# The Poker Squares Challenge

Todd W. Neller

# What is the Poker Squares Challenge?

- A semester-long contest where Gettysburg College students (individuals and/or teams) compete to develop the best time-limited Poker Squares playing program.
- Outline:
  - Learn how to play
  - Play
  - Discuss strategy
  - Present possible computational approaches
  - Contest details

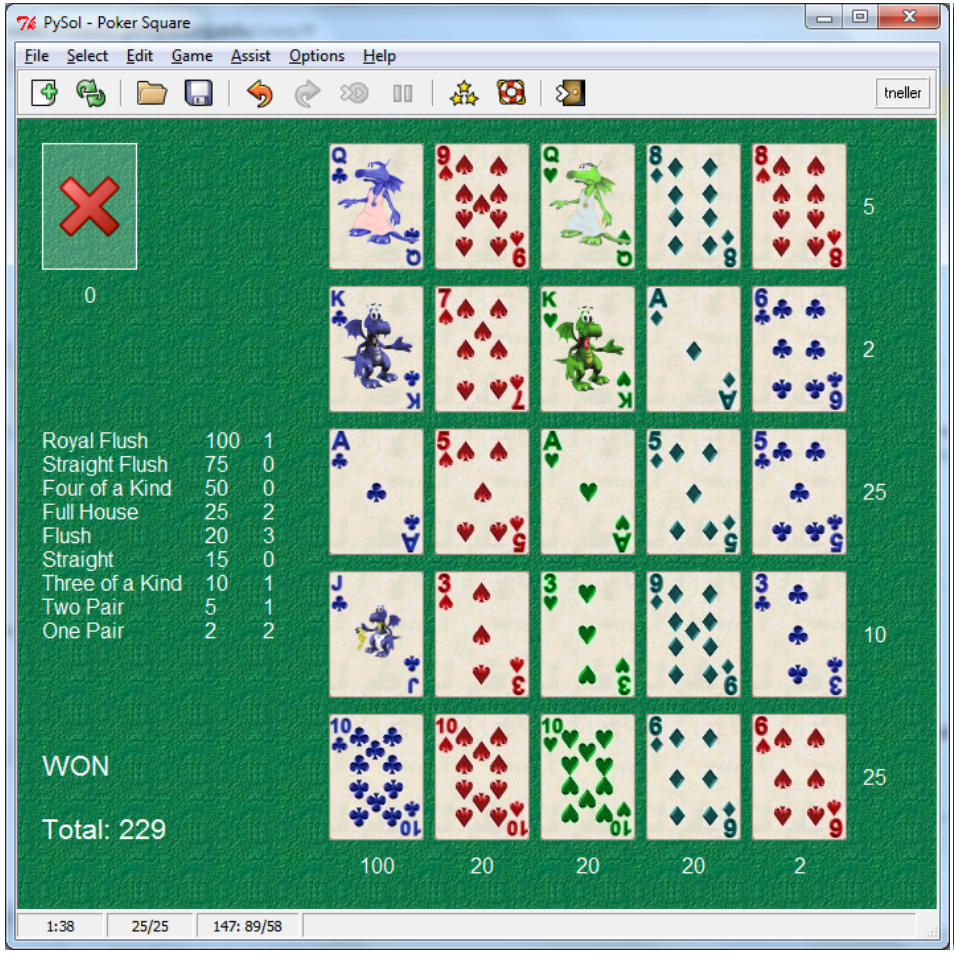
# Poker Squares

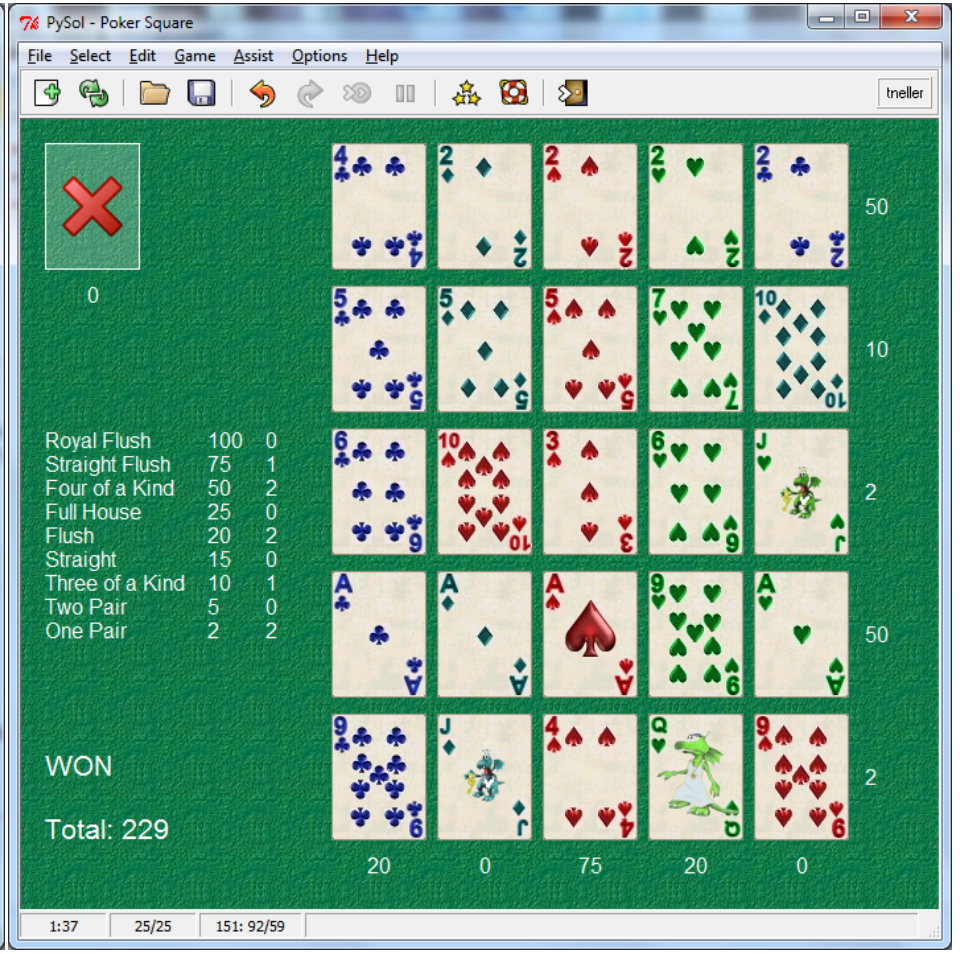
- Materials:
  - shuffled standard (French) 52-card card deck,
  - paper with 5-by-5 grid, and
  - pencil
- Each turn, a player draws a card and writes the card rank and suit in an empty grid position.
- After 25 turns, the grid is full and the player scores each grid row and column as a 5-card poker hand according to the American point system.

# American Point System

| Poker Hand | Points | Description | Example |
|---|---|---|---|
| Royal Flush | 100 | A 10-J-Q-K-A sequence all of the same suit | 10♣, J♣, Q♣, K♣, A♣ |
| Straight Flush | 75 | Five cards in sequence all of the same suit | A♦, 2♦, 3♦, 4♦, 5♦ |
| Four of a Kind | 50 | Four cards of the same rank | 9♣, 9♦, 9♥, 9♠, 6♥ |
| Full House | 25 | Three cards of one rank with two cards of another rank | 7♠, 7♣, 7♦, 8♥, 8♠ |
| Flush | 20 | Five cards all of the same suit | A♥, 2♥, 3♥, 5♥, 8♥ |
| Straight | 15 | Five cards in sequence; Aces may be high or low but not both | 8♣, 9♠, 10♥, J♦, Q♣ |
| Three of a Kind | 10 | Three cards of the same rank | 2♠, 2♥, 2♦, 5♣, 7♠ |
| Two Pair | 5 | Two cards of one rank with two cards of another rank | 3♥, 3♦, 4♣, 4♠, A♣ |
| One Pair | 2 | Two cards of one rank | 5♦, 5♥, 9♣, Q♠, A♥ |
| High Card | 0 | None of the above | 2♦, 3♣, 5♠, 8♥, Q♦ |

# Scoring Examples

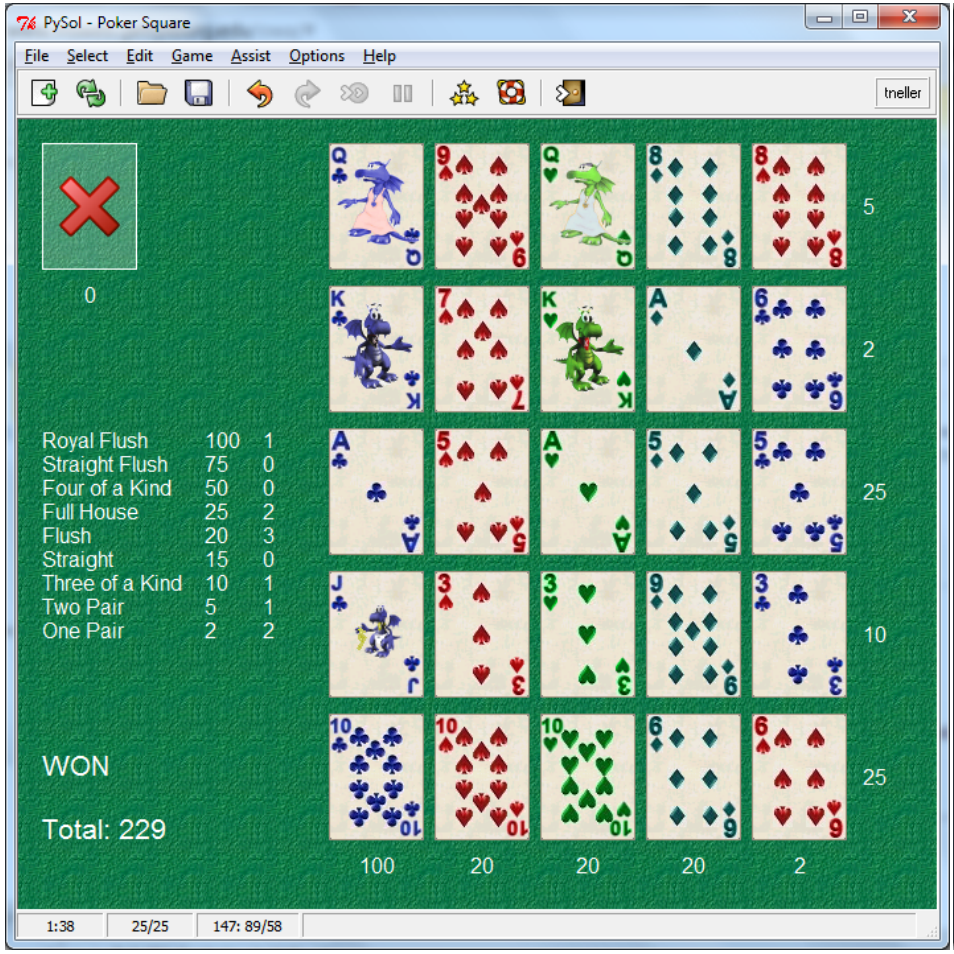


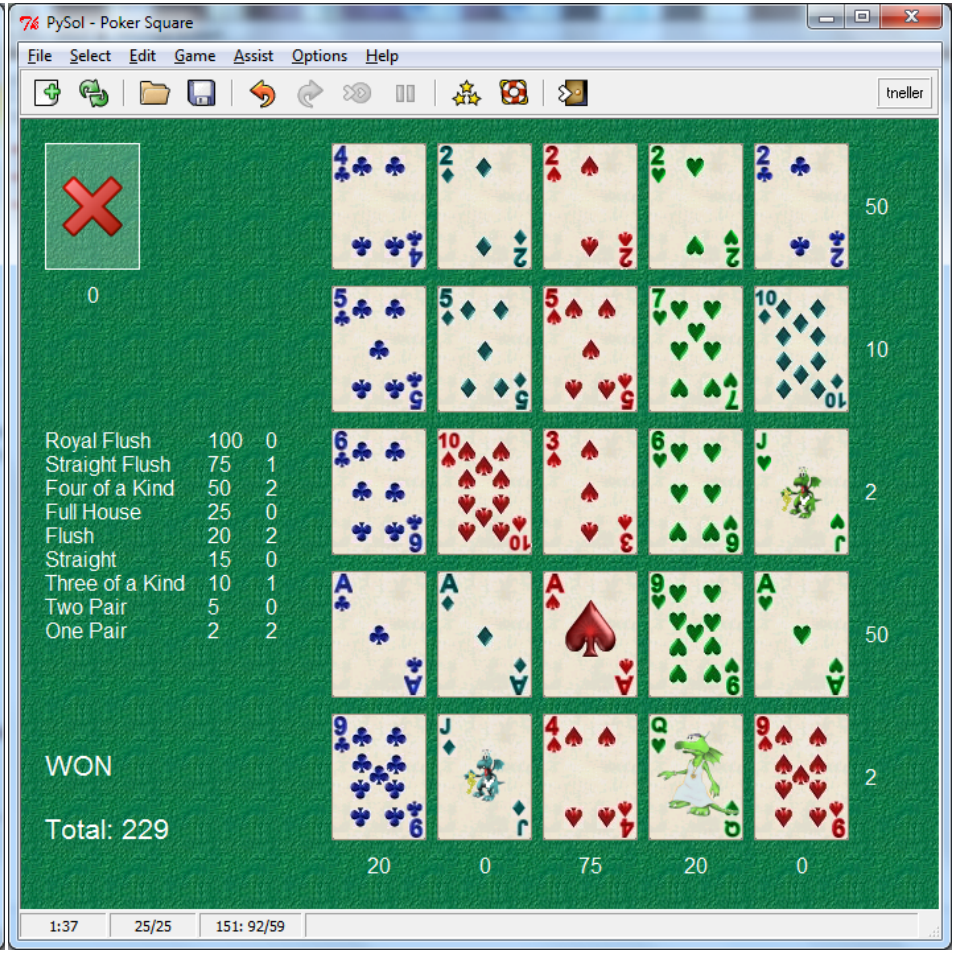



# Let's Play!

| Poker Hand | Points | Description | Example |
|---|---|---|---|
| Royal Flush | 100 | A 10-J-Q-K-A sequence all of the same suit | 10♣, J♣, Q♣, K♣, A♣ |
| Straight Flush | 75 | Five cards in sequence all of the same suit | A♦, 2♦, 3♦, 4♦, 5♦ |
| Four of a Kind | 50 | Four cards of the same rank | 9♣, 9♦, 9♥, 9♠, 6♥ |
| Full House | 25 | Three cards of one rank with two cards of another rank | 7♠, 7♣, 7♦, 8♥, 8♠ |
| Flush | 20 | Five cards all of the same suit | A♥, 2♥, 3♥, 5♥, 8♥ |
| Straight | 15 | Five cards in sequence; Aces may be high or low but not both | 8♣, 9♠, 10♥, J♦, Q♣ |
| Three of a Kind | 10 | Three cards of the same rank | 2♠, 2♥, 2♦, 5♣, 7♠ |
| Two Pair | 5 | Two cards of one rank with two cards of another rank | 3♥, 3♦, 4♣, 4♠, A♣ |
| One Pair | 2 | Two cards of one rank | 5♦, 5♥, 9♣, Q♠, A♥ |
| High Card | 0 | None of the above | 2♦, 3♣, 5♠, 8♥, Q♦ |

# Strategy Discussion



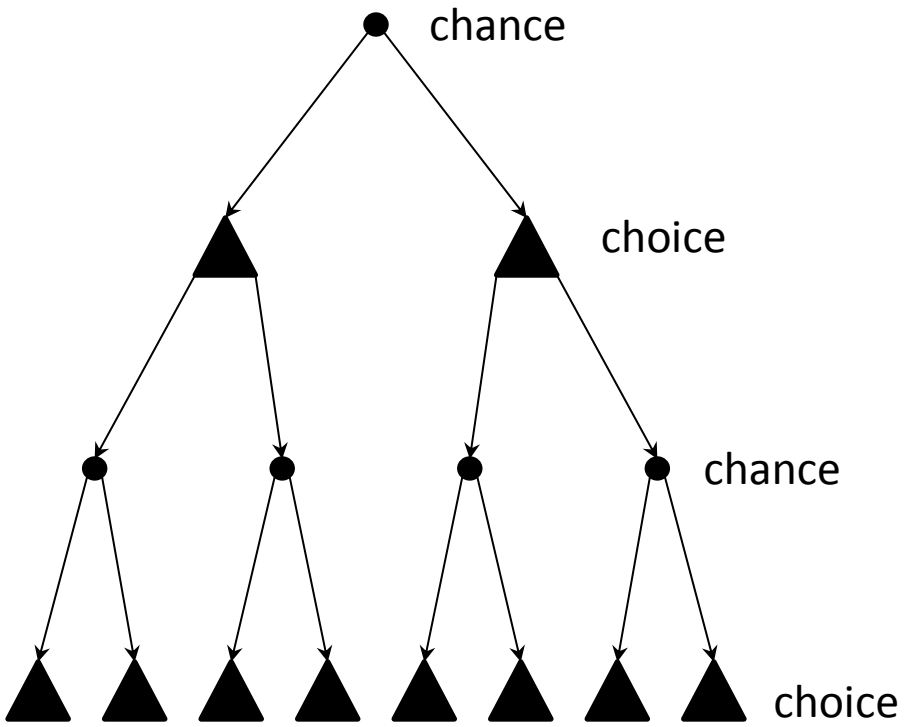



# Possible Computational Approaches

- Rule-based: hard code an algorithm (e.g. decision tree) for the placement of cards
  - Example: Place cards so as to maximize potential column flushes and row rank repetitions
- Simple Monte Carlo:
  - For each possible play, shuffle remaining cards and simulate a number of random/rule-based playouts.
  - Choose the play that yields the best average result.
- More complex Monte Carlo play is possible.

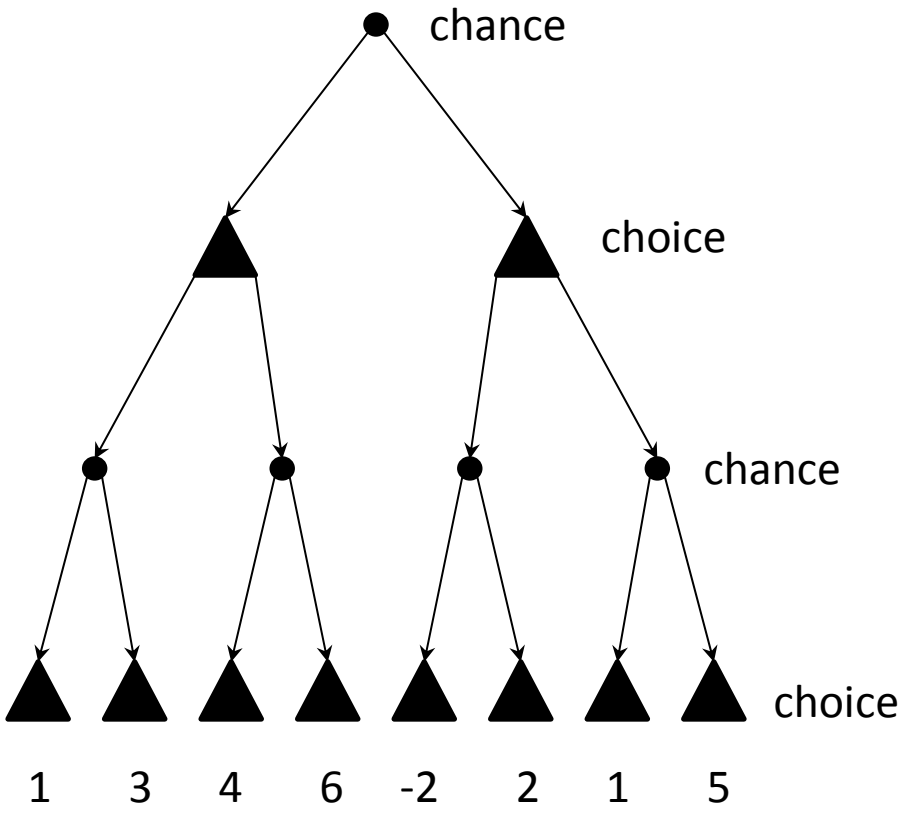
# Structure of the Game

- The game is structured as an alternating sequence of *chance nodes* and player *choice nodes*.
  - Each card draw is a probabilistic event where any remaining card is drawn with equal probability.
  - Each player *action* is a commitment to a card placement.



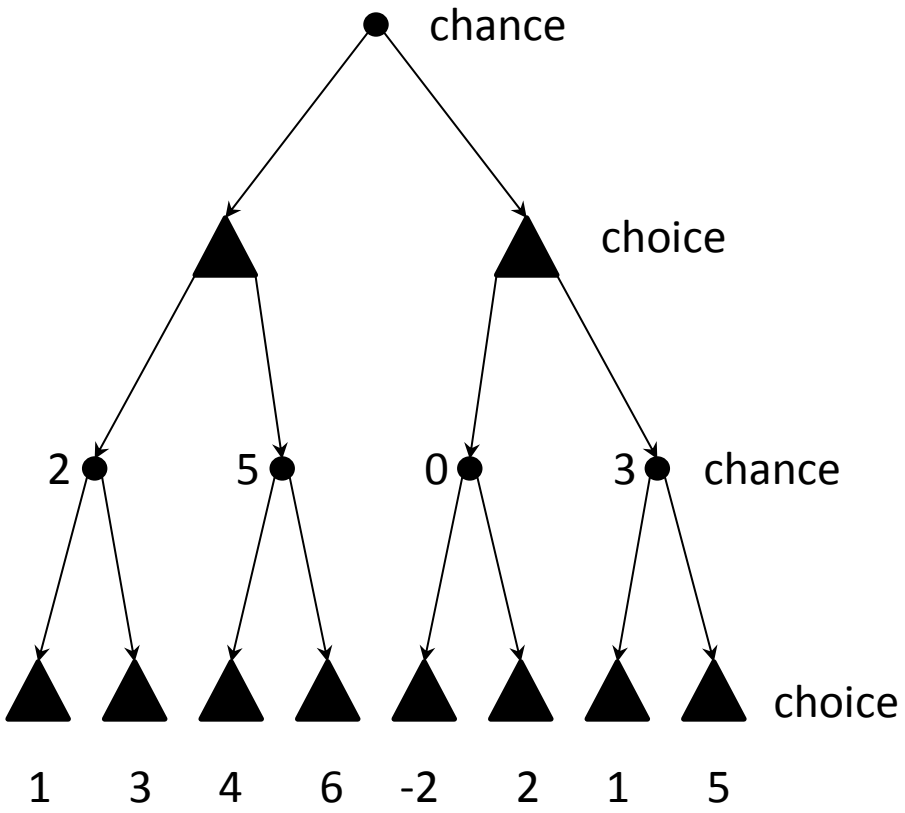
# Expectimax Example

- Assume:
  - all chance events are equiprobable
  - numbers indicate node utility (e.g. score)
- What is the expected value of the root chance node?



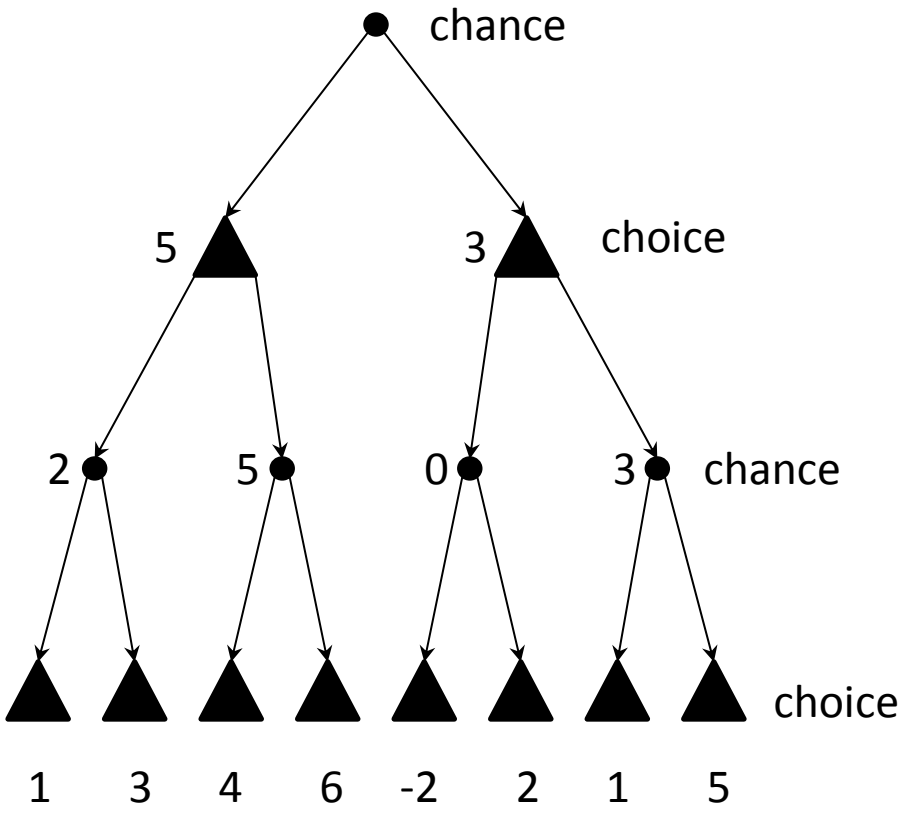
# Expectimax Example

- Assume:
  - all chance events are equiprobable
  - numbers indicate node utility (e.g. score)
- What is the expected value of the root chance node?



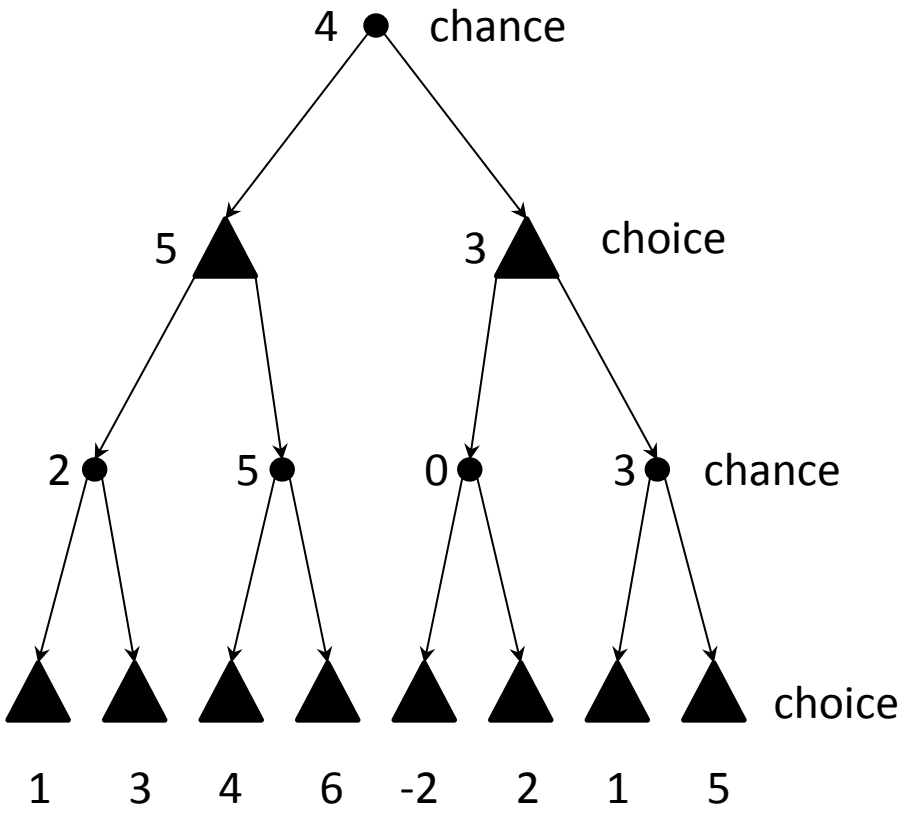
# Expectimax Example

- Assume:
  - all chance events are equiprobable
  - numbers indicate node utility (e.g. score)
- What is the expected value of the root chance node?



# Expectimax Example

- Assume:
  - all chance events are equiprobable
  - numbers indicate node utility (e.g. score)
- What is the expected value of the root chance node?



# Game Tree Size

- How big is the Poker Squares game tree?
  - Root chance node: 52 possible cards
  - 52 depth-1 choice nodes: 25 possible placements
  - 52x25 depth-2 chance nodes: 51 possible cards
  - 52x25x51 depth-3 choice nodes: 24 possible placements
  - …
  - $52!/27! \times 25! = 52!/(27 \times 26) \cong 1.15 \times 10^{65}$ nodes
  - Although:
    - Different draw/play sequences can lead to the same state.
    - Rows/columns may be reordered without affecting score.
  - Still, we will not be able to evaluate entire expectimax trees except for much smaller end-game situations.

# Static Evaluation

- Another approach: optimize static evaluation
  - Static evaluation: a measure of the relative goodness/badness of a partially filled grid.
  - Simple depth-1 greedy play: place a card so as to achieve the best static evaluation of the resulting board
  - More generally, compute depth-*n* expectimax for small *n*, using static evaluation at the depth limit.
  - Still, *n* must remain small for fast tree evaluation.

# Monte Carlo Sampling

- We can reduce the branching factor and evaluate more deeply and approximately by *sampling.*
- Chance events and/or actions may be sampled:
  - At each chance node, average a sample drawn from the given probability distribution.
  - At each choice node, maximize a sample of the possible actions.
- However, we'd like to sample better plays more often to discern which is the best.

# Monte Carlo Tree Search (MCTS)

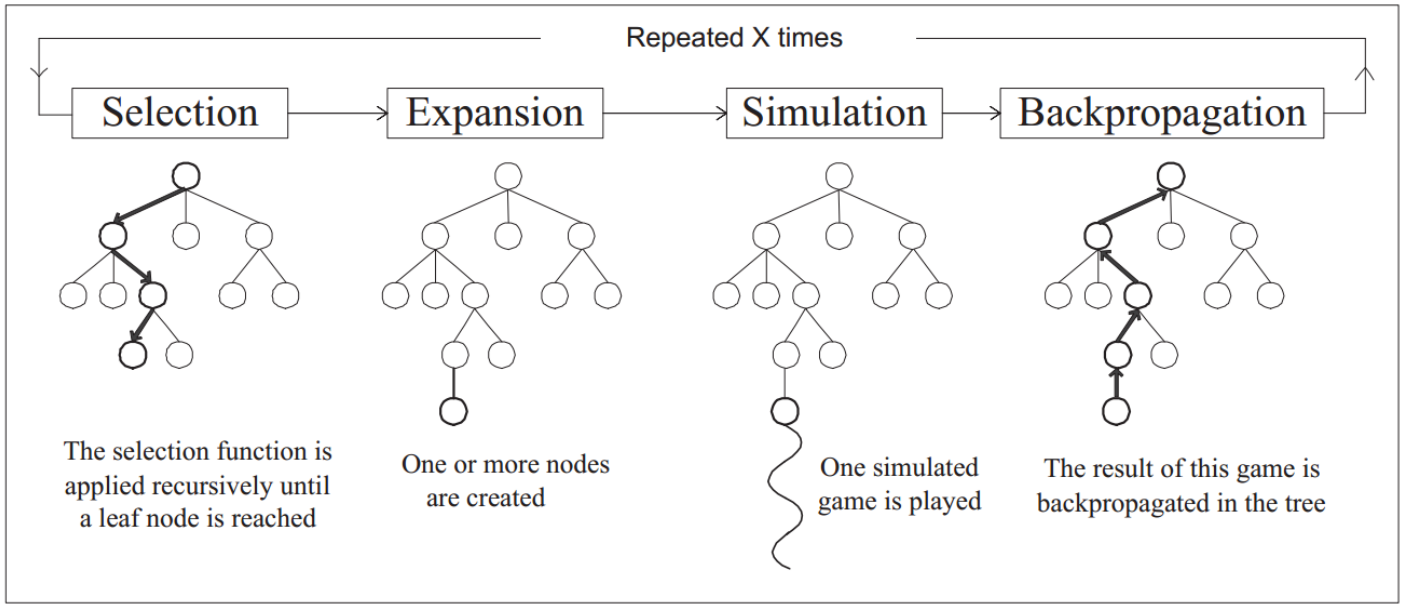



Figure from http://www.personeel.unimaas.nl/g-chaslot/papers/newMath.pdf

- Monte Carlo Tree Search details are beyond the scope of this talk, but
  - UCT is a popular form of MCTS: L. Kocsis, C. Szepesvari. *Bandit based Monte-Carlo Planning.*
  - Richard Lorentz has recently had success adapting UCT to a game with similar structure: R. Lorentz. *An MCTS Program to Play EinStein Würfelt Nicht!*

# Combining Static Evaluation and MCTS

- One can also combine the ideas of static evaluation and MCTS by
  - Limiting depth of MCTS playouts, and
  - Using static evaluations instead of terminal evaluations
- Many different approaches are possible
  - The better the static evaluation, the less the need for tree search.
  - Perfect static evaluation → use simple greedy play!

# Contest Details

- From http://tinyurl.com/pokersqrs, download:
  - Card.java: basic card object
  - PokerSquares.java: game simulator, player tester
  - PokerSquaresPlayer.java: simple player interface
  - RandomPokerSquaresPlayer.java: random player
- Run RandomPokerSquaresPlayer to see random game.
- Run PokerSquares to see RandomPokerSquaresPlayer test.
  - Mean score: 14.4, standard deviation: 7.6
- Each game is limited to 1 minute. A player taking longer than 1 minute on a game scores 0 for that game.

# Dickinson Contest Suggestion

- Mid-semester trial contest:
  - Submissions due before spring break to local coordinator, results available after break.
- End-semester contest:
  - Submissions due  at the end of the 2nd-to-last week of classes, results available in the last week of classes.
- Prizes?

# Be Encouraged

- Don’t let the complexity of some of these approaches discourage you from trying. This is an open problem; the best approach is unknown. Remember the **KISS principle**.
- Recall that random play has a mean score of 14.4 with a standard deviation of 7.6.
- A *very* simple player of mine with a **15-line** getPlay method has a **mean score of 81.1** with a standard deviation of 16.8. Can you guess what it does?

# Resources and References

- Gettysburg College Poker Squares Page: http://tinyurl.com/pokersqrs
  - References
  - Rules and play grids
  - Contest code
- Monte Carlo Tree Search (MCTS):
  - L. Kocsis, C. Szepesvari. *Bandit based Monte-Carlo Planning.*
  - http://www.mcts.ai/?q=mcts
- MCTS application to similar problem: R. Lorentz. *An MCTS Program to Play EinStein Würfelt Nicht!*