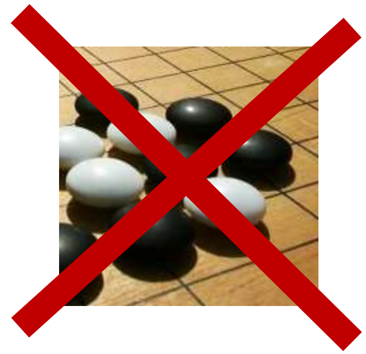
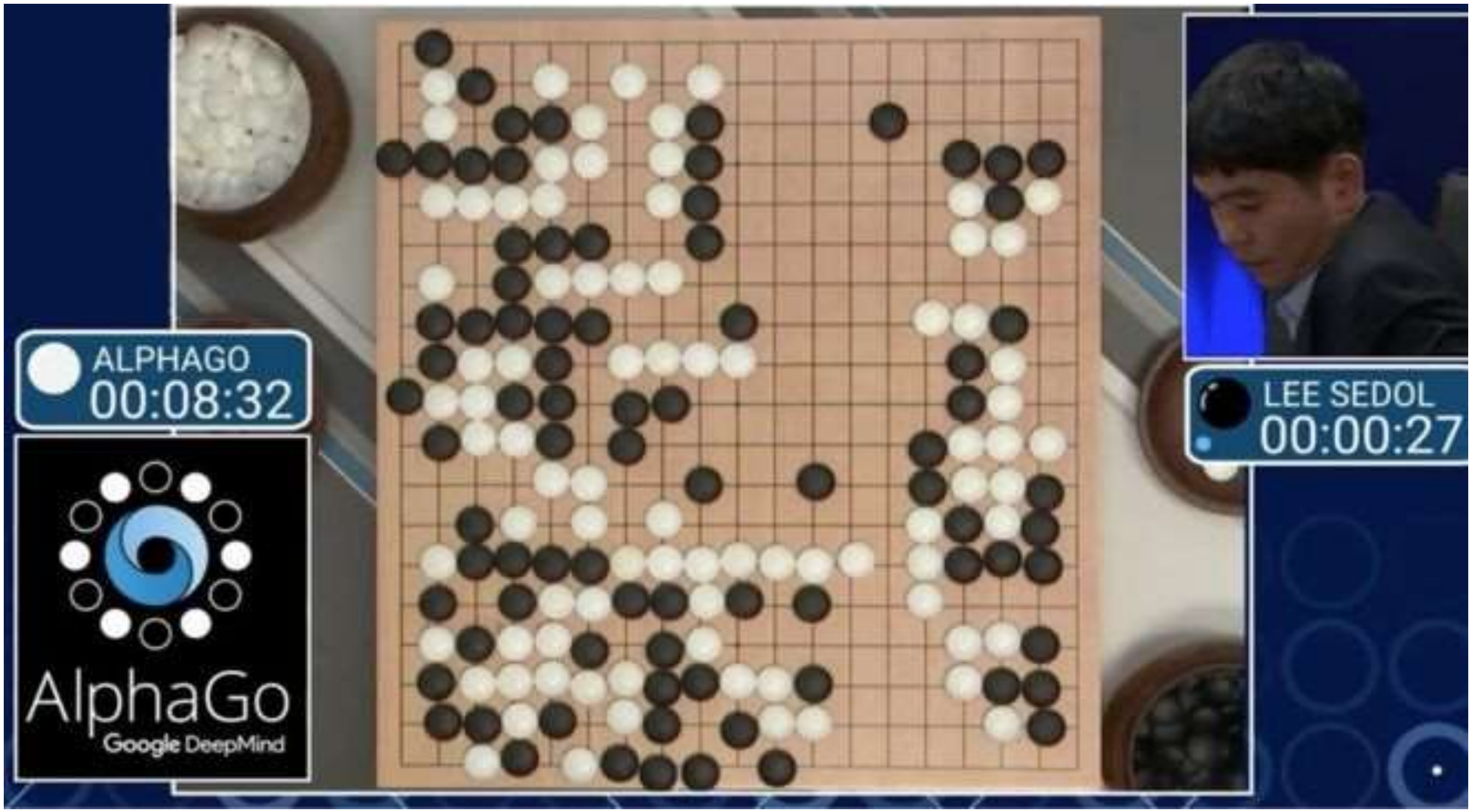
Endgame solving, and *Libratus*, the first superhuman AI for 2-player no-limit Texas hold'em

Tuomas Sandholm

Imperfect-information games



AlphaGo



In principle, AlphaGo techniques extend to other **perfect-information** games

Perfect-information games

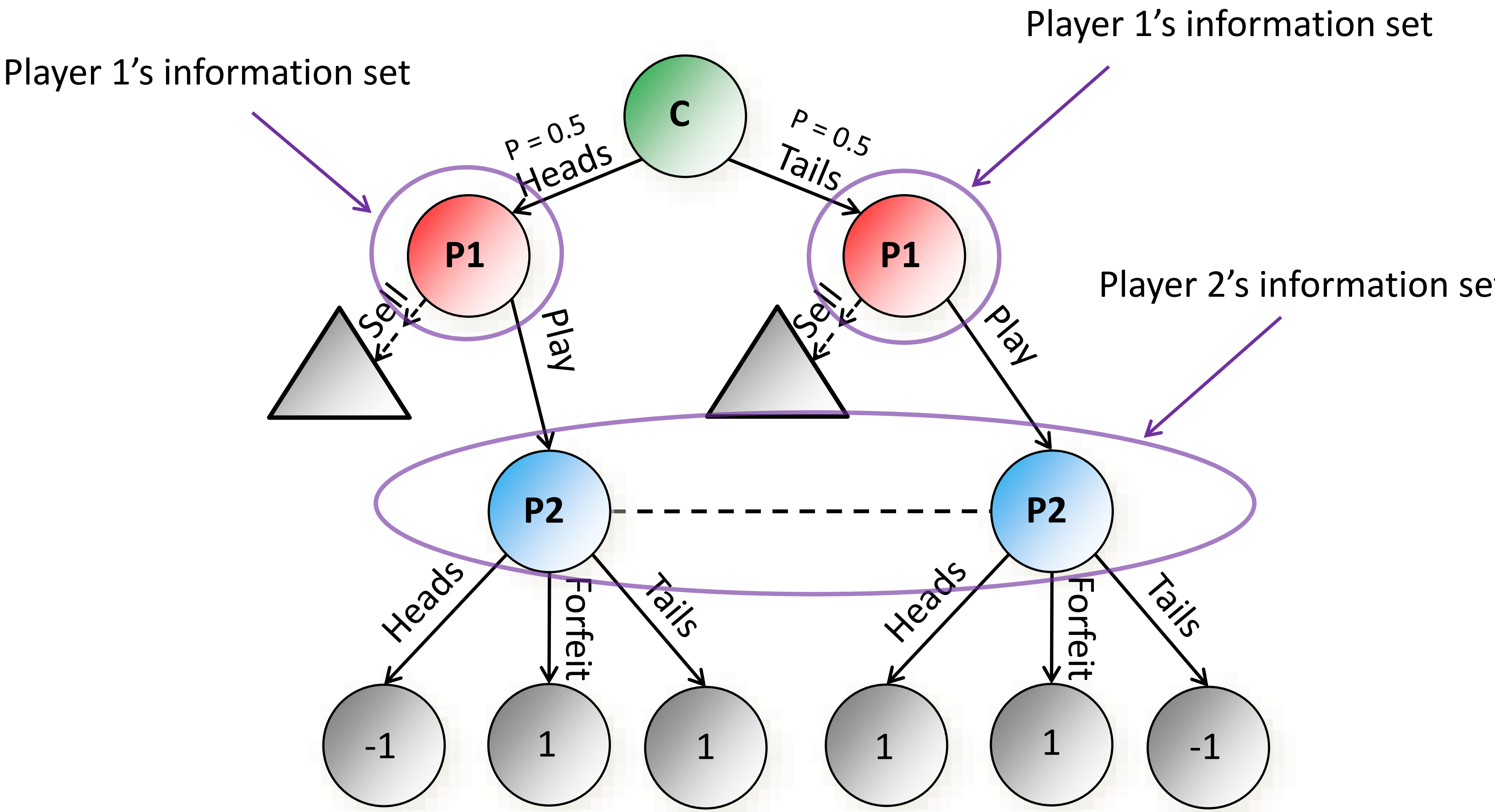
Sicilian Defense

Queen's Gambit

- Subgames can be solved with information from the subgame only
- This is **not true** in imperfect-information games

Imperfect-information games

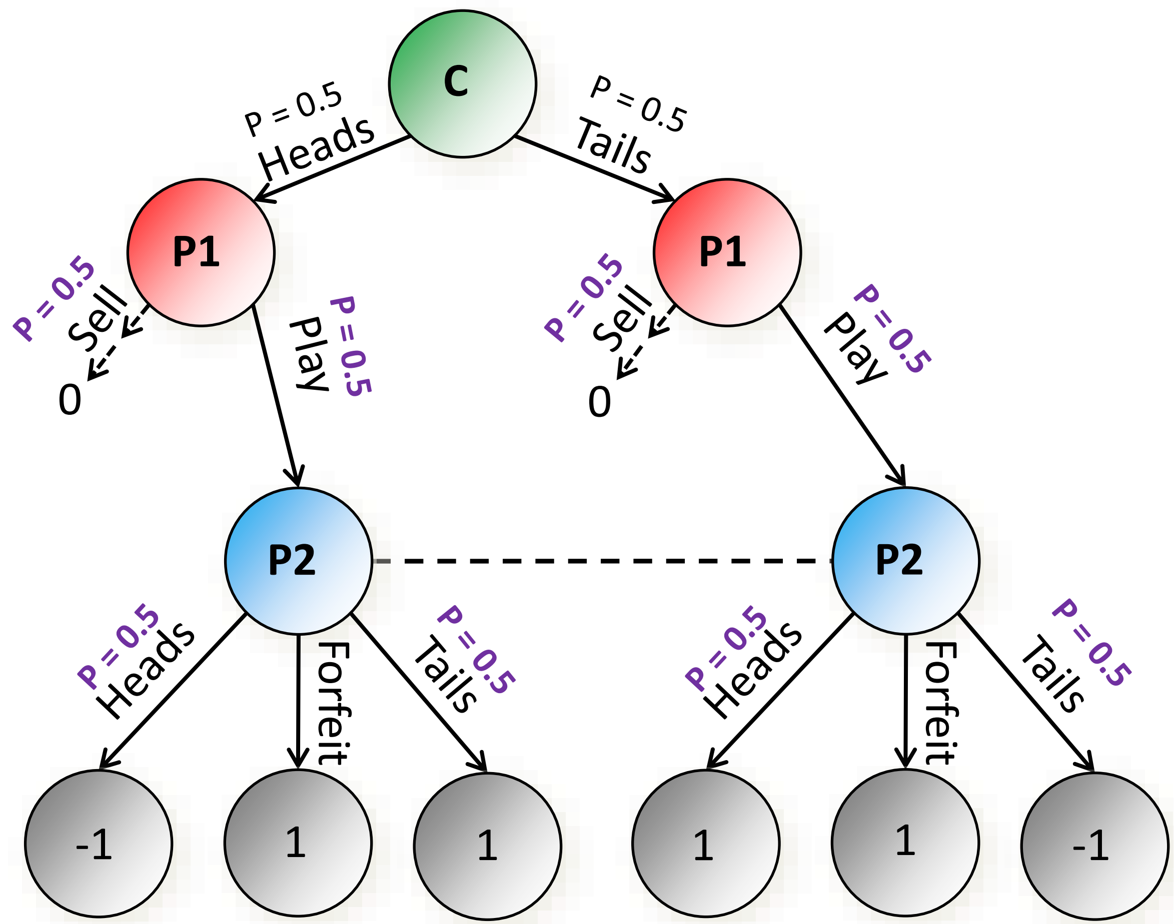
Example game: "Coin toss"

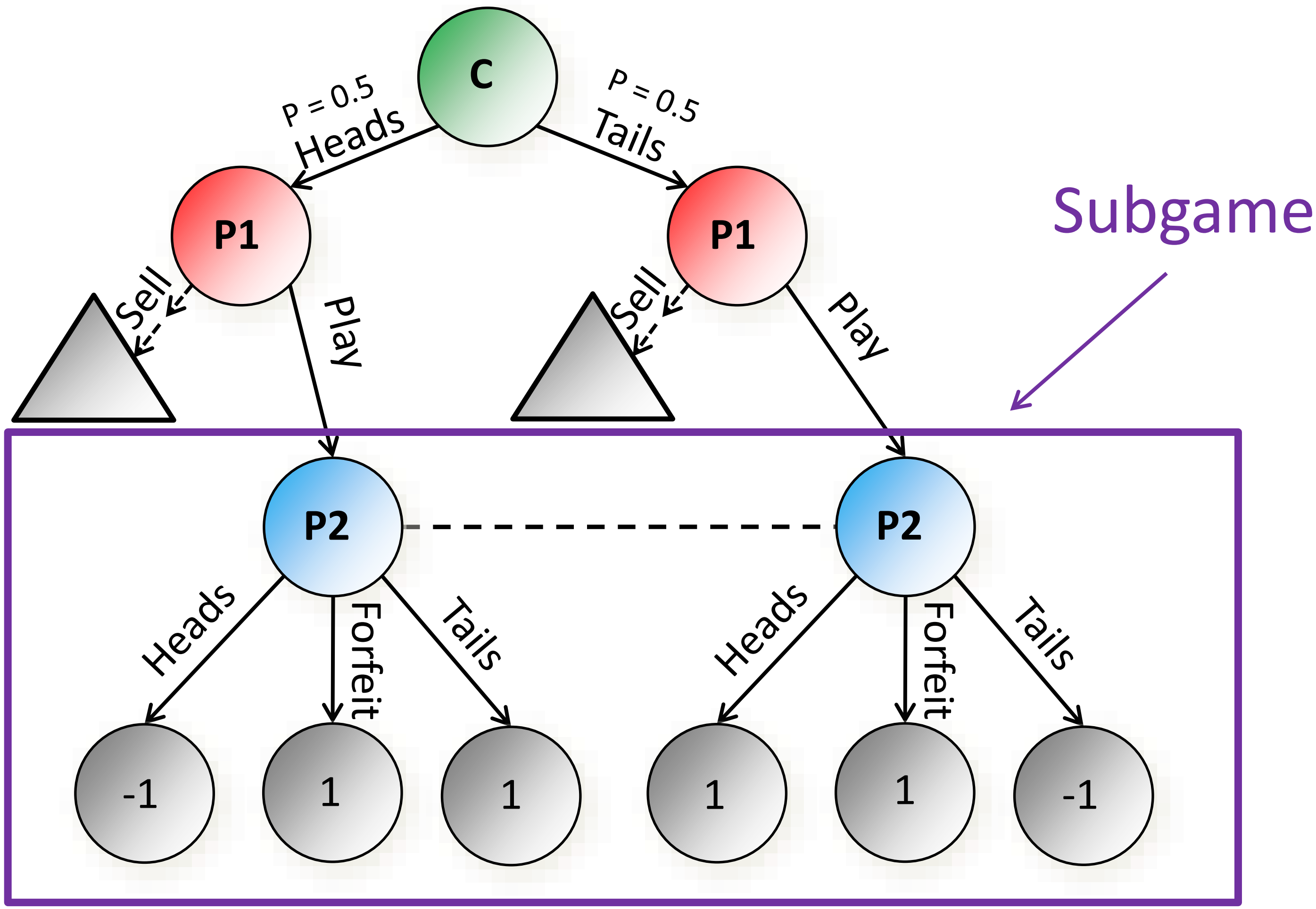


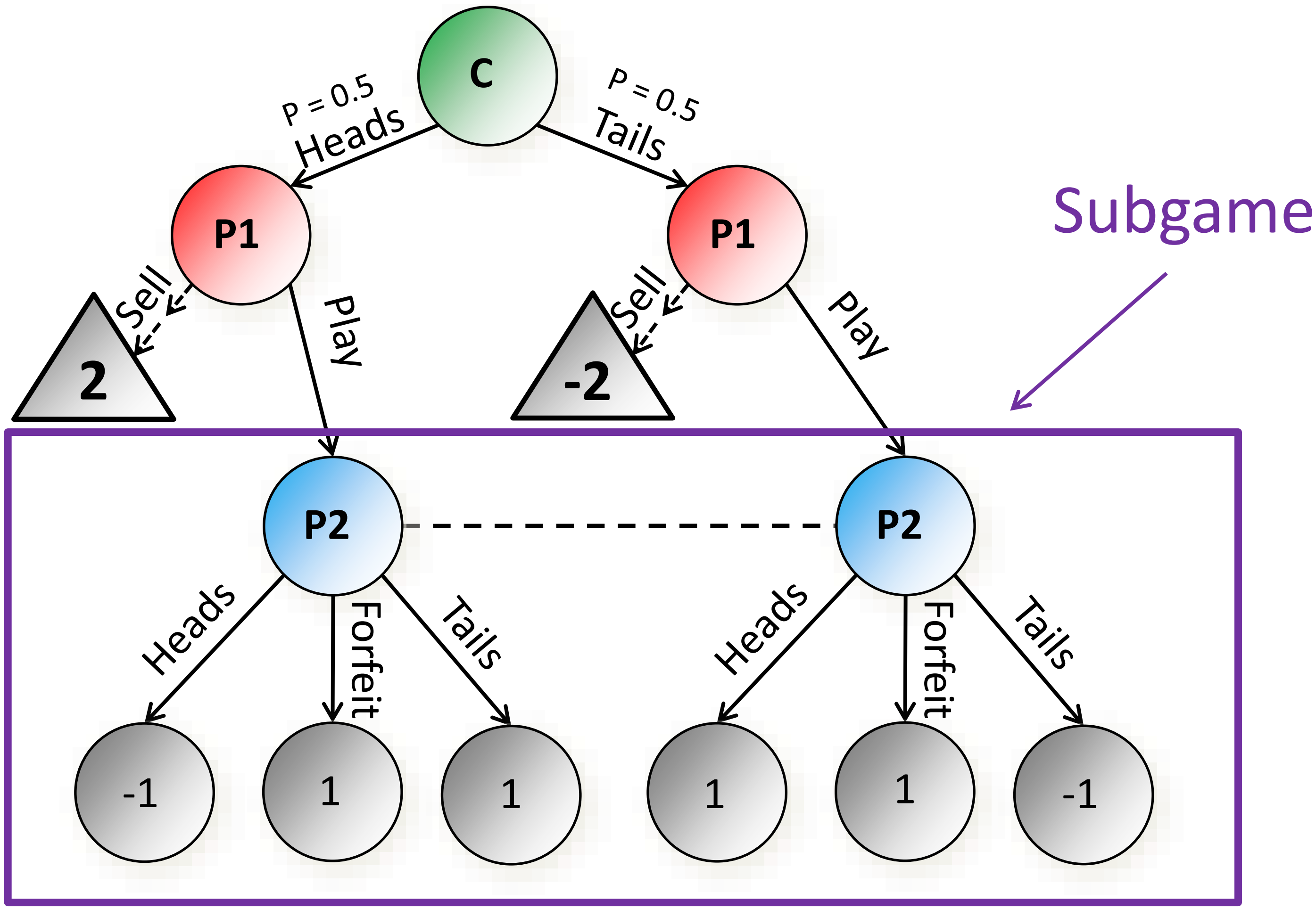
What is rational play?

Nash Equilibrium: a profile of strategies in which no player can improve by deviating (beliefs derived from strategies using Bayes rule). **Robust**

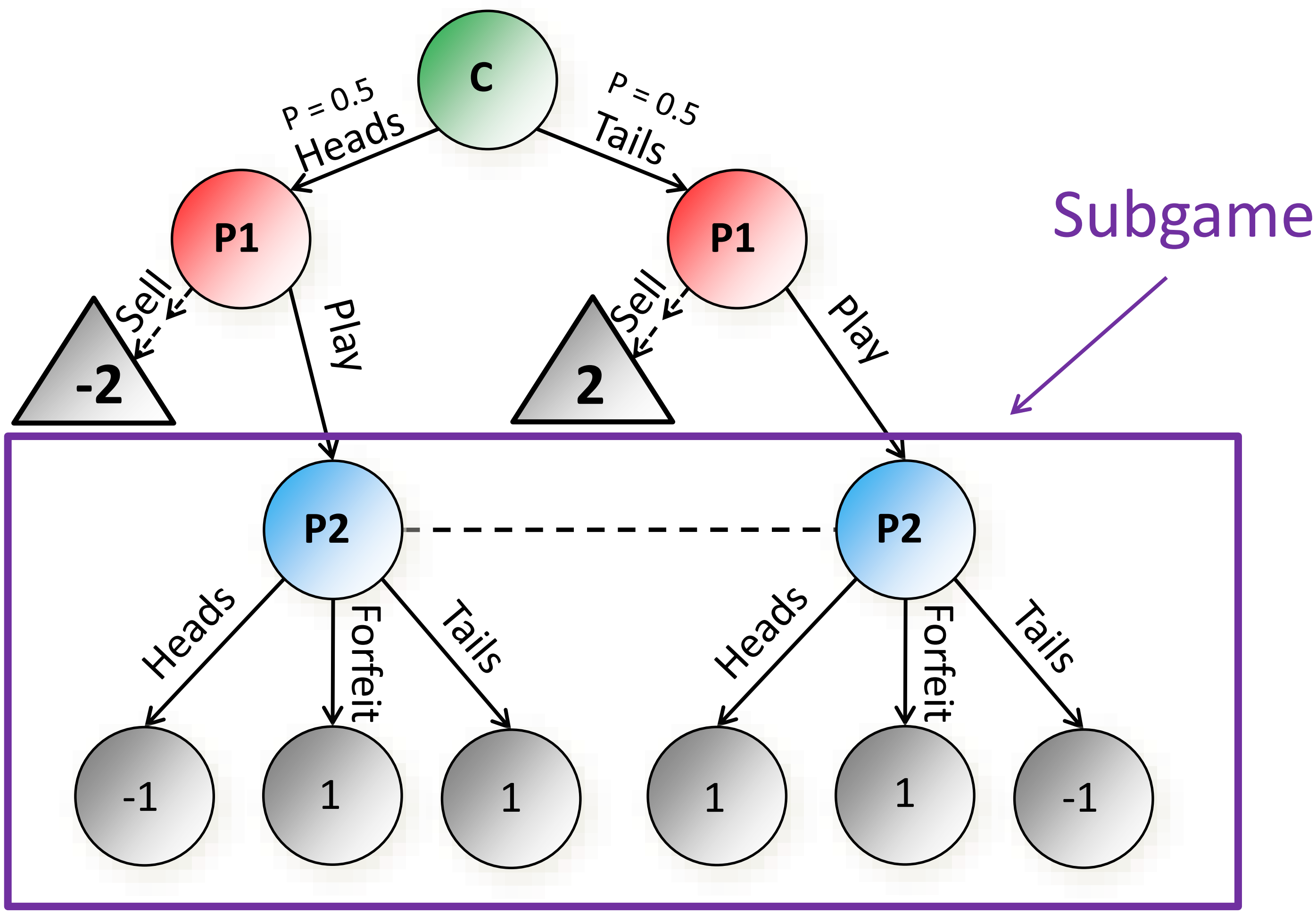
ϵ-Nash Equilibrium: No player can improve by more than ϵ




Subgame
C
P = 0.5
Heads
P = 0.5
Tails
P1
P1
Sell
Play
Sell
Play
P2
P2
Heads
Forfeit
Tails
Heads
Forfeit
Tails
-1
1
1
1
1
-1



C
P = 0.5
Heads
P = 0.5
Tails
P1
P1
Sell
Play
Sell
Play
2
-2
Subgame
P2
P2
Heads
Forfeit
Tails
Heads
Forfeit
Tails
-1
1
1
1
1
-1



C
P = 0.5
Heads
P = 0.5
Tails
P1
P1
Sell
Play
Sell
Play
-2
2
Subgame
P2
P2
Heads
Forfeit
Tails
Heads
Forfeit
Tails
-1
1
1
1
1
-1

Tackling imperfect-info games

- Application-independent techniques that algorithmically create the strategy

- Techniques for perfect-info games don’t apply

- Challenges
  – Uncertainty about what others and chance will do
  – Hidden state => need to interpret signals
    => use game theory

Poker

- Recognized challenge problem in game theory and AI
  - [Nash 1950]
  - [Kuhn 1950]
  - [Newman 1959]
  - [Waterman 1970]
  - [Zadeh 1977]
  - [Caro 1984]
  - [Pfeffer & Koller 1995]
  - [Billings *et al.* 1998]
  - [Schaeffer *et al.* 1999]
  - [Shi & Littman 2001]
  - [Billings *et al.* 2003]
- Tremendous progress since 2005
  - Rhode Island Hold'em solved (10^9 nodes) [Gilpin & Sandholm 2005]
  - Annual Computer Poker Competition 2006-2018
  - Limit Texas Hold'em near-optimally solved (10^{13} decisions) [Bowling *et al.* 2015]

Heads-up no-limit Texas hold’em

- Was the main ***benchmark and challenge problem*** in AI for imperfect-information games

- 10^{161} situations

- Mostly played on the Internet
  - Also in World Series of Poker, NBC Heads-Up Championship, etc.
  - Featured in *Casino Royale* and *Rounders*

- “Purest form of poker”

- No prior AI has beaten top humans

Texas hold'em

Chance deals 2 cards to each player

Round of betting

Chance deals 3 shared cards

Round of betting

Chance deals 1 shared card

Round of betting

Chance deals 1 shared card

Round of betting

Brains vs AI Rematch

- *Libratus* (= our AI) against four of the ***best*** heads-up no-limit Texas Hold'em specialist pros



- 120,000 hands over 20 days in January 2017
- $200,000 divided among the pros based on performance
- Conservative experiment design

Conservative experiment design to favor humans

- Large number of hands
- Humans got to choose:
  - #days, break days, times of day, breaks between sessions—even dynamically
  - Two tabling
  - 4-color deck
  - Hot keys, adjustable dynamically
  - Specific hi-res monitors, their own mice
  - Twitch chat on vs off
  - Play in public vs private within each pair
- 200 big blinds deep
- No use of timing tells
- Action history displayed
- Hand histories given to both sides every evening, including hands opponent folded
- Humans allowed to:
  - Use computers and any programs to analyze
  - Collaborate and coordinate actions (except within each hand)
  - Get outside help (e.g., Doug Polk)
- Humans allowed to think as long as they want
- Mis-click hands canceled
- Ginseng ☺

Final result

- Libratus beat the top humans in this game by a lot
  - 147 mbb/hand
  - Statistical significance 99.98%, i.e., 0.0002
  - Each human lost to Libratus

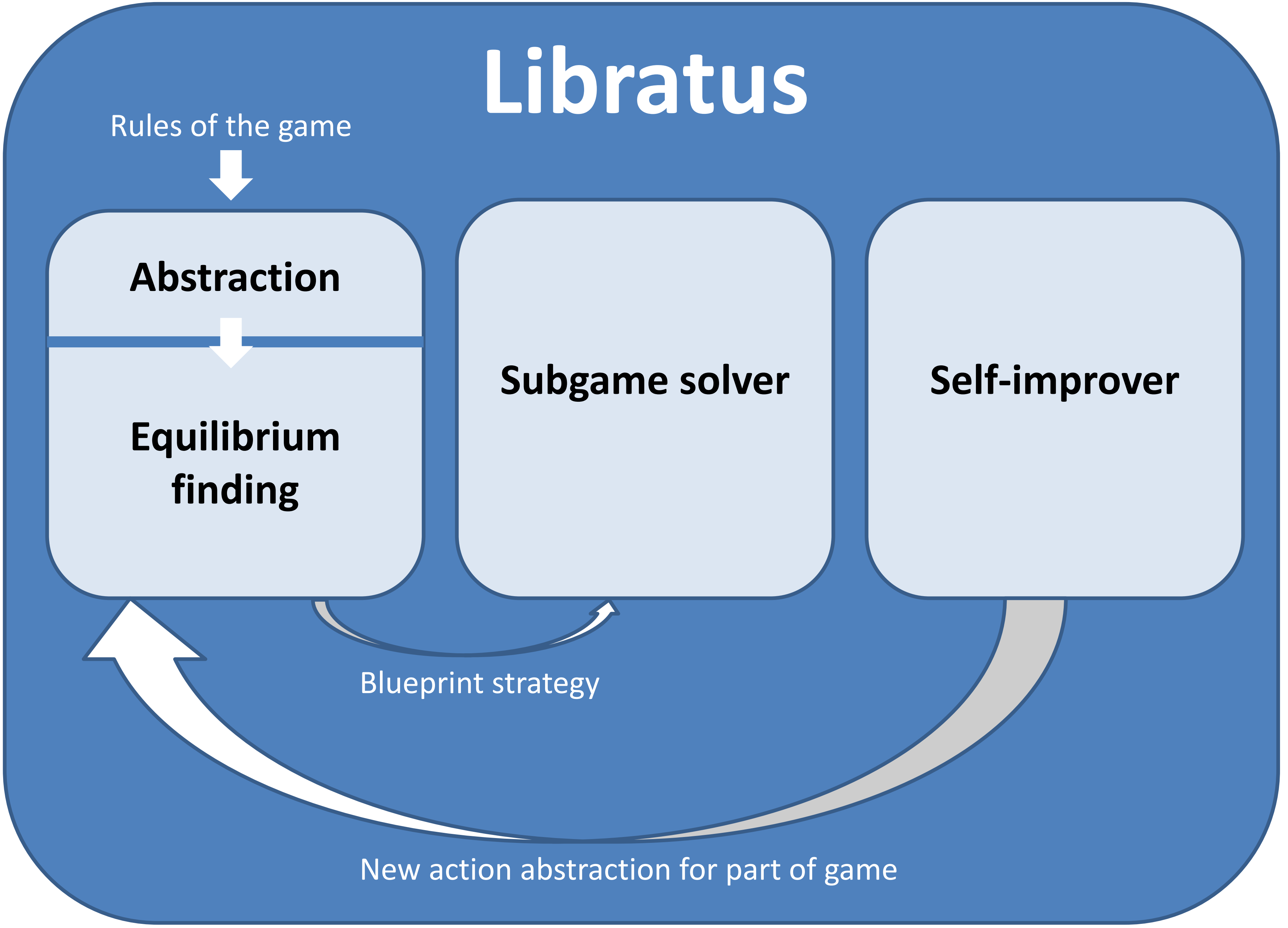
Why is game-theoretic AI better than machine learning for these problems?

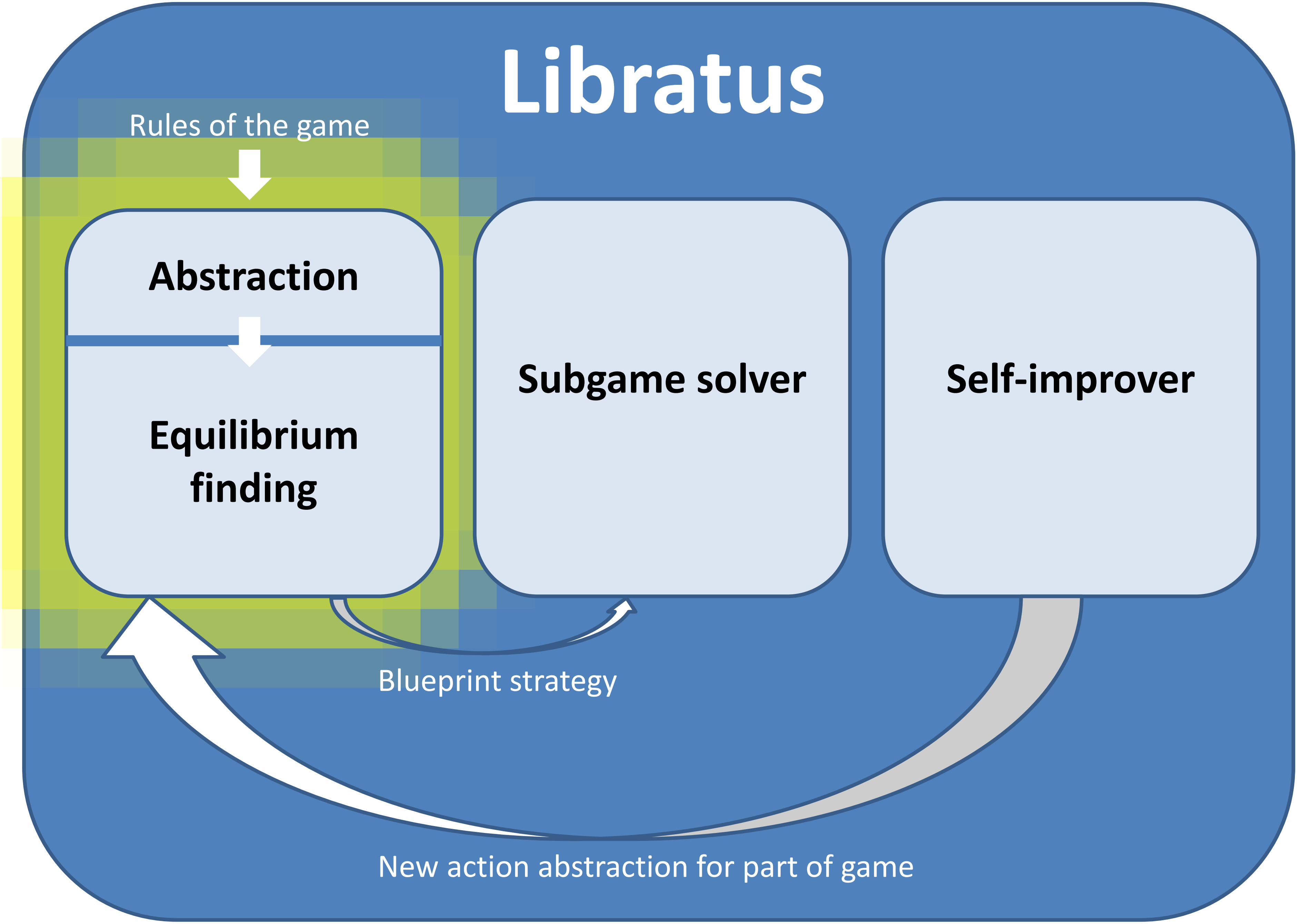
1. Requires no data
2. Doesn't assume opponent will continue to behave the same way as in the past
3. Not exploitable (even if opponent knows our strategy)
   - 36,000 hands against 6 Chinese poker players
     - WSOP bracelet winner
     - Expertise in computer science & ML
     - They studied Libratus's hand histories in advance
   - AI won by 220 mbb/hand
     - Won each of the 9 sessions
     - Also beat each human individually
   - Demonstrated that this approach is not frail
     - Minmax theorem proves this for exact Nash equilibrium. Our experiments showed it for computational approximations
     - Unlike what has been found with ML approaches (e.g., for Go, DOTA2, and Starcraft II)

How does *Libratus* work?

[Brown & Sandholm, *Science* 2018]


Bridges supercomputer


Libratus
Rules of the game
Abstraction
Equilibrium finding
Subgame solver
Self-improver
Blueprint strategy
New action abstraction for part of game


Libratus
Rules of the game
Abstraction
Equilibrium finding
Subgame solver
Self-improver
Blueprint strategy
New action abstraction for part of game

Abstraction in Libratus

- Abstracting chance's actions (cards in poker)
  - Same algorithm that we used in *Tartanian8* [Brown, Ganzfried & Sandholm, AAMAS-15]
    - Like the state-of-the-art state-abstraction algorithm for centralized equilibrium finding presented in class, except distributed based on the public flop cards so that any one sample stays within one compute node (blade)
  - But much finer abstraction
    - 1^{st} and 2^{nd} betting round: no abstraction
    - 3^{rd} betting round: 55M card histories -> 2.5M buckets
    - 4^{th} betting round: 2.4B card histories -> 1.25M buckets

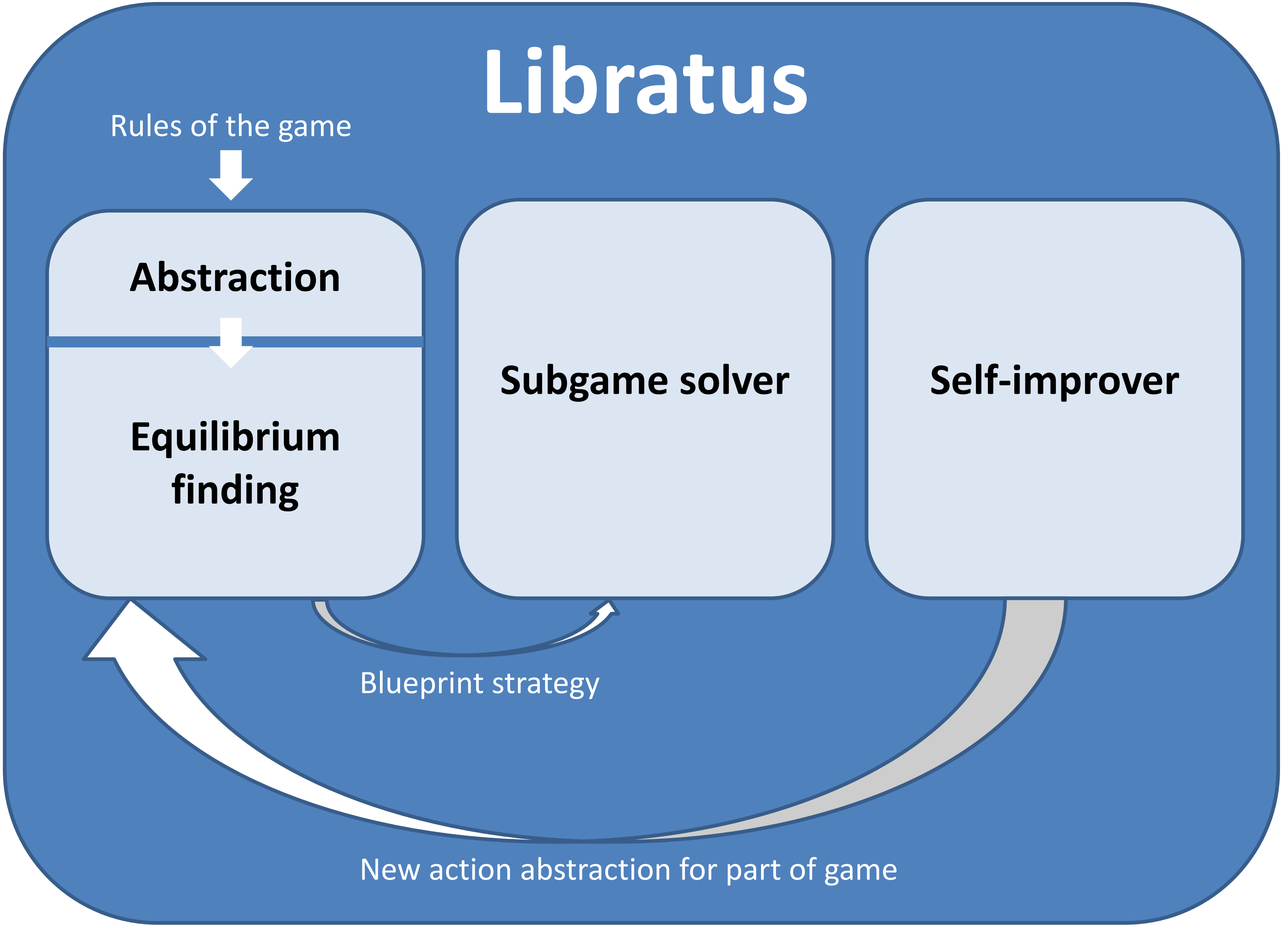
- Abstracting player's actions (bet sizes in poker)
  - Largely based on what top humans and AIs do
  - Added radical bet sizes
  - Optimized some of the bet sizes in the early parts of the tree [Brown & Sandholm, AAAI-14]

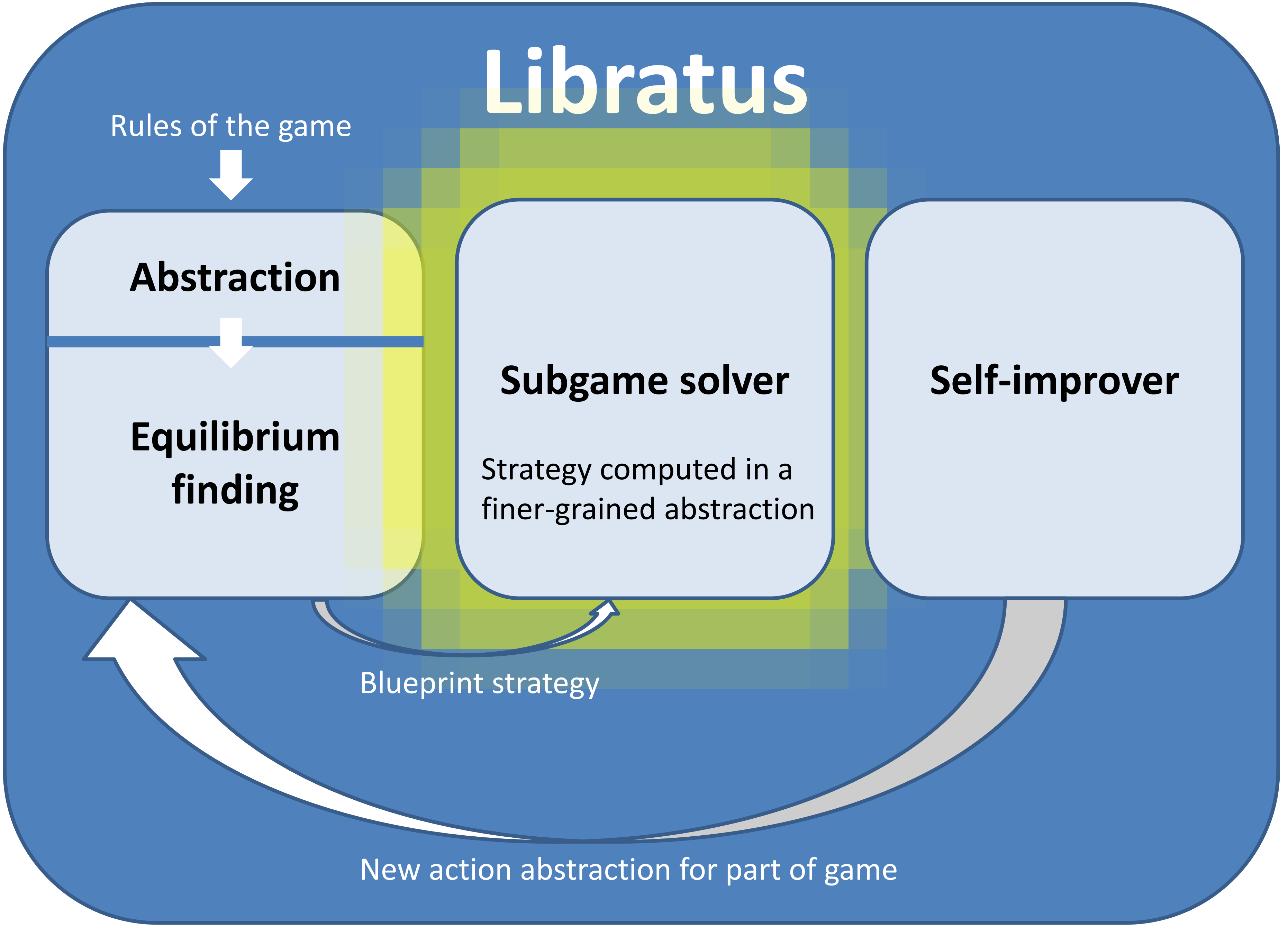
Our equilibrium-finding algorithm

- Improvement on Monte-Carlo Counterfactual Regret Minimization [Lanctot *et al. NIPS*-09]
- Starts visiting less often paths where our own actions don't look promising (similar to Brown & Sandholm NIPS-15 paper and AAAI-17 workshop paper)
  => Speedup => can solve larger abstractions
- Also, the imperfect-recall abstraction, in effect, becomes finer grained
  => Better solution quality
- Distributed across 1 + 195 compute nodes
  - Distribution along game tree, not "embarrassingly parallel"

Systems structuring & our usage

- ***Bridges*** supercomputer
  - ~$17 million (including running it for its lifetime)
  - Architected by Hewlett Packard Enterprise (HPE) & Pittsburgh Supercomputing Center
  - Heterogeneous architecture
  - We used the part that has 800 HPE Apollo 2000 servers, each with 28 cores and 128GB RAM
  - We officially used ~24 million core hours for Libratus (Jan 2016-Jan 2017)
  - But we used only 14 of the 28 cores on each node because that was fastest
  - We were the biggest user of Bridges in that timeframe (used about half)
- Blueprint runs typically used 1 + 195 nodes
  - Typically ~1-8 weeks per run
- Each endgame solver used 50 nodes
  - Typically 30-60 seconds per run
- Each self-improver run used 196-600 nodes
  - Typically for 8-30 hours per run
- C++, Open-MP for parallelism within each server, MPI for distributed computing
- 2.6 PB disk storage
  - Multiple strategies
  - Snapshots (balance in snapshotting)
  - Connections by Intel Omni-Path
  - Intel Lustre file system
- During the competition, we had three locations connected by Internet:
  - Front end running on a browser at Rivers casino
  - Poker server running on a Dell rack server at CMU
  - AI running on Bridges at Pittsburgh Supercomputing Center (in an industrial basement in Monroeville)


Libratus
Rules of the game
Abstraction
Equilibrium finding
Subgame solver
Self-improver
Blueprint strategy
New action abstraction for part of game

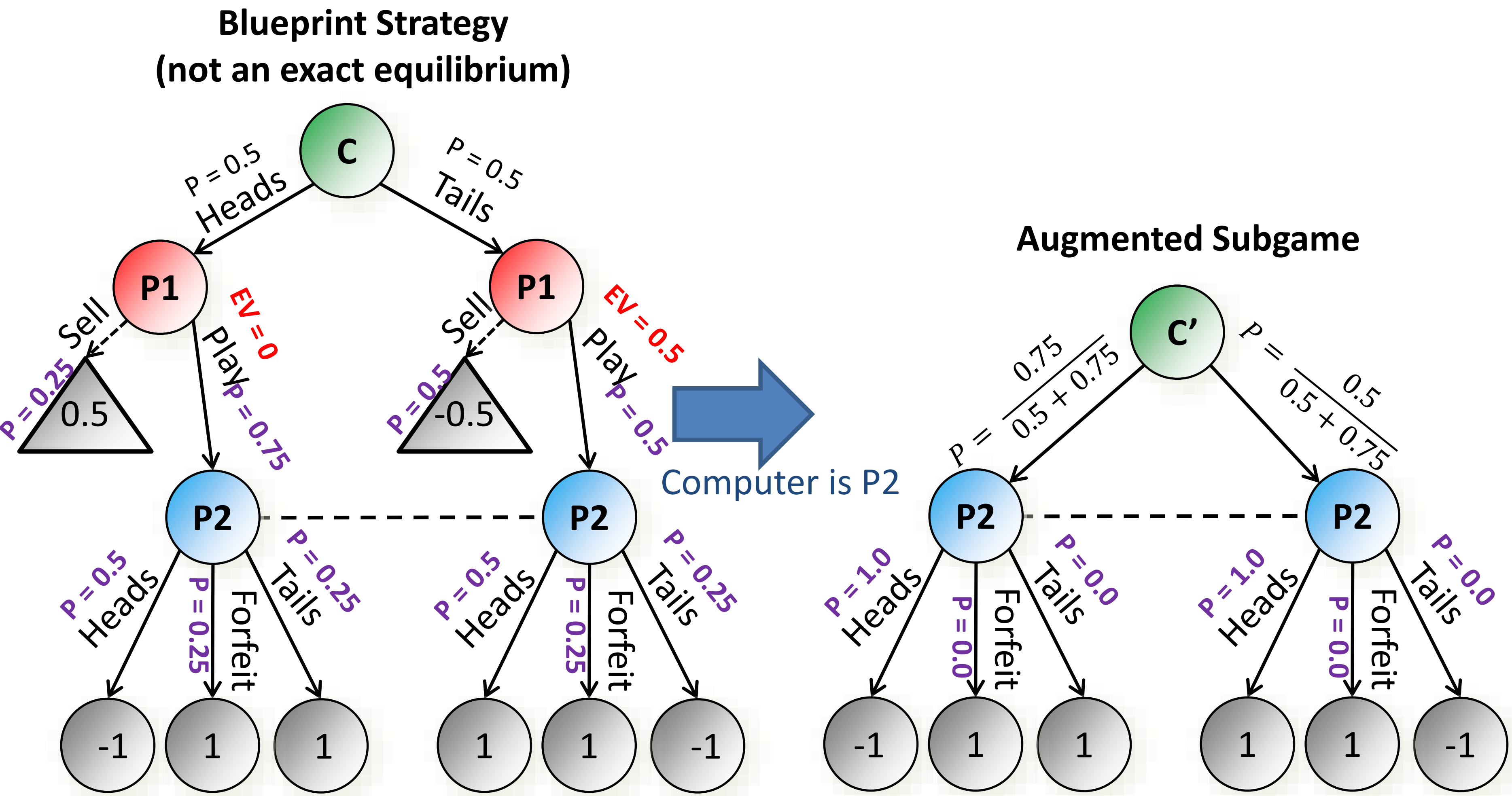

Libratus
Rules of the game
Abstraction
Equilibrium finding
Subgame solver
Strategy computed in a finer-grained abstraction
Self-improver
Blueprint strategy
New action abstraction for part of game

Subgame solving during game play

- Want to solve reached "subgames" in finer abstraction
- …but in imperfect-information games, subgames can't be solved independently
- => Solve the whole game in a coarse abstraction for a "blueprint strategy" that gives context for solving the current subgame in a finer abstraction
- This is the most important technique in *Libratus*, 1st AI to beat top pros in 2-player no-limit Texas hold'em (10^{161} information sets) [Brown & Sandholm *Science* 2018]

Bayesian subgame solving

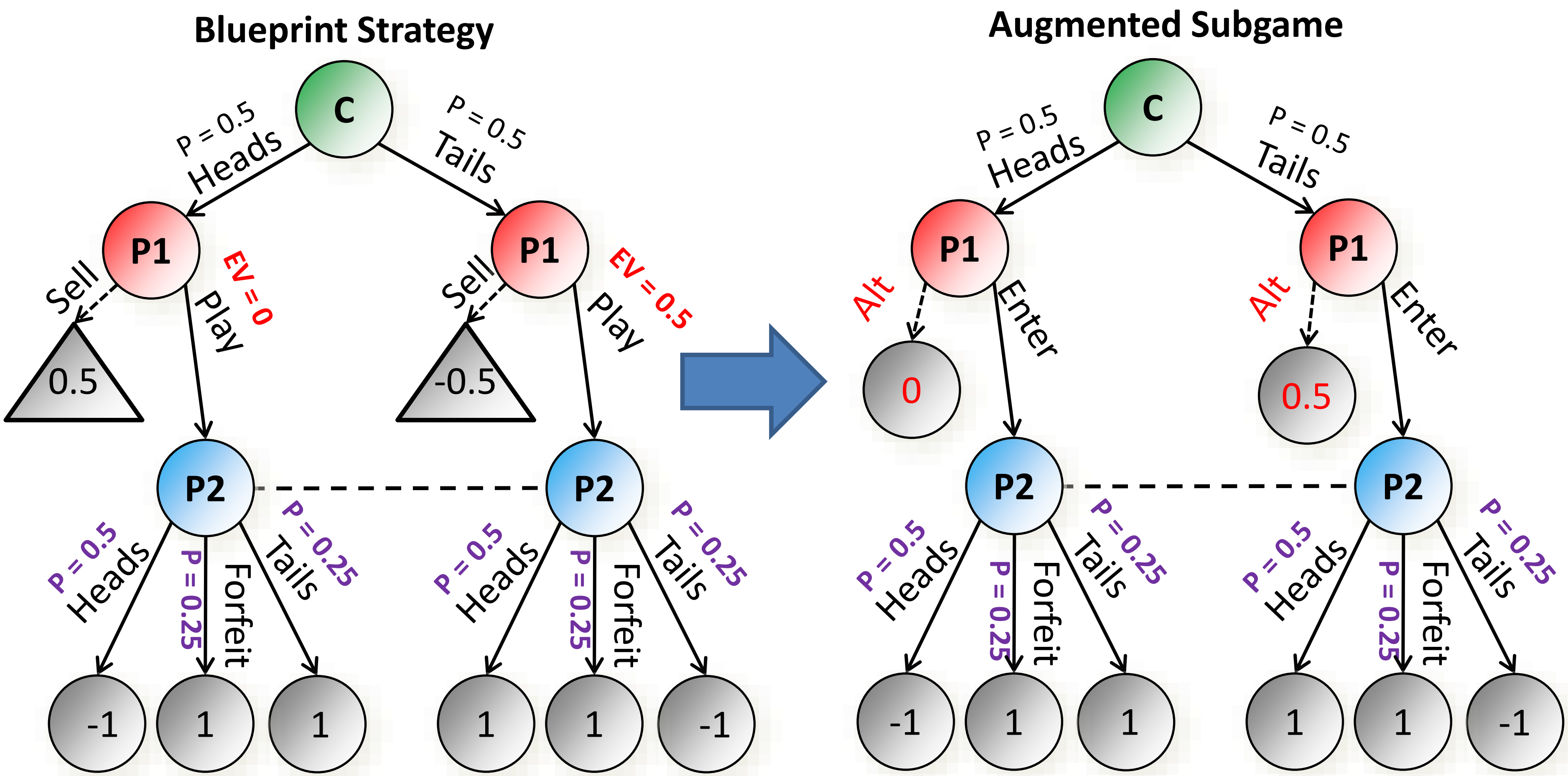
[Gilpin & Sandholm, *AAAI*-06, *AAMAS*-07; Ganzfried & Sandholm, *AAMAS*-15]



- No theoretical guarantees: unsafe
- Does well in practice for some domains

Re-solve refinement [Burch *et al*. AAAI-14]

- P1 can choose between entering the subgame or taking the EV (according to the blueprint) of the subgame
- Makes sure opponent's EV for entering the subgame is no higher than in the blueprint strategy
  => **Safety theorem.** Strategy is no more exploitable than blueprint strategy
- But may miss obvious opportunities for improvement (e.g., not forfeiting)

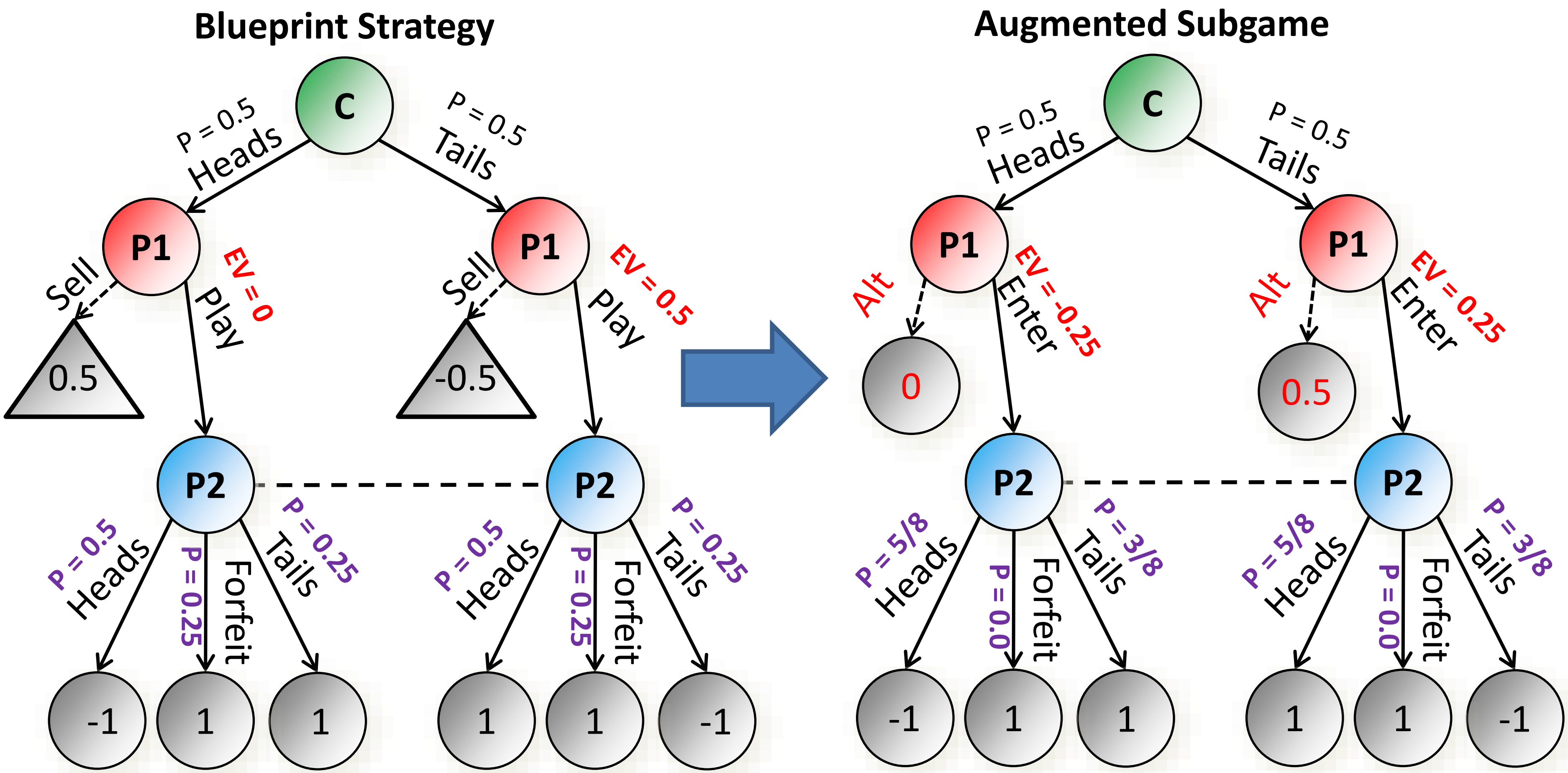


Maxmargin refinement [Moravcik *et al.*, *AAAI*-16]

Similar to Re-solve, but punishes P1 as much as possible for choosing Enter rather than Alt

$Margin_{Heads} = EV[Alt_{Heads}] - EV[Enter_{Heads}]$

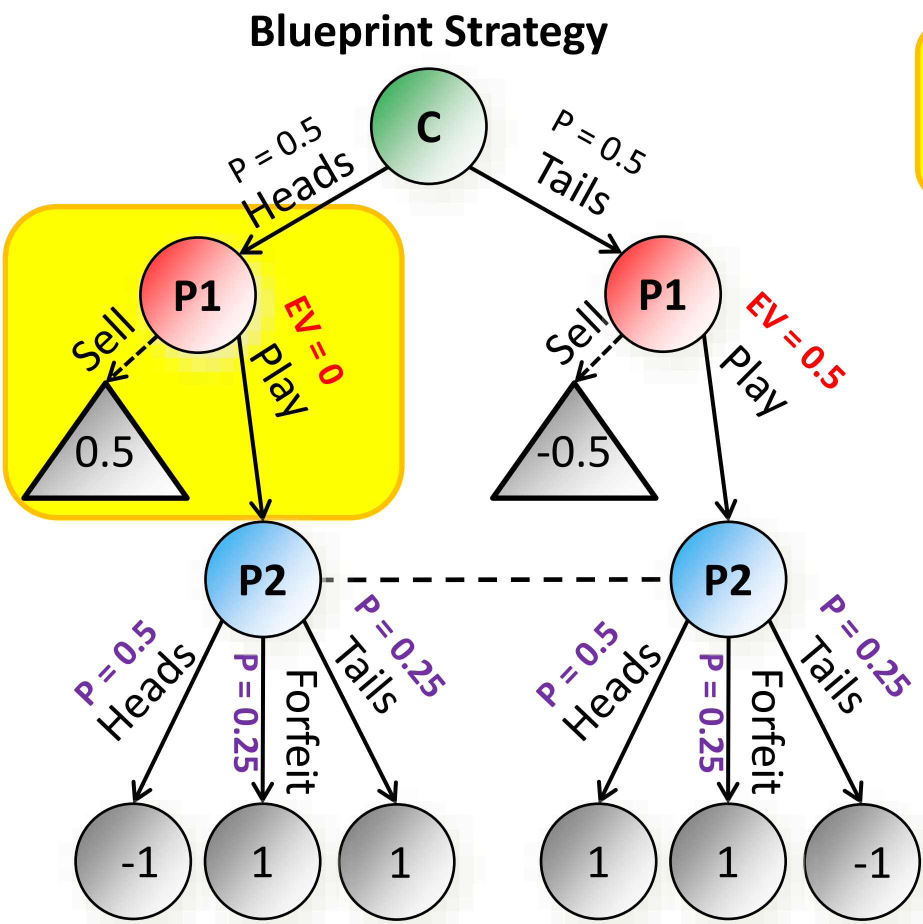
Maximizes the minimum margin (Re-solve simply attempts to make all margins nonnegative)



Problem: While we focus on reducing P1's EV for Heads in the subgame to -0.25, P1 can just Sell for 0.5 in Heads

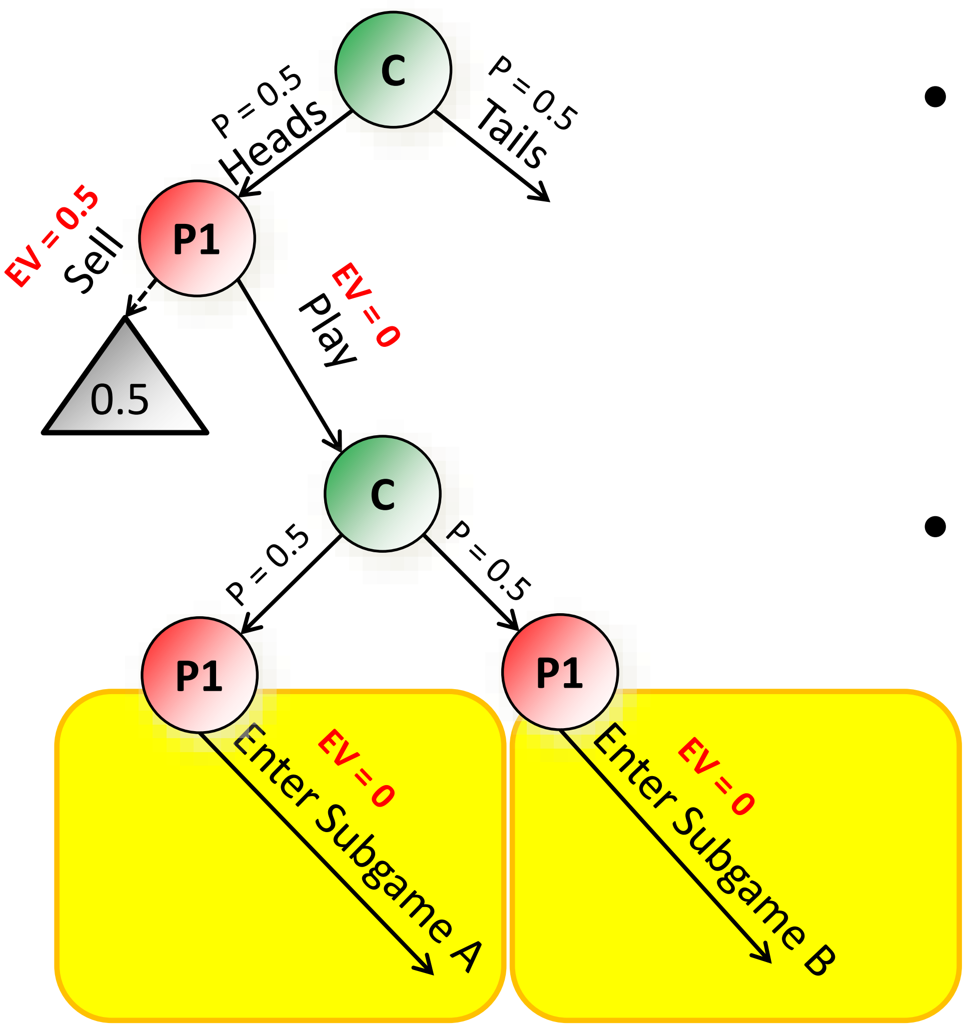
Reach-maxmargin refinement

[Brown & Sandholm, *AAAI*-17 workshop, *NeurIPS*-17, *Science*-18]



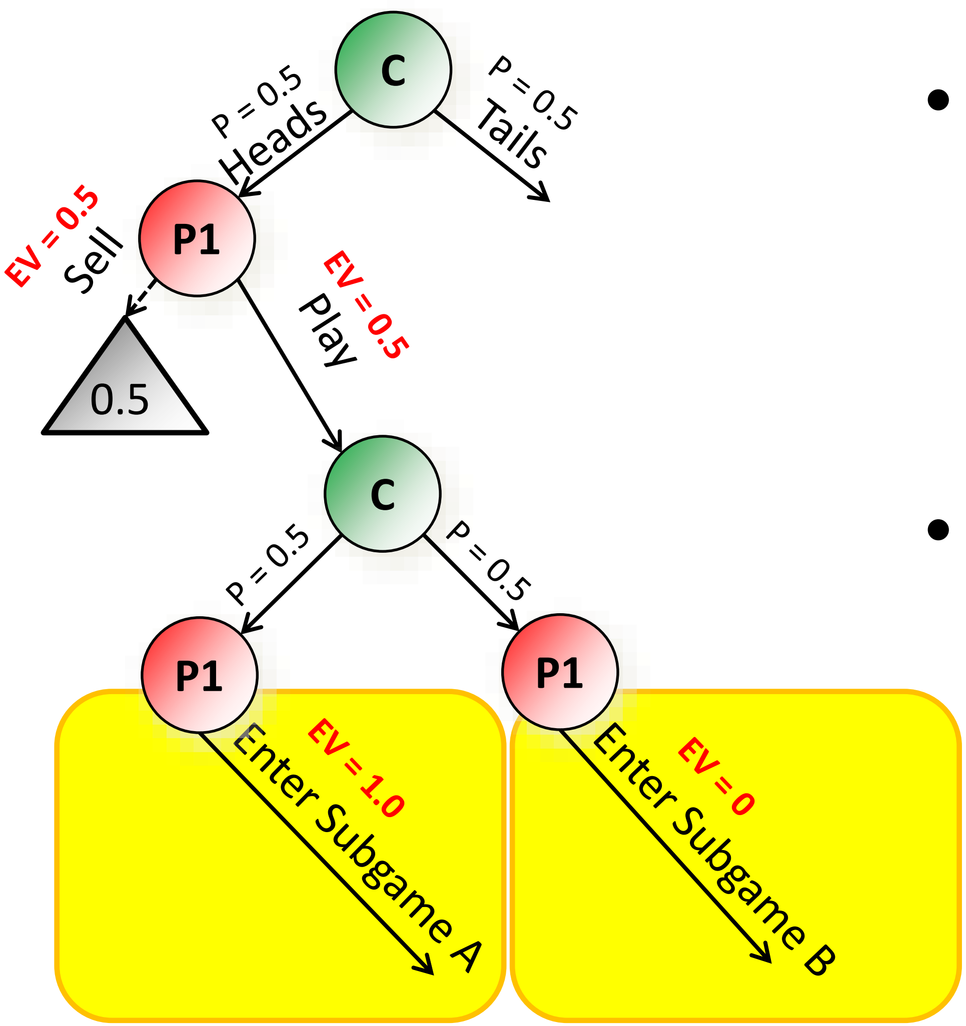
- If P1 chooses Play following Heads, P1 is **gifting** us 0.5
- So, in Augmented Subgame, we can increase the alternative payoff following Heads by 0.5, because choosing Play would still be a mistake for P1 there
- Thus the Gadget Game solver focuses on reducing P1's EV for other types she may have

Reach-maxmargin refinement: multiple subgames



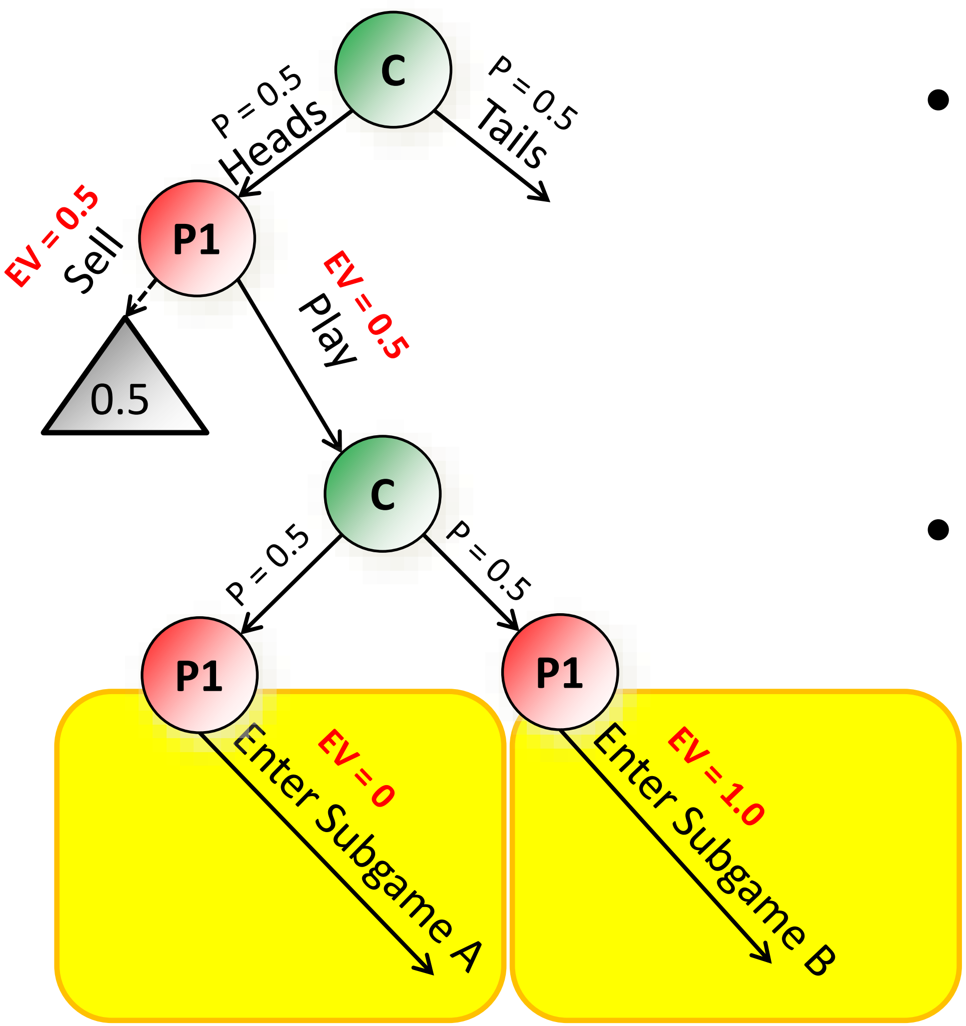
- If multiple subgames are refined, off-path EVs might not remain constant
- Solution: split gifts among subgames by probability subgame is reached

Reach-maxmargin refinement: multiple subgames



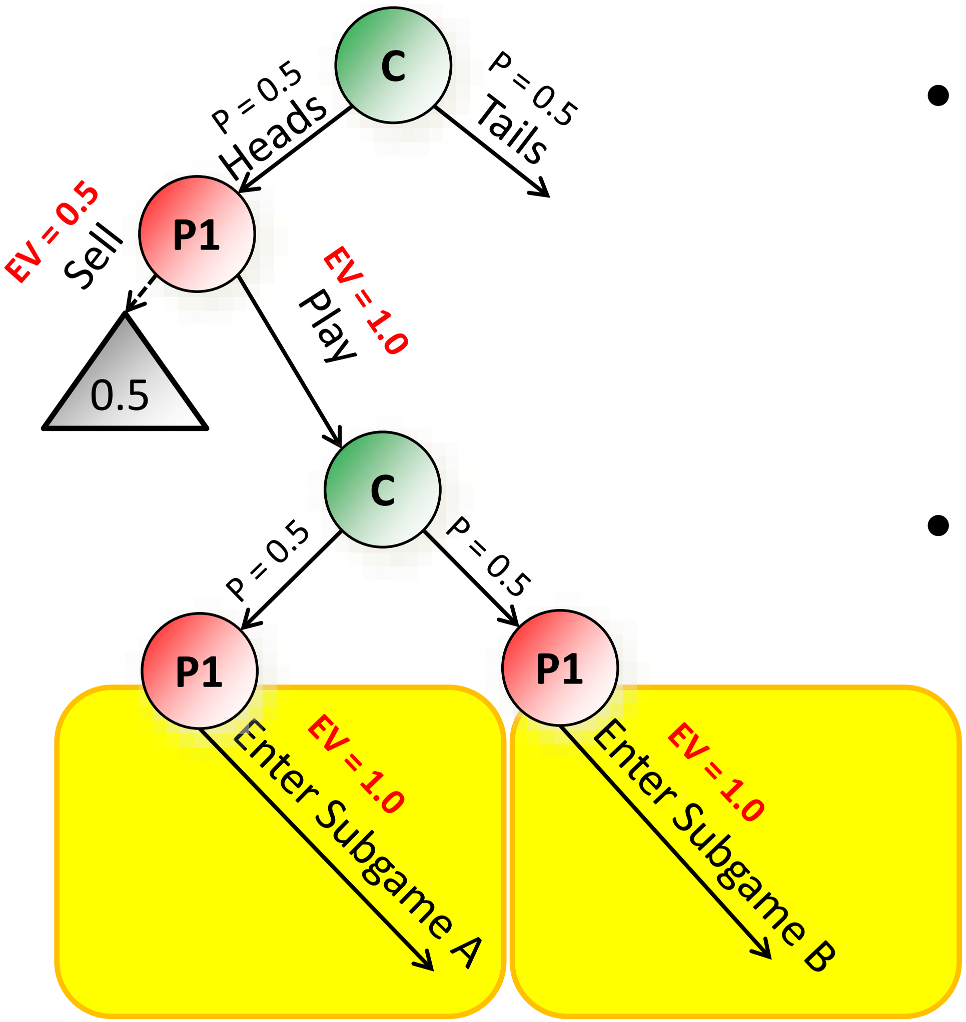
- If multiple subgames are refined, off-path EVs might not remain constant
- Solution: split gifts among subgames by probability subgame is reached

Reach-maxmargin refinement: multiple subgames



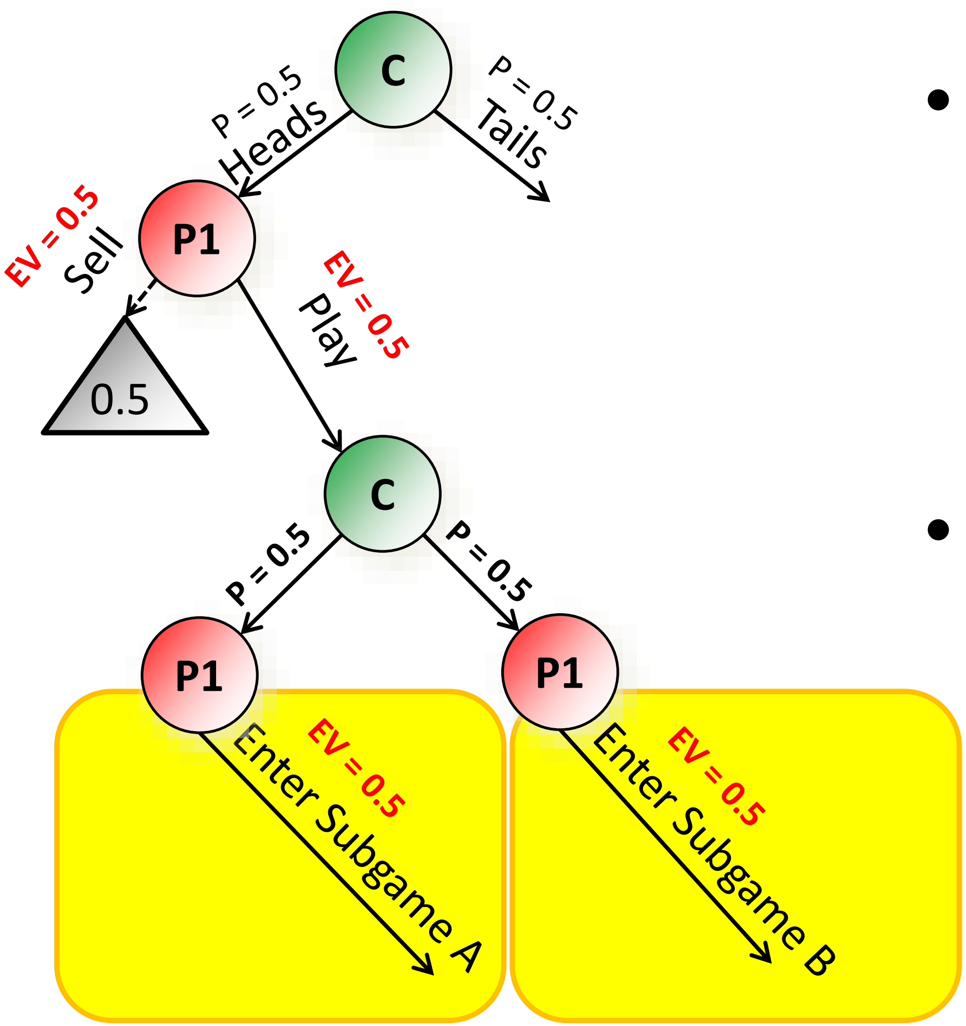
- If multiple subgames are refined, off-path EVs might not remain constant
- Solution: split gifts among subgames by probability subgame is reached

Reach-maxmargin refinement: multiple subgames



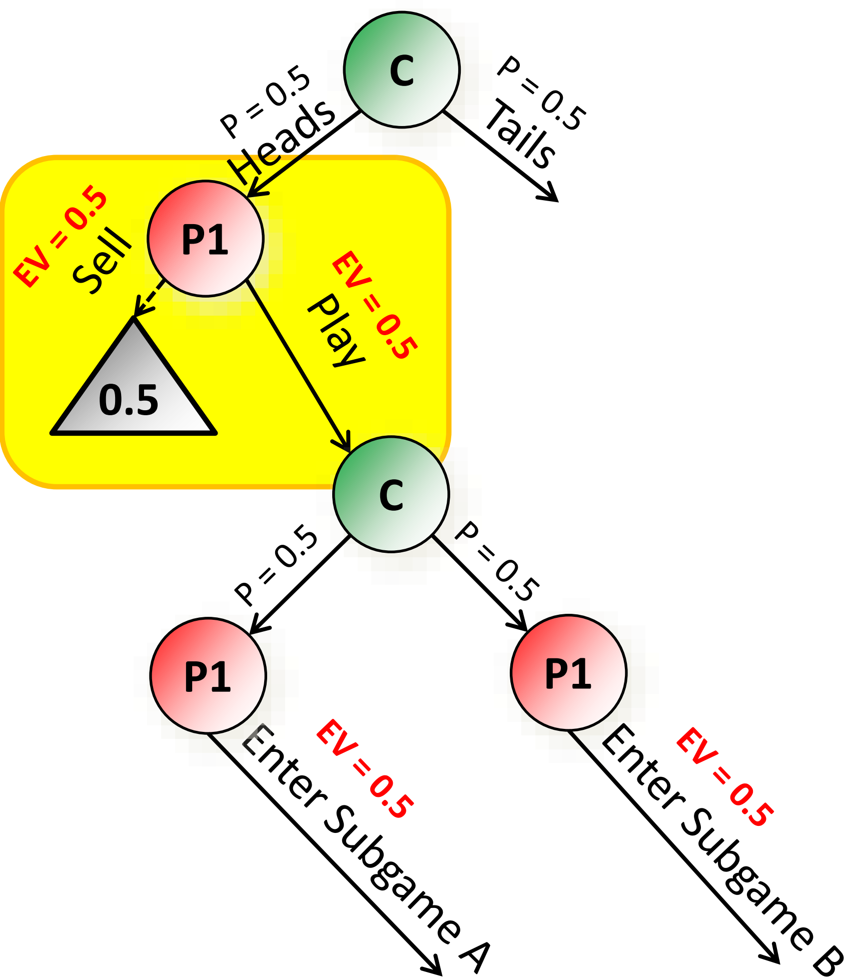
- If multiple subgames are refined, off-path EVs might not remain constant
- Solution: split gifts among subgames by probability subgame is reached

Reach-maxmargin refinement: multiple subgames



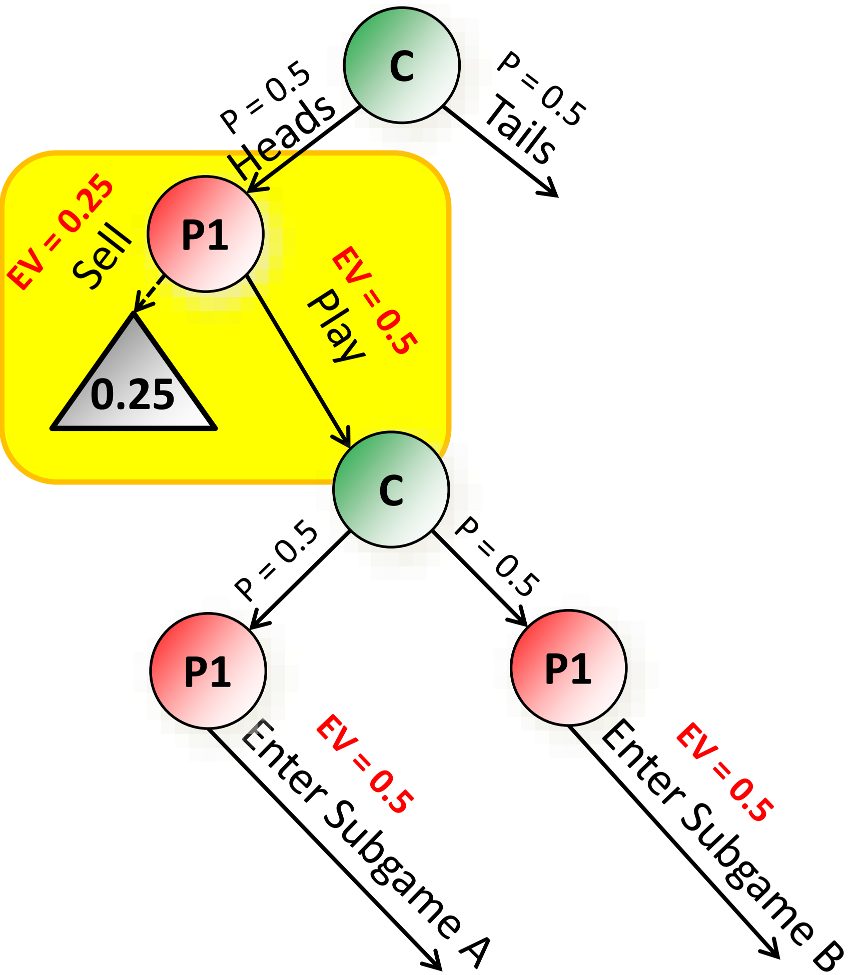
- If multiple subgames are refined, off-path EVs might not remain constant
- Solution: split gifts among subgames by probability subgame is reached

Reach-maxmargin refinement: multiple subgames



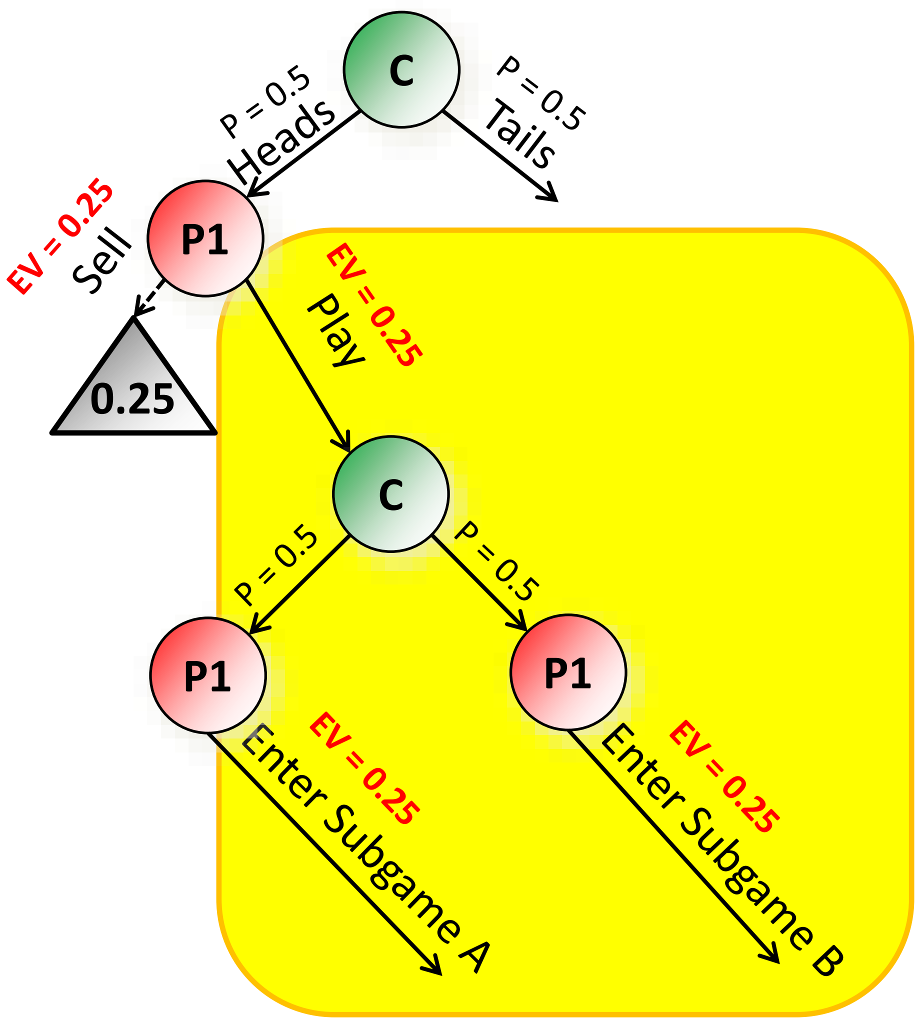
- Gifts might not be as large as we thought, because the subgames they come from will be improved
- Solution: substitute a lower bound on the gift

Reach-maxmargin refinement: multiple subgames



- Gifts might not be as large as we thought, because the subgames they come from will be improved
- Solution: substitute a lower bound on the gift

Reach-maxmargin refinement: multiple subgames



- Gifts might not be as large as we thought, because the subgames they come from will be improved
- Solution: substitute a lower bound on the gift

Medium-scale experiments on subgame solving within action abstraction

	Small Game Exploitability	Large Game Exploitability
Blueprint Strategy	91.3 mbb / hand	41.4 mbb / hand
Unsafe Subgame Solving	5.51 mbb / hand	397 mbb / hand
Re-solve Refinement	81.2 mbb / hand	36.3 mbb / hand
Maxmargin Refinement	9.36 mbb / hand	6.12 mbb / hand
Reach-Maxmargin Refinement	**8.26 mbb / hand**	**5.50 mbb / hand**

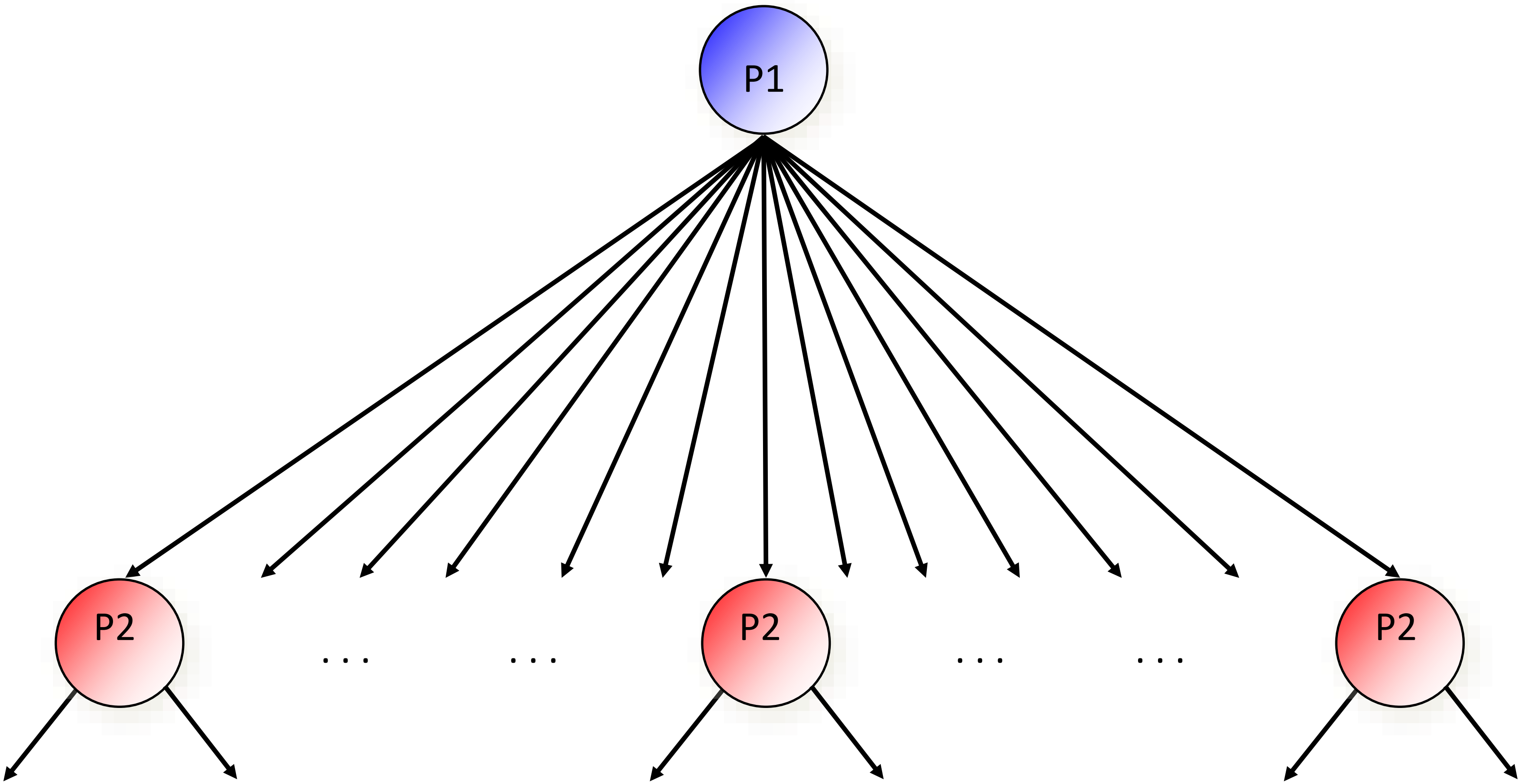
New ideas in subgame solver

NIPS-17 best paper award

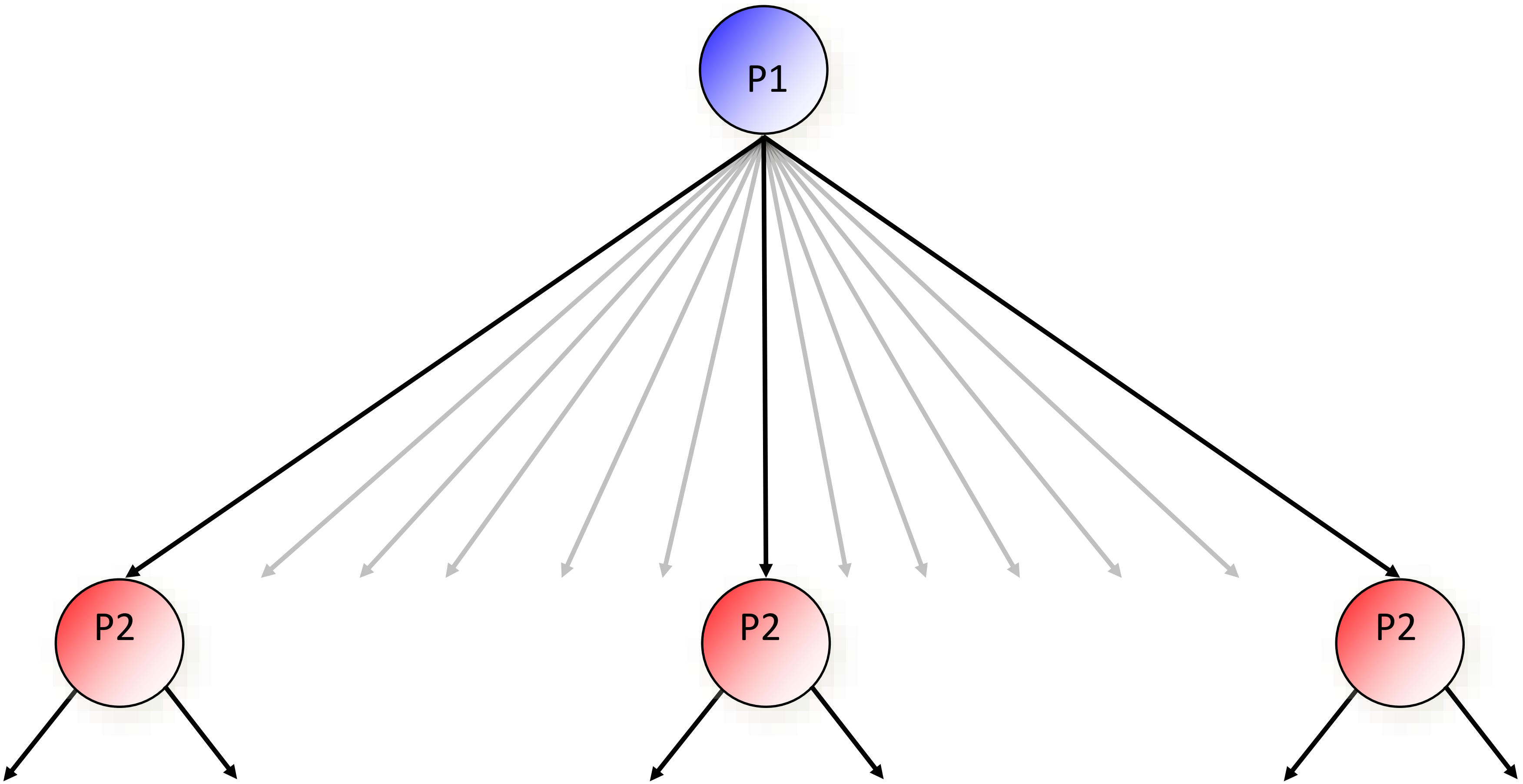
- Provably *safe* subgame solving taking into account opponent's mistakes in the hand so far
- Nested subgame solving
- Subgame solving starts much earlier
- No card abstraction in the subgame
- Changed our action abstraction between hands

New ideas in subgame solver

- Provably *safe* subgame solving taking into account opponent’s mistakes in the hand so far
- Nested subgame solving
- Subgame solving starts much earlier
- No card abstraction in the subgame
- Changed our action abstraction between hands

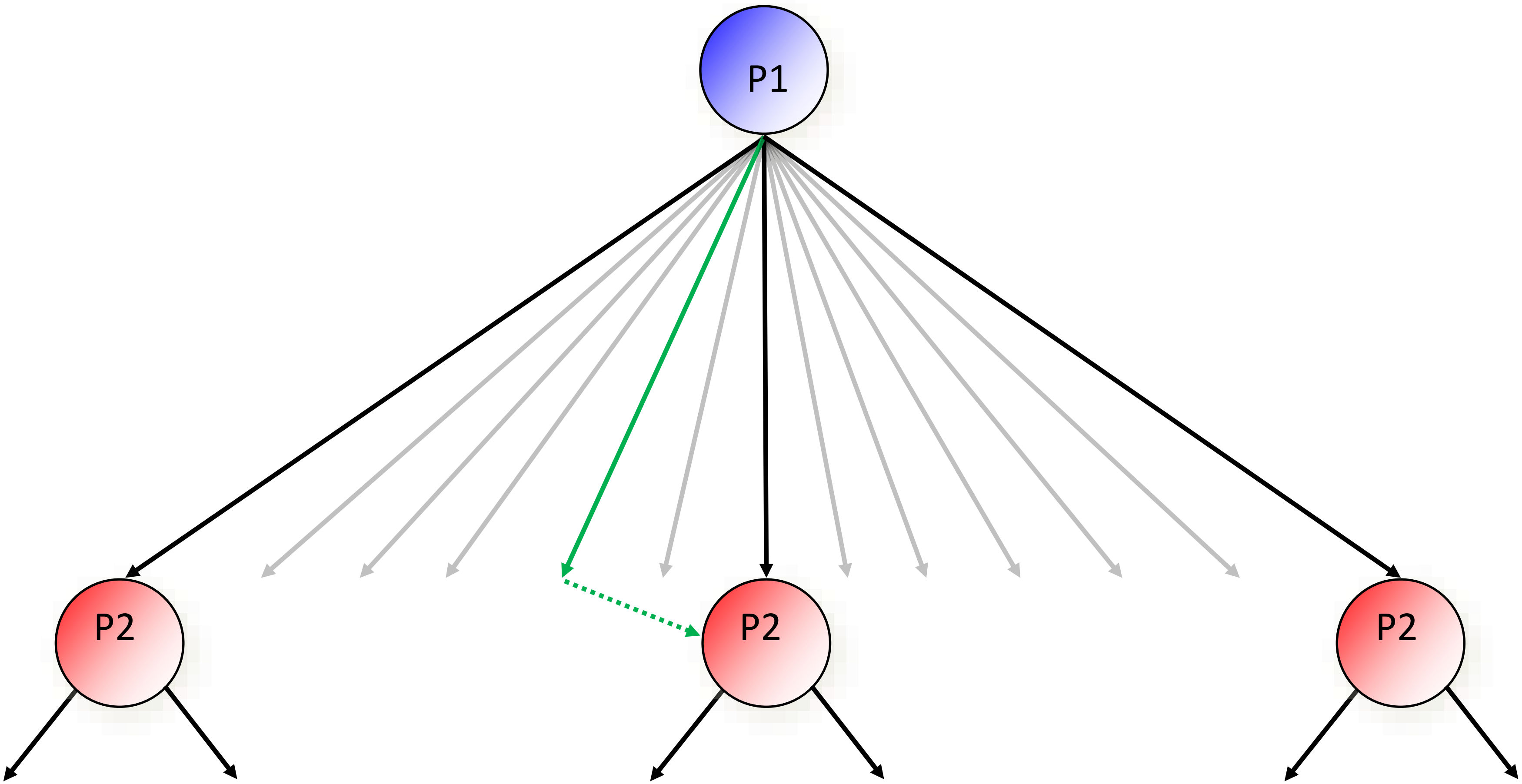

P1
P2
P2
P2
. . .
. . .
. . .
. . .

Action abstraction



[Gilpin, Sandholm & Sørensen, *AAMAS*-08], [Hawkin et al., *AAAI*-11, *AAAI*-12], [Brown & Sandholm, *AAAI*-14]

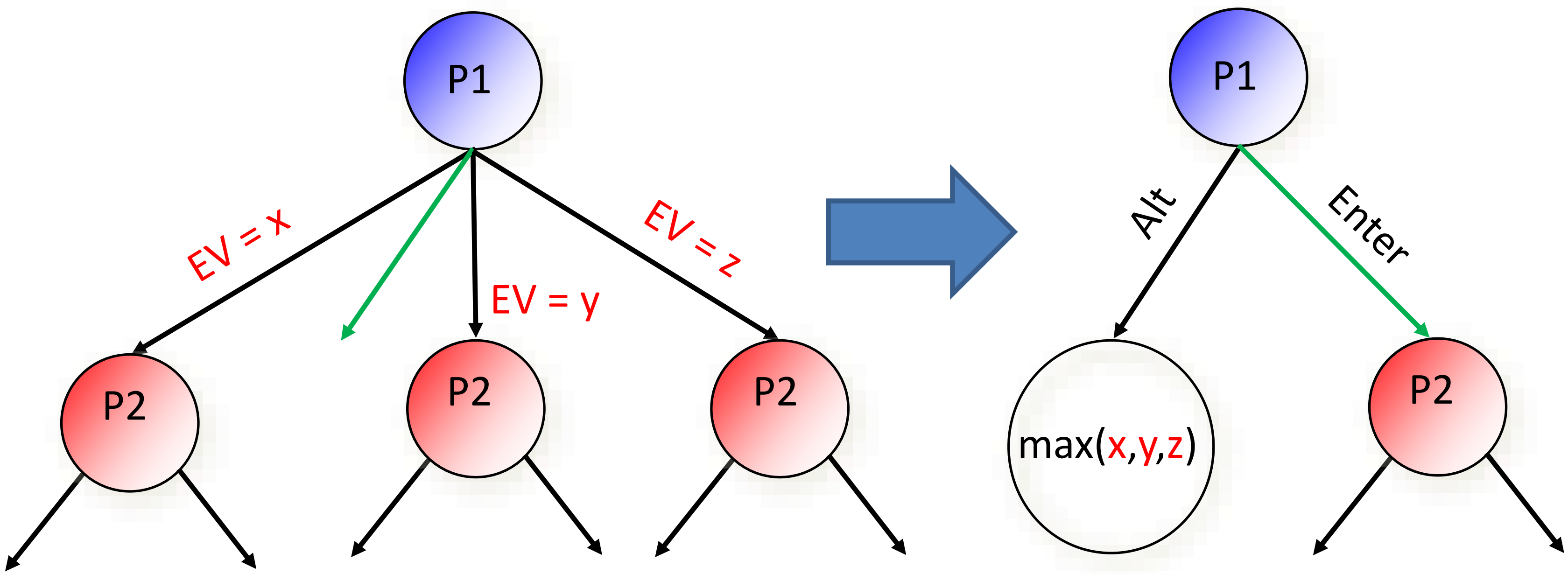
Action translation



[Gilpin, Sandholm & Sørensen, *AAMAS*-08], [Schnizlein et al., *IJCAI*-09], [Ganzfried & Sandholm, *IJCAI*-13]

Nested subgame solving

- Idea: Solve a subgame in real time for the off-tree action taken



- **Theorem.** Say the blueprint is an ε-equilibrium of the abstraction. If EV[Enter] ≤ EV[Alt] for all P1 types, then the strategies form an ε-equilibrium to the new abstraction that includes the new action
- Can be repeated for every subsequent off-tree action (typically in finer and finer abstraction)

[Brown & Sandholm, *AAAI*-17 workshop, *NeurIPS*-17]

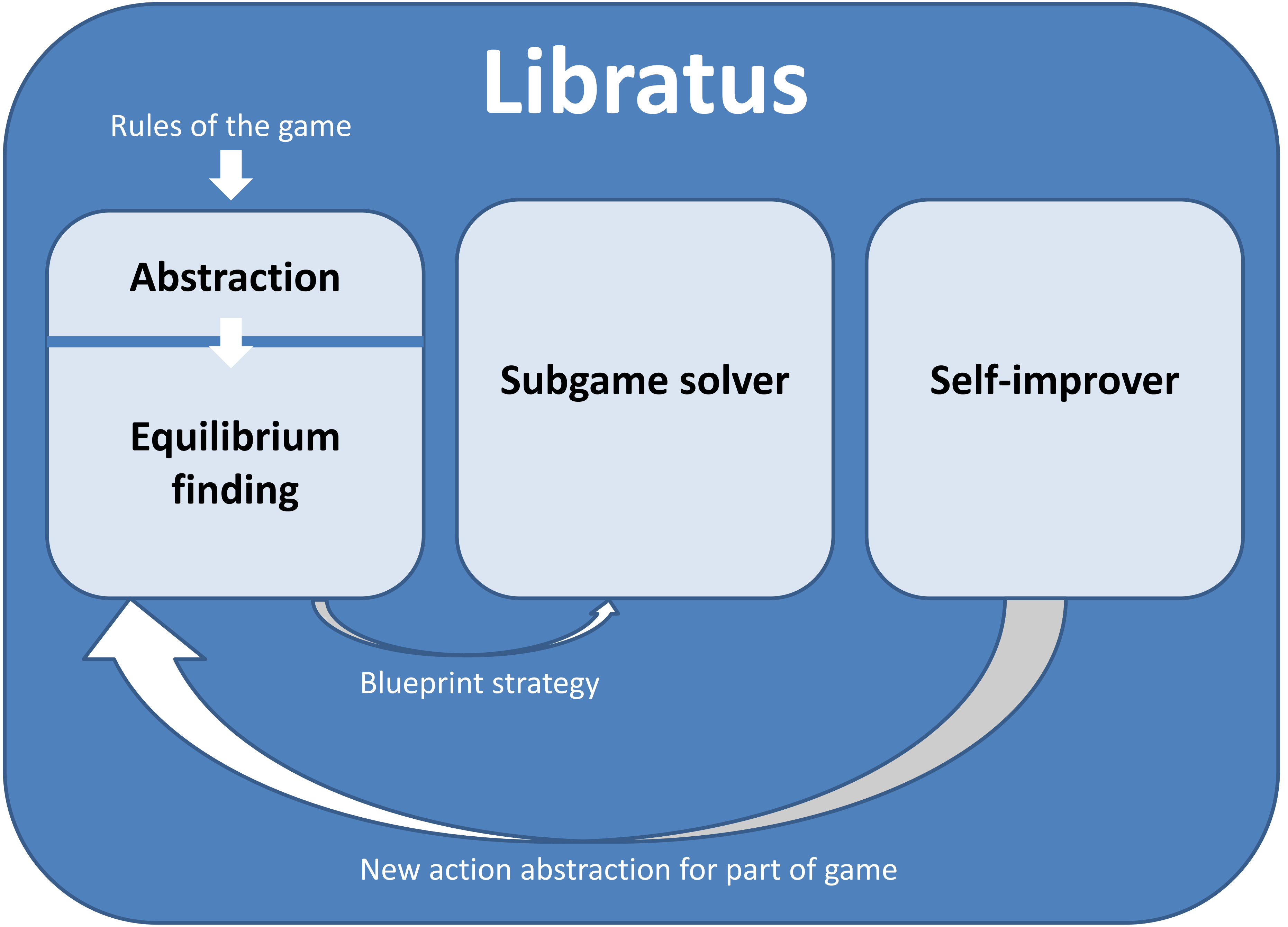
Medium-scale experiments on *nested* subgame solving

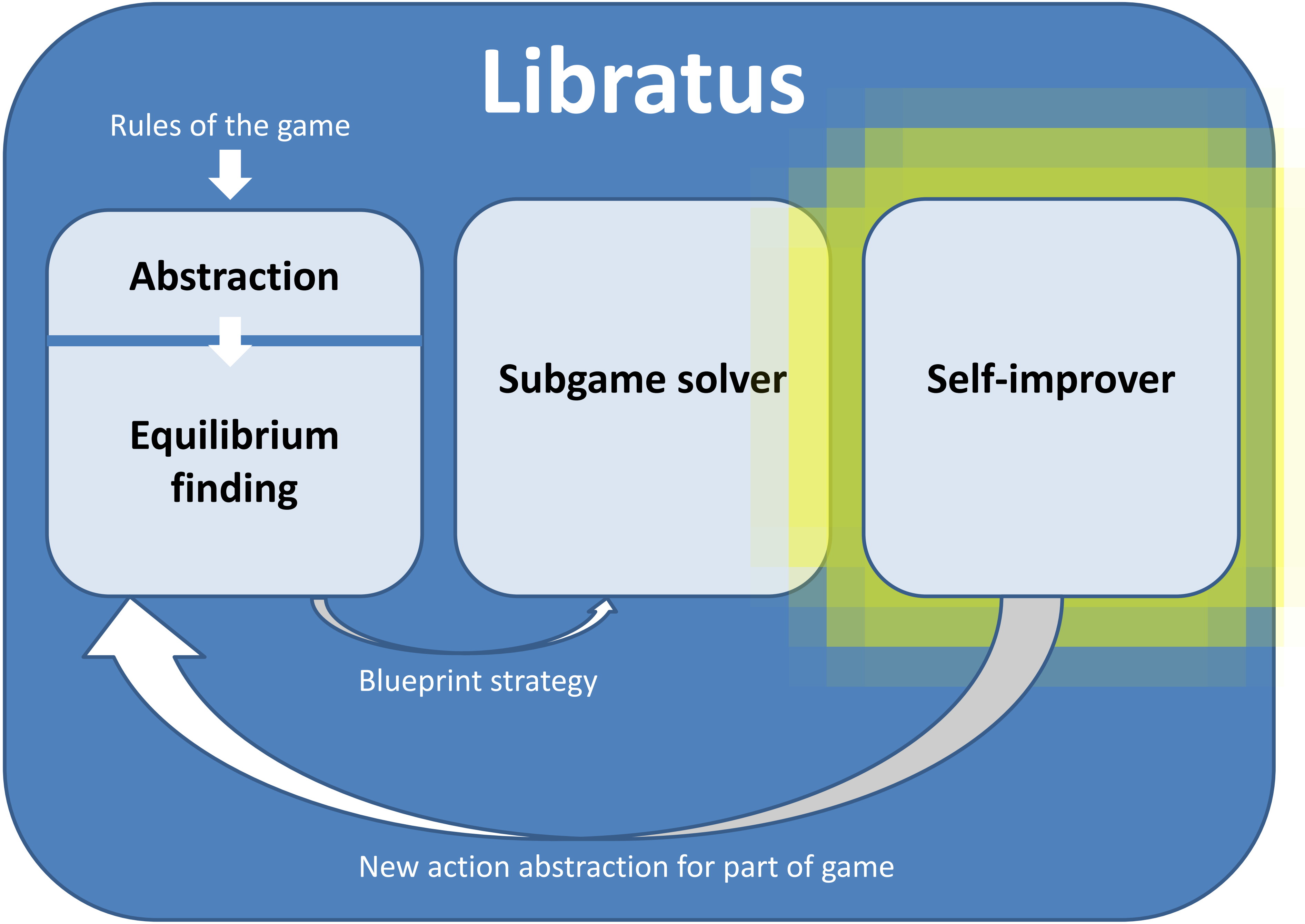
	Exploitability
Randomized Pseudo-Harmonic Mapping [Ganzfried & Sandholm IJCAI-13]	1465 mbb / hand
Nested Re-solve Refinement	**150.2 mbb / hand**
Nested Unsafe Refinement	**148.3 mbb / hand**
Nested Maxmargin Refinement	**122.0 mbb / hand**
Nested Reach-Maxmargin Refinement	**119.1 mbb / hand**

New ideas in subgame solver

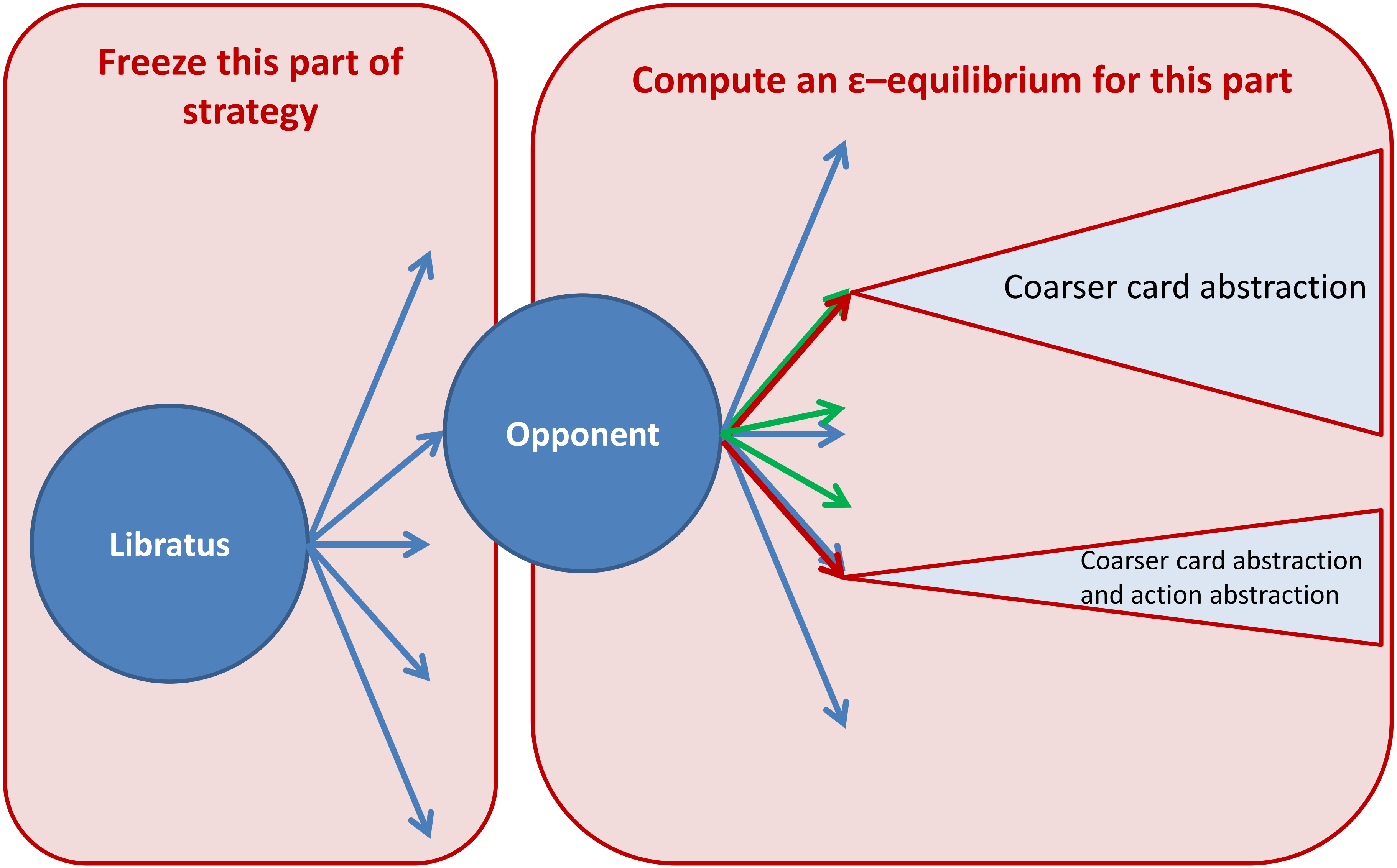
- Provably *safe* subgame solving taking into account opponent's mistakes in the hand so far
- Nested subgame solving
- Subgame solving starts much earlier
- No card abstraction in the subgame
- Changed our action abstraction between hands

Libratus's "balance" and use of "blockers"


Libratus
Rules of the game
Abstraction
Equilibrium finding
Subgame solver
Self-improver
Blueprint strategy
New action abstraction for part of game


Libratus
Rules of the game
Abstraction
Equilibrium finding
Subgame solver
Self-improver
Blueprint strategy
New action abstraction for part of game

Filling holes in the action tree

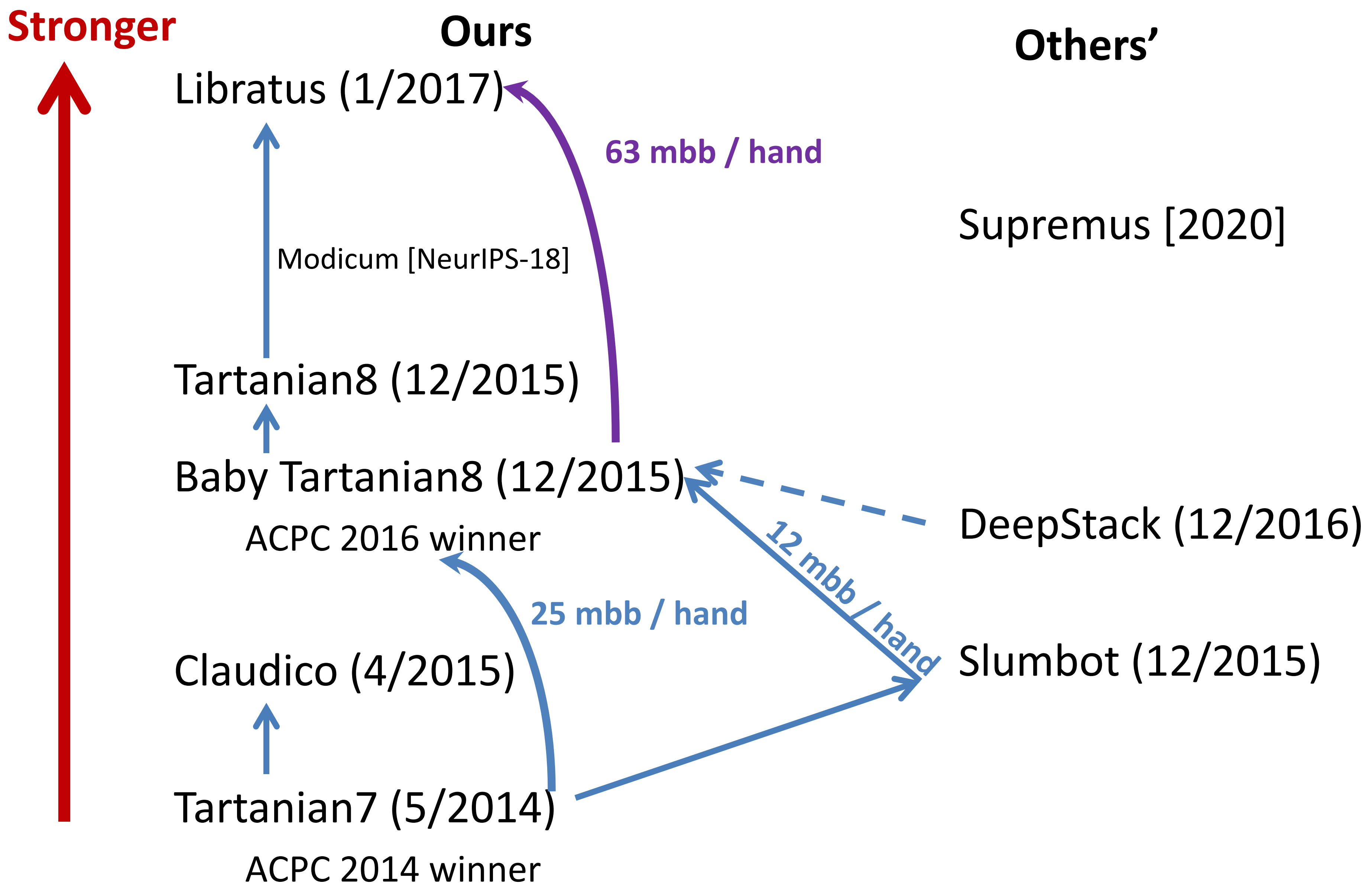


We do this for top *k* holes

Libratus fixing its own weaknesses

Libratus fixing its own weaknesses…

Head-to-head strength of top AIs



Observations about Libratus's play

- Strengths:
  – Small bets & huge bets & huge all-ins
  – Multiple bet sizes in any one situation
  – "Limping", "donk betting"
  – "Perfect balance"
  – Mixed strategy
  – Probability distributions over players' hands; not just "range-based"
  – Near-perfect subgame play; great use of "blockers"
  – Different bet sizings used in subgames
- Weaknesses?
  – No opponent exploitation

Is safe (equilibrium) play timid/boring?