Cumulated Effects in Learning

Érik Martin-Dorel Sergei Soloviev

ACADIE team
IRIT laboratory
Université de Toulouse

30 March 2018

Positioning of this talk

- Starting point: a game theory result, formally verified in Coq
- Methodology: probabilistic analysis of entire classes of games
- Objective: discuss implications of this result, possible generalizations and connection with learning

Outline

1. A Coq theory of bool. games with random formulas as payoff functions
2. Discussion on modeling cumulative effects in learning

Boolean games

- A lot of literature (since 2001)

Boolean games

- A lot of literature (since 2001)
- Particular case of win/lose games (only 2 outcomes):

Boolean games

- A lot of literature (since 2001)
- Particular case of win/lose games (only 2 outcomes):
- Strategies of players are vectors of bits. In a 2-player setting:
  - Alice controls k bits
  - Bob controls $n - k$ bits

Boolean games

- A lot of literature (since 2001)
- Particular case of win/lose games (only 2 outcomes):
- Strategies of players are vectors of bits. In a 2-player setting:
  - Alice controls k bits
  - Bob controls $n-k$ bits

$\rightsquigarrow$ Game represented by a Boolean function $F : \underbrace{\mathbf{2}^k \times \mathbf{2}^{n-k}}_{\simeq\ \mathbf{2}^n} \rightarrow \mathbf{2}$

Boolean games

- A lot of literature (since 2001)
- Particular case of win/lose games (only 2 outcomes):
- Strategies of players are vectors of bits. In a 2-player setting:
  - Alice controls k bits
  - Bob controls $n-k$ bits

$\rightsquigarrow$ Game represented by a Boolean function $F : \underbrace{\mathbf{2}^k \times \mathbf{2}^{n-k}}_{\simeq\, \mathbf{2}^n} \to \mathbf{2}$

- Alice wins (with strat. a) against Bob (with strat. b) iff $F(a,b) = 1$

Methodology

- As part of our project FAGames (Formal analysis of games using ITP):

Methodology

- As part of our project FAGames (Formal analysis of games using ITP):
- Use probability to explore entire classes of games

Methodology

- As part of our project FAGames (Formal analysis of games using ITP):
- Use probability to explore entire classes of games
  - Assume the type of a game is known, but not its parameters in advance
  - Randomly pick a game in the considered class
  - Estimate the probability of various situations (A has a winning strategy, B has a winning strategy, no player has a winning strategy...)

Methodology

- As part of our project FAGames (Formal analysis of games using ITP):
- Use probability to explore entire classes of games
  - Assume the type of a game is known, but not its parameters in advance
  - Randomly pick a game in the considered class
  - Estimate the probability of various situations (A has a winning strategy, B has a winning strategy, no player has a winning strategy...)
- The size of the considered class (number of n-var. Boolean functions) grows very fast (2^{2^n}) $\rightsquigarrow$ exhaustive computation does not scale

Methodology

- As part of our project FAGames (Formal analysis of games using ITP):
- Use probability to explore entire classes of games
  - Assume the type of a game is known, but not its parameters in advance
  - Randomly pick a game in the considered class
  - Estimate the probability of various situations (A has a winning strategy, B has a winning strategy, no player has a winning strategy...)
- The size of the considered class (number of n-var. Boolean functions) grows very fast (2^{2^n}) $\rightsquigarrow$ exhaustive computation does not scale
- Derive symbolic results (that can then be numerically evaluated)

Overview of the library

- Coq library "`RandBoolGames`"
- Based on SSReflect/MathComp (`fintype`, `finfun`, `finset`, `bigop`) as well as on the `infotheo` library [Affeldt et al.]
- 3.1k lines of Coq code – automation: introduce a new tactic "`under`"
- https://sourcesup.renater.fr/coq-bool-games/

Probability setting

- Focus on $\Omega := \mathbf{2}^{\mathbf{2}^n}$ and $\mathcal{S} := \mathcal{P}(\Omega) = \mathbf{2}^{\mathbf{2}^{\mathbf{2}^n}}$

Probability setting

- Focus on $\Omega := \mathbf{2}^{\mathbf{2}^n}$ and $\mathcal{S} := \mathcal{P}(\Omega) = \mathbf{2}^{\mathbf{2}^{\mathbf{2}^n}}$
- Which probability distribution?

Probability setting

- Focus on $\Omega := \mathbf{2}^{\mathbf{2}^n}$ and $\mathcal{S} := \mathcal{P}(\Omega) = \mathbf{2}^{\mathbf{2}^{\mathbf{2}^n}}$
- Which probability distribution?
- Prior choice: assign proba. to bool. functions or to bool. formulas?

Probability setting

- Focus on $\Omega := \mathbf{2}^{\mathbf{2}^n}$ and $\mathcal{S} := \mathcal{P}(\Omega) = \mathbf{2}^{\mathbf{2}^{\mathbf{2}^n}}$
- Which probability distribution?
- Prior choice: assign proba. to bool. functions or to bool. formulas?

Probability setting

- Focus on $\Omega := \mathbf{2}^{\mathbf{2}^n}$ and $\mathcal{S} := \mathcal{P}(\Omega) = \mathbf{2}^{\mathbf{2}^{\mathbf{2}^n}}$
- Which probability distribution?
- Prior choice: assign proba. to bool. functions or to bool. formulas?
- First step: consider any $\mathbb{P} : \mathcal{S} \to [0, 1]$

First result

Theorem (Pr_ex_winA)

For any finite probability space $(\Omega, \mathcal{S}, \mathbb{P})$, the probability that there exists some strategy $a = (a_1, \ldots, a_k)$ of A that is winning satisfies:

$$\mathbb{P}(\exists a.\, \mathrm{win}_A(a)) = \sum_{m=1}^{2^k} (-1)^{m-1} \sum_{\substack{J \subseteq \mathbf{2}^k \\ \mathrm{Card}\, J = m}} \mathbb{P}\left(\bigcap_{a \in J} W_a \right),$$

denoting for any $a \in \mathbf{2}^k$,

$$\mathrm{win}_A(a) := \forall b \in \mathbf{2}^{n-k}.\ F(a,b) = 1$$
$$\omega_a := \{v \in \mathbf{2}^n \mid v_1 = a_1 \wedge \cdots \wedge v_k = a_k\} \in \Omega$$
$$W_a := \{\omega \in \Omega \mid \omega_a \subseteq \omega\} \in \mathcal{S}.$$

Specialization of the probability setting

- We define $\mathbb{P}_{n;p}$ as a finite Bernoulli process:
  - we construct bool. functions F from the truth-set $F^{-1}(\{1\})$ as follows:

Specialization of the probability setting

- We define $\mathbb{P}_{n;p}$ as a finite Bernoulli process:
  - we construct bool. functions F from the truth-set $F^{-1}(\{1\})$ as follows:
  - we consider all vectors $v \in \mathbf{2}^n$
  - for each v, we decide with proba. p if it belongs to the truth-set of F

  $\rightsquigarrow$ 2^n independent Bernoulli trials with parameter p

Specialization of the probability setting

- We define $\mathbb{P}_{n;p}$ as a finite Bernoulli process:
  - we construct bool. functions F from the truth-set $F^{-1}(\{1\})$ as follows:
  - we consider all vectors $v \in \mathbf{2}^n$
  - for each v, we decide with proba. p if it belongs to the truth-set of F

  $\rightsquigarrow$ 2^n independent Bernoulli trials with parameter p

Remark

This setting subsumes the simpler case where all functions have the same elementary probability (just take $p = \frac{1}{2}$)

Results I

Theorem (Pr_ex_winA_Bern)

For all $p \in [0,1]$ and for all integers n, k, the probability that player A has a winning strategy is:

$$\mathbb{P}_{n;\,p}(\exists a \in \mathbf{2}^k.\ \mathrm{win}_A(a)) = 1 - \left(1 - p^{2^{n-k}}\right)^{2^k}.$$

Results I

Theorem (Pr_ex_winA_Bern)

For all $p \in [0,1]$ and for all integers n, k, the probability that player A has a winning strategy is:

$$\mathbb{P}_{n;\,p}(\exists a \in \mathbf{2}^k.\ \mathrm{win}_A(a)) = 1 - \left(1 - p^{2^{n-k}}\right)^{2^k}.$$

Proba. that a winning strategy exists neither for A nor for B ($n = 10$)

p\k	1	2	3	4	5	6	7	8	9
0.25	1.52e-184	5.11e-43	1.37e-6	**0.525**	**0.997**	**≈1**	**0.998**	0.367	4.46e-15
0.5	1.07e-64	6.68e-8	**0.606**	**0.999**	**≈1**	**0.999**	**0.606**	6.68e-8	1.07e-64
0.75	4.46e-15	0.367	**0.998**	**≈1**	**0.997**	**0.525**	1.37e-6	5.11e-43	1.52e-184

(**bold values** > 0.5)

Results I

Theorem (Pr_ex_winA_Bern)

For all $p \in [0,1]$ and for all integers n, k, the probability that player A has a winning strategy is:

$$\mathbb{P}_{n;\,p}(\exists a \in \mathbf{2}^k.\ \mathrm{win}_A(a)) = 1 - \left(1 - p^{2^{n-k}}\right)^{2^k}.$$

Proba. that a winning strategy exists neither for A nor for B ($n = 10$)

p\k	1	2	3	4	5	6	7	8	9
0.25	1.52e-184	5.11e-43	1.37e-6	**0.525**	**0.997**	**≈1**	**0.998**	0.367	4.46e-15
0.5	1.07e-64	6.68e-8	**0.606**	**0.999**	**≈1**	**0.999**	**0.606**	6.68e-8	1.07e-64
0.75	4.46e-15	0.367	**0.998**	**≈1**	**0.997**	**0.525**	1.37e-6	5.11e-43	1.52e-184

(**bold values** > 0.5)

The order of moves matters: no winning strat $\wedge$ B plays first $\Rightarrow$ A can always win!
But what happens if player A just knows a few bits from player B's strategy?

Results II

Theorem (Pr_ex_winA_knowing_Bern)

For all $p \in [0,1]$ and for all integers n, k, s satisfying $0 \leq s \leq n-k \leq n$, the probability of guaranteed win for player A knowing s choices of player B among his $n-k$ variables is:

$$\mathbb{P}_{n;p}\left(\forall b_{1:s} \in \mathbf{2}^s.\ \exists a \in \mathbf{2}^k.\ \mathrm{win}_A(a \mid b_{1:s})\right) = \underbrace{\left(1-\left(1-p^{2^{n-k-s}}\right)^{2^k}\right)^{2^s}}_{g(s)}.$$

Results II

Theorem (Pr_ex_winA_knowing_Bern)

For all $p \in [0,1]$ *and for all integers* n, k, s *satisfying* $0 \leq s \leq n-k \leq n$, *the probability of guaranteed win for player* A *knowing* s *choices of player* B *among his* $n-k$ *variables is:*

$$\mathbb{P}_{n;p}\left(\forall b_{1:s} \in \mathbf{2}^s.\ \exists a \in \mathbf{2}^k.\ \mathrm{win}_A(a \mid b_{1:s})\right) = \underbrace{\left(1-\left(1-p^{2^{n-k-s}}\right)^{2^k}\right)^{2^s}}_{g(s)}.$$

Theorem (phi_ineq)

For any $p \in (0,1)$, $n,k \in \mathbb{N}^*$ *such that* $0 \leq s \leq n-k$, *if*

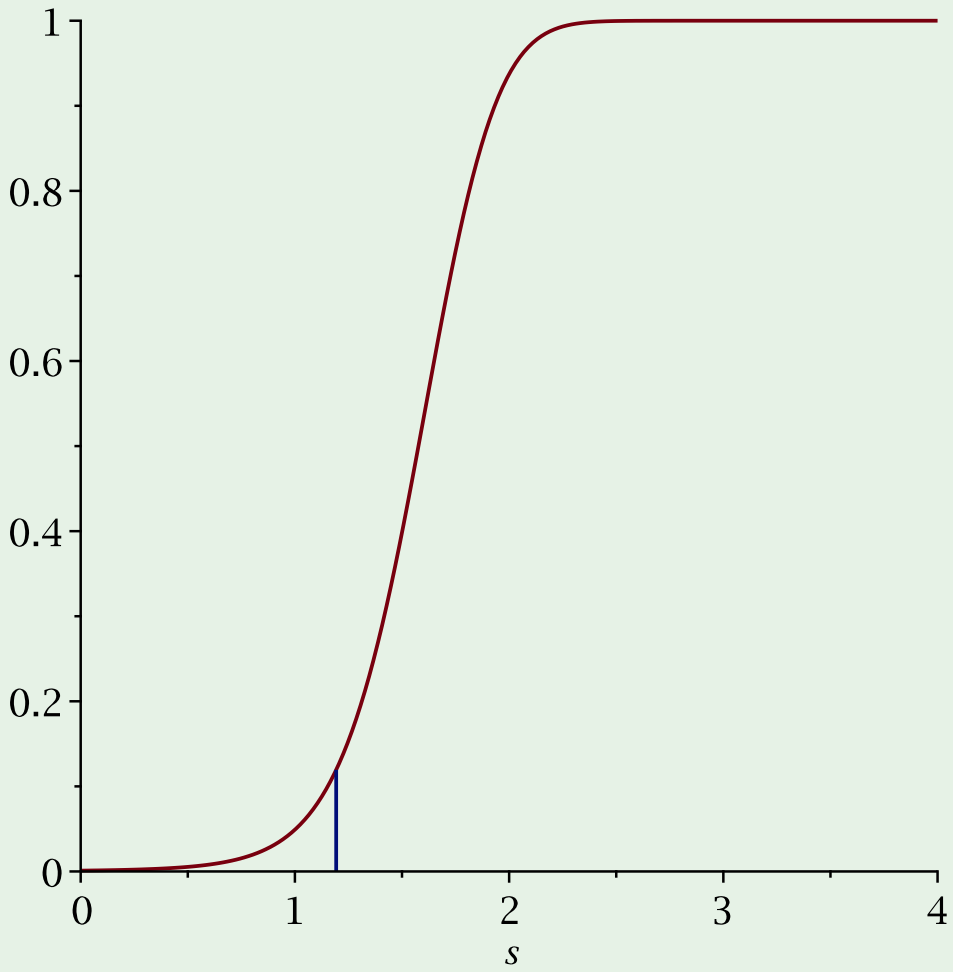
$$s \leq (n-k) - \log_2(k+1) + \log_2(|\log_2 p|) \tag{1}$$

then

$$\phi(s) := g(s) - g(0) > \left(2^{(k-1)2^s} - 2^k\right) p^{2^{n-k}}.$$

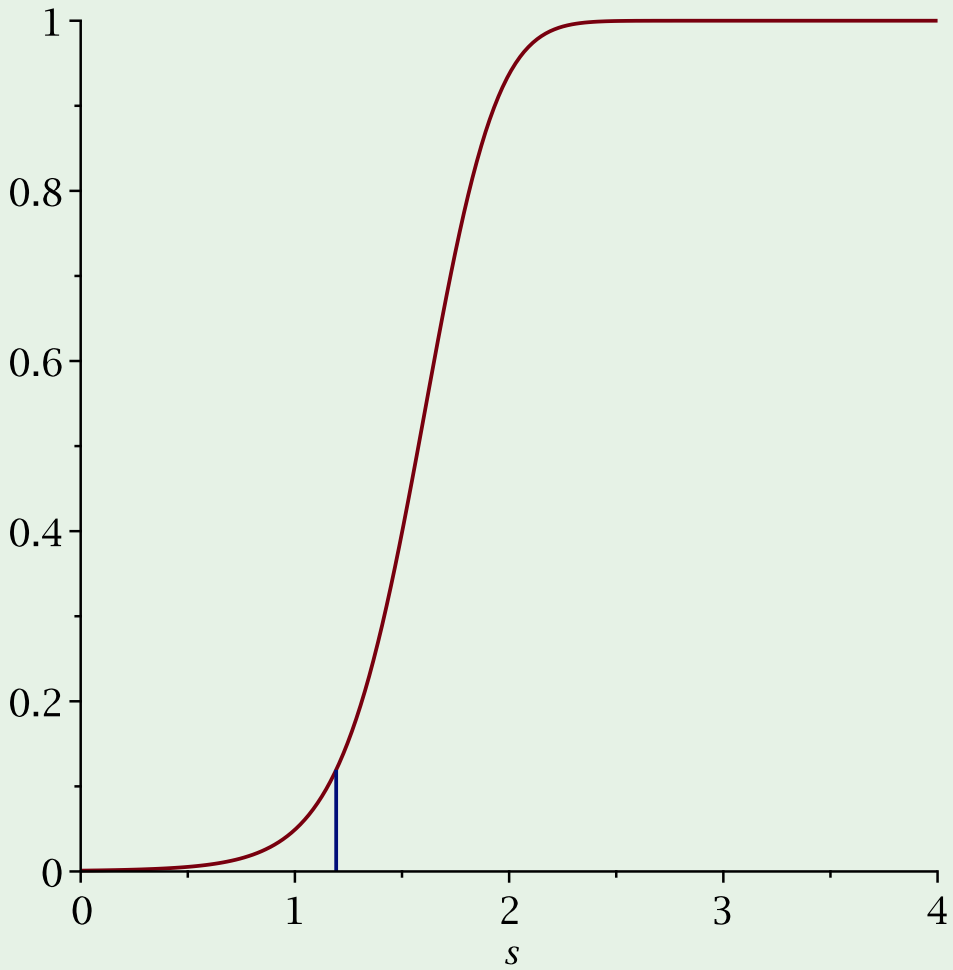
Growth rate of guaranteed win w.r.t knowledge on player B

Example (Graph of $g(s)$ for parameters $p = \frac{1}{2}$, $n = 10$, and $k = 6$)



Growth rate of guaranteed win w.r.t knowledge on player B

Example (Graph of $g(s)$ for parameters $p = \frac{1}{2}$, $n = 10$, and $k = 6$)



The blue vertical line indicates the largest $s \in \mathbb{R}$ satisfying (1)

Wrap-up

1. (Quantitative arg. that) the order of moves matters in Boolean games

Wrap-up

1. (Quantitative arg. that) the order of moves matters in Boolean games
2. Difference between *the information required to win* and *full knowledge on the opponent's strategies*

Wrap-up

1. (Quantitative arg. that) the order of moves matters in Boolean games
2. Difference between *the information required to win* and *full knowledge on the opponent's strategies*
3. The probability of guaranteed win grows much faster than usual 2^s where s denotes the quantity of information (number of extra bits) known by the player

Wrap-up

1. (Quantitative arg. that) the order of moves matters in Boolean games
2. Difference between *the information required to win* and *full knowledge on the opponent's strategies*
3. The probability of guaranteed win grows much faster than usual 2^s where s denotes the quantity of information (number of extra bits) known by the player

$\rightsquigarrow$ can this be related to other phenomena as well as to learning?

Can we learn Boolean functions?

Let $\mathcal{C}$ the concept class of n-var. Boolean functions.
Several approaches (all related to supervised learning):

Can we learn Boolean functions?

Let $\mathcal{C}$ the concept class of n-var. Boolean functions.
Several approaches (all related to supervised learning):

PAC learning [Valiant]

- $X = \mathbf{2}^n$, $\mathcal{C} = (X \to \mathbf{2})$, $\mathcal{D}$ distribution over X
- unknown $c \in \mathcal{C}$ (target concept)
- access to data points $(x, c(x))$ (for random samples $x \in X$ w.r.t. $\mathcal{D}$)
- specification of the "computational complexity" of learning

Can we learn Boolean functions?

Let $\mathcal{C}$ the concept class of n-var. Boolean functions.
Several approaches (all related to supervised learning):

PAC learning [Valiant]

- $X = \mathbf{2}^n$, $\mathcal{C} = (X \to \mathbf{2})$, $\mathcal{D}$ distribution over X
- unknown $c \in \mathcal{C}$ (target concept)
- access to data points $(x, c(x))$ (for random samples $x \in X$ w.r.t. $\mathcal{D}$)
- specification of the "computational complexity" of learning

Active learning – membership queries – ACRE learning [Lowd et al.]

- $X = \mathbf{2}^n$, $\mathcal{C} = (X \to \mathbf{2})$
- access to black-box fun. $c \in \mathcal{C}$, $a : X \to \mathbb{R}_+$ (adversarial cost fun.) and $x_+, x_- \in X$ s.t. $c(x_+) = 1$ and $c(x_-) = 0$
- (efficiently) find $x_0 \in X$ s.t. $c(x_0) = 0$ and $a(x_0) = \min_{x|c(x)=0} a(x)$

Possible extensions of our approach

- Let us notate $\mathrm{BF}(n) = (\mathbf{2}^n \to \mathbf{2})$ and $\mathrm{BG}(n;k) = (\mathbf{2}^k \times \mathbf{2}^{n-k} \to \mathbf{2})$

Possible extensions of our approach

- Let us notate $\mathrm{BF}(n) = (\mathbf{2}^n \to \mathbf{2})$ and $\mathrm{BG}(n;k) = (\mathbf{2}^k \times \mathbf{2}^{n-k} \to \mathbf{2})$
- Several possible scenarios:
  - $F \in \mathrm{BF}(n)$ is fixed
    1. Players A and B pick a strategy $x = (a, b)$ and evaluate $F(x)$
    2. They memorize the datum $(x, F(x))$
    3. Repeat at step 1 with the same function and try to guess F

    $\rightsquigarrow$ no adversarial context!

Possible extensions of our approach

- Let us notate $\mathrm{BF}(n) = (\mathbf{2}^n \to \mathbf{2})$ and $\mathrm{BG}(n;k) = (\mathbf{2}^k \times \mathbf{2}^{n-k} \to \mathbf{2})$
- Several possible scenarios:
  - $F \in \mathrm{BF}(n)$ is fixed
    1. Players A and B pick a strategy $x = (a, b)$ and evaluate $F(x)$
    2. They memorize the datum $(x, F(x))$
    3. Repeat at step 1 with the same function and try to guess F

    $\rightsquigarrow$ no adversarial context!
  - Player A plays repeatedly against player B with random $F \in \mathrm{BG}(n;k)$
    1. Players A and B choose a strategy (a, b) and evaluate $F(a, b)$
    2. Player A memorizes (part of) the profile strategy and the outcome
    3. Repeat at step 1 with (another) function F and try to get $F(a, b) = 1$

    $\rightsquigarrow$ what are the conditions on info./choices to make the scenario realistic?

Ignorance scenario

- Consider a large population where all members play simultaneously a game against some opponent, called Ignorance (e.g. each plays some Boolean game with a randomly selected formula of some class)

Ignorance scenario

- Consider a large population where all members play simultaneously a game against some opponent, called Ignorance (e.g. each plays some Boolean game with a randomly selected formula of some class)
- Ignorance may always choose an optimal strategy at his side, he has an unlimited computational power.

Ignorance scenario

- Consider a large population where all members play simultaneously a game against some opponent, called Ignorance (e.g. each plays some Boolean game with a randomly selected formula of some class)
- Ignorance may always choose an optimal strategy at his side, he has an unlimited computational power.
- But he does not know the choices of the members of the population.

Ignorance scenario

- Consider a large population where all members play simultaneously a game against some opponent, called Ignorance (e.g. each plays some Boolean game with a randomly selected formula of some class)
- Ignorance may always choose an optimal strategy at his side, he has an unlimited computational power.
- But he does not know the choices of the members of the population.
- Some of them still may win (they may even have a universal winning strategy in their personal game)

Ignorance scenario

- Consider a large population where all members play simultaneously a game against some opponent, called Ignorance (e.g. each plays some Boolean game with a randomly selected formula of some class)
- Ignorance may always choose an optimal strategy at his side, he has an unlimited computational power.
- But he does not know the choices of the members of the population.
- Some of them still may win (they may even have a universal winning strategy in their personal game)
- Assume now that those who win pass to the next level and at this level, may know in advance one bit of the strategies chosen by Ignorance against themselves, and moreover, may help others, for example by telling their friends (defined by some relation) one bit of the strategies that Ignorance plays against them.

Ignorance scenario

- Consider a large population where all members play simultaneously a game against some opponent, called Ignorance (e.g. each plays some Boolean game with a randomly selected formula of some class)
- Ignorance may always choose an optimal strategy at his side, he has an unlimited computational power.
- But he does not know the choices of the members of the population.
- Some of them still may win (they may even have a universal winning strategy in their personal game)
- Assume now that those who win pass to the next level and at this level, may know in advance one bit of the strategies chosen by Ignorance against themselves, and moreover, may help others, for example by telling their friends (defined by some relation) one bit of the strategies that Ignorance plays against them.
- This will increase their probability to win. The win will then let them pass to the next level, and so on.

Collective learning and modeling distributed ITP

- Proving mathematics results: a one-player game?

Collective learning and modeling distributed ITP

- Proving mathematics results: a one-player game?
- Interaction with a tactic-based ITP: roughly

```
type tactic = goal -> (goal list) option
```

Collective learning and modeling distributed ITP

- Proving mathematics results: a one-player game?
- Interaction with a tactic-based ITP: roughly

```
type tactic = goal -> (goal list) option
```

⇝ Doesn't fit "as is" in our Boolean games framework

Collective learning and modeling distributed ITP

- Proving mathematics results: a one-player game?
- Interaction with a tactic-based ITP: roughly
  `type tactic = goal -> (goal list) option`

⇝ Doesn't fit "as is" in our Boolean games framework

- Consider instead `eval : goal -> tactics -> bool` to evaluate the "proof strategies" down to the "leafs" (possibly using ATPs)?

Collective learning and modeling distributed ITP

- Proving mathematics results: a one-player game?
- Interaction with a tactic-based ITP: roughly
  `type tactic = goal -> (goal list) option`

⇝ Doesn't fit "as is" in our Boolean games framework

- Consider instead `eval : goal -> tactics -> bool` to evaluate the "proof strategies" down to the "leafs" (possibly using ATPs)?
- In this context we might want to have a player whose role is dedicated to state definitions and conjecture properties?

Collective learning and modeling distributed ITP

- Proving mathematics results: a one-player game?
- Interaction with a tactic-based ITP: roughly
  `type tactic = goal -> (goal list) option`

⇝ Doesn't fit "as is" in our Boolean games framework

- Consider instead `eval : goal -> tactics -> bool` to evaluate the "proof strategies" down to the "leafs" (possibly using ATPs)?
- In this context we might want to have a player whose role is dedicated to state definitions and conjecture properties?

⇝ Analogy with "developer" and "tester" roles in software engineering!

Collective learning and modeling distributed ITP

- Proving mathematics results: a one-player game?
- Interaction with a tactic-based ITP: roughly
  `type tactic = goal -> (goal list) option`

⇝ Doesn't fit "as is" in our Boolean games framework

- Consider instead `eval : goal -> tactics -> bool` to evaluate the "proof strategies" down to the "leafs" (possibly using ATPs)?
- In this context we might want to have a player whose role is dedicated to state definitions and conjecture properties?

⇝ Analogy with "developer" and "tester" roles in software engineering!

- If one aims to get a distributed ITP platform that scales as much as S.O., the "social" interactions (discussions/reward/reputation/etc.) as well as the user-friendliness of the interface are maybe as important as the "logical" aspects.

Conclusion and perspectives

Wrap-up

- A Coq/SSReflect library of random Boolean games
- Use probability to study properties on an entire class of games
- Winning $\neq$ Learning

Conclusion and perspectives

Wrap-up

- A Coq/SSReflect library of random Boolean games
- Use probability to study properties on an entire class of games
- Winning $\neq$ Learning

Observation

Small changes in probabilistic parameters may have huge effects in cases of iterated events · queues · phase transitions · chain reactions · positive feedback... and certainly in other domains?

Conclusion and perspectives

Wrap-up

- A Coq/SSReflect library of random Boolean games
- Use probability to study properties on an entire class of games
- Winning $\neq$ Learning

Observation

Small changes in probabilistic parameters may have huge effects in cases of iterated events · queues · phase transitions · chain reactions · positive feedback... and certainly in other domains?

Perspectives

- Refine the model sketched in this talk $\rightsquigarrow$ amenable to formal methods?
- Consider other distributions/methods for constructing Boolean games
- Consider more general settings such as n-player Boolean games
- Strengthen the "`under`" tactic and propose to integrate it in MathComp