COMPSCI 767: Intelligent Software Agents
Approaches for building autonomous poker agents

Jonathan Rubin
University of Auckland Game AI Group

http://www.cs.auckland.ac.nz/research/gameai

Overview

- Texas Hold'em
- Types of Strategies
- Approaches
  - Knowledge based systems
  - Imperfect information game tree search
  - Monte-Carlo Simulation
  - Game Theoretic approaches
  - Evolutionary algorithms
  - Case based reasoning
- Recommended websites and papers

Introduction

Introduction

- Give brief introduction to main approaches in computer poker
- Highlight appropriate papers where more information can be obtained

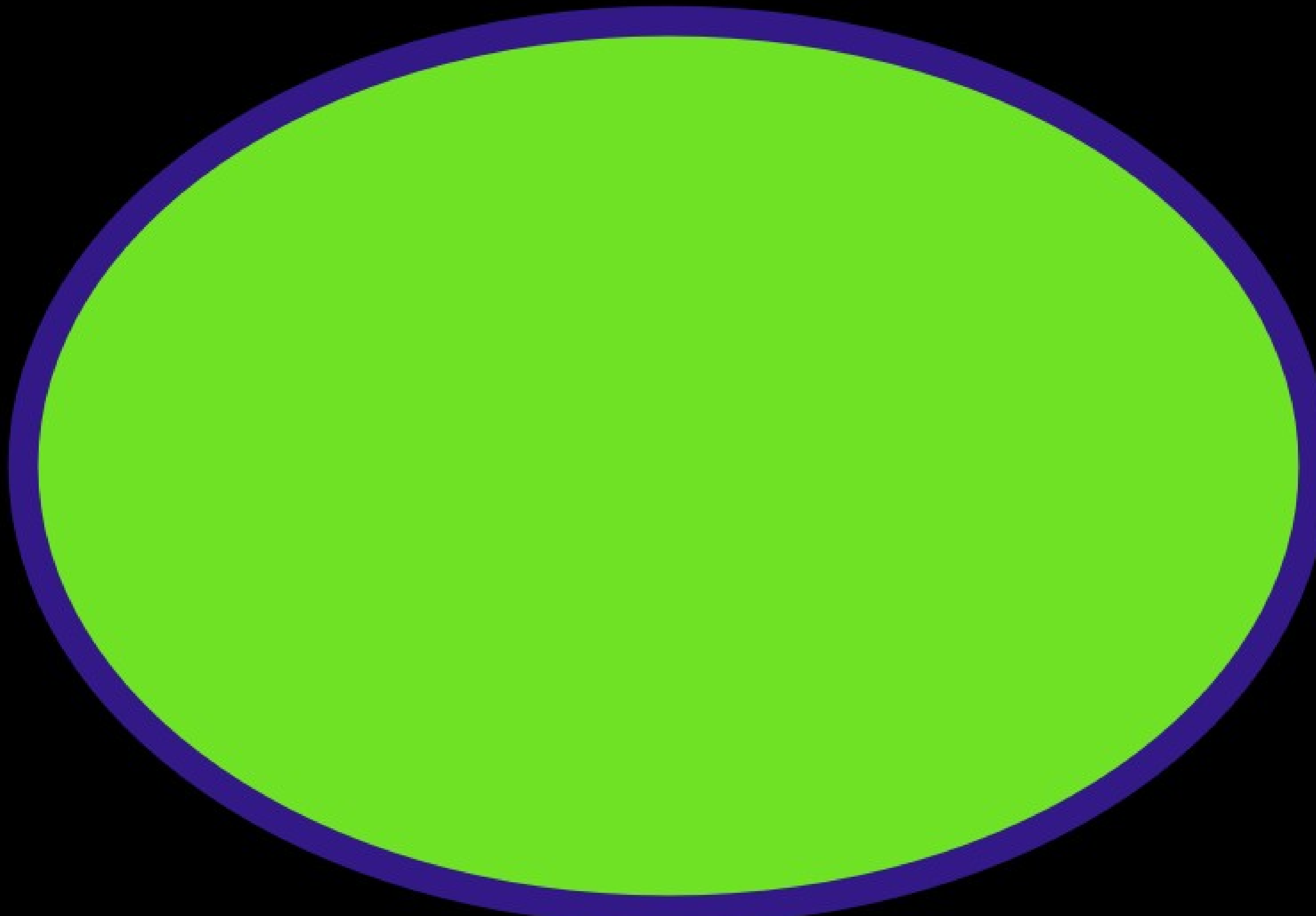
AI and Games

- Fun!
- Well defined rules and boundaries
- Clear goals and objectives
- Sophisticated strategies and tactics
- Embedded performance metrics

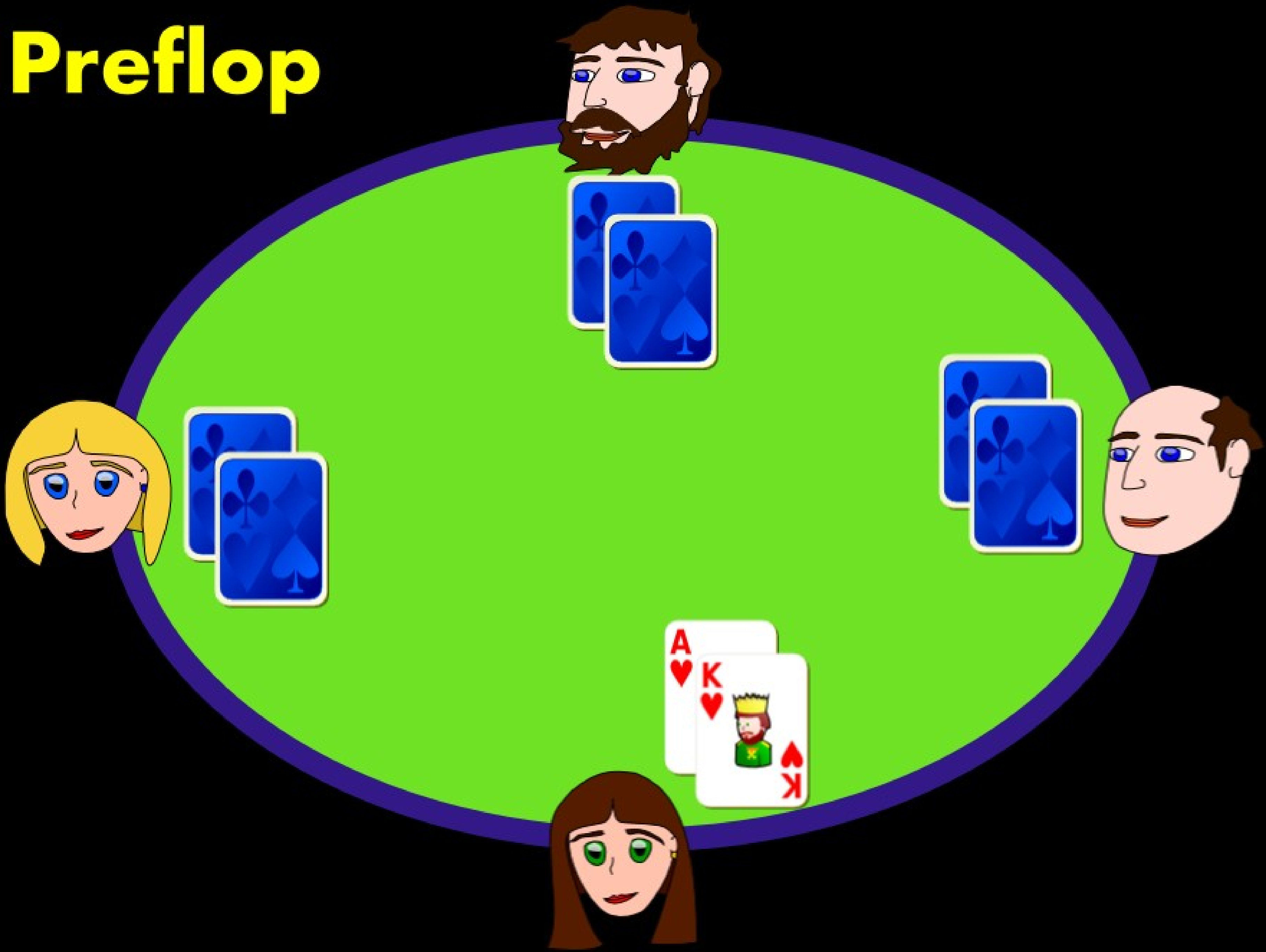
The Poker Domain

- Identified as challenging domain for AI research
- Imperfect Information
  - Other players hidden cards
- Chance events
  - Random dealing of cards
- Increasingly popular
  - AAAI Computer Poker Competition

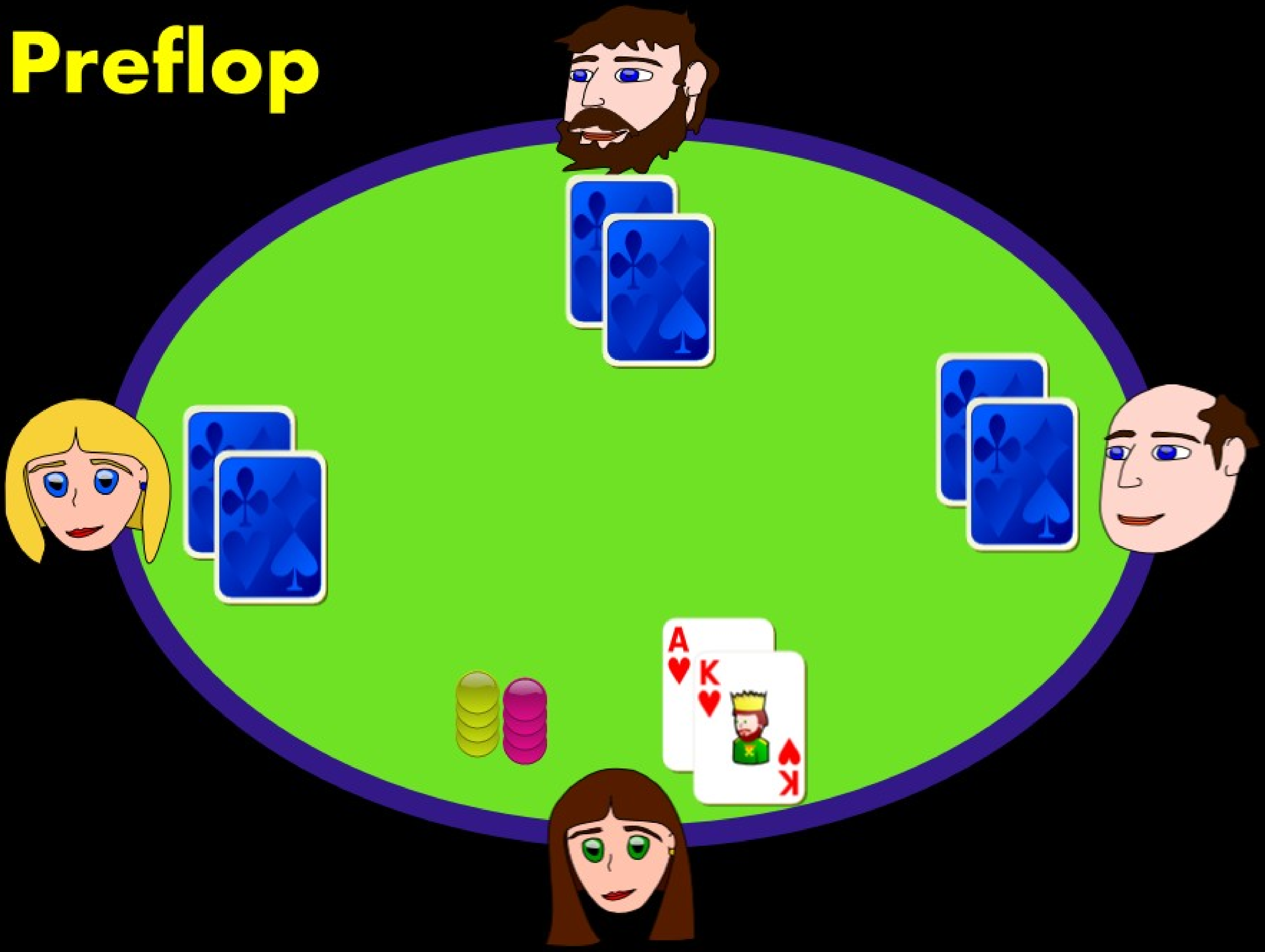
The Rules of Texas Hold'em



Preflop



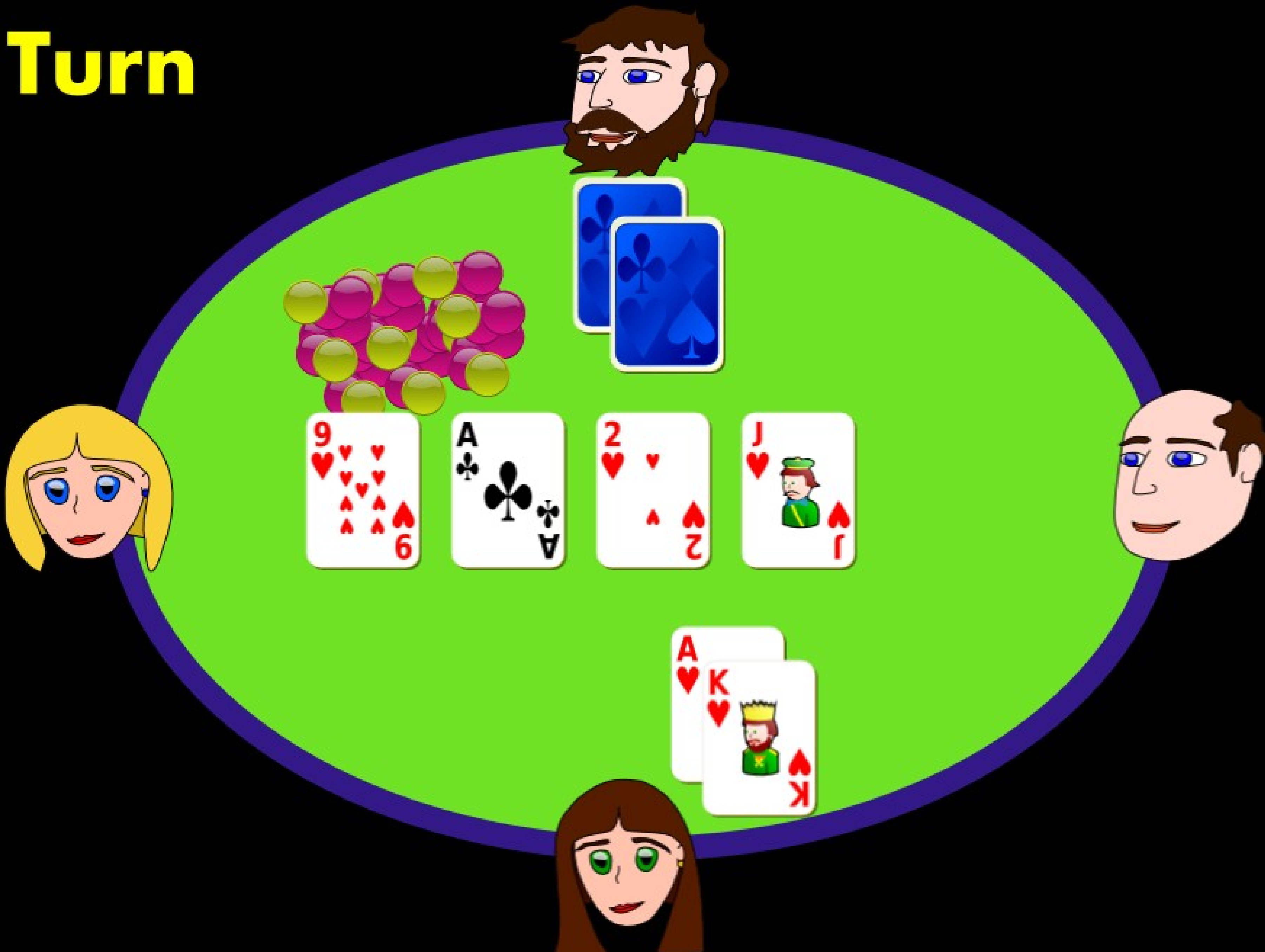
Preflop



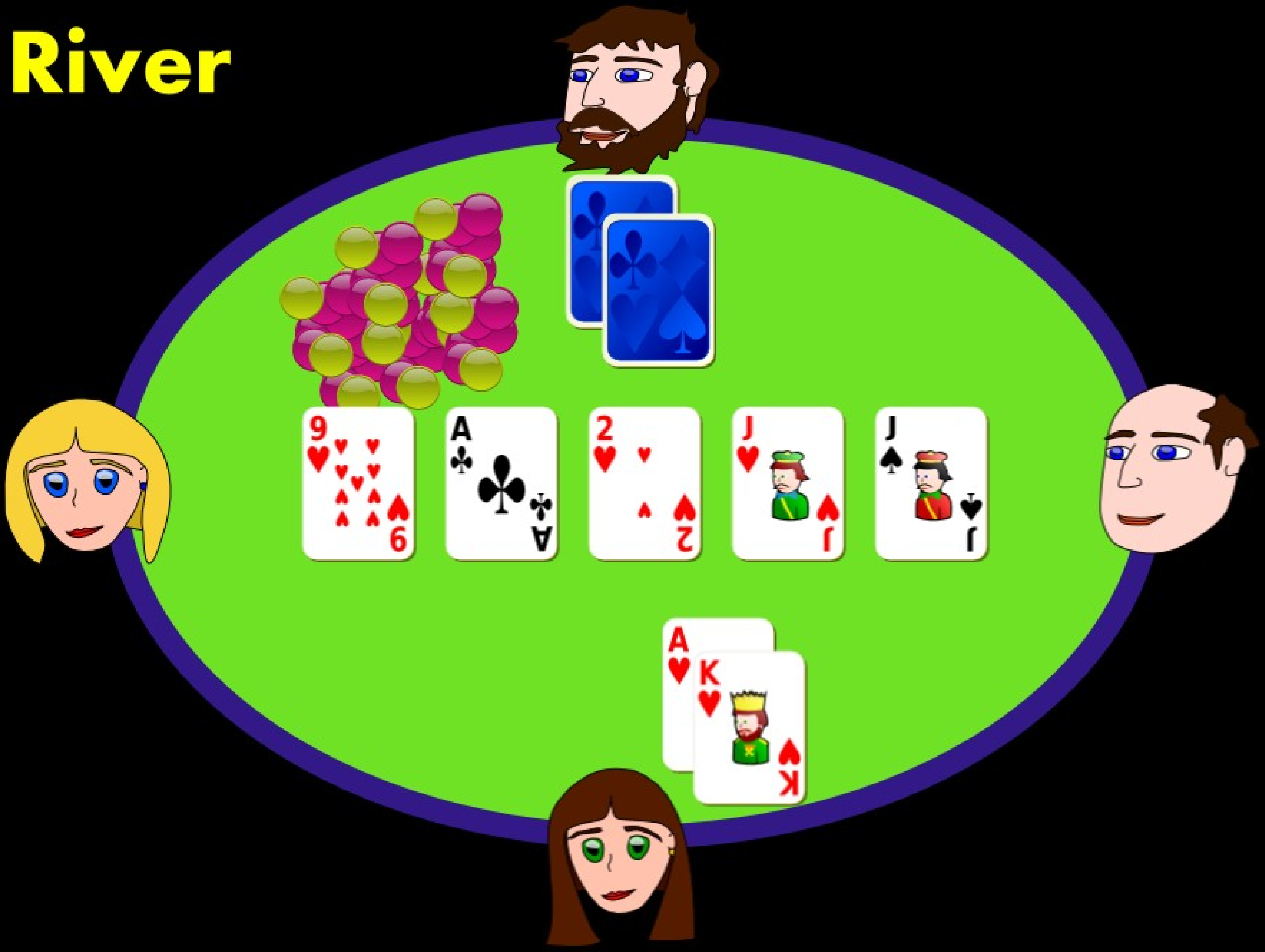
Preflop

Flop

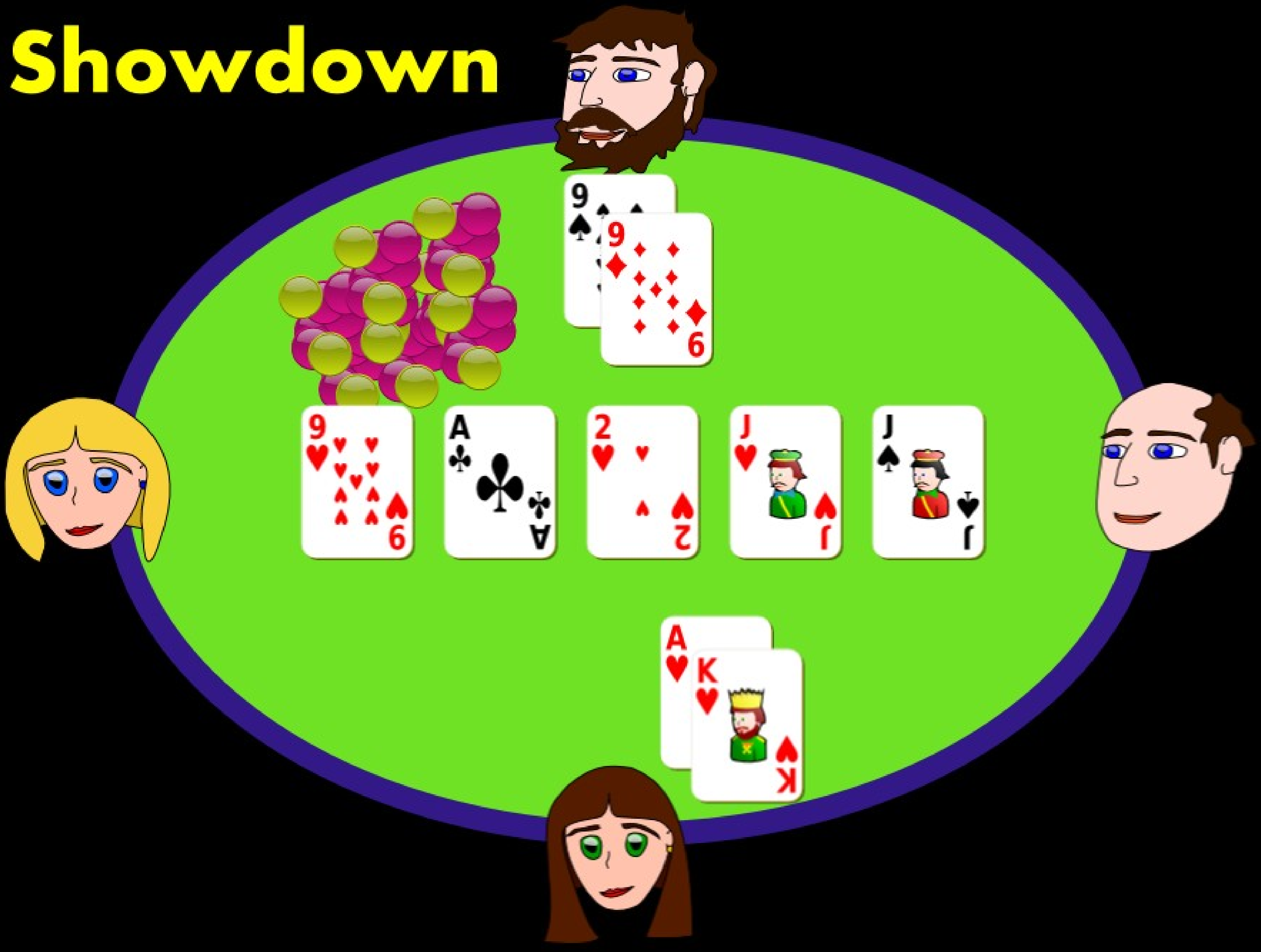
Turn



River



Showdown



Texas Hold'em

- Current Focus
  - Heads up (2 players)
  - Limit betting
    - $2/$4 Hold'em

Poker Strategies

A Poker Strategy

- At every decision point a probability triple is required that indicates the proportion of the time a player should either fold, call or raise

$$(f,c,r) \rightarrow (0,\ 0.5,\ 0.5)$$

Types of Strategies

- *e*-Nash Equilibria
  - Robust strategies that attempt not to lose to any type of opponent
- Exploitive Strategies
  - Attempts to react to an opponent's play in a way that allows maximum exploitability of that opponent
  - Requires opponent modeling

Rock-Paper-Scissors Example

- Nash equilibrium
  - *(R,P,S) → (1/3, 1/3, 1/3)*
  - The Nash player will never lose against any player in the long run

Rock-Paper-Scissors Example

- Nash equilibrium
  - *(R,P,S) → (1/3, 1/3, 1/3)*
  - The Nash player will never lose against any player in the long run
- Along comes Jimmy who only ever plays Paper

Rock-Paper-Scissors Example

- The Nash player will continue to play
  - *(1/3, 1/3, 1/3)*
  - Lose 33%, Win 33%, Draw 33%
  - The Nash player will still only draw against Jimmy

Rock-Paper-Scissors Example

- However because we know Jimmy's strategy, an exploitive player would be better off using the strategy
  - *(0, 0, 1)*
  - i.e. a best response that maximally exploits Jimmy at every decision point
- Now, against Jimmy the exploitive player will win
  - Consequence is that the exploitive player plays off the equilibrium, and is hence subject to potential exploitation itself

Approaches to creating poker agents

Knowledge-Based Systems

- Rule-based expert systems
- Formula-based expert systems

Rule-Based Expert System

- Collection of if-then rules

```
Action preflopAction(Hand hand, GameState state){
    if( state.betsToCall > 2 &&
        state.playersInHand > 1 &&
        state.relativePosition > 0.8 &&
        hand.AAo){
            return getAction(new Triple(0.0, 0.05, 0.95));
    } else if...
}
```

Formula-Based Expert System

- Accepts a collection of (possibly weighted) inputs
- Outputs a probability triple

$$f(x_1, x_2, \ldots, x_n) \Rightarrow (f,c,r)$$

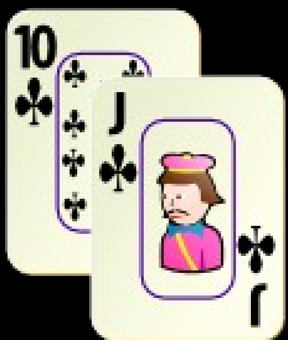
- Inputs are things like

Hand rank

Pot Odds

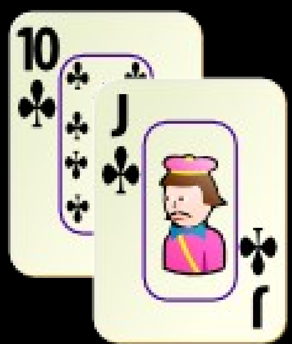
Immediate Hand Rank

What is the quantitative value of this hand?

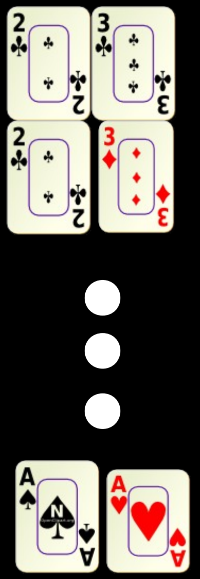


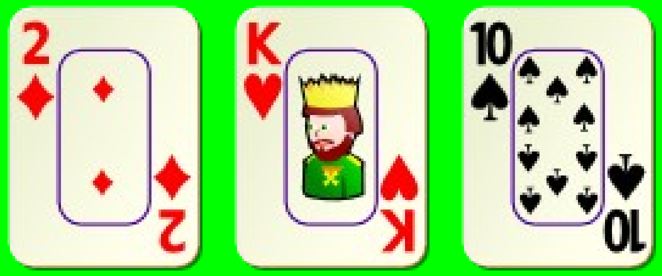
Flop

Immediate Hand Rank



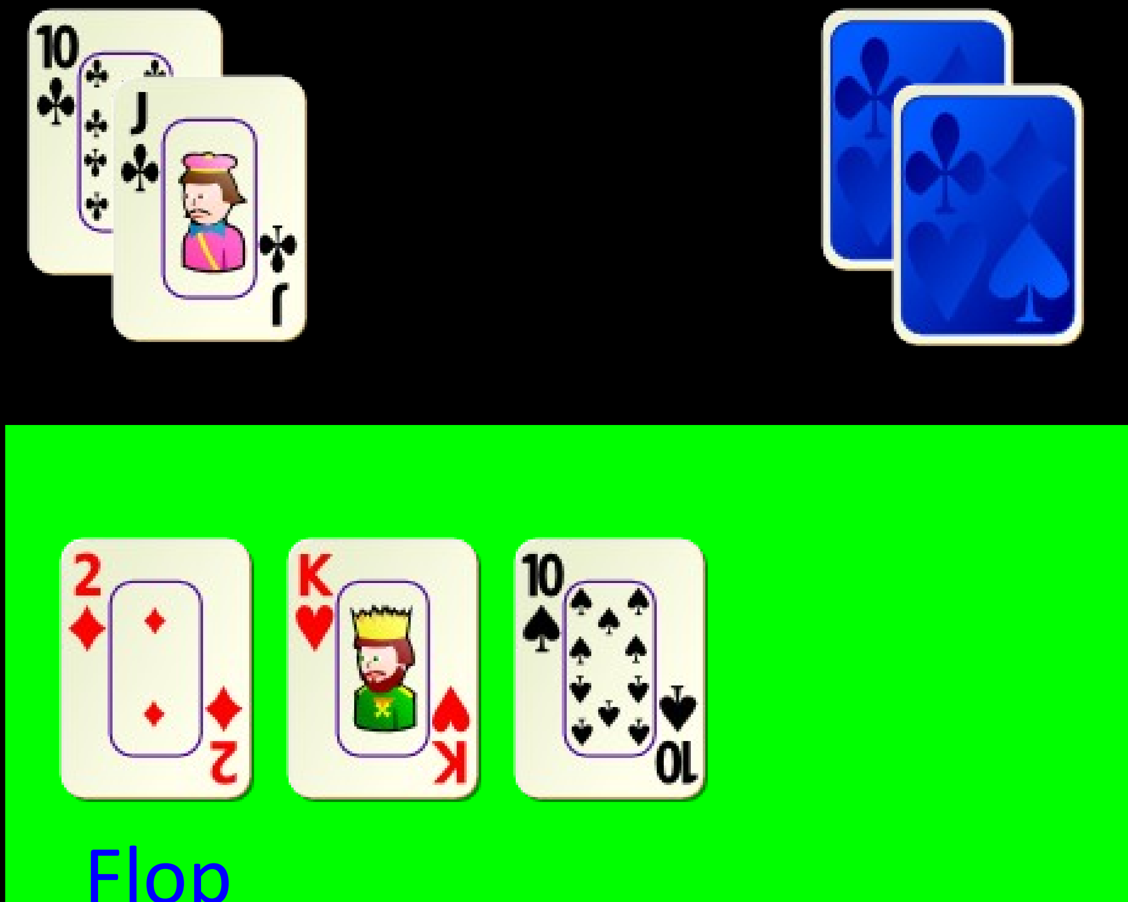
1081 Possible
Opponent Holdings





Flop

Immediate Hand Rank



Out of 1081

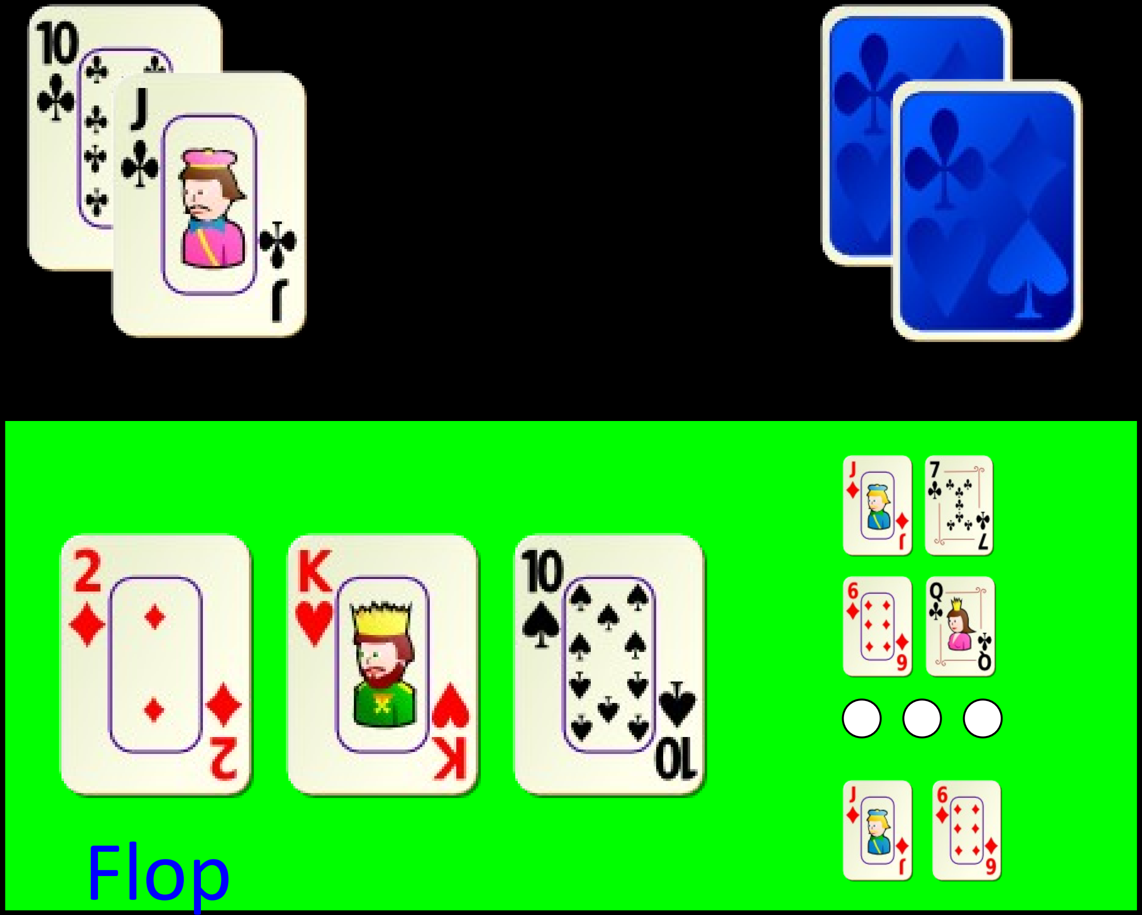
Win: 899 times
Tie: 6 times
Lose: 176 times

IHR = (win + tie/2) / total

IHR = 899 + 3 / 1081

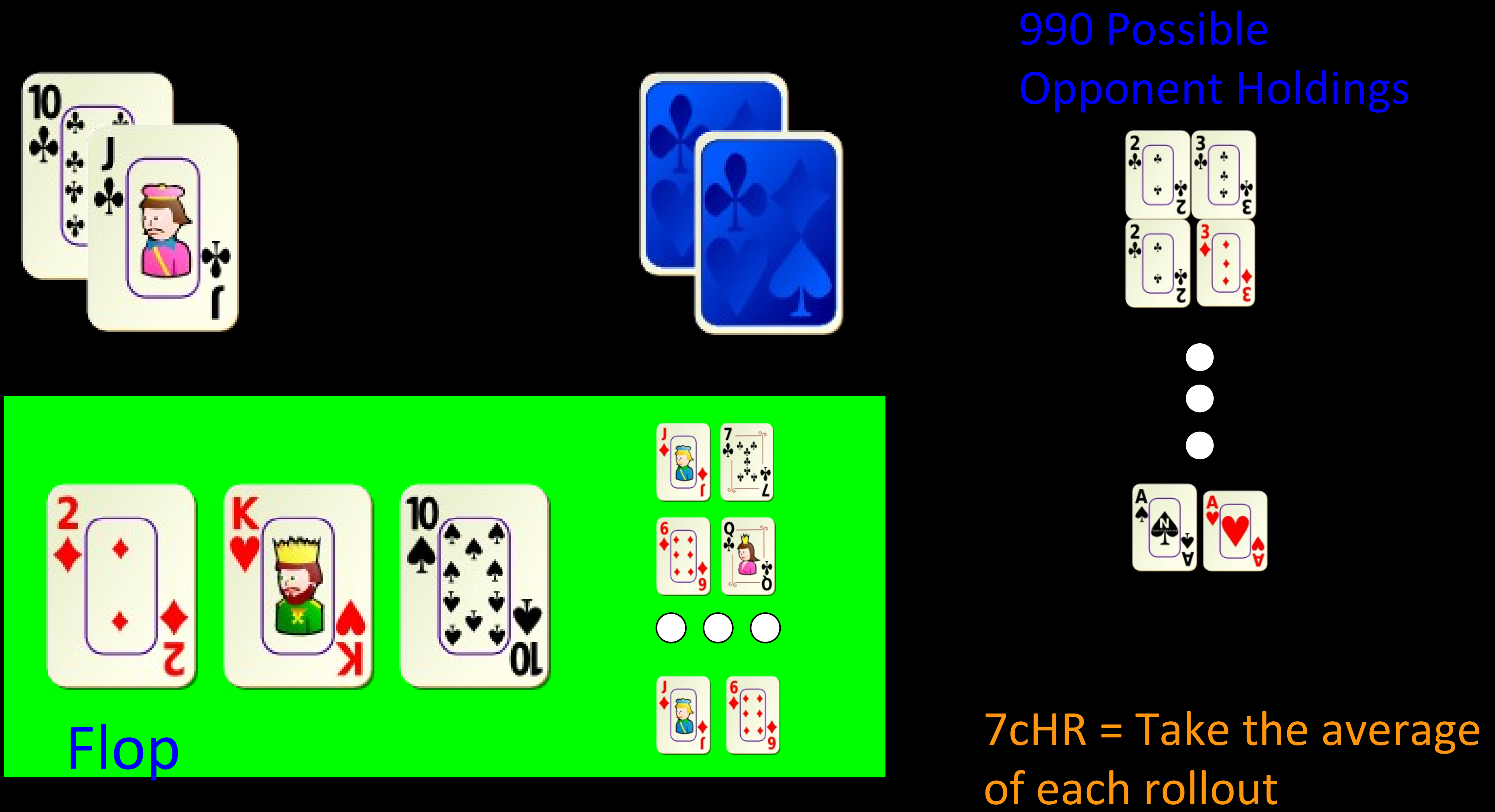
= 0.834413

7-card Hand Rank



7cHR = Take the average of each rollout

7-card Hand Rank



Knowledge-Based Systems

- Pros

  Easy to implement

- Cons

  Require many rules

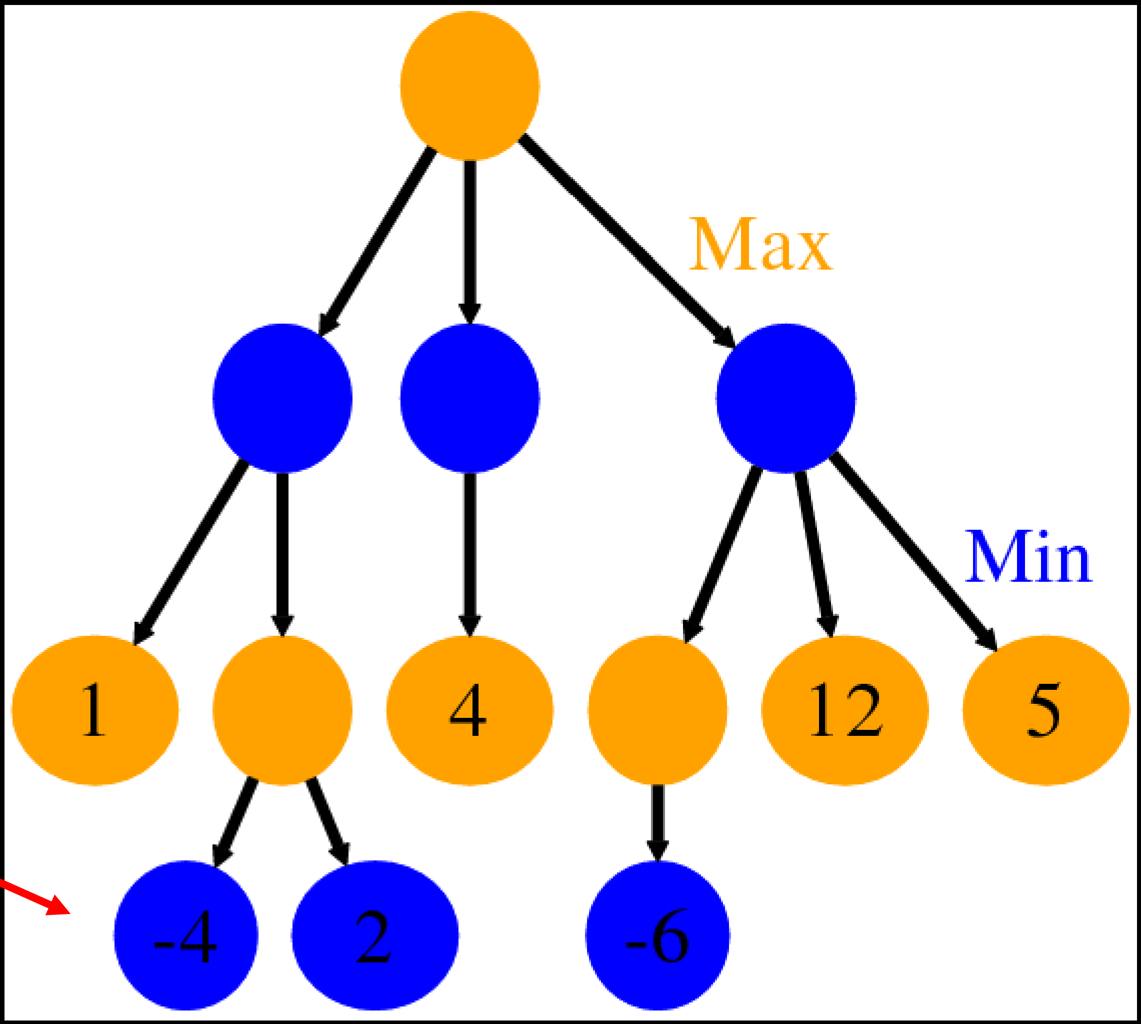
  Difficult to maintain

Imperfect Information Game Tree Search

- Similar to perfect information minimax search
- Requires opponent model

Imperfect Information Game Tree Search

Minimax Review:

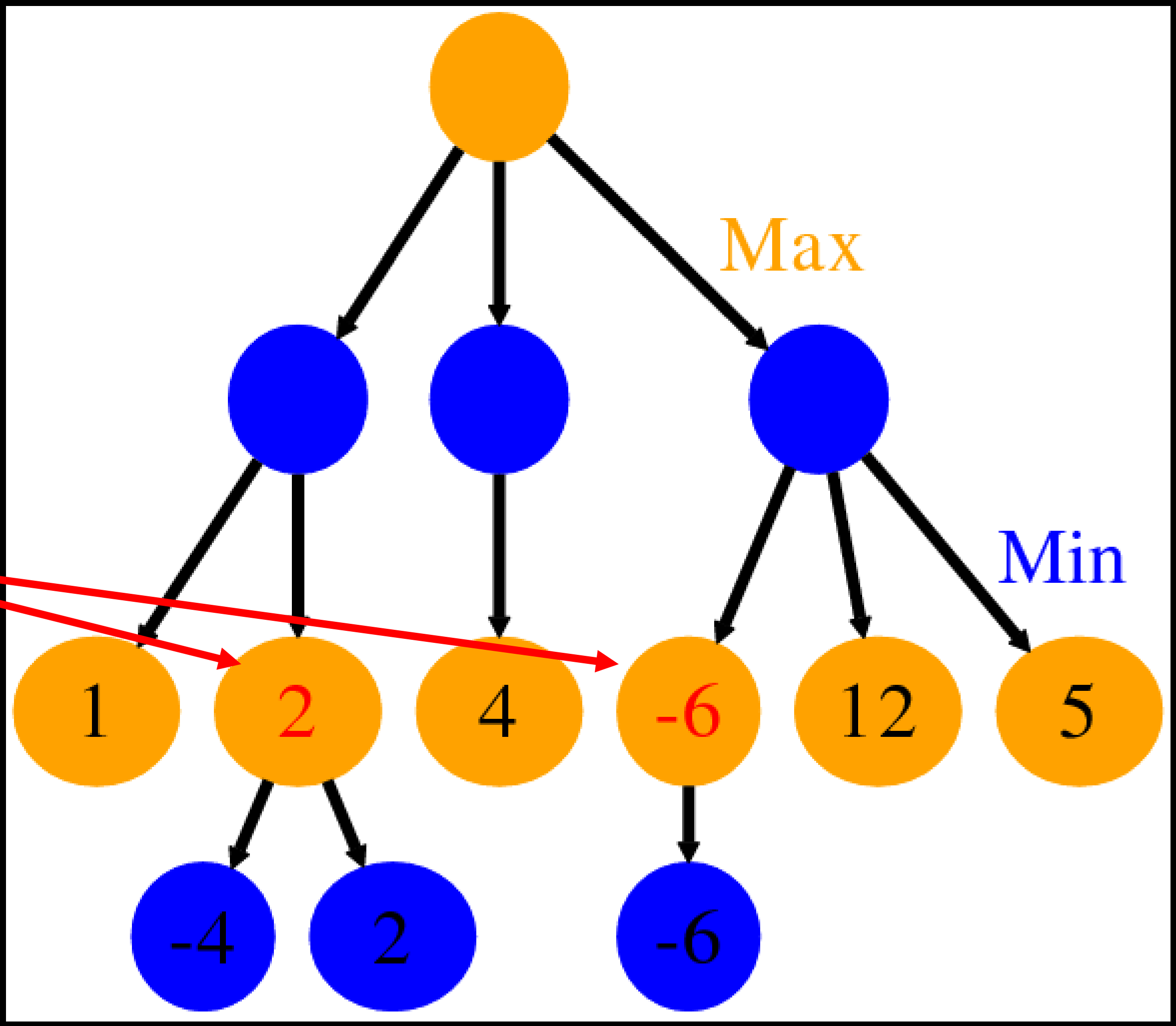


From terminal game states or evaluation function

Imperfect Information Game Tree Search

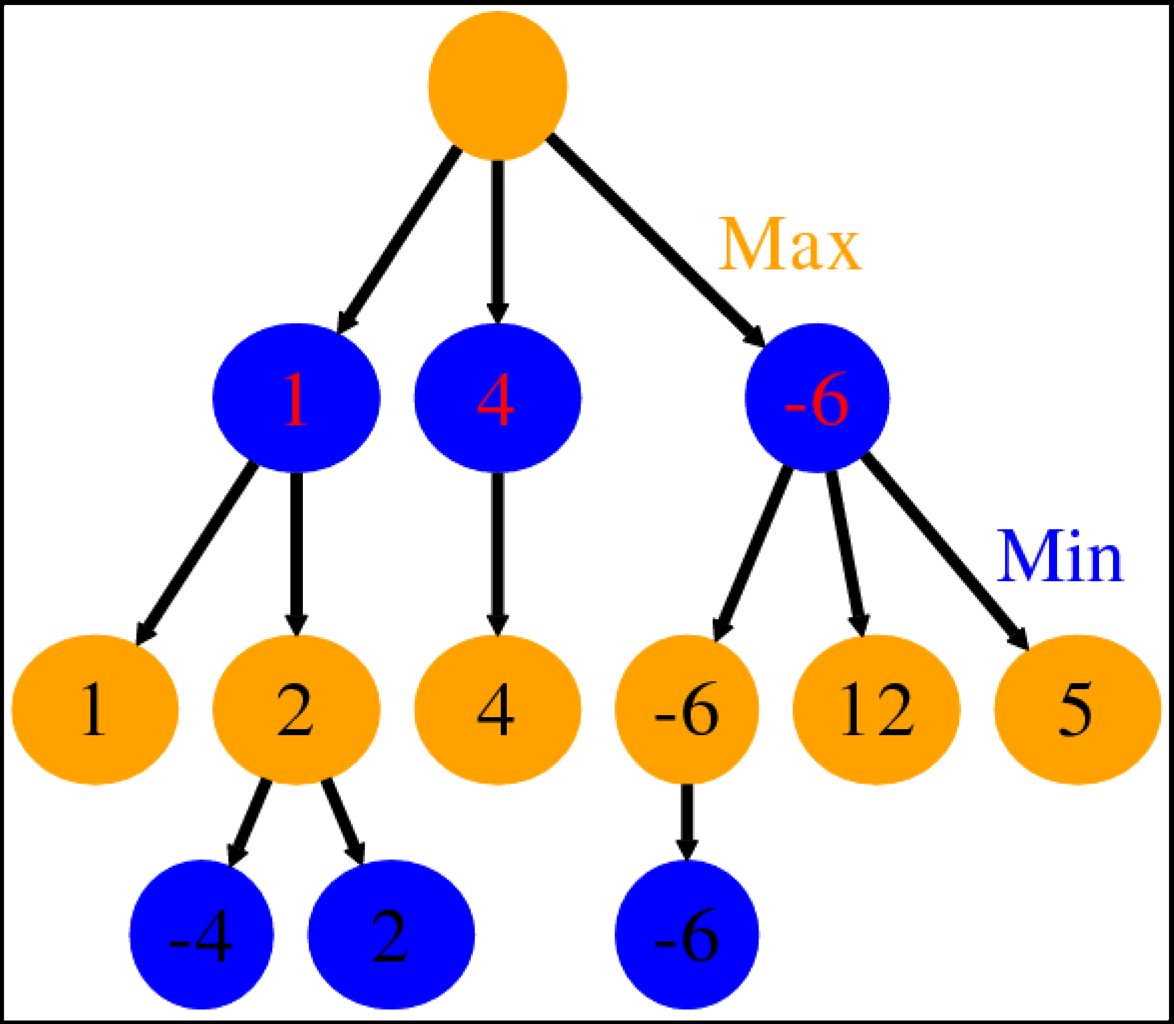
Minimax Review:

Backpropagate values



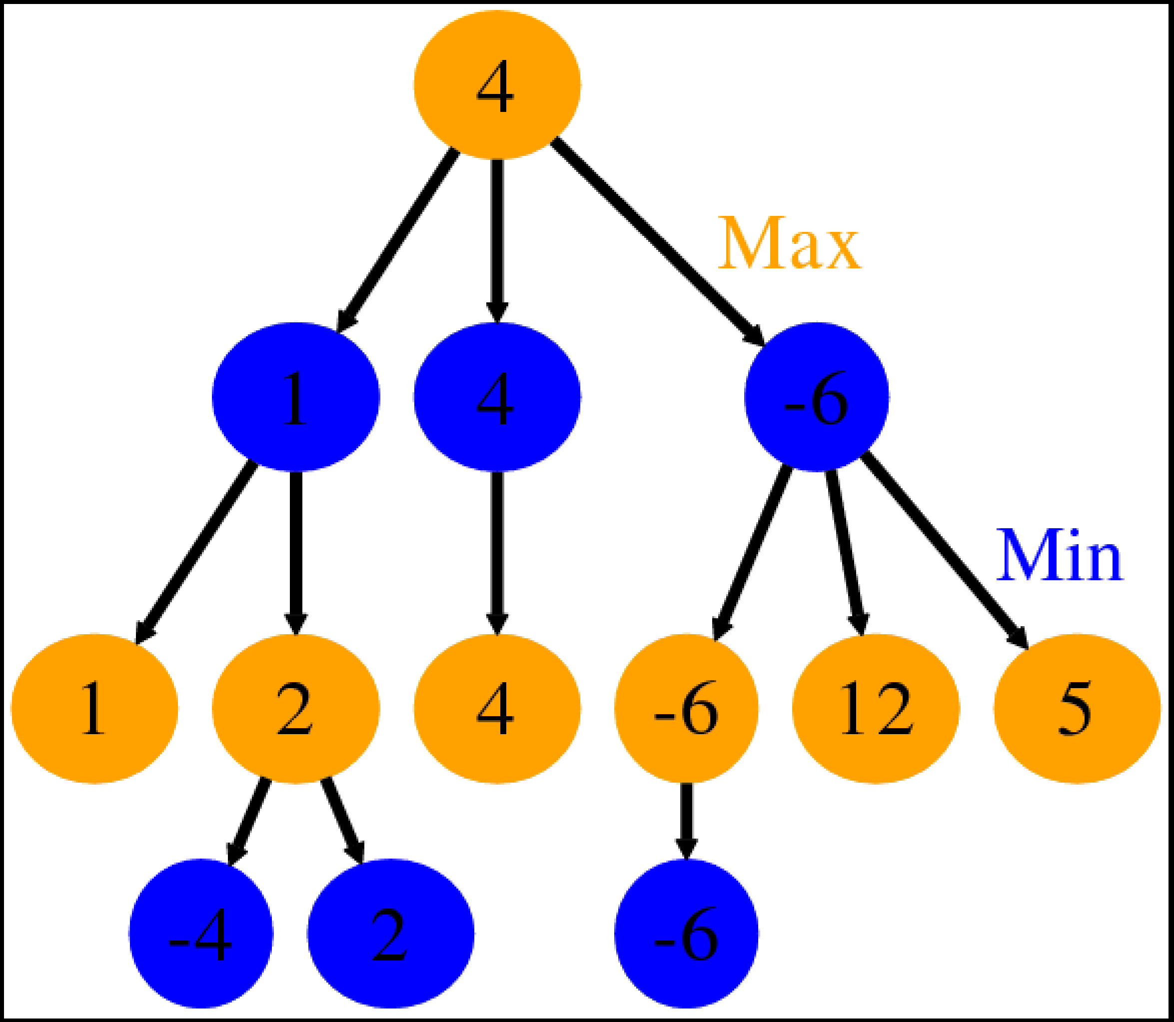
Imperfect Information Game Tree Search

Minimax Review:

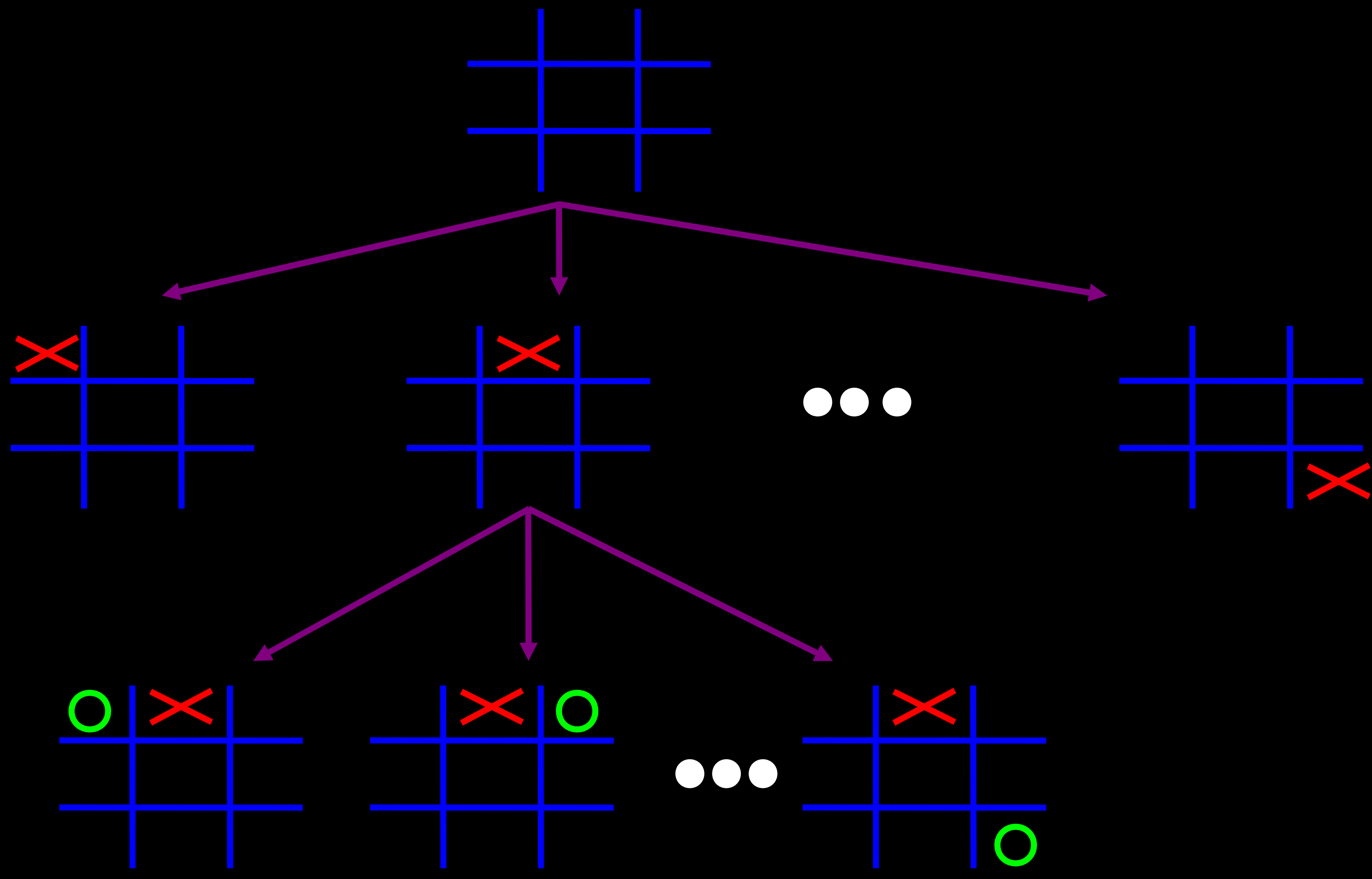


Imperfect Information Game Tree Search

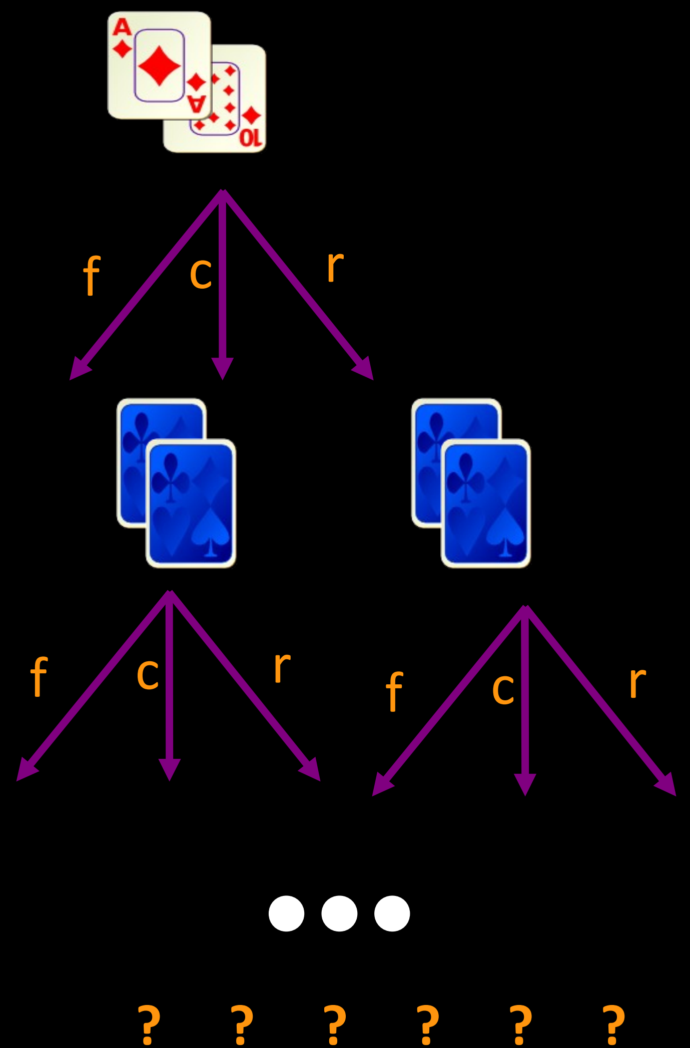
Minimax Review:



Imperfect Information Game Tree Search



Imperfect Information Game Tree Search



Imperfect Information!

Imperfect Information Game Tree Search

- Opponent's cards determine their strategy
- Unknown information

  Which betting decisions the opponent will make?

  Probability of winning at showdown?
- Opponent Model

  “fill in” for the missing information

  Can now assign values to leaf nodes and backpropagate - *miximax*

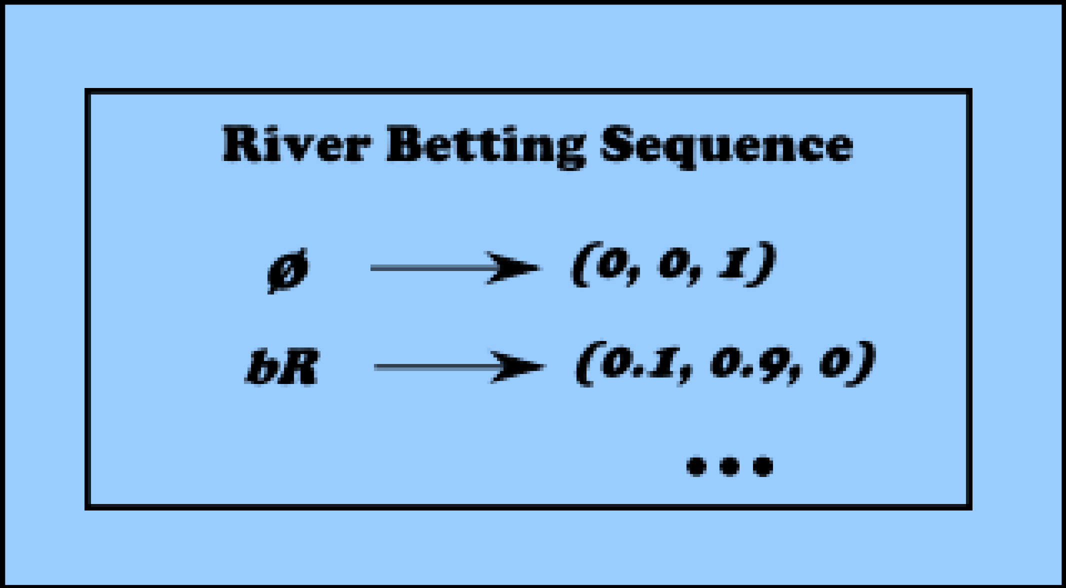
Imperfect Information Game Tree Search

- Example
  - Two players
  - On the last betting round
  - Each player has already contributed 5 bets each
  - Further bets are in increments of 2
  - Hand strength values are grouped into 1 of 5 buckets
    - 1 being the lowest bucket, 5 the highest
- Player A = Us
- Player B = Opponent (acts first)

Imperfect Information Game Tree Search

- Opponent Model

Prediction of B's actions on the river

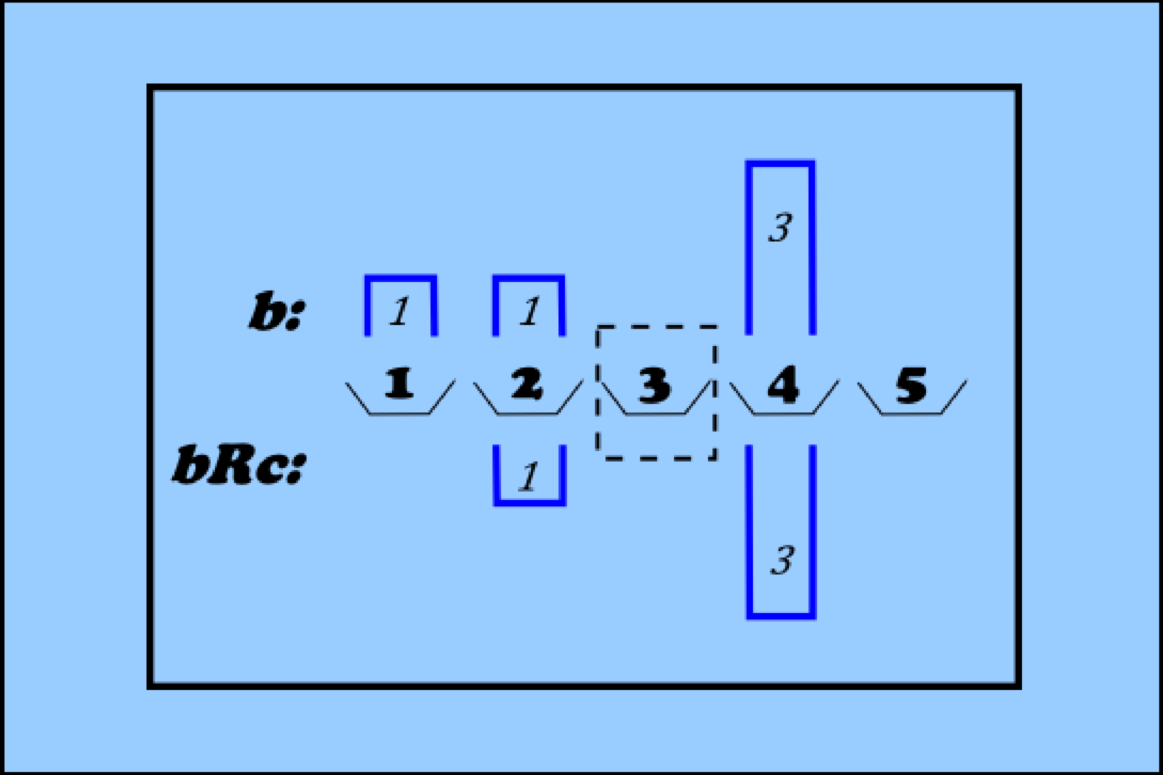


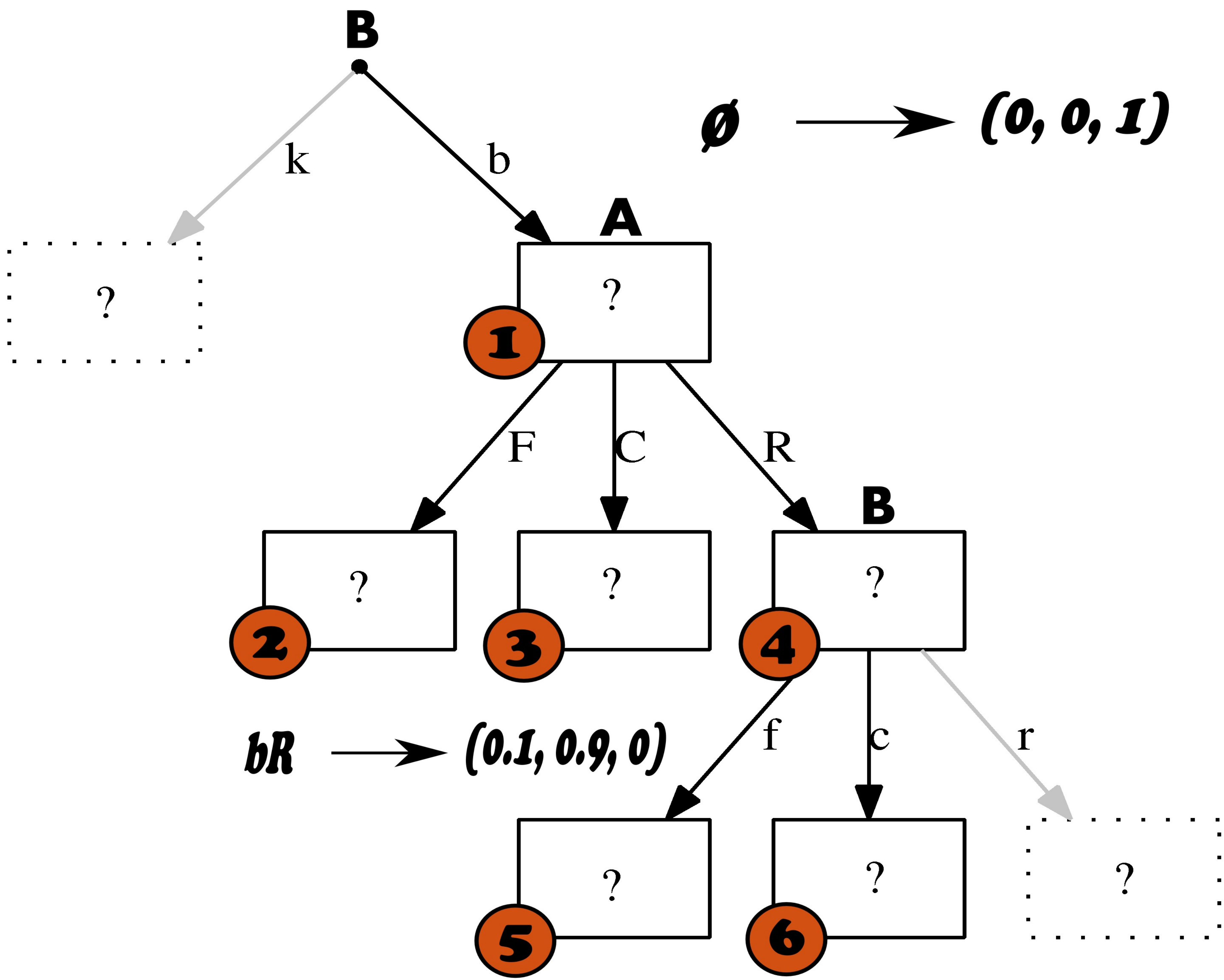
Imperfect Information Game Tree Search

- Opponent Model

  The buckets Player B has held in the past

  We currently have a hand in bucket 3





B
k
b
?
A
?
1
F
C
R
B
?
?
?
2
3
4
f
c
r
?
?
?
5
6
ø → (0, 0, 1)
bR → (0.1, 0.9, 0)

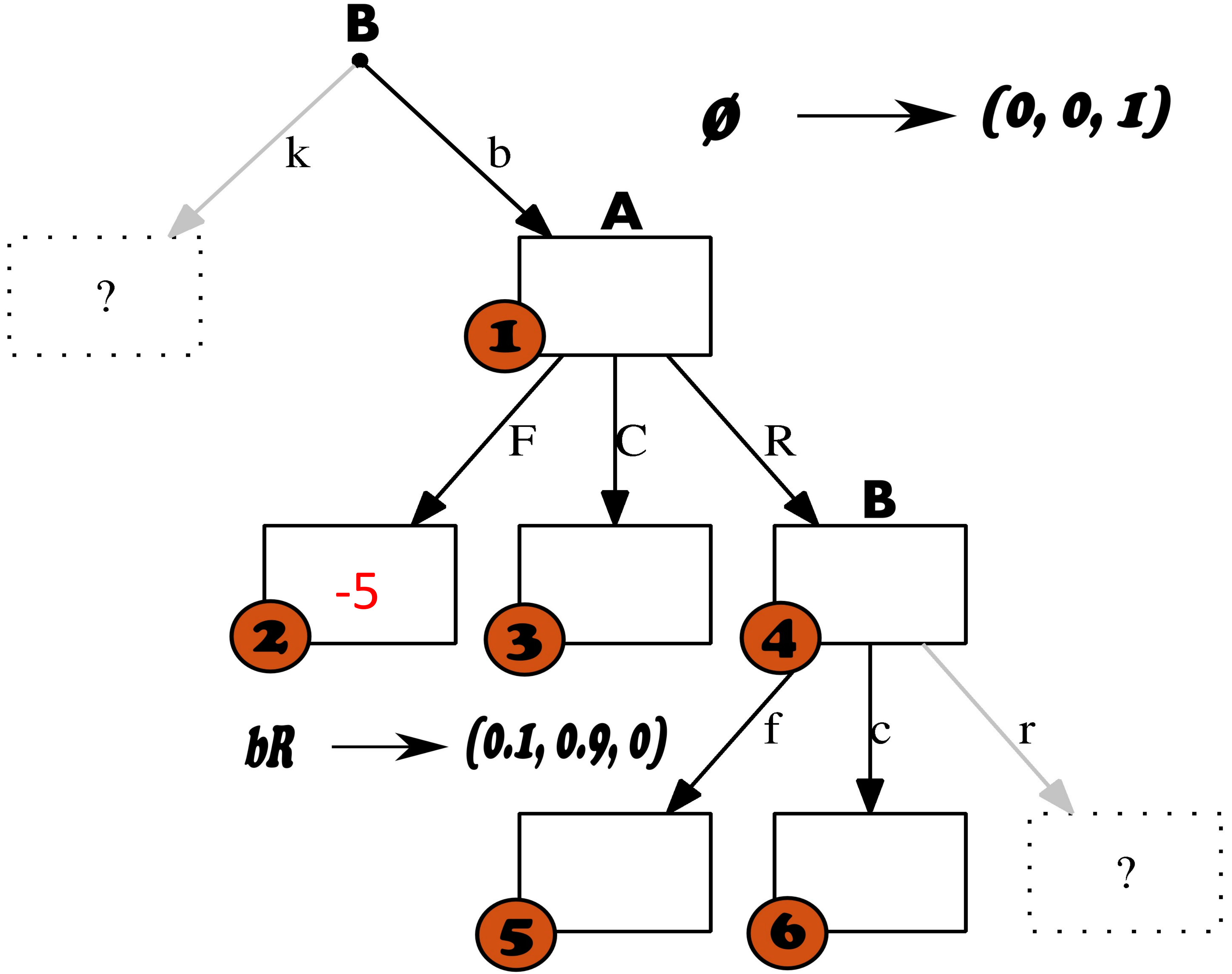
Imperfect Information Game Tree Search

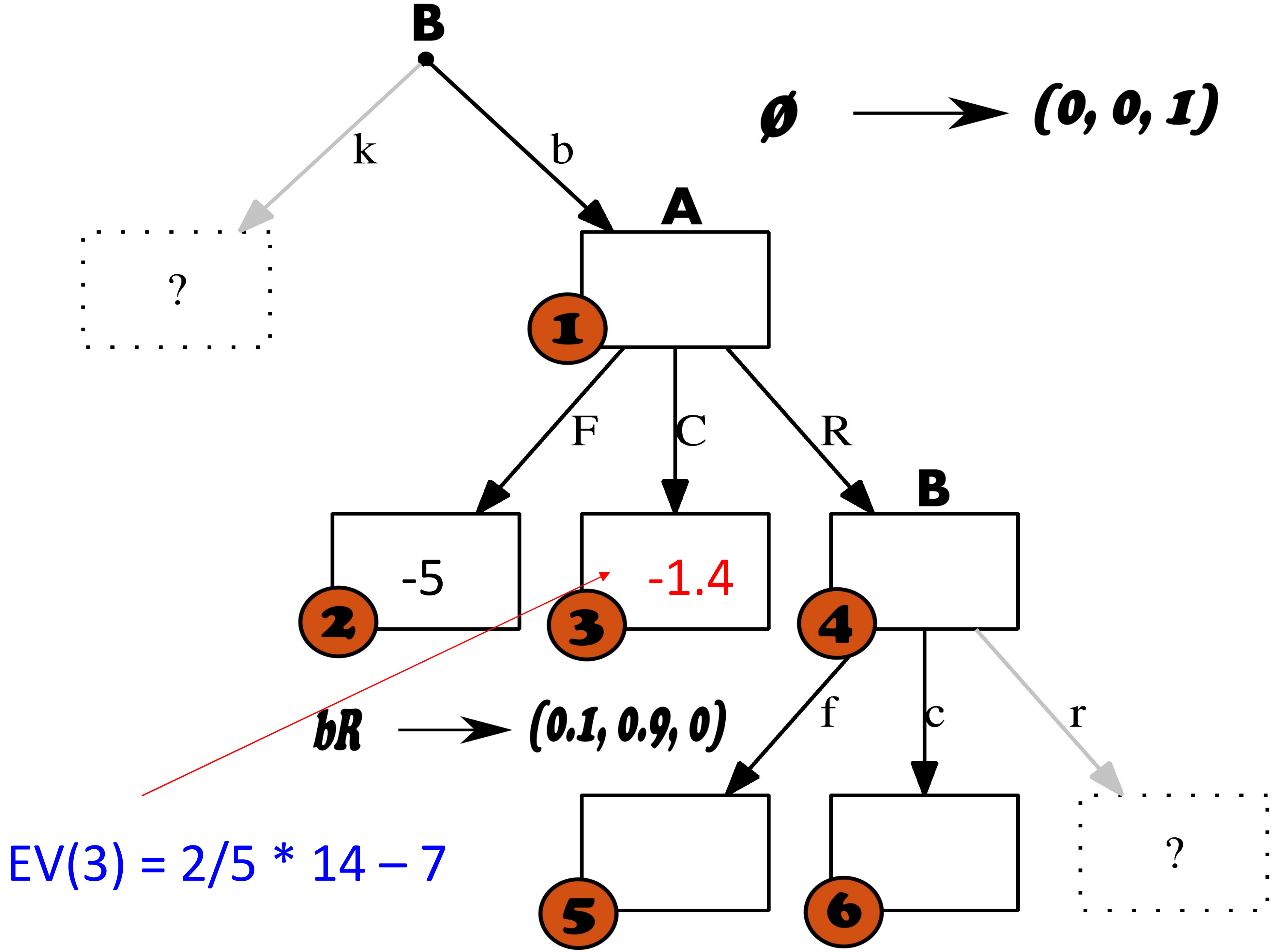
- Perform depth first search on the game tree
- Leaf nodes

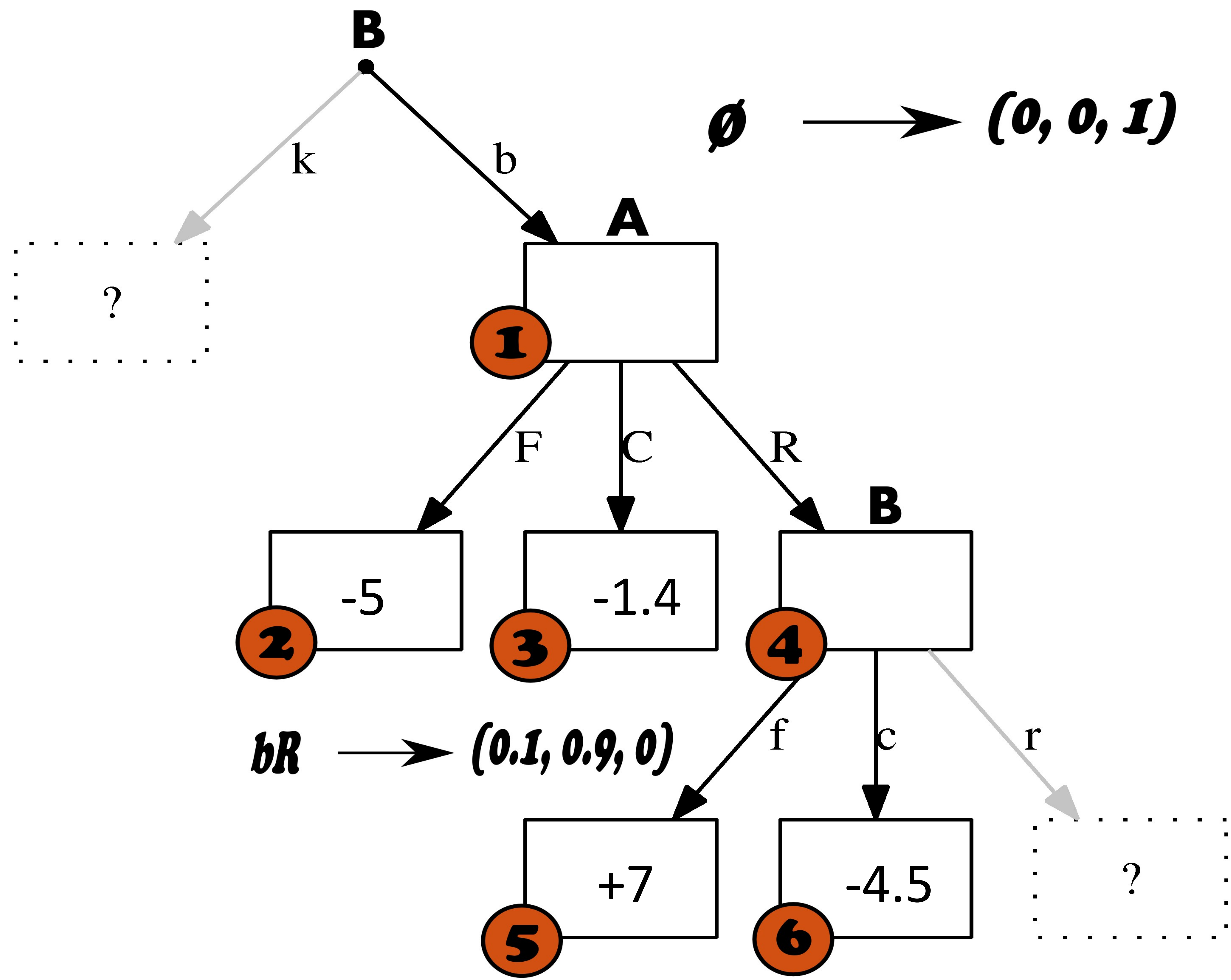
  *EV(x) = Pr(Win) * TotalPot – PlayerInvestment*
- Opponent choice nodes

  Mix from opponent model
- Player choice nodes

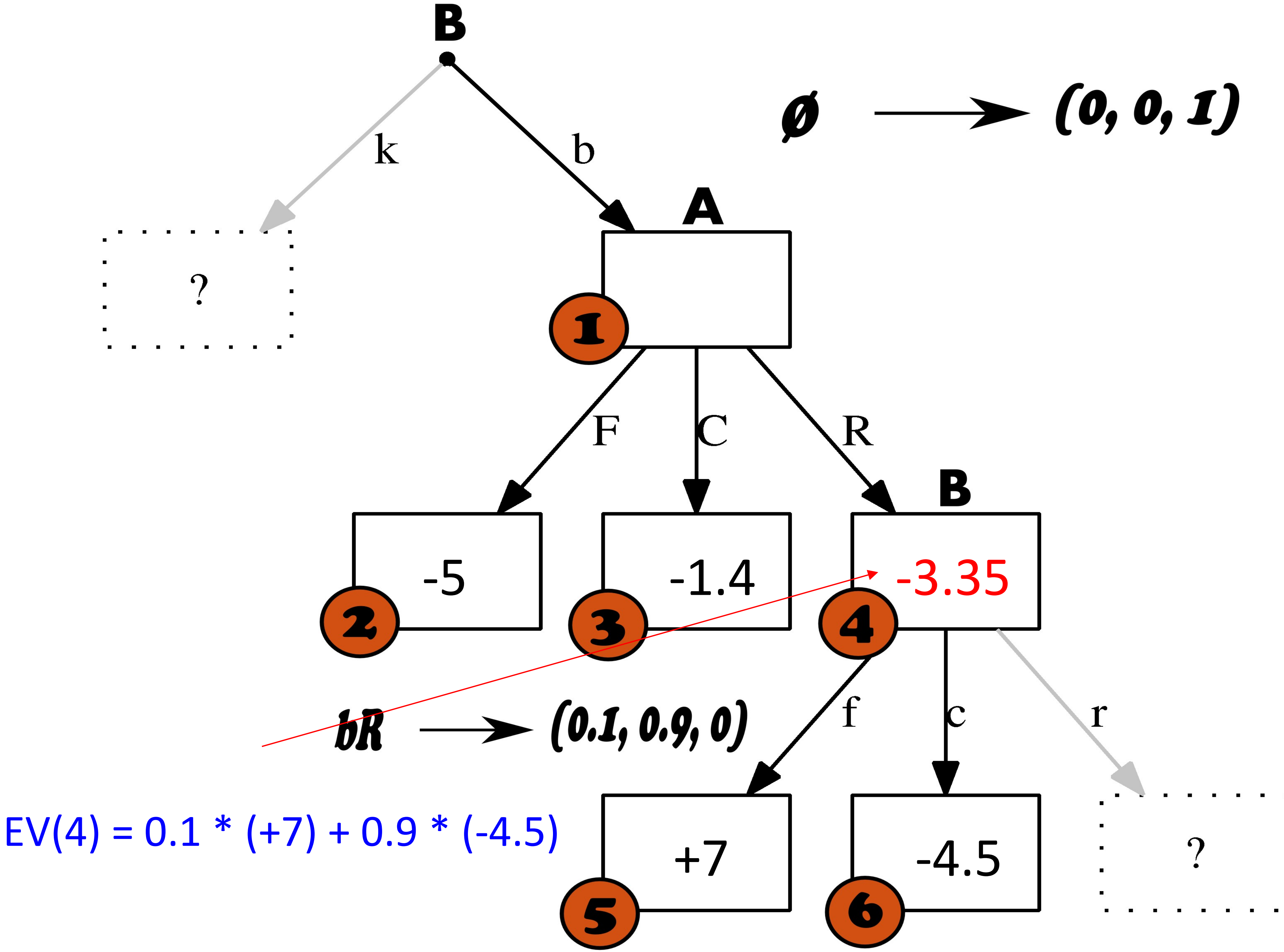
  Choose max child node

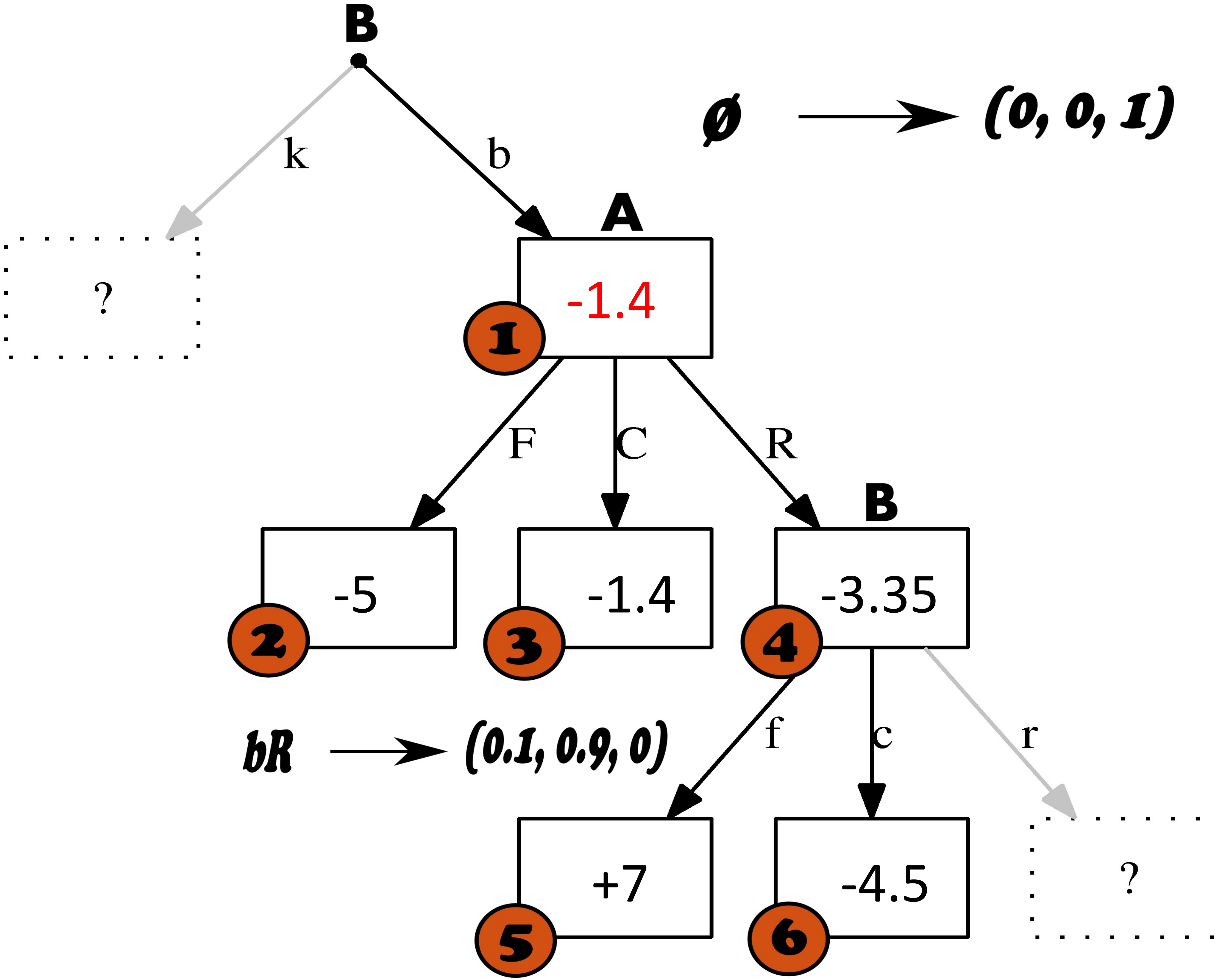

B
k
b
Ø
(0, 0, 1)
?
A
1
F
C
R
B
2
-5
3
4
bR
(0.1, 0.9, 0)
f
c
r
5
6
?


B
k
b
?
ø → (0, 0, 1)
A
1
F
C
R
-5
2
-1.4
3
B
4
bR → (0.1, 0.9, 0)
f
c
r
EV(3) = 2/5 * 14 – 7
5
6
?



B
k
b
?
A
1
F
C
R
-5
-1.4
B
2
3
4
f
c
r
+7
-4.5
?
5
6
ø → (0, 0, 1)
bR → (0.1, 0.9, 0)


B
k
b
?
A
Ø → (0, 0, 1)
1
F
C
R
B
-5
-1.4
-3.35
2
3
4
bR → (0.1, 0.9, 0)
f
c
r
+7
-4.5
?
5
6
EV(4) = 0.1 * (+7) + 0.9 * (-4.5)


B
k
b
?
A
-1.4
1
∅ → (0, 0, 1)
F
C
R
-5
-1.4
-3.35
B
2
3
4
bR → (0.1, 0.9, 0)
f
c
r
+7
-4.5
?
5
6

Imperfect Information Game Tree Search

- Pros

  Adaptive

  Exploitive

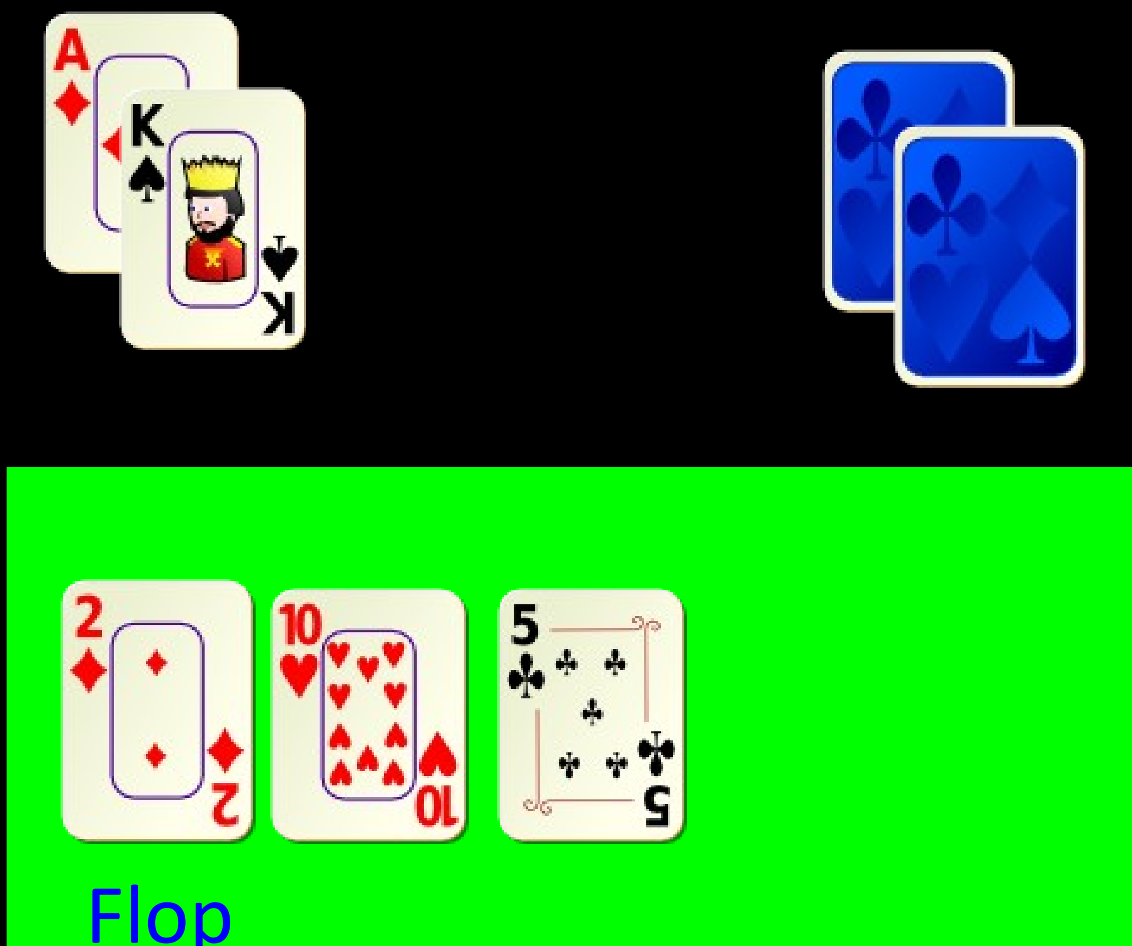
- Cons

  Generating game tree on the fly can take a long time

Monte-Carlo Simulation

- Alternative game tree search procedure
- Uses Monte-Carlo sampling to predict EVs at choice nodes

Monte-Carlo Simulation



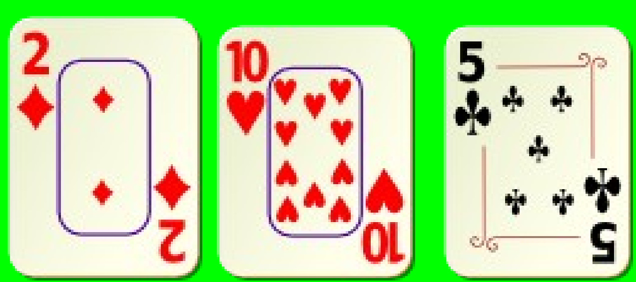
Which action has the greatest EV?

Fold: $0
Check/Call: ??
Bet/Raise: ??

$10/$20 Limit Hold'em

Monte-Carlo Simulation

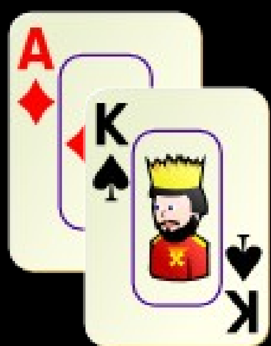
First guess opponent's hand

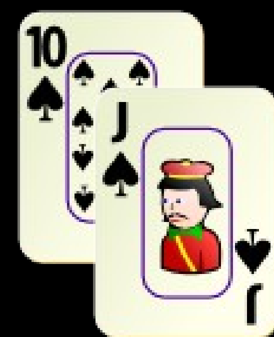


Flop

$10/$20 Limit Hold'em

Monte-Carlo Simulation





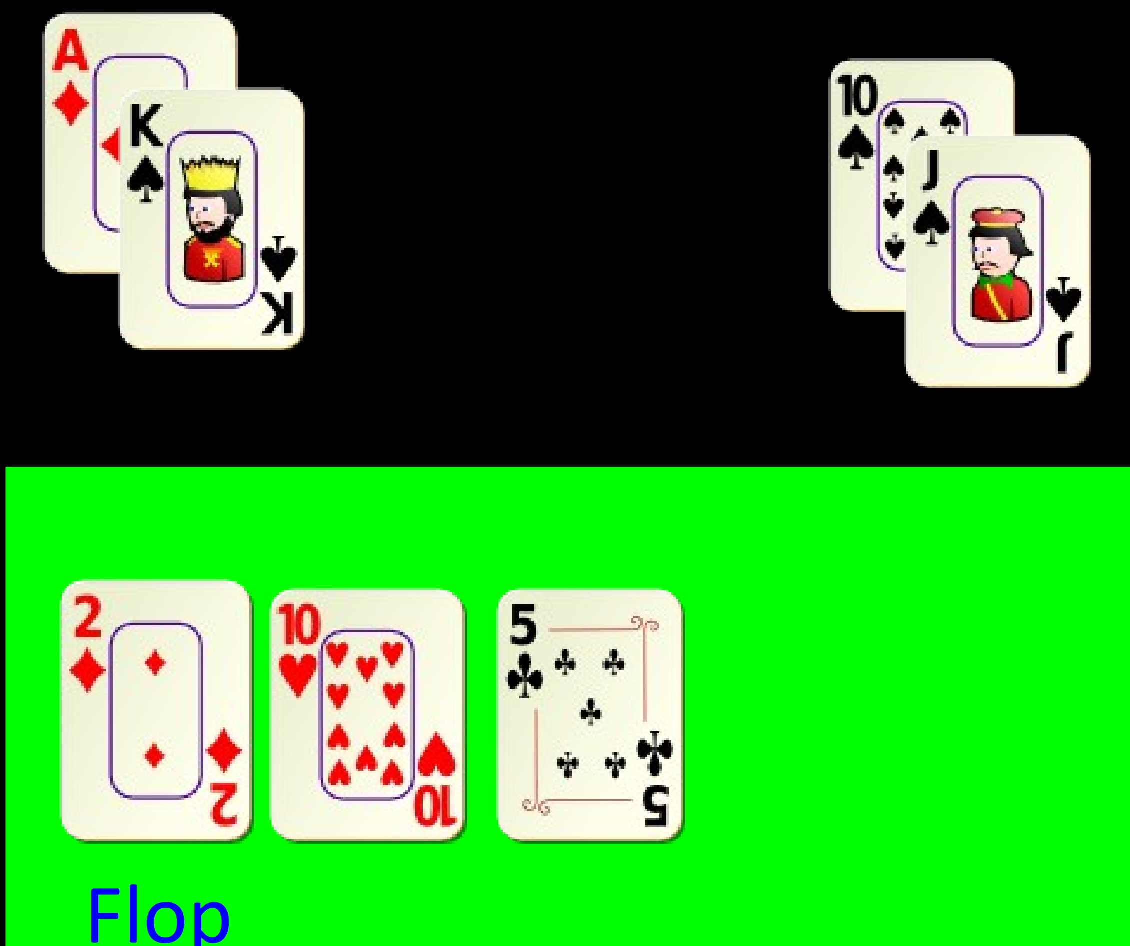
First guess opponent's hand

Flop

$10/$20 Limit Hold'em

Monte-Carlo Simulation

Then simulate actions until leaf node

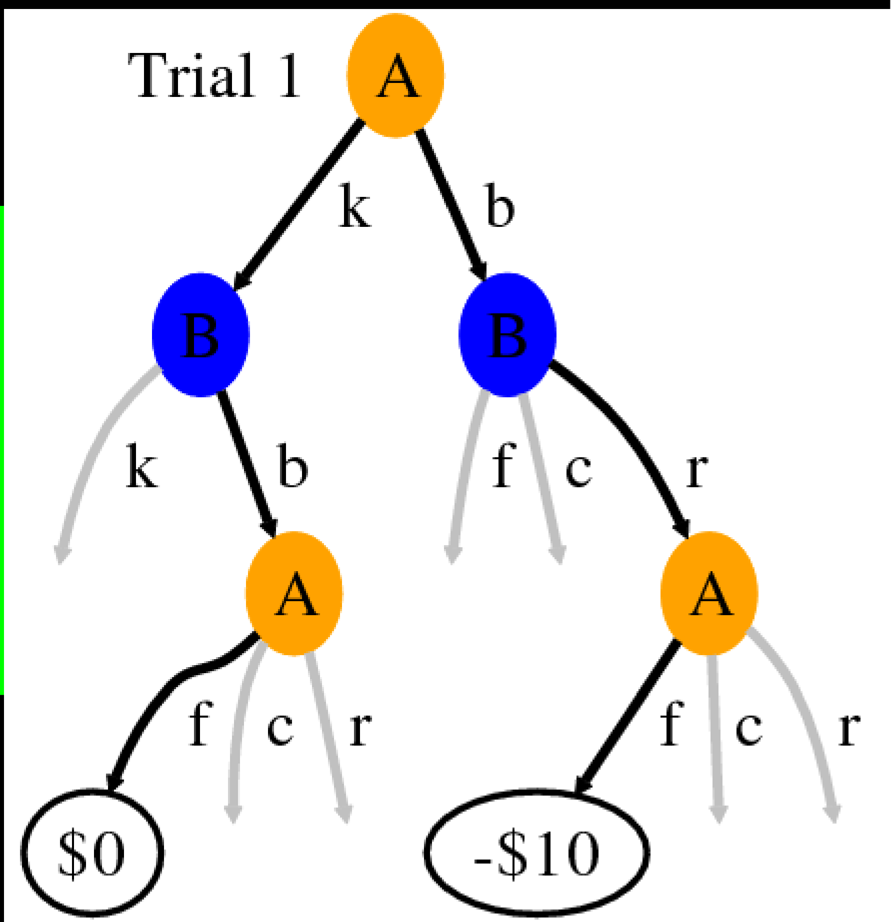


$10/$20 Limit Hold'em

Monte-Carlo Simulation

Then simulate actions until leaf node

Flop



$10/$20 Limit Hold'em

Monte-Carlo Simulation

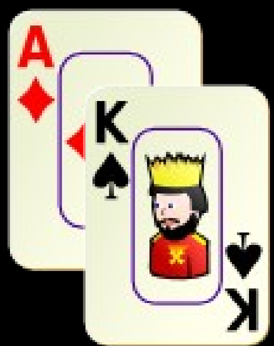
- EVs after trail 1

  Fold: \$0

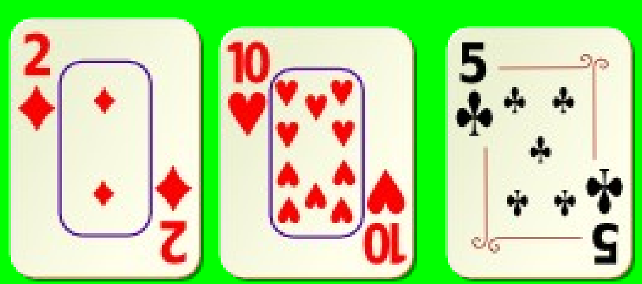
  Check/Call: \$0

  Bet/Raise: -\$10

Monte-Carlo Simulation



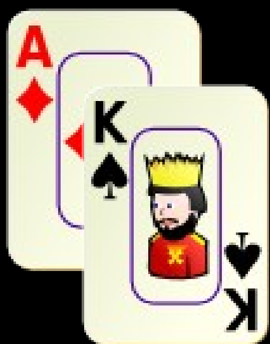
Repeat

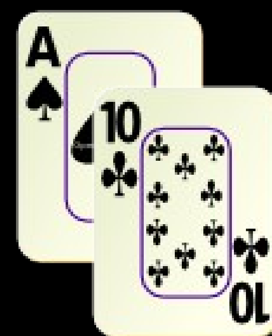


Flop

$10/$20 Limit Hold'em

Monte-Carlo Simulation



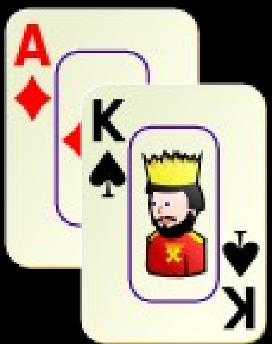


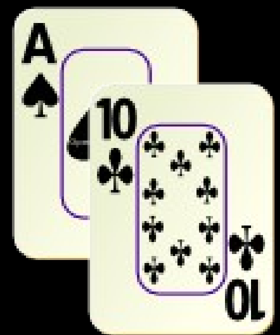
Repeat

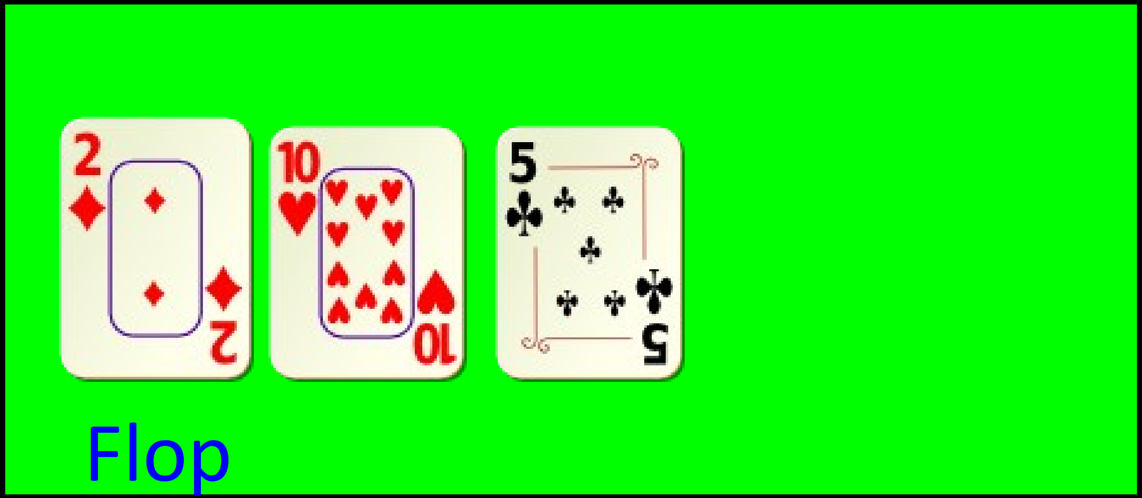
Flop

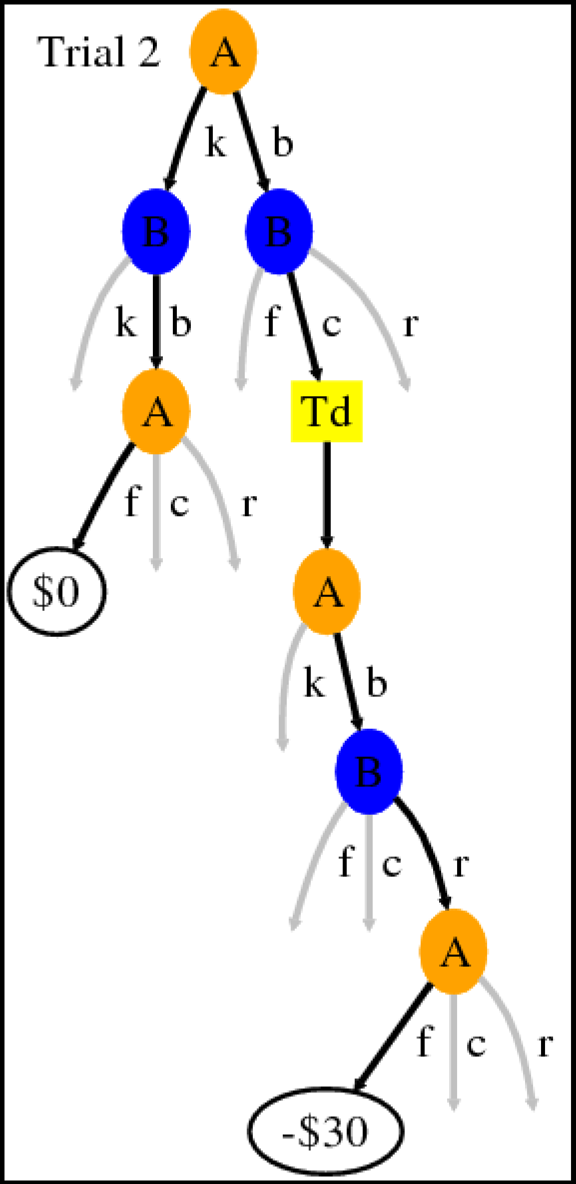
$10/$20 Limit Hold'em

Monte-Carlo Simulation



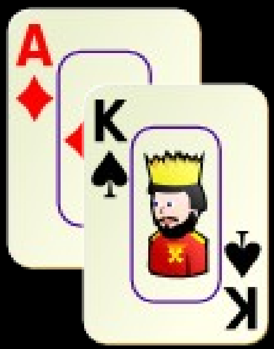


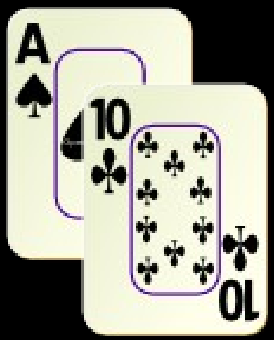


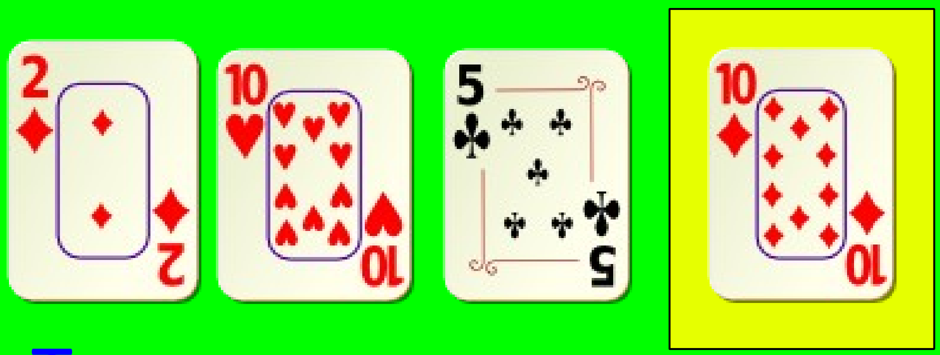


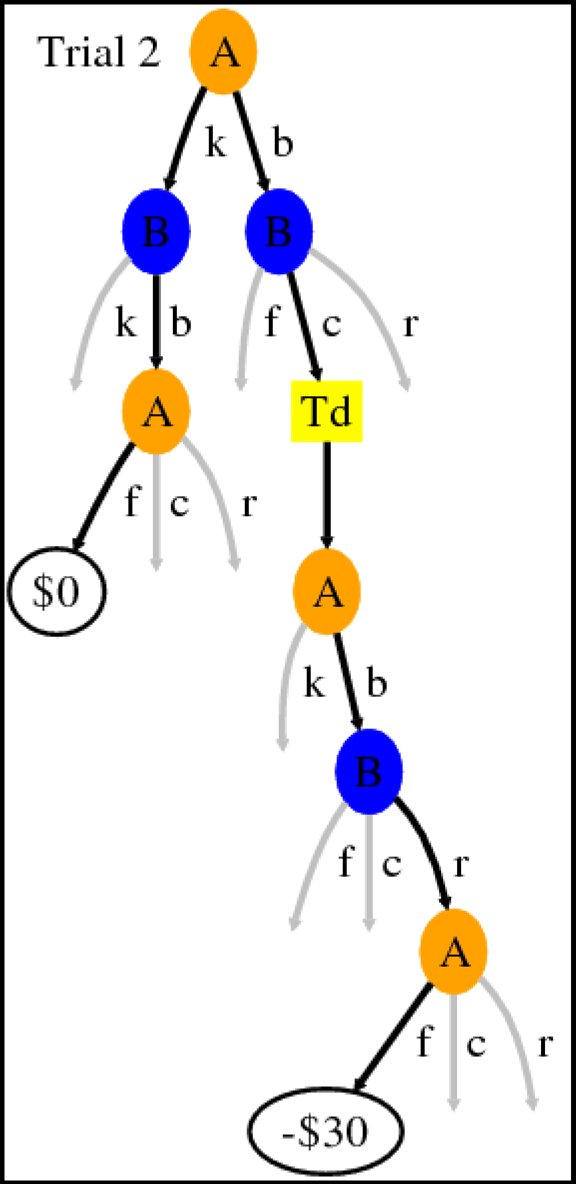
$10/$20 Limit Hold'em

Monte-Carlo Simulation









Turn

$10/$20 Limit Hold'em

Monte-Carlo Simulation

- EVs after trail 2

Fold: \$0

Check/Call: \$0

Bet/Raise: -\$20

Monte-Carlo Simulation

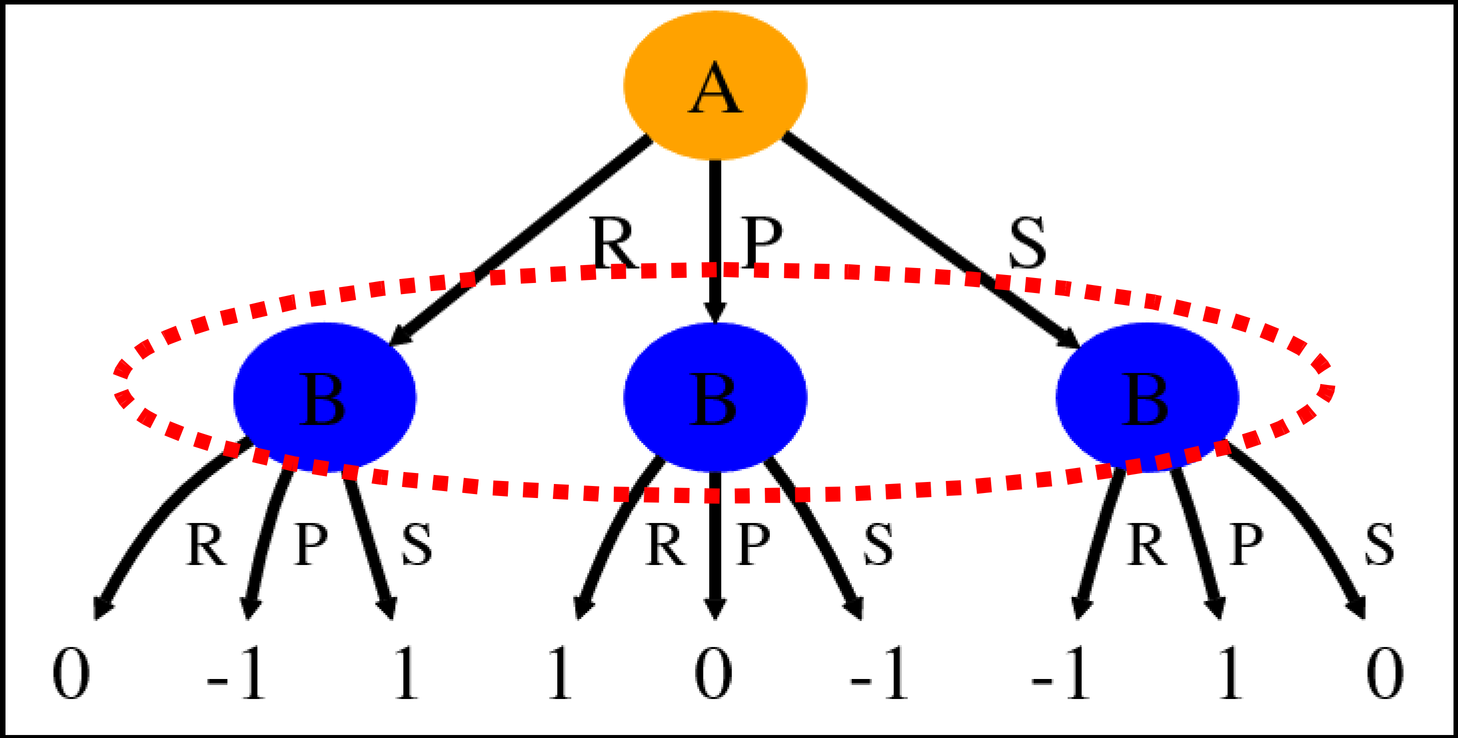
- EVs after 1000s of trails
- Converge to stable values
- Choose action with greatest EV.
- The better the opponent model the more accurate the EVs that are calculated.

Monte-Carlo Simulation

- Pros
  - Adaptive
  - Emergent sophisticated plays e.g. check-raise
- Cons
  - Sensitive to bias

Game Theoretic Approaches

- Extensive form



Game Theoretic Approaches

- Normal form

$$\begin{array}{cc} & \begin{array}{ccc} R & P & S \end{array} \\ \begin{array}{c} R \\ P \\ S \end{array} & \begin{pmatrix} 0 & -1 & 1 \\ 1 & 0 & -1 \\ -1 & 1 & 0 \end{pmatrix} \end{array}$$

Game Theoretic Approaches

- Payoff matrix acts as constraint within a linear program

e.g

$$\text{minimize } y_1 + \cdots + y_m$$
$$\text{subject to } \textstyle\sum_{i=1}^{m} m_{ij} y_i \geq 1, \text{ for } 1 \leq j \leq n$$

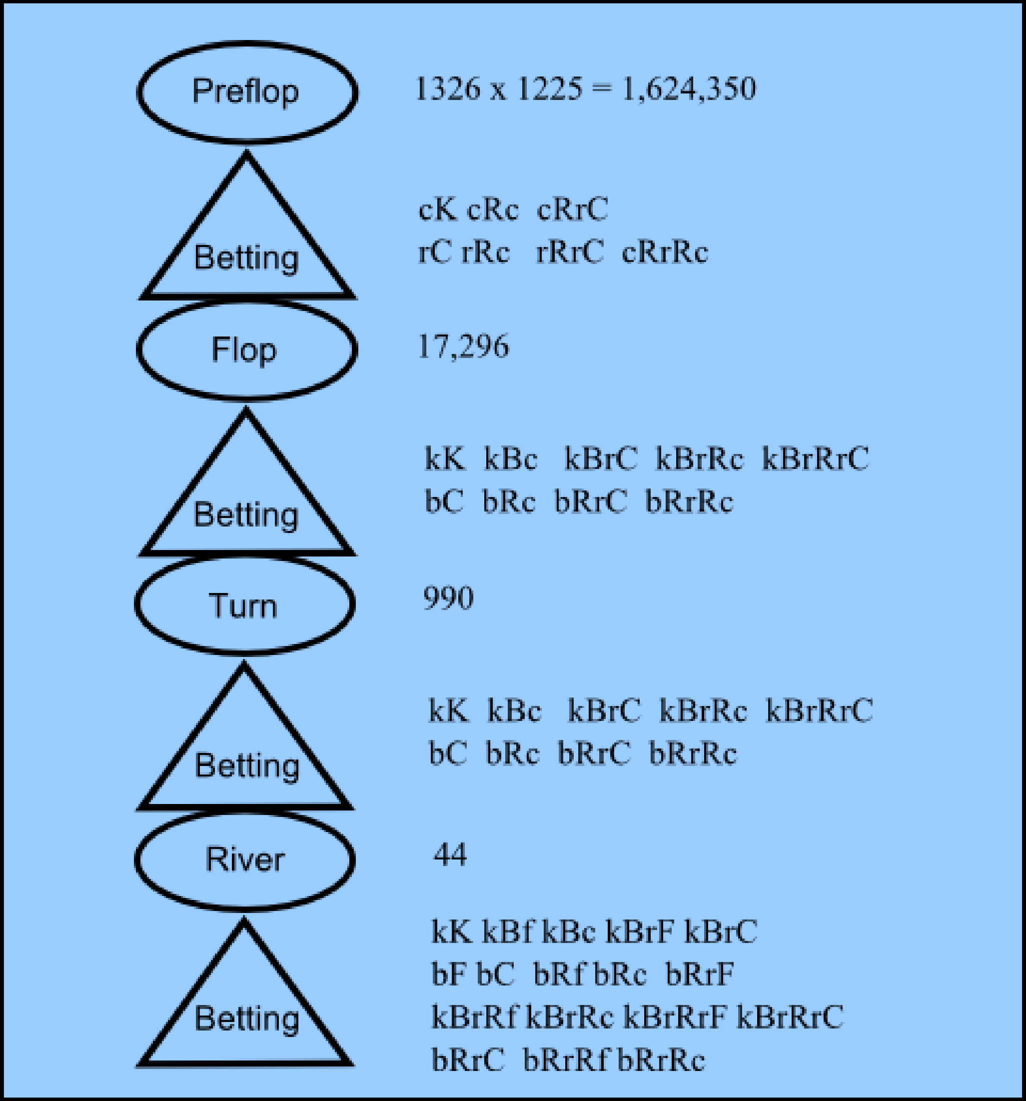
- Solve LP e.g. simplex method
- Solving LP gives a mixed strategy

e-Nash Equilibrium

- A Nash equilibrium can easily be computed for Rock-Paper-Scissors
- However, the poker game tree is much to large to find exact Nash equilibria
  - Abstractions required
- Can only approximate Nash-equilibria
  - *e*-Nash Equilibria
  - *e* specifies a lower bound on how exploitable the equilibrium strategy is

Game Theoretic Approaches

- 2-Player Hold'em Game Tree



Game Theoretic Approaches

- Normal form increases exponentially with size of extensive form game tree
- Sequence form representation
- Requires abstractions
  - Bucketing
    - Grouping strategically similar hands together
  - Restricting the allowed number of raises
  - etc...

e-Nash Equilibrium

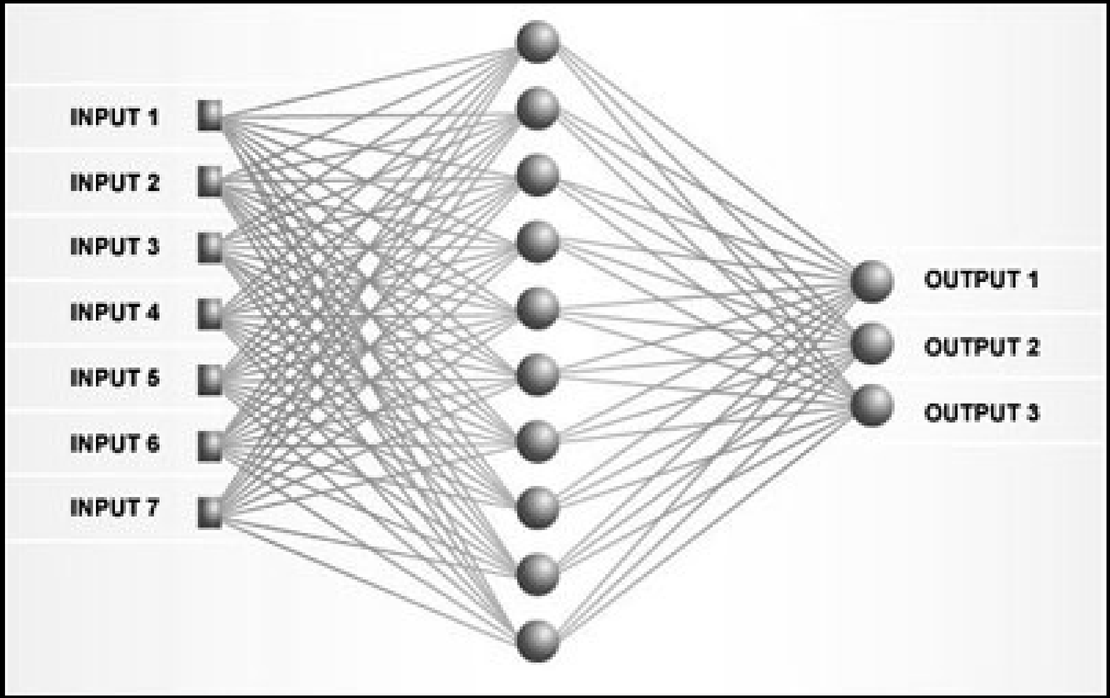
- Linear Programming
  - Constructs matrices that act as constraints within an optimization problem
- Iterative approaches
  - Basic idea: Two players begin with arbitrary strategies, play many repetitions of a game and modify their strategies in a way that improves their strategy against their opponent.
  - As the number of iterations increases the strategies approach a Nash equilibrium
  - e.g. *Fictitious Play, Counterfactual Regret Minimization*

Game Theoretic Approaches

- Pros
  - Produce solid players that restrict their own exploitability
- Cons
  - Large time and space requirements
  - Not exploitive of weak opponents

Artificial Neural Networks

- Specify appropriate inputs
- Design multi-layer network
- Outputs for fold / call / raise
- Train network



Artificial Neural Networks

- Pros

  Simple approach

- Cons

  Relies on good training data

Evolutionary Algorithms

- Genetic algorithms
  - Selection, crossover and mutation procedures
  - Population of neural networks
- Evolve an ANN based on maximising a fitness function

Evolutionary Algorithms

- Pros

  Evolves strong players via self-play

- Cons

  Time consuming

  Need to handle multi-objective optimization

Our Approach

Goal

- Investigate whether hand histories from strong poker players can be reused within a Case-Based Reasoning framework to achieve a similar performance?

A Memory-Based Approach

- Casper (CASe-based Poker playER)

  Past poker agent for 10-player Texas Hold'em

- Sartre (Similarity Assessment Reasoning for Texas hold'em via Recall of Experience)

  Our latest agent

  Specialised for heads-up limit hold'em

Sartre

- Overview

  Cases are attribute-value pairs

  Separate case-bases are used for each different round (preflop, flop, turn, river)

  When a decision is required a case is created to describe the current state of the game and the appropriate case-base is searched to find similar cases

  The solution of the similar cases are reused for the current situation

Sartre

- Latest Case Representation

Attribute	Type	Example
1. Hand Type	Class	*Missed, Pair, Two-Pair, Set, Flush,Flush-Draw, Straight-Draw, …*
2. Betting Sequence	String	*rc-c, crrc-crrc-cc-r, …*
3. Board Texture	Class	*No-Salient, Flush-Possible, Straight-Possible, Flush-Highly-Possible, …*
Solution **Outcome**	Triple Triple	*(0, 0.5, 0.5), …* *(-inf, 4.3, 15.6), …*

Sartre

- Similarity Metrics

  Each feature requires local similarity metric

  0.0 = entirely dissimilar, 1.0 = exactly similar

- Hand Type & Board Texture

  Map to same category then similarity = 1.0, otherwise 0.0

- Betting Sequence

  Sequences with the same number of bets/raises considered more similar

Sartre

- Training Data

  Trained on data from the best agents in past Computer Poker Competitions

Experiments

- 2009 IJCAI Computer Poker Competition
    - Participated in limit hold'em competition
    - 13 competitors
    - 2 divisions
        - Bankroll
        - Equilibrium

Results

- 2009 IJCAI Computer Poker Competition

Limit bankroll division

Place	Agent	sb/h
1	MANZANA	0.186
2	Hyperborean-BR	0.116
3	GGValuta	0.110
4	Hyperborean-Eqm	0.116
5	Rockhopper	0.103
6	**Sartre**	**0.097**
7	Slumbot	0.096
8	GS5	0.082
9	AoBot	-0.002
10	dcurbHU	-0.07
11	LIDIA	-0.094
12	GS5Dynamic	-0.201

Results

- 2009 IJCAI Computer Poker Competition

Limit equilibrium division

Place	Agent
1	GGValuta
2	Hyperborean-Eqm
3	MANZANA
4	Rockhopper
5	Hyperborean-BR
6	Slumbot
7	**Sartre**
8	GS5
9	AoBot
10	GS5Dynamic
11	LIDIA
12	dcurbHU
13	Tommybot

Case-Based Reasoning

- Pros

  Simple approach
  Ability to learn over time

- Cons

  Relies on good training data

Recommended Websites

- **Poker ai.org**

  http://pokerai.org/pf3/index.php

  Community of people interested in poker AI

  Forums – good place to pose questions

  Includes large collection of papers related to poker AI

- **University of Alberta Computer Poker Research Group**

  http://poker.cs.ualberta.ca/

  Homepage of Alberta computer poker research group

  Includes theses and papers published by its members

- **Coding the Wheel (Hand Evaluator Roundup)**

  http://www.codingthewheel.com/archives/poker-hand-evaluator-roundup

  Comprehensive listing of publicly available poker hand evaluators

Theses & Papers

- Imperfect information game tree search

Darse Billings, et. al. (2004)

Game-tree search with adaptation in stochastic imperfect-information games.

Computers and Games, 4th International Conference, CG 2004, pp 21 – 34

(*includes clear example of miximax search procedure)

Terence Schauenberg (2006)

Opponent Modelling and Search in Poker

MSc , University of Alberta

Patrick McCurley (2009)

An Artificial Intelligence Agent for Texas Hold'em Poker

Undergraduate Dissertation, University of Newcastle Upon Tyne

(*good starting point for tree search based agents)

Theses & Papers

- Monte-Carlo Simulation

  Darse Billings, et. al. (2002)
  The challenge of poker
  Artificial Intelligence Journal pp 201 - 240

  Immanuel Schweizer, et. al. (2009)
  An Exploitative Monte-Carlo Poker Agent
  Tech Report, Technische Universität Darmstadt

Theses & Papers

- Game Theoretic Approaches – Sequence form + abstractions

Darse Billings, et. al. (2003)

Approximating Game-Theoretic Optimal Strategies for Full-scale Poker

IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence pp 661 – 668

Rickard Andersson (2006)

Pseudo-Optimal Strategies in No-Limit Poker

MSc, Umea University

(*good introductory examples for LP construction via the sequence form)

Theses & Papers

- Game Theoretic Approaches – CFRM

  Michael Johanson (2007)

  Robust Strategies and Counter-Strategies: Building a Champion Level Computer Poker Player

  MSc, University of Alberta

  Martin Zinkevich, et. al. (2007)

  Regret Minimization in Games with Incomplete Information

  Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, NIPS 2007

Theses & Papers

- Artificial Neural Networks

    Aaron Davidson (2002)

    Opponent Modeling in Poker: Learning and Acting in a Hostile and Uncertain Environment

    MSc, University of Alberta

    (*section on neural networks for opponent modeling)

- Evolutionary Algorithms

    Jason Noble (2002)

    Finding Robust Texas Hold'em Poker Strategies Using Pareto Coevolution and Deterministic Crowding

    Proceedings of the 2002 International Conference on Machine Learning and Applications - ICMLA 2002

Theses & Papers

- Case-Based Reasoning

  Jonathan Rubin and Ian Watson (2009)

  A Memory-Based Approach to Two-Player Texas Hold'em

  AI 2009: Advances in Artificial Intelligence, 22nd Australasian Joint Conference pp 465 - 474

Thank you!

To challenge Sartre go to:

www.cs.auckland.ac.nz/poker