Evolutionarily stable strategies in Kuhn poker

Leo Muller

2140043

Project Dissertation

Department of Computer Science

Adran Gyfrifidureg

26th April 2024

Declaration

Statement 1

This work has not been previously accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed	....	Leo Muller (2140043)
Date	....	

Statement 2

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by citations giving explicit references. A bibliography is appended.

Signed	....	Leo Muller (2140043)
Date	....	

Statement 3

The University's ethical procedures have been followed and, where appropriate, ethical approval has been granted.

Signed	....	Leo Muller (2140043)
Date	....	

Abstract

This paper will explore the outcomes of running simulations on populations of evolving decision trees engaged in repeated games of Kuhn Poker. The project endeavours to understand how individuals in the population will adapt, evolve, and develop strategies in competitive environments, ultimately seeking to ascertain whether an evolutionarily stable strategy may emerge within the population.

In order to ascertain whether evolutionary stable strategy has been adopted by the population, an understanding of the rules and optimal strategies for Kuhn poker must established. Using the rules as a guideline, a system capable of facilitating a simulation environment for repeated games of Kuhn poker and a genetic algorithm capable of representing and evolving populations of players will be developed. This system will then be used to prove the convergence of the player population towards an evolutionarily stable strategy or, in a sub-optimal case, an evolutionarily stable state.

Acknowledgements

I would like to thank Dr Arno Pauly for the help he provided me in his role as project supervisor. His encouragement and guidance during the planning faze of my project played a major role in my choice of final topic for the third year project. Without his guidance, I would have been unable to discover many key points of research, without which I would have lacked the necessary information required to write my dissertation.

I would also like to thank my close friend Logan Evans for the aid he provided me while researching poker. His knowledge of poker, among other gambling games, proved to be greatly influential in my decision to choose Kuhn poker as the game around which I would build my project.

Contents

List of Figures

1 Introduction

In this project, I will explore the outcomes of running simulations on populations of evolving decision trees, which will play repeated games of Kuhn poker against each other. Depending on their success in these games, they'll procreate, mutate or die out. This project will look at the possible emergence of an evolutionarily stable strategy that may be adopted by the population.

1.1 Motivation

The concept of evolutionarily stable strategies is central to understanding the stability of strategic behaviors within evolving populations. They represent strategies that, once established, resist invasion by emerging strategies, rendering them evolutionary stable. This project is motivated by a desire to potentially derive real-world applications from and understanding the dynamics of strategy evolution.

Understanding evolutionarily stable strategies in Kuhn poker has broader implications. It sheds light on real world scenarios where frequency dependent strategies play a role, such as cooperation, mate choice, and competitive interactions. In these scenarios, the fitness of a strategy depends not only on its absolute performance but also on its prevalence within the population

1.2 Aims and objectives

The primary aim of this project is to explore the outcomes of running simulations on populations of evolving decision trees engaged in repeated games of Kuhn Poker. The project endeavors to understand how individuals in the population will adapt, evolve, and develop strategies in competitive environments, ultimately seeking to ascertain whether an evolutionarily stable strategy may emerge within the population.

The main objectives of this project are:

1. *Population performance evaluation*: Establish criteria for evaluating whether an evolutionarily stable strategy has been adopted by the population.
2. *Simulation framework development*: Implement a simulation framework capable of facilitating recurring games of Kuhn poker for a population.
3. *Genetic algorithm development*: Design and develop a genetic algorithm capable of representing and evolving populations of decision trees for playing Kuhn poker.
4. *Program execution*: Use the developed systems to produce an output population in order to determine the dominant strategies used. Investigate whether the population converges toward an evolutionarily stable strategy or if multiple strategies coexist.

2 Solution concepts

A solution concept is a formal rule used for making game-play predictions known as ”solutions”. Solutions describe which strategies the players will adopted by players and, therefore, predict the outcome of the game. Equilibrium concepts are the most commonly used solution concepts, Nash equilibrium being the most famous.

2.1 Nash equilibrium

In game theory, the Nash equilibrium is the traditional solution concept for non-cooperative games involving multiple players. In a Nash equilibrium, each player is assumed to have common knowledge of the equilibrium strategies of the other players, and there does not exist a player that would gain anything by changing their own strategy. [6]

Given a two player game, let $E(S,T)$ represent the payoff for playing strategy S against strategy T. The strategy is a Nash equilibrium only if for both players and for any strategy T:

$E(S,S) \geq E(T,S)$

Using this definition, a strategy $T \neq S$ can score equally well, but not better to S. In this case, a Nash is presumed to be stable. This is presumed on the assumption that there is no incentive for players to adopt T instead of S.

2.2 Evolutionarily stable strategy

An evolutionarily stable strategy is an equilibrium refinement of the Nash equilibrium. Thus, once established by a population, natural selection will prevent any strategies that emerge from mutation from replacing it.

The conditions for a strategy $S \neq T$ to be an evolutionarily stable strategy are:

1. $E(S,S) \geq E(T,S)$, and
2. $E(S,T) > E(T,T)$

The first condition states that the strategy is a Nash equilibrium and the second condition establishes that the population of players who play strategy S has an advantage when playing against T. [8]

3 A simplified two-person poker

Poker is zero-sum imperfect-information game which can be played by two or more participants. It also incorporates a move by nature in the form of a dealer, this is interesting as it means the players must adapt to handle random possibilities.

Kuhn poker is a simplified variant of poker developed by Harold W. Kuhn as a simple model two-player game, amenable to a complete game-theoretic analysis. [1]

The list of changes that facilitate these features of Kuhn poker are as follows:

- The deck for Kuhn poker only consists of three playing cards, a King, Queen, and Jack.
- The players can only bet in increments of one unit;
- The ability of the player to raise has been removed.

In order to limit the complexity of the decision trees, the interactions for Kuhn poker will also be changed. Instead of being able to *bet*, *check*, *call*, and *fold*, the players will have the choices of **bet** and **pass**. Here **bet** functions as both *bet* and *call*, and **pass** functions as both *check* and *fold*. Therefore, the second **bet** made by any player will function as a *call*, and any **pass** made in response to a **bet** functions as a *fold*.

Figure 1 shows a tree of all moves that the player can make in a game Kuhn poker.

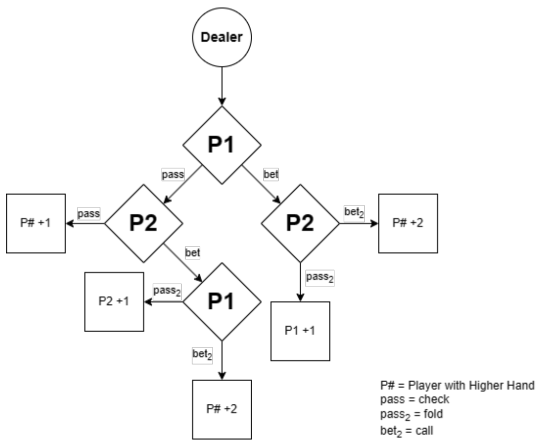

Figure 1: Tree for game of Kuhn poker

3.1 Optimal strategy

In game theory, a game can have a pure or a mixed strategy Nash equilibrium. When using a mixed strategy a probability is assigned to each pure strategy. This allows for a player to stochastically select a pure strategy.

In his paper Harold W. Kuhn describes an optimal mixed-strategy Nash equilibrium for Kuhn poker. [1]

First it is given that no player will decide either to bet on with a jack or to pass with a king when confronted by a bet.

Player 1's strategy may be described by saying that player 1 may pass on a king in the first round with arbitrary probability, but then he must bet on a jack in the first round one third as often, while the probability with which he bets on a queen in the second round is one third more than the probability with which he bets on a jack in the first round. If player 1 holds a queen he may as well pass in the first round, deciding to bet in the second if confronted by a bet.

Player 2 has a single solution which instructs him to bet one third of the time when holding a jack and confronted by a pass and to bet one third of the time when holding a queen and confronted by a bet. Player 2 may as well pass when holding a queen and confronted by a pass, and should bet when holding a king and confronted with a pass.

4 Genetic algorithm

For this project, a genetic algorithm is being utilised to evolve player populations. In computer science, a genetic algorithm is a population based metaheuristic optimisation algorithm that is a member of a larger family of evolutionary algorithms. In order to generate solutions to optimisation and search problems, genetic algorithms use operators inspired by biological evolution, such as selection, crossover, and mutation. [5]

Figure 2 shows a flowchart representing the genetic algorithm for this project.

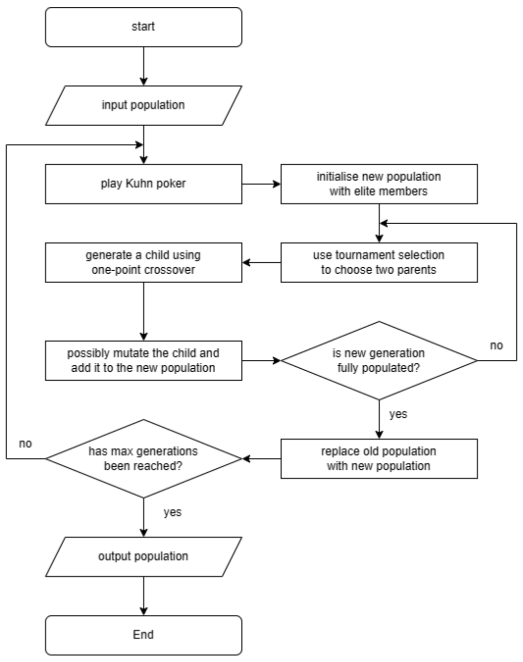
start
input population
play Kuhn poker
initialise new population with elite members
use tournament selection to choose two parents
generate a child using one-point crossover
possibly mutate the child and add it to the new population
is new generation fully populated?
no
yes
replace old population with new population
has max generations been reached?
no
yes
output population
End

Figure 2: Flowchart of Genetic algorithm

The genetic algorithm implemented in python for this project A follows the provided figure 2 shown above.

Candidate solutions to the optimisation problem play the role of individuals in a population. A genetic algorithm will use a fitness function to determine the quality of the given solutions.[2] In this project, Kuhn poker can serve as the fitness function. After playing a game, an individual will either earn or lose points depending on whether or not they won. Therefore, we can assume that individuals with higher scores after playing repeated games of Kuhn poker are higher quality than those with fewer points.

Individuals in the population will be given a starting bankroll with which they will play poker. Upon loosing a sufficient number of games, a player's bankroll may become zero at which point the individual will be removed from the population. Any survivors of this process will be added to any successive generations, this is known as elitism and is an optional stage in the selection process.

4.1 Genetic representation

Design solutions are referred to as chromosomes in evolutionary algorithms, and are commonly represented as a string of numbers.

The pure strategies available to the players can be genetically represented with the ordered list $(x_1, x_2, x_3, y_1, y_2, y_3)$ where $(x_i = 0, 1, 2; y_j = 0, 1, 2, 3)$. The instructions contained in x_i are for card i and are deciphered by expanding x_i in the binary system, the first figure giving directions for the first round of betting, the second giving directions for the second, with 0 meaning pass and 1 meaning bet. Similarly, to decode y_j, one expands in the binary system, the first figure giving directions when confronted by a pass, the second when confronted by a bet, with 0 meaning pass and 1 meaning bet. An example chromosome $(2, 0, 1, 2, 0, 1) = (10, 00, 01, 10, 00, 01)$ would mean that, when playing as player 1, the player should bet on a jack in the first round, always pass with a queen and wait until the second round to bet on a king. Alternatively, when playing as player 2, the player should pass except when holding a jack and confronted by a pass or holding a king and confronted by a bet. [1]

4.2 Algorithm operators

Selection is the stage of a genetic algorithm in which individual chromosomes are chosen from the population for crossover. Selection mechanisms may also be used to choose candidate solutions for the next generation. This method of retaining the well performing individuals and appending them to the generation unchanged is elitist selection.[3] In this project's genetic algorithm, unsuccessful players are culled in the Kuhn poker faze, this systems version of a complex fitness function. Parents are then chosen from the surviving population using tournament selection. The fitness of an individual is decided by their total winnings from playing poker.

The next operator in a genetic algorithm is crossover. Here genetic information from the parents, chosen in the selection faze, are combined to generate a new offspring. This process os analogous to the crossover that occurs during sexual reproduction. This project uses one point crossover to combine parent solutions, this is done by randomly selecting a point on both parents' chromosomes. Bits to the left of that point from the first parent are merged with bits to the right of it from the second parent. This results in a child, that carries some genetic information from both parents.[3] For example, if parent one had the chromosome $(2, 0, 2, 0, 1, 3)$ and parent two had the chromosome $(0, 0, 1, 0, 0, 3)$ and the crossover point was the second bit, the child's chromosome would be $(2, 0, 1, 0, 0, 3)$. Solutions can also be generated by cloning individuals from the previous generation, which is a process analogous to asexual reproduction. These cloned solutions may then be mutated before being added to the population. In order to reduce instability from needless mutation, the implemented genetic algorithm does not use this method of reproduction.[3]

The final operator in a genetic algorithm is mutation. Mutation of children in the population maintains diversity of the chromosomes. It is analogous to the process of biological mutation. For this project, single point mutation has been used. This process involves randomly selecting a bit from a child's chromosome, this bit would usually be bit flipped. Since the chromosomes in this project are represented with real numbers, the chosen point is instead replaced with a randomly generated number that conforms to the requirements of the chromosome format.[3] For example, the child $(2, 0, 1, 0, 0, 3)$ could be mutated by replacing it's second digit with a 1, thus producing a new child $(2, 1, 1, 0, 0, 3)$. Mutation isn't guaranteed to occur, instead it occurs with a fixed probability which may be defined freely. A mutation operator helps the genetic algorithm to avoid local minima by preventing the population of chromosomes from becoming overly similar to each other which would slow, or possibly even stop, the convergence to the global optimum.[3]

5 Results

In order to prove that Kuhn poker has an evolutionarily stable strategy, a majority of the population would have to adopt the mixed strategy Nash equilibrium for Kuhn poker. This group of the population would then have to maintain themselves as the majority without fluctuation.

Figure 3 is a bar chart (created using the matplotlib library [4]) which displays the percentage of the population that is using a pure strategy derived from the mixed strategy for Kuhn poker at given observation points, these observations being once every 100 generations. Figure 3 shows that, once adopted, mixed strategy Nash equilibrium derived pure strategies make up a consistent majority of the population. This would suggest that the population has adopted an evolutionarily stable strategy.

5.1 Evolutionarily stable state

In figure 3 it shows that pure strategies derived from mixed strategies make up a consistent majority of the population. However, around observation 20, there is a substantial decrease in the percentage of the population using the optimal strategy. In this instance, the population has maintained a majority of players that use the mixed strategy Nash equilibrium. In earlier runs of the genetic algorithm, populations have been produced where the majority was temporarily lost. This enables the possibility that the population is in an evolutionarily stable state rather than possessing an evolutionarily stable strategy.

An evolutionarily stable state describes a population that is susceptible to being displaced by alternative or mutant strategies but will return to its previous composition even after being disturbed. [7]

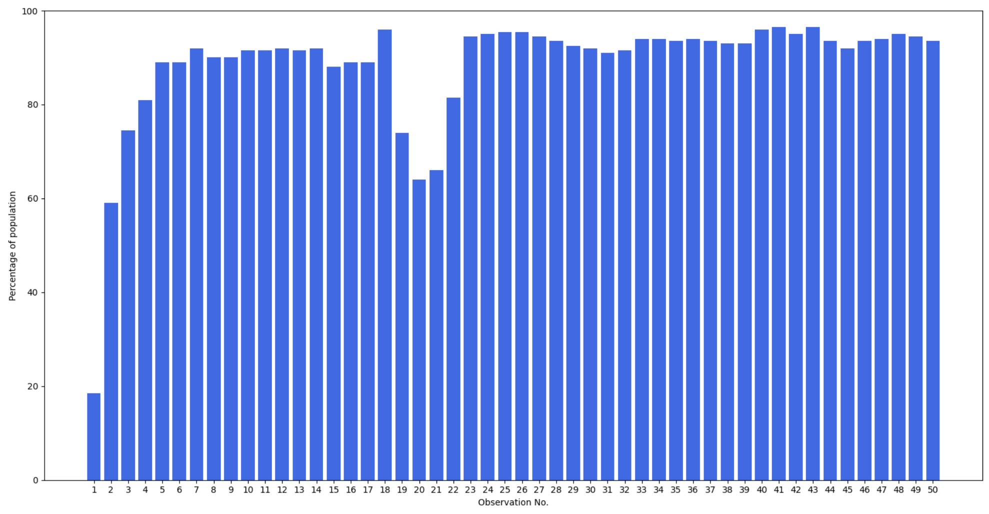

Figure 3: Bar chart displaying results from algorithm

6 Summary

This project endeavoured to ascertain whether a population engaged in repeated games of Kuhn poker would adopt an evolutionarily stable strategy. The first step in proving this, was to design a simulation framework that would be capable of facilitating the evolution of populations of decision trees and mediating repeated games of Kuhn poker. A genetic algorithm was chosen for the purpose of population and evolution management. This required that individuals in the population be genetically represented as chromosomes. Despite the population being comprised of decision trees, the simplicity of the Kuhn poker game tree allowed for the decision trees o be stored as strings of integers. Additionally, the exact design of these string could easily be implemented using the encoding format defined by Harold W. Kuhn in his paper. A simulation environment for playing Kuhn poker which could read the chromosome strings as decisions was then designed.

In order to utilise the implemented system, research had to be complete to define criteria to establish whether an evolutionarily stable strategy was present in the population. An understanding of solution concepts and the optimal strategies of Kuhn poker had to be developed for this. The most important points here are that, Kuhn poker possess a mixed strategy Nash equilibrium and that an evolutionarily stable strategy is an equilibrium refinement of a Nash equilibrium. Thus, if the mixed strategy Nash equilibrium proved to be impermeable when adopted by the simulated population, then it would be proven that Kuhn poker possess an evolutionarily stable strategy.

When executed, the genetic algorithm A produced an output 3. Given this data, the existence of an evolutionarily stable state can be proven. In order to prove the emergence of an evolutionarily stable strategy, further testing would need to be completed. This would require improving the simulation system to provide more data on the populations generated form the genetic algorithm in it’s output.

References

[1] H. F. Bohnenblust et al. *Contributions to the Theory of Games (AM-24), Volume I*. Red. by H. W. Kuhn and A. W. Tucker. Princeton University Press, 1952. ISBN: 978-0-691-07934-9. URL: https://www.jstor.org/stable/j.ctt1b9rzq1 (visited on 28/10/2023).

[2] *Introduction to Evolutionary Computing*. URL: https://link.springer.com/book/10.1007/978-3-662-44874-8 (visited on 24/04/2024).

[3] *Main page - Introduction to Genetic Algorithms - Tutorial with Interactive Java Applets*. URL: https://www.obitko.com/tutorials/genetic-algorithms/index.php (visited on 25/04/2024).

[4] *Matplotlib — Visualization with Python*. URL: https://matplotlib.org/ (visited on 24/04/2024).

[5] Melanie Mitchell. *An Introduction to Genetic Algorithms*. The MIT Press, 2nd Mar. 1998. ISBN: 978-0-262-28001-3. DOI: 10.7551/mitpress/3927.001.0001. URL: https://direct.mit.edu/books/book/4675/An-Introduction-to-Genetic-Algorithms (visited on 24/04/2024).

[6] Martin J. Osborne and Ariel Rubinstein. *A course in game theory*. Cambridge, Mass: MIT Press, 1994. 352 pp. ISBN: 978-0-262-15041-5 978-0-262-65040-3.

[7] John Maynard Smith. *Evolution and the Theory of Games*. Cambridge: Cambridge University Press, 1982. ISBN: 978-0-521-28884-2. DOI: 10.1017/CBO9780511806292. URL: https://www.cambridge.org/core/books/evolution-and-the-theory-of-games/A3BDF54AF5C6297E308AB15BBEF45E48 (visited on 24/04/2024).

[8] Bernhard Thomas. ‘On evolutionarily stable sets’. In: *Journal of Mathematical Biology* 22.1 (1st June 1985), pp. 105–115. ISSN: 1432-1416. DOI: 10.1007/BF00276549. URL: https://doi.org/10.1007/BF00276549 (visited on 30/10/2023).

A Implementation of genetic algorithm

```python
import kuhn
import random

POPULATION_SIZE = 100
MUTATION_RATE = 0.2
MAX_GENERATIONS = 100

def mutation(child):
    new_child = list(child)
    n = random.randint(0, 5)
    if (n < 3):
        i = random.randint(0, 2)
    elif (n > 2):
        i = random.randint(0, 3)

    new_child[n] = str(i)
    return "".join(new_child)

def crossover(x, y):
    c = random.randint(1, 4)
    child = x[:c] + y[c:]
    return child

#this algorthm uses the tournement selection method
def selection(population):
    tournament_size = POPULATION_SIZE // 10
    tournament = random.sample(population, tournament_size)
    return max(tournament, key=lambda x: x[1])

#the replication method from genetic programming has been used to
   promote stability
def replication(population):
    new_population = []
    for i in population:
        if (i[1] > 0):
            new_population.append([i[0], kuhn.BANKROLL])
    return new_population

def algorithm(population):
    for _ in range(MAX_GENERATIONS):
        population = kuhn.poker(population)
        new_population = replication(population)
        while (len(new_population) < POPULATION_SIZE):
            x = selection(population)
            y = selection(population)
            child = crossover(x[0], y[0])
            if random.random() < (MUTATION_RATE):
                child = mutation(child)
            new_population.append([child, kuhn.BANKROLL])
        population = new_population
    return population
```
Listing 1: Implementation of genetic algorithm.