NICK'S BLOG

WEDNESDAY, JULY 18, 2007

Algorithm for evaluating poker hands

I've never actually seen the full description in one place on the net, so I thought I'd do a public service.

First, if you're playing a game with extra cards, like Hold'em or 7 stud, you first use recursion thusly: Iterate through the set of cards, removing one at a time and recurse. From the recursion results, save the best hand and return it.

Once you get down to 5 cards, you first take a histogram of the card ranks. That is, for each rank in the hand, count how often it appears. Sort the histogram by the count backwards (high values first).

If the histogram counts are 4 and 1, then the hand is quads.

If the histogram counts are 3 and a 2, then the hand is a boat.

If the histogram counts are 3, 1 and 1, then the hand is a set.

If the histogram counts are 2, 2 and 1, then the hand is two pair.

If the histogram has 4 ranks in it, then the hand is one pair.

Next, check to see if the hand is a flush. You do this by iterating through the cards to see if the suit of a card is the same as its neighbor. If not, then the hand is not a flush. Don't return a result yet, just note whether or not it's a flush.

Next, check for straights. Do this by sorting the list of

SECTIONS

poker (133)
tv (114)
technology (100)
ham (76)
iphone (62)
politics (44)
mac (30)
funny (22)
arduino (17)
web (15)
sports (14)
movies (12)
home_improvement (11)
misc (11)
music (11)
toast-r-reflow (11)
food (10)
unix (9)
java (8)
auto (4)
business (3)
administrivia (2)
blogosphere (2)

BLOG ARCHIVE

▶ 2014 (1)
▶ 2013 (20)

cards. Then subtract the rank of the bottom card from the top card. If you get 4, it's a straight. At this point, you must also check either for the wheel or for broadway, depending on whether your sort puts the ace at the top or bottom. I would expect most folks to put the ace at the top of the sort, since it is usually a high card. So to check for the wheel, check to see if the top card is an ace and the 2nd to top card is a 5. If so, then it is the wheel.

If the hand is a straight and a flush, it's a straight-flush. Otherwise if it's one or the other, you can return that.

If we haven't matched the hand by now, it's High Card.

Once you know what the hand is, it is fairly straightforward to compare two hands of the same type. For straights, you simply compare the top card. For flushes, you iterate down through a reverse sort of the ranks until you find a higher card or run out. For the histogram-based hands, you start at the beginning of the reverse-sorted count list and compare the ranks (two pair is tricky - you must always evaluate the higher of the two pairs for each hand first - the histogram won't help, then the lower pair, then the kicker).

Checking for a low (for high/low or Razz) is a little easier. Sort the hand by rank. Iterate through the list. If any card is bigger than an 8 or is the same rank as it's neighbor, it's not a low hand.

POSTED BY NICK AT 10:16 AM
LABELS: POKER

16 COMMENTS:

Anonymous said...

This is exactly what I was looking for.
I'm trying to build a texas hold 'em game in python (just for fun and learning) and I needed some clue on how to calculate a poker hand.

So, thank you very much for sharing.

AUGUST 6, 2007 AT 3:23 AM

Thomas said...

- ▶ 2011 (12)
- ▶ 2010 (86)
- ▶ 2009 (144)
- ▶ 2008 (127)
- ▼ 2007 (125)
  - ▶ December (7)
  - ▶ November (6)
  - ▶ October (12)
  - ▶ September (17)
  - ▶ August (21)
  - ▼ July (11)
    - Poker at the Commerce Casino
    - You don't know Jack on the web!
    - Algorithm for evaluating poker hands
    - MST3K mark 2
    - Metallica IS Spinal Tap
    - Was this a bad play?
    - A solution to the GPS-less iPhone
    - Heads-up position
    - WPT Ca$h camp
    - Victory!
    - Little things in iPhone
  - ▶ June (27)
  - ▶ May (24)

SUBSCRIBE TO

Posts

Comments

LINKS

My Home Page
My YouTube page

I have written such a code that is similar to this (including calculating for 7 card hands). The problem I have with it is that it is very very slow; I use it to calculate odds of making hands given x hands. The down side is that to do this, I have to calculate 16000 hands to get a good representation of odds, and that takes 4 secs or so to do. Compare this to software out there called "pokerstove" and you will see I am way slow. His is able to calculate the odds of 5 different players hands preflop in about 12 seconds using 4.4billion games. I will be happy to share the code in java if interested, but I am looking for a streamlined algorythmn.

APRIL 14, 2008 AT 5:34 PM

PJ said...

That much I'd gotten on my own, but the part I'm having trouble with is finding the odds for partial hands. Say I'm playing 7 Card Stud, and at 4th street I have my four cards, my opponent's two upcards, and a list of all the cards in the deck I haven't seen yet. Is there a good way to find the odds of my hand beating his at showdown, assuming his hole cards and all the cards not yet dealt are chosen randomly from the list of remaining cards? The only way I know is to just try a large number of combinations, but that can't be the best way.

JULY 30, 2008 AT 1:55 AM

Nick said...

When I have needed to do this, I've used the Monte Carlo method. You simply run the hand 1000 times and see how many times it wins. It's brute force, and it's only an approximation, but surprisingly it doesn't take terribly long to do it with modern CPUs.

JULY 30, 2008 AT 9:14 AM

Nick said...

By the way, everyone, there is another algorithm out there that uses a giant array of pre-computed hand values that is much, much faster than my method. The description of his method and source

My Lovely Wife Scarlet

English Wikipedia

Bay 101

Card Player Hold'Em Odds Calculator

FOLLOWERS

Join this site
with Google Friend Connect

Members (6)

Already a member? Sign in

SEARCH THIS BLOG

Search

code is here.

JULY 30, 2008 AT 9:17 AM

Ottepsir said...

Thanks for helping me. I've only found a little mistake...I think...in the algorithm of straight...it is not sufficient to subtract the rank of the bottom card with the rank of the top one and obtain 4, because the rule doesn't work in the case that we have a Pair in the middle...

MARCH 17, 2009 AT 4:27 AM

Nick said...

You must be doing things in the wrong order, then. The histogram based check will eliminate all hands with pairs, sets or quads. You only need to check for straights and flushes with non-paired hands.

MARCH 17, 2009 AT 5:31 AM

Anonymous said...

I have developed a method that is much more efficient than this, which ends up assigning values to hands.

sort the cards by rank.
shift and subtract
store result in 1x4 array, p0
Turn on non-zero values to 1, p1
look for number of 0's in the array (and placement determines strength in given category.)
examples:
{1,0,1,1} would be a pair
{1,0,0,1} would be trips
{1,0,0,0} would be quads
{0,1,1,0} would be 2-pair
{0,1,0,0} would be full-house
This is much like the histogram method, but you can weight the positions and rankings with values, thereby assigning actual values to hands, making ranking multiple hands very efficient. You have to work in base 13...
for example
13^4 is the foundation for pairs

13^5 is the foundation for 2-pair
... etc.
13^10 would be foundation for full house
...
I did this in Mathematica one afternoon...
I can sort ~ 5,000,000 hands in 11 seconds

if you want more details... e-mail me.

poker AT deckhole DOT COM

APRIL 9, 2009 AT 8:12 PM

Nick said...

That doesn't look tremendously more efficient to me. You still have to sort the hand and check each card against its adjacent neighbor for equality (that's going to be faster than subtracting the two and then substituting 1 for non-zero values). You also wind up with situations where there are, in fact, two different ways to see a boat: {0,0,1,0} and (0,1,0,0}. And, like my method, you still have to have a special case for flushes and straights.

No, the really, *really* efficient way to do poker hand evaluation is the one I linked to in the 4th comment. But understanding that method requires you to understand some serious math. My method is less efficient, but more understandable.

APRIL 10, 2009 AT 8:53 AM

kromped said...

"Next, check for straights. Do this by sorting the list of cards. Then subtract the rank of the bottom card from the top card. If you get 4, it's a straight."

Also check if the array contains duplicates. If you have two pairs, and a single card that is low enough so that the high card subtract will return 4 which is not a straight.

DECEMBER 3, 2009 AT 4:54 PM

Nick said...

Nope. You would have caught hands with pairs in them earlier in the process when doing the histogram operation. There's no need to check for straights or flushes on any hands with a pair.

DECEMBER 3, 2009 AT 4:58 PM

cdrick said...

I did a version a bit similar that looks more optimal to me. I'll test later (but in Smalltalk - squeak version, which is slower that java), I can simulate 16000 in 8 seconds. I guess in Cit should be around the second. I'll try later.

Basically, I think the recursion is useless (having the reverse "histogram" reverses is enough).

The tip is that if we have to find a combination on 7 cards max, a flush imply no possible quad. Therefore, only straighted flush are better.

Then, another optimization is to sort and count colors in one iteration.

Last point, is I notice that 25% of time is spent is shuffling cards. So my next optimization will be regarding this aspect.

cu

FEBRUARY 20, 2010 AT 8:54 AM

supertunaman said...

Thanks for this, it is much appreciated.

I implemented this in C for a project of mine involving Texas Hold'em and IRC. I can process 5,000,000 5-card hands in less than 5 seconds with no optimizations. With -O2, it got through 5,000,000 in about 1.3 seconds. Great for my purposes!

If you're on github, check out supertunaman/ircpoker. that's my project.

JANUARY 13, 2011 AT 7:49 PM

spitfire said...

Thank you very much for posting this up. It was very helpful and thorough.

AUGUST 4, 2011 AT 8:46 PM

supermouse said...

I have just finished writing a texas holdem %win calculator in excel vba, you enter the number of players then your hole cards then it calcs you odds of winning by the river. Once you enter the flop cards it calcs your odds of having the best hand by the turn and the river, it does this also for the turn and river cards..... the thing is i'm stunned by the speed that is claimed by the posters here, obviously excel vba is very slow, it takes about 12 secs to calc odds of winning by the river card (21 possible combinations of hand with 7 cards) with 6 players but that's only running 20 simulations! so not very accurate! i need >1000 simulations to be reliably accurate. i only know vba i've never heard of '-O2' what would be the easiest/quickest operating language to learn to do this ?

SEPTEMBER 10, 2011 AT 2:51 AM

Jon Molnar said...

I've been working on a Texas Hold'em minigame for Skyrim that needed a memory-efficient algorithm that could be expressed in the Papyrus scripting language. This was just the ticket. Thanks!

OCTOBER 31, 2014 AT 8:48 AM

Post a Comment

LINKS TO THIS POST

Create a Link

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)