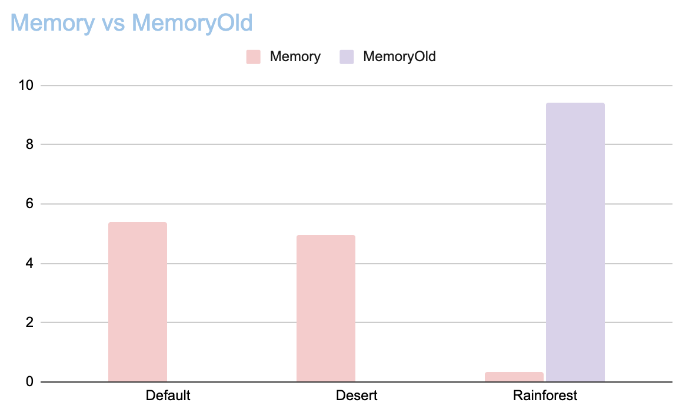
Strengths and Weakness

Memory did the best in the desert as compared to other teams. As conditions worsened from rainforest to desert, Memory performed increasingly better. We hypothesize that this is because we focused on creating a player that would do well in different configurations. As such, in any given configuration our player performs strictly worse than a player optimized for that specific configuration. As the competition increases in the better conditions where other teams focus their strategy and where it is more likely to survive in general, our organism does worse. However, in a climate like the desert where other teams did not focus their strategies and where it is more likely to die in general, our player does worse. This is proven by the fact that in the singleton configuration where there are strictly less players on the board or in a big configuration with more room and thus less competition, Memory tends to do even better than in smaller or more dense configurations.

Another reason for the exceptional desert performance could be because of an interesting point another team made in class. Earlier on in the desert, conditions were very poor and there was little food. However, as time goes on there tends to be more available food spawned on the board so the environment is more similar to a default configuration. Therefore, our adaptive strategy that makes use of a desert strategy when an organism's P is low and a default strategy when its P is high is very well suited for a configuration like the desert where conditions fluctuate.

Another configuration that Memory did well with the goldmines hidden configurations that we did not know about in advance. In this configuration, food is likely to double, each cell can store a lot of food, but appearance of food is low. We survived the most of any team at 61%. We believe this can be because in a way the low appearance of food can mimic a desert type of game trajectory. In the beginning of the game, there is a low amount of food on the board. Our organism's more conservative early game approach can survive this tough start much like we see in the desert. However, as food doubles the amount of food is higher overall. Therefore, adopting the P prediction for each organism can help adjust our strategy based on if we hit one of those gold mines or if we are more in a local desert.

MemoryOld survived better in the rainforest than Memory as we made the reproduction threshold lower for this player. Otherwise, MemoryOld did not perform well as we optimized for reproduction and which isn't always the best thing to prioritize in the desert and default configurations. We believe Memory performed poorly in the rainforest for the exact reason MemoryOld did well. Memory has a conservative approach to reproduction because we wanted it to survive the initial desert drought i.e. very rough conditions with low food. However, MemoryOld does the opposite and takes a very aggressive approach to reproduction which is possible to do in the rainforest where food is abundant and overextending or reproducing very quickly has less adverse effects.



Picture 7. Performance of Memory vs MemoryOld in various configurations

SUMMARY OF CONTRIBUTIONS -Norris

Individual Contributions

Below is a summary of the different contributions made by each team member, including those that were collaborative efforts and including the nontechnical aspects of this project:

Team Member (alphabetical)	Contributions
Norris	Implemented P Estimation Implemented Organism Policy strategies Adjusted reproduction thresholds based on the P estimations
Rebecca	Implemented initial strategy with basic assumptions and code structure Implemented our final strategy that made use of P estimation to adopt different players' strategies Adjusted reproduction and movement thresholds for MemoryOld based on running test tournaments in different configurations against various opponents
Ruxue	Adjusted dynamic reproduction threshold Adjusted movement and reproduction strategy Implemented random move after consecutive stagnations Implemented memory strategy
Shared Contributions	Slides and report Team meetings, brainstorming ideas, and staying on track with milestones

Team Contributions to the Class Project

The two most significant contributions our team made to the class were:

1. Balancing Exploration and Exploitation in P Estimation

Our emphasis on the concept of reinforcement learning was pivotal in our approach. We recognized the necessity of comprehensively understanding the environment before committing to a specific algorithm. In the initial stages, we allocated resources to gather information about the game's parameters. Subsequently, we utilized the predicted parameters to guide the decision-making process of our organisms. Our efforts resulted in the development of a robust p estimation calculation, which notably contributed to our success within the class.

2. Implementation of a Memory Algorithm

As mentioned earlier, we innovated a novel algorithm that significantly enhanced our organism's efficiency in locating food. This algorithm involved storing the locations of food encountered by the organism and returning to those specific locations after consuming other food sources. By adopting this approach, we effectively minimized the time spent exploring unfamiliar territories, thereby enabling the organism to focus on exploiting the immediate vicinity that yielded the highest rewards.

FUTURE DIRECTION AND LIMITATIONS

There are several potential directions for advancing this project:

1. Developing specific algorithms for different game configurations

When we initially approached the game with a generalized strategy using P prediction and policy making, our algorithm, although not tailored to any specific configuration, encountered challenges in excelling in particular settings (except the desert). In retrospect, innovating distinct algorithms for each proposed configuration discussed in class might have yielded better results. Integrating these diverse algorithms into the organism's code and adapting them based on environmental parameters could have been beneficial. For instance, integrating an algorithm optimized for exploiting the boom-and-bust cycle in the desert, alongside a conservative movement strategy, with another algorithm focused on rapid reproduction and creating food gardens to hinder rival organisms would have been a perfect balance between the configurations.

2. Enhancing P and Q estimation

As previously mentioned, we encountered difficulties in accurate P estimation. These challenges stemmed from external factors, such as increased inaccuracy due to a swarm of organisms on the board, and some inherent limitations of the game, like the inability to detect directions beyond the cardinal points. To address this issue, one viable approach could involve maintaining the state of the board for each organism, similar to what Group 1 implemented. This method would provide the most precise predictions for both P and Q. However, it is essential to consider that this approach might significantly amplify the code's processing time, potentially leading to notable lag. Therefore, a consideration for future development would be determining the optimal balance between the amount of information retained and the computational efficiency required for accurate P and Q predictions.

3. Enhancing policy formulation

Our existing policy was a simple empirical equation intended to define the reproduction threshold. Although it was effective to a certain extent, an improvement in the estimation of variables such as P, Q, M, and N could have enabled us to devise more intricate policies. These refined policies could have been subsequently tested to identify the most effective one. Given more time for the project, conducting multiple tests with diverse variables could have facilitated the identification of the most optimal policy.

ACKNOWLEDGMENT

There are many ideas, pieces of code and general ideas we'd like to credit our classmates with:

Code:

We took Team 1's BH strategy and NotSoSlow strategy to use with our P estimation. As mentioned before, we took Group 1's NotSoSlow strategy because they work particularly well in the desert; it was conservative first before reproducing and moved a lot later when food built up. We also took their BH strategy because they do particularly well in a default configuration. It prioritizes fast reproduction which means more food taken means more reproduction and has a swarming effect on the board

Ideas:

We also took some more general approaches to the problem from class discussion like p prediction and developing early late game strategies. We would also like to mention Team 6 for their adaptive player. While they use a very different more natural selection type of approach to adaptive performance, we still took inspiration from that and towards striving for better performance in different configurations.

CONCLUSION

In conclusion, our strategic implementation manifests a robust performance spectrum, excelling notably in resource-scarce environments such as deserts, achieving commendable third-place efficiency in abundant ecosystems like rainforests, yet presenting only median results in standard settings. These performance metrics can be primarily attributed to the timing of our 'p' estimation, which culminates just shy of the final iteration, coupled with our strategic focus on achieving a balance across diverse environmental conditions.

It's imperative to acknowledge that the pursuit of a universal, one-size-fits-all parameter set that thrives in all conceivable environments is a complex, if not unattainable, endeavor. The efficiency of our memory strategy, while generally high, is notably contingent on the abundance of food resources within the environment; it thrives in conditions of plenty and wanes in paucity.

One of the pivotal strengths of our approach is the inherent flexibility ingrained in our 'p' estimation methodology. This adaptability facilitates a nuanced, organism-specific calibration of strategies and parameters, thereby ensuring a bespoke optimization that resonates with the unique circumstances and exigencies of each organism. This not only enhances survival probabilities but also propels the overall fitness and thriving capacity of the population.

Looking forward, a more segmented strategic approach could further enhance performance metrics. By tailoring distinct, environment-specific strategies and fostering the capability to seamlessly transition between them or ingeniously integrate them as situational variables dictate, we could leverage a more dynamic adaptation model. However, such an approach would necessitate an extension in developmental timelines, allowing for meticulous strategy formulation and rigorous testing phases.

Ultimately, while the quest for a universally optimal strategy remains elusive, the insights garnered from this analysis are invaluable. They underscore the intricate dance between genetic fixedness and environmental fluidity and highlight the imperative of strategic adaptability and dynamic decision-making protocols in the relentless pursuit of evolutionary success.