Frank Cremen, Ben Mautner, Chris Olzinski
ECE 251 – Senior Project, Fall 2002
LUPoker with Artificial Intelligence
Submitted 12/9/2002 to Prof. Munoz-Avila

Summary:

The purpose of this report is to document and describe our Senior project for ECE 251. We have designed and coded a Java application which implements a Texas Hold'em style poker game, which we call LUPoker. The purpose of this is application is to allow the user to practice and learn poker through simulation of nine other tight (good) players by artificial intelligence.

Introduction:

Texas Hold'em is a variant of the normal five card poker that most people know and enjoy. Amongst serious poker players it is the game of choice- the world champion is decided over a game of hold'em poker. There are several different types of hold'em including limit, pot limit, and no limit. The strategies for each of these types is varied. This software concentrates on the limit type of hold'em poker.

Description of the Game:

Each player starts in Texas Hold'em with two randomly dealt cards. They then have a choice of whether to continue with these cards or two fold. Since the rest of the decisions for the hand are based on the decisions you make with these first two cards the standards by which you choose to play must be pretty high.

After the players decide if they are going to continue with their first two cards and the betting, raising and calling (collectively referred to as 'action') is complete, the dealer then deals three community cards face up. These three cards, commonly called the 'flop,' can be used by all the players in any manner that they choose, so for one player these community cards may make a straight, for another player they may make two pair, and for yet another they may make nothing at all.

After these cards are dealt there is another round of betting. Then there is another card dealt face up and another round of betting. Some quick points about the is fourth community card. The amount of each bet doubles after this card comes out. There is only one card to come out so not many hands will be "made" by this card. There is only one card left to come out so players are going to be betting a lot on this round to get people out that don't have "made" hands. This card is called the "turn" card.

After the betting is concluded on the fourth card, the fifth and final community card is dealt. This card is called the "river" card. The last betting round starts, and the bets on this round are of the same amount as on the "turn" card. After the betting concludes the remaining players Show their hands and determine the best one and the winner gets the money.

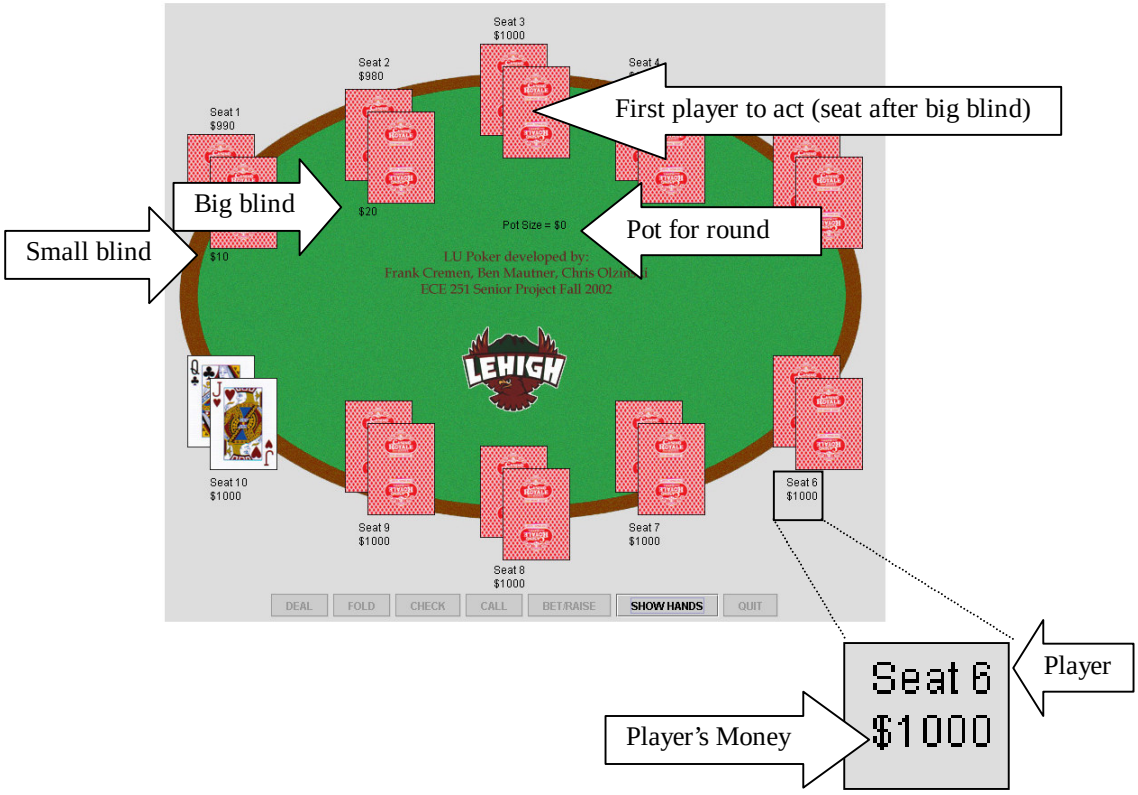
Now for some general points on the game. The hand rankings are as follows:

1. Royal Flush
2. Straight Flush
3. Four of a Kind
4. Full House
5. Flush
6. Straight
7. Tree of a Kind
8. Two Pair
9. Pair
10. Ace High

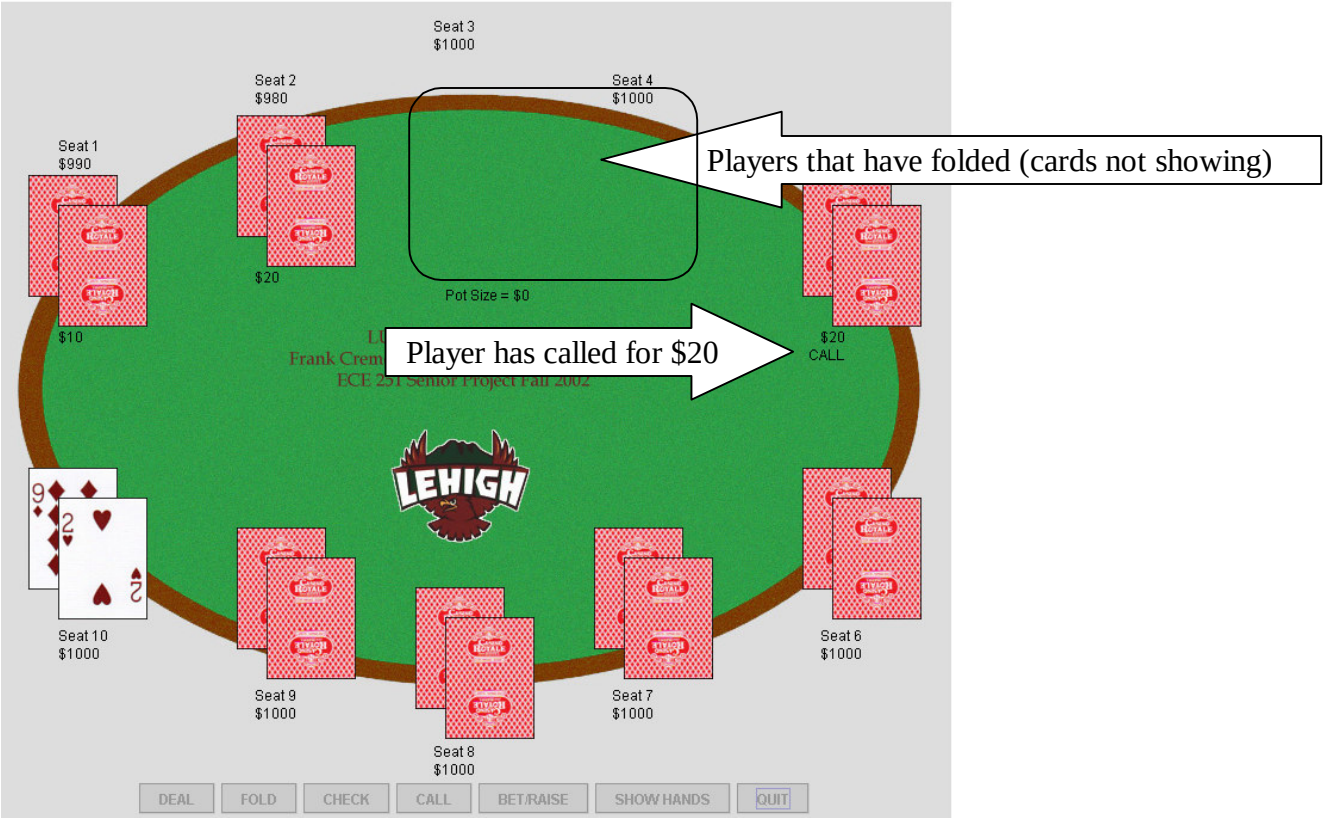
User interface (user manual):

The program starts out by a user opening up a HTML page, or an applet from the Java SDK. When it opens, the user will see the following:

When the user wants to begin playing, he should click the 'DEAL' button (magnified above). The deal button will begin the play of the game by first dealing out the cards to each of the ten players. The cards will be face down for each of the nine artificial intelligence players, and face up for the user. The small blind and the big blind will be appropriately assigned and play will begin with the player in the third position. The initial dealing of the cards can be seen below.

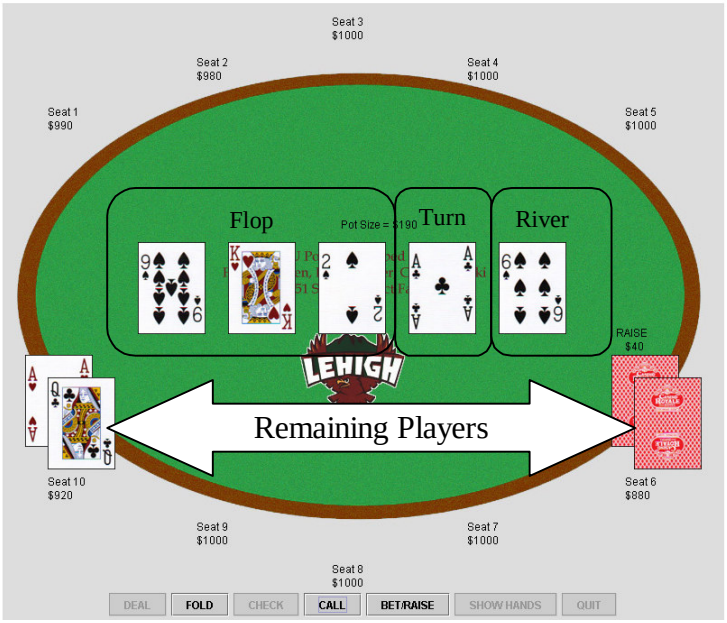


Each player has a seat position and their own money, which are displayed on the outside of the table for each player. (Seen above) Then the action of the game begins to take place. You will soon see players making their respective moves. These actions will be visible in many ways. As seen below, if a player FOLDS, or decides not to play in the hand, his cards will disappear and his action will appear near the inner, table side of his position. If a player decides to place a bet, his action will also be seen, the amount of money that he bet will also be seen and his cards will remain visible. The flow of the game moves in a clockwise manner from player to player.

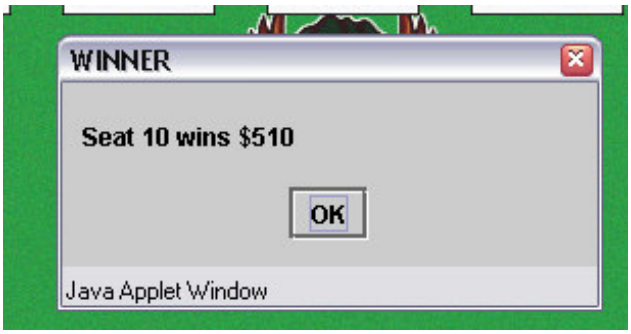


When it is the users turn, the buttons at the bottom of the screen which the player can use will become active (black lettering, not gray), seen below.

The user can click any of the above active buttons and play will continue to the other AI players. Then the flop will be dealt. After the flop is dealt, the remaining players will again act. Then the turn will be dealt and again the remaining players will act and lastly the river will be dealt and the remaining players will again act.



After this last round of acting is done, a winner is determined and a window will pop-up saying who won and how much money they won.



When this happens, it can be seen that the winning player has their money amount updated with their winnings. The user then must click OK in order to be able to continue play. Then the user has the option to view the cards that were just played by the last remaining players by clicking the now active SHOW HANDS button twice. To play another hand the user can now click the DEAL button again and another had of LUPoker will begin!

Description and Functionality of the software:

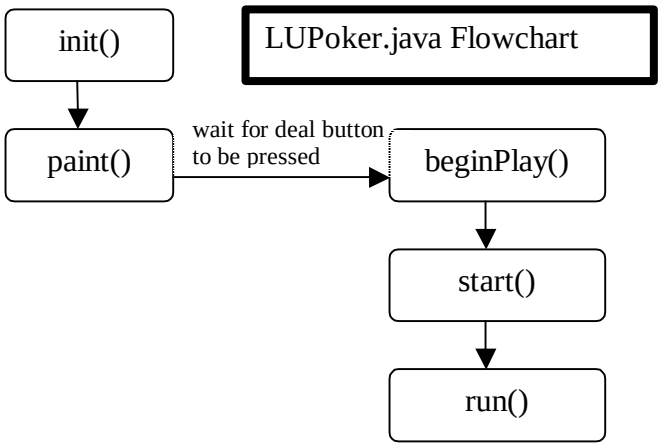
The software was written in Java using the Sun ONE Studio 4 IDE. You can download a free version of the development suite that we used to write this from the following location: http://java.sun.com/j2se/1.4.1/download.html
You want to download the executable file of **Sun™ ONE Studio 4 update 1, Community Edition - J2SE™ 1.4.1 Cobundle.** This is all you need for development.

The main class is LUPoker, and the other classes CONSTANT, Card, Dealer, Groups, Player and StartingHand are all used to make the code easy to read, write, and reuse.

LUPoker.java

This software was done by implementing the interface as an applet and not as an application. Because it was done by implementing an applet there is no main function. When the applet is first started the function *init()* is called. In *init()* the players are created and the buttons on the applet are initialized. After *init()* is done executing *paint()* is automatically called. The *paint()* function does all the drawing of the cards in their appropriate locations.

Once *paint()* is done executing, the program waits for the user to hit the deal button. When any button is pressed the function *actionPerformed()* is invoked. After the deal button is pressed *beginPlay()* is called and the game is then started by calling the function *start()* which automatically calls *run()* which is in the inner class called PlayerThread. All of the game processing is done in the function *run()*.



Player.java

This class is used to represent a player in the game. Each player has a set of its own properties, including: type, hand, Booleans for actions taken and others. The main functions in this class are:
normalActFlop(),normalActPreFlop(),normalActPreFlopBigBlind(), normalActPreFlopEarly(), normalActPreFlopLate(), normalActPreFlopMiddle(), normalActPreFlopSmBlind(),normalActTurn(),normalActRiver().

These functions do the artificial intelligence decision making for the AI players in the game.

Another important function in class Player is *handValue()*. This function returns a numerical ranking for a hand and also assigns an appropriate string for each hand.

Dealer.java

This class is used to shuffle the cards, deal out the cards to each player, and then to deal the flop, turn and river cards. This class also provides getter functions to return the appropriate cards. This class with also move the 'button' right. The button is used to identify where the blinds are located.

Kinds of players

- User, controlled by player interactions of the GUI
- Normal Artificial Intelligence Player
    - Strategy is based on a very good player
    - Decisions are done through logic trees

Possible extensions and future work:

- Adding different types of players (bad, average)
- AI can calculate hand probability of having the best hand compared to other players
- Make it so that you can play game over network, other users
- Adding the calculation to split the pot when two or more players have the same hand. This code can be added in the function compareHands() in the Dealer class.