Project 1 - Texas Hold 'em

or

How I learned poker

Martin T. H. Sandsmark

Basic structure of the code

My poker simulator and bot is structured into mainly six classes:

1. **Card class:** This is a very simple class representing a simple card.
2. **Deck class**: This is simply an ordered set of cards, also containing functions to detect different poker hands (such as straight flush or two pairs), calculate the absolute hand strength as defined in the hand out, and also an overloaded operator to be able to easily compare different hands, as well as some private helper functions.
3. **Player class**: This is a class representing a player. The most important function here is the "asses(Table *table)", which returns what action the player wants to take, given a table state (represented by the Table class, see below). It also maintains other state relevant to the player, like the amount of money it has, how many time it has betted in a given round, how many times it has won or tied, as well as whether it has folded this round. The default assess function represents the Phase I player.
4. **RolloutPlayer class:** This is a subclass of player, which overrides the virtual assess function and uses both pre-computed and on-the-fly statistical simulations to calculate the probability of winning with a given hand (and community cards). This is the Phase II player.
5. **Table class:** This represents a "table", that is cards on the table, order of players, the pot, how much the last person bet, etc. It also contains functions for running games, hands and betting rounds. It handles calling each player and controlling their actions, checking who wins, handing out the pot, taking in money from bets, etc.
6. **PreFlop class:** This class is basically a memory structure for storing the pre-computed statistical information from pre-flop rollouts, helper functions for doing the actual

computation, as well as storing and retrieving the data from comman-separated files (CSV-files).

Betting decisions

Phase I player: If it is before the flop has been dealt, there is a 50/50 chance of the player folding immediately. If it doesn't fold, has an offensive strategy and the last bet on the table was less than 150, it raises with 50 before the flop. If it doesn't have an offensive strategy, it is before the flop, and it hasn't folded it will simply call.
If it is after the fold it will use the static and absolute hand strength evaluation to see what action it should take (with only slight different thresholds between aggressive and defensive gameplay).

Phase II player: This player creates a PreFlop object, which in turns either loads statistical data from a file or generates it on the fly, if the file for some reason isn't available.
It then uses that statistical information to assess its chances of winning in the pre-flop betting round, and based on that decides whether to fold, call or raise. The aggressive player attempts to "bluff" by slightly altering the thresholds for the different actions (i. e. self-deluding about the actual worth of the cards).
It also uses pot odds, which is a representation of the expected value vs. how much the player has to bet. If the expected winning is less than the amount it has to bet, it will fold.

Results - Phase I

Round 1:

Player	Bluffing	Wins	Ties	Money at end
offensive4	true	416	166	122625
conservative3	false	4	0	-115200
offensive2	true	410	167	110750
conservative1	false	3	1	-114175

Round 2:

Player	Bluffing	Wins	Ties	Money at end

offensive4	true	411	160	95200
conservative3	false	5	0	-106950
offensive2	true	415	160	114250
conservative1	false	9	0	-98500

Round 3:

Player	Bluffing	Wins	Ties	Money at end
offensive4	true	434	160	149050
conservative3	false	8	0	-101450
offensive2	true	401	160	72800
conservative1	false	4	0	-116400

Round 4:

Player	Bluffing	Wins	Ties	Money at end
offensive4	true	416	146	87900
conservative3	false	8	0	-105200
offensive2	true	424	160	131575
conservative1	false	5	1	-110275

Round 5:

Player	Bluffing	Wins	Ties	Money at end
offensive4	true	389	175	63700
conservative3	false	6	0	-106450
offensive2	true	427	175	160950
conservative1	false	3	0	-114200

Results - Phase II

Round 1:

Player	Bluffing	Wins	Ties	Money at end
rolloutplayer4	false	2	0	-136700
rolloutplayer3	false	3	0	-141350
trolloutplayer2	true	233	78	198842
trolloutplayer1	true	215	77	104468
offensive8	true	216	80	131193
conservative7	false	5	0	-119700
offensive6	true	210	79	86250
conservative5	false	4	0	-115050

Round 2:

Player	Bluffing	Wins	Ties	Money at end
rolloutplayer4	false	8	0	-126500
rolloutplayer3	false	6	0	-112900
trolloutplayer2	true	231	83	190534
trolloutplayer1	true	222	76	134370
offensive8	true	205	71	44768
conservative7	false	3	0	-125600
offensive6	true	210	78	96269
conservative5	false	8	0	-93000

Round 3:

Player	Bluffing	Wins	Ties	Money at end
rolloutplayer4	false	6	0	-118600
rolloutplayer3	false	3	0	-140850
trolloutplayer2	true	220	79	125249
trolloutplayer1	true	195	75	4275
offensive8	true	219	72	122191
conservative7	false	2	0	-134800
offensive6	true	234	89	229916
conservative5	false	10	0	-79450

Round 4:

Player	Bluffing	Wins	Ties	Money at end
rolloutplayer4	false	4	0	-117850
rolloutplayer3	false	6	0	-128400
trolloutplayer2	true	230	78	184269
trolloutplayer1	true	238	70	204055
offensive8	true	184	64	-58530
conservative7	false	4	0	-116950
offensive6	true	219	74	134719
conservative5	false	8	0	-93350

Round 5:

Player	Bluffing	Wins	Ties	Money at end

rolloutplayer4	false	6	0	-108150
~~rolloutplayer3~~	~~false~~	~~6~~	~~0~~	~~-119650~~
trolloutplayer2	true	229	74	161591
trolloutplayer1	true	203	74	29348
offensive8	true	240	66	216265
conservative7	false	6	0	-104950
offensive6	true	207	74	57257
conservative5	false	2	0	-123750

We see from the results that bluffing is very effective in poker (that truthfully following your own estimated chances of winning are not necessarily good). We also see that rollout gives better results (the reason it is only marginally better is probably because I am not familiar with poker, and have been unable to set up good thresholds for the different actions it can choose between).