

State-of-the-Art Practical Game Abstraction

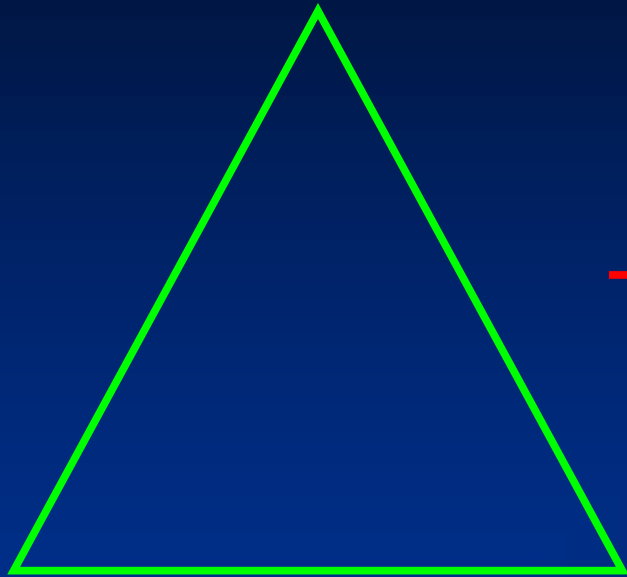
Tuomas Sandholm

Automated game abstraction

[Gilpin & Sandholm, EC-06/*J. of the ACM* 2007, AAAI-06...]

Now used basically by all competitive Texas Hold'em programs

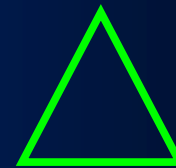
Original game



Abstraction algorithm



Abstracted game



Equilibrium-finding
algorithm



Nash equilibrium

Reverse model



Nash equilibrium

Lossless game abstraction

Information filters

- **Observation:** We can make games smaller by filtering the information a player receives
- Instead of observing a specific signal exactly, a player instead observes a **filtered set** of signals
 - *E.g.*, receiving signal $\{A\spadesuit, A\clubsuit, A\heartsuit, A\diamondsuit\}$ instead of $A\heartsuit$

Signal tree

- Each edge corresponds to the revelation of some signal by nature to at least one player
- Our abstraction algorithm operates on it
 - Doesn't load full game into memory

Isomorphic relation

- Captures the notion of strategic symmetry between nodes
- Defined recursively:
 - Two leaves in signal tree are **isomorphic** if for each action history in the game, the payoff vectors (one payoff per player) are the same
 - Two internal nodes in signal tree are **isomorphic** if their children are isomorphic
 - *Challenge*: permutations of children
 - *Solution*: custom **perfect matching** algorithm between children of the two nodes such that only isomorphic children are matched

Abstraction transformation

- Merges two isomorphic nodes that are siblings
- **Theorem.** *If a strategy profile is a Nash equilibrium in the abstracted (smaller) game, then its interpretation in the original game is a Nash equilibrium*

GameShrink algorithm

- **Bottom-up pass:** Run DP to mark isomorphic pairs of nodes in signal tree
- **Top-down pass:** Starting from top of signal tree, perform the transformation for siblings where applicable
- **Theorem.** Conducts all these transformations
 - $\tilde{O}(n^2)$, where n is #nodes in *signal tree*
 - Usually highly *sublinear* in game tree size

Solved Rhode Island Hold'em poker

- AI challenge problem [Shi & Littman 01]
 - 3.1 billion nodes in game tree
- Without abstraction, LP has 91,224,226 rows and columns => unsolvable
- *GameShrink* runs in one second
- After that, LP has 1,237,238 rows and columns (50,428,638 non-zeros)
- Solved the LP
 - CPLEX *barrier* method took 8 days & 25 GB RAM
- **Exact** Nash equilibrium
- **Largest incomplete-info game solved by then by over 4 orders of magnitude**



Lossy game abstraction

Example game for the rest of this lecture:

Texas hold'em poker



Nature deals 2 cards to each player



Round of betting



Nature deals 3 shared cards



Round of betting



Nature deals 1 shared card



Round of betting



Nature deals 1 shared card



Round of betting

- 2-player Limit has $\sim 10^{18}$ nodes
- 2-player No-Limit has $\sim 10^{165}$ nodes
- Losslessly abstracted game too big to solve
 $\Rightarrow$ abstract more
 $\Rightarrow$ lossy

First abstraction algorithm applied to Texas hold'em [Gilpin & Sandholm, AAAI-06]

- *GameShrink* can be made to abstract more by not requiring a perfect matching => lossy
 - for speed of the matching we used a faster matching heuristic:
 $|wins_{node1} - wins_{node2}| + |losses_{node1} - losses_{node2}| < k$
 - Greedy => lopsided abstractions

Better and more scalable approach for lossy abstraction than GameShrink:

[Gilpin & Sandholm, AAMAS-07]

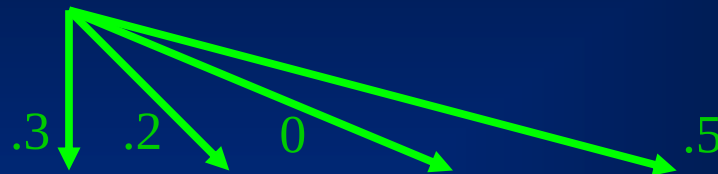
- Operates in signal tree of **one** player's signals & common signals at a time (i.e., no longer in signal tree of both player's signals)
 - This'll be the case also in the state-of-the-art algorithm described later
- “Clustering + IP”:
 - For every betting round i , tell the algorithm how many buckets K_i it is allowed to generate
 - This determines the size of the abstraction, and should be set based on the available computational resources for the equilibrium computation
 - For the first betting round, run **k_1 -means clustering** to bucket the nodes
 - In each later round i , run an **IP** to determine how many children each parent should be allowed to have so the total number of children doesn't exceed K_i
 - The value of allowing a parent to have k children is done by running k -means clustering for different values of k under each parent before running the IP

Potential-aware abstraction

- All prior abstraction algorithms had probability of winning (assuming no more betting) as the similarity metric
 - Doesn't capture *potential*
- Potential not only positive or negative, but “multidimensional”
- We developed an abstraction algorithm that captures potential ...
[Gilpin, Sandholm & Sørensen, AAI-07; Gilpin & Sandholm, AAI-08]

Bottom-up pass to determine abstraction for round 1

Round r-1



Round r



- Clustering using L_1 norm
 - Predetermined number of clusters, depending on size of abstraction we are shooting for
- In the last (4th) round, there is no more potential $\Rightarrow$ we use probability of winning (assuming rollout) as similarity metric

Determining abstraction for round 2

- For each 1st-round bucket i :
 - Make a bottom-up pass to determine 3rd-round buckets, considering only hands compatible with i
 - For $k_i = 1, 2, \dots, \max$
 - Cluster the 2nd-round hands into k_i clusters
 - based on each hand's histogram over 3rd-round buckets
- IP to decide how many children each 1st-round bucket may have, subject to $\sum_i k_i \leq K_2$
 - Error metric for each bucket is the sum of L_2 distances of the hands from the bucket's centroid
 - Total error to minimize is the sum of the buckets' errors
 - weighted by the probability of reaching the bucket

Determining abstraction for round 3

- Done analogously to how we did round 2

Determining abstraction for round 4

- Done analogously, except that now there is no potential left, so clustering is done based on probability of winning (assuming rollout)
- Now the potential-aware abstraction has been computed!

Important ideas for practical lossy abstraction 2007-13

- Integer programming [Gilpin & Sandholm, AAMAS-07]
- Potential-aware [Gilpin, Sandholm & Sørensen, AAI-07; Gilpin & Sandholm, AAI-08]
- Imperfect recall [Waugh et al., SARA-09. Johanson et al., AAMAS-13]

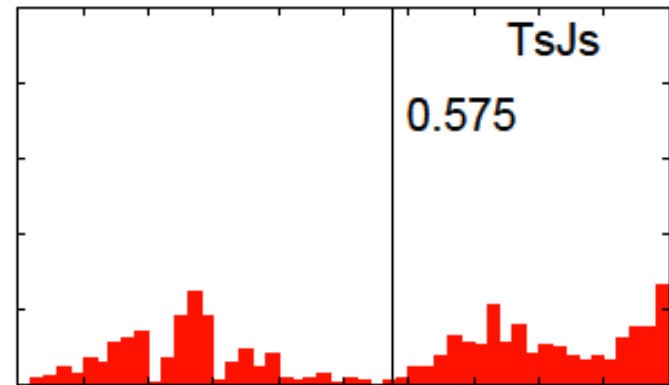
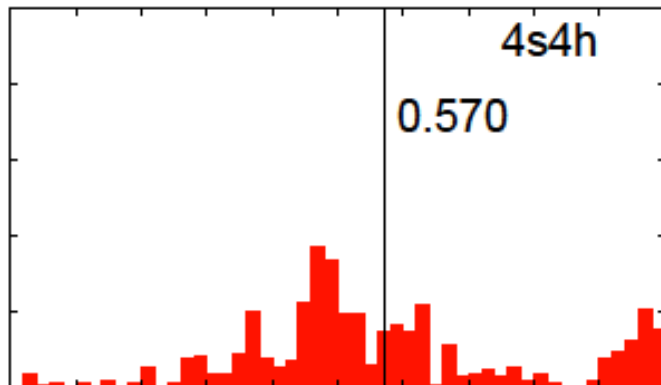
STATE OF THE ART:

Potential-Aware Imperfect-Recall Abstraction
with Earth Mover's Distance in Imperfect-Information Games

[Ganzfried & Sandholm, AAAI-14]

Expected Hand Strength (EHS)

- EHS (aka equity) is the probability of winning (plus $\frac{1}{2}$ x probability of tying)
 - against a uniform random draw of private cards for the opponent,
 - assuming a uniform random rollout of the remaining public cards
- Early poker abstraction approaches used EHS (or EHS exponentiated to some power) to cluster hands [e.g., Billings et al., IJCAI-03; Gilpin & Sandholm, AAAI-06; Zinkevich et al., NIPS-07; Waugh et al., SARA-09]
- EHS fails to account for the **distribution** of hand strength
 - 4s4h and TsJs have very similar EHS (0.575 and 0.570), but 44 frequently has EHS in [0.4,0.6] and rarely in [0.7,0.9], while the reverse is true for TsJs



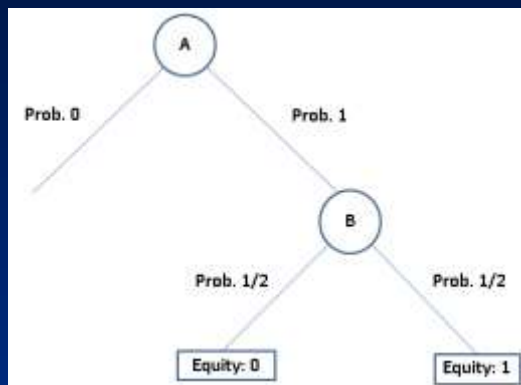
Distribution-aware abstraction

- Takes into account the full distribution of hand strength. Uses earth-mover's distance (EMD) as distance metric between histograms
 - EMD: “minimum cost of turning one pile into the other, where cost is amount of dirt moved times the distance by which it is moved”
- EMD can be computed in linear time for 1D setting, but more challenging in higher dimensions
- Prior best approach used distribution-aware abstraction with imperfect recall for flop and turn rounds. The histograms were over equities after all public cards are dealt (assuming uniform random hand for opponent) [Johanson et al., AAMAS-13]

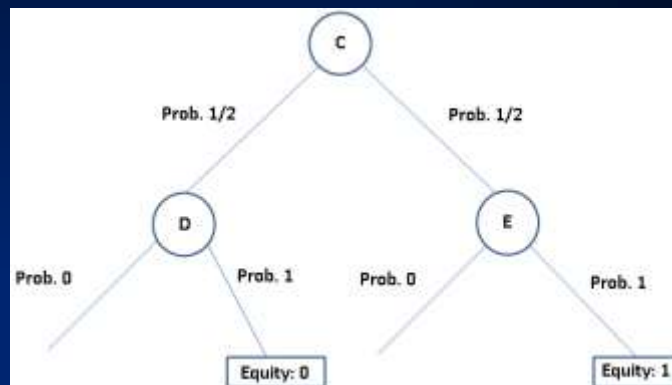
Potential-aware abstraction

- Hands can have very similar distributions over strength at the end, but realize the equity at different ways/rates
- Potential-aware abstraction** [Gilpin, Sandholm & Soerensen, AAAI-07] considers all future rounds, not just final round
- In distribution-aware abstraction, histograms are over cardinal equities
- In potential-aware abstraction, histograms are over non-ordinal next-round states
=> must compute EMD in higher-dimensional space

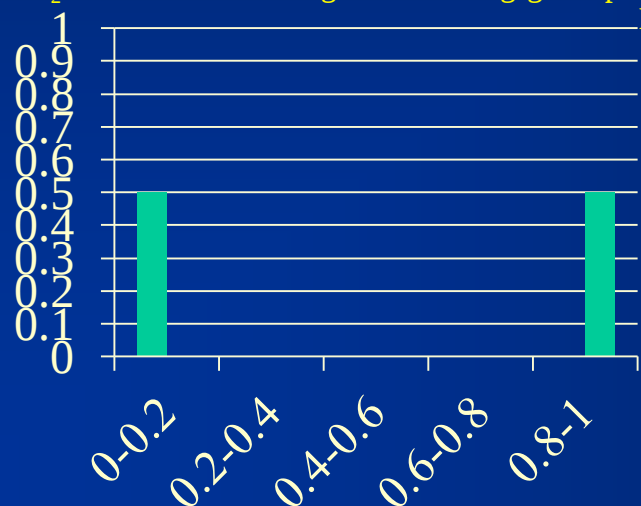
Private signal x_1



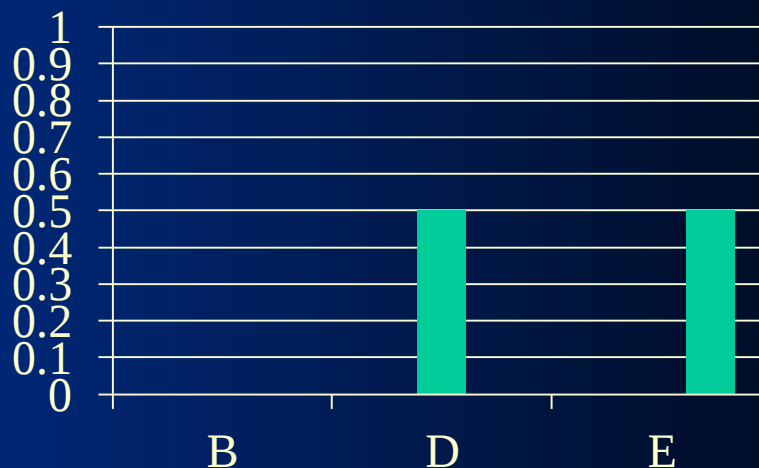
Private signal x_2



x_1 and x_2 have the same histogram assuming game proceeds to the end



Histogram for private signal x_2 at round 1 over non-ordinal information states at round 2



Algorithm for potential-aware imperfect-recall abstraction with EMD

- Perform bottom-up pass of the tree, using histograms over distributions of clusters at next round
 - EMD is now in multi-dimensional space, where the ground distance is assumed to be the (next-round) EMD between the corresponding cluster means
- Best implementation of EMD is far too slow for Texas Hold'em. We developed a fast custom heuristic for approximating it in this setting
- Using our algorithm to compute the abstraction for the flop round, we beat best prior abstraction algorithm
- **Notes:**
 - No need to perform multiple bottom up passes like in potential-aware abstraction before, due to imperfect recall
 - No need for IP, due to imperfect recall

Conclusions

- Domain-independent techniques
- Automated lossless information abstraction: exactly solved 3-billion-node game
- Lossy information abstraction is key to tackling large games like Texas Hold'em. Main progress 2007-2013: integer programming, potential-aware, imperfect recall
- State of the art from our 2014 paper:
 - First information abstraction algorithm that combines potential aware and imperfect recall
- Future research
 - Applying these techniques to other domains
 - Abstraction techniques that have theoretical bounds (discussed next time) and good practical performance