Learning equilibria in monoidal computer

(5 dubious statements about outsmarting)

Dusko Pavlovic
University of Hawaii

LAP 2016
Dubrovnik, September 2016

Outline

Outline

Dubious Statement 1

Dubious Statement 2

Dubious Statement 3

Dubious Statement 4

Dubious Statement 5

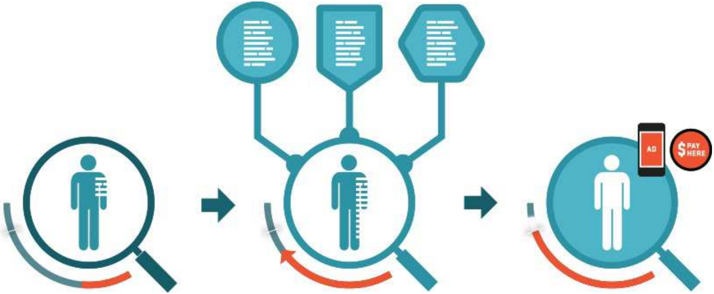
Is there a logic of social behaviors?

Is there a logic of social behaviors?

And the herd of swine rushed towards the steep bank and threw themselves in the sea.

Matthew 8:30-37
Mark 5:1-20
Luke 8:27-38

Behaviors are counted, predicted and sold



Artificial Intelligence has been achieved

Not just computers behaving like people
but also people behaving like computers

Working hypotheses

- math of social interactions
  = game theory

Working hypotheses

- math of social interactions
  = game theory

- people $\subseteq$ computers
  $\Longrightarrow$ social behaviors $\subseteq$ computations

Working hypotheses

- math of social interactions
  = game theory

- people $\subseteq$ computers
  $\implies$ social behaviors $\subseteq$ computations

- **logic of social behaviors**
  = game theory + computation

Trouble: game theory + computation = 

- simple games have
  complex strategies
  - Rabin (1957): BR undecidable
  - Blass (1972): BR at any hyperarithmetic level

Trouble: game theory + computation = 

- simple games and regular (FSM implemented) strategies have
  complex strategy search space
  - Gilboa, Ben-Porath (1980s): BR search unfeasible
  - Nachbar (1990s): equilibrium undecidable

Every problem is a game

Every problem is a game

Problem (Landau)

Are there infinitely many n such that $n^2 + 1$ is prime?

Every problem is a game

Game

- Alice picks a_1.
- Bob picks b_1
- Alice picks a_2.
- Bob picks b_2
- Alice picks a_3.
- Alice wins if $(a_1 + b_1)^2 + 1 = (a_2 + 2)(a_3 + 2)$

Every problem is a game

Strategies

- If n is the greatest number with $n^2 + 1$ prime, then Alice picks $a_1 > p$,
  — and wins.

Every problem is a game

Strategies

- If n is the greatest number with $n^2 + 1$ prime, then Alice picks $a_1 > p$, — and wins.
- If there are infinitely many primes $p = n^2 + 1$, then Bob picks b_1 so that $(a_1 + b_1)^2 + 1$ is prime, — and wins.

Every problem is a game

Strategies

- If n is the greatest number with $n^2 + 1$ prime, then Alice picks $a_1 > p$,
  — and wins.
- If there are infinitely many primes $p = n^2 + 1$, then Bob picks b_1 so that $(a_1 + b_1)^2 + 1$ is prime,
  — and wins.
- Since Landau's Problem is open, **we don't know who wins**, or **how hard it is to win.**

Dubious Statement 1

Every problem is a game.

Corollary

- Finding simple games with hard strategies
  is like finding simple statements with hard proofs.
  - It shouldn't stop us.

Corollary

- Finding simple games with hard strategies
  is like finding simple statements with hard proofs.
  - It shouldn't stop us.

- The fact that game theory + computation = 
  is the usual proof search challenge.
  - It's fun!

Corollary

- Finding simple games with hard strategies
  is like finding simple statements with hard proofs.
  - It shouldn't stop us.

- The fact that game theory + computation = 
  is the usual proof search challenge.
  - It's fun!

- So lets try
  - **social behaviors = game theory + computation**

Outline

Game theory in one slide

$$\frac{\displaystyle \frac{\displaystyle \frac{\prod_{i=1}^{n} A_i \xrightarrow{u_k} \mathbb{R}}{\prod_{k\neq i=1}^{n} A_i \overset{BR_k}{\rightsquigarrow} A_k}}{\prod_{i=1}^{n} A_i \xrightarrow{\;BR=\langle BR_k \circ \pi_k \rangle_{i=1}^{n}\;} \prod_{i=1}^{n} A_i}}{}$$

$$1 \overset{NE}{\rightsquigarrow} \prod_{i=1}^{n} A_i$$

$$1 \overset{NE}{\rightsquigarrow} \prod_{i=1}^{n} A_i \overset{BR}{\rightsquigarrow} \prod_{i=1}^{n} A_i$$

Position games

$$A \times X \xrightarrow{\langle u_k, s_k \rangle} \mathbb{R} \times X$$

$$\overline{\qquad A_{-k} \times X \overset{BR_k}{\rightsquigarrow} A_k \qquad}$$

$$\overline{\qquad A \times X \overset{BR = \langle BR_k \circ \pi_k \rangle_{i=1}^{n}}{\rightsquigarrow} A \qquad}$$

$$\overline{\begin{array}{ccc} X & \overset{NE}{\rightsquigarrow} & A \\ & \underset{\langle NE, \mathrm{id} \rangle}{\searrow} \quad \underset{BR}{\nearrow} & \\ & A \times X & \end{array}}$$

where

$$A = \prod_{i \in n} A_i \qquad X = \prod_{i \in n} X_i \qquad A_{-i} = \prod_{\substack{k \in n \\ k \neq i}} A_k$$

Dubious Statement 2

Utility is a red herring in game theory: only used to derive strategies.

Dubious Statement 2

Utility is a red herring in game theory: only used to derive strategies.

Game theory is a theory of fixed points of strategies.

Dubious Statement 7

Categories help with strategies.

Category of strategies

Definition

Let $\mathcal{C}$ be a cartesian category, and $\Delta : \mathcal{C} \longrightarrow \mathcal{C}$ a commutative monad over it.

The category $\mathcal{S} = \mathcal{S}_{\Delta \mathcal{C}}$ of Δ-strategies over $\mathcal{C}$ consists of

- players $A = \langle M_A, S_A \rangle \in \mathcal{C}^2$
- strategies $(A \stackrel{\Phi}{\longrightarrow} B) \in \mathcal{C}\big(M_A \times S_B,\ \Delta(M_B \times S_B)\big)$

Category of strategies

Composition

$$\frac{A \xrightarrow{\Phi} B \qquad\qquad B \xrightarrow{\Psi} C}{A \xrightarrow{\Phi;\Psi} C}$$

is given by

$$(\Phi;\Psi)_{a\gamma c\gamma'} = \sum_{\beta b} \Phi_{a\beta b\beta} \cdot \Psi_{b\gamma c\gamma'}$$

Category of strategies

Composition

$$\frac{A \xrightarrow{\Phi} B \qquad\qquad B \xrightarrow{\Psi} C}{A \xrightarrow{\Phi;\Psi} C}$$

is given by

$$\begin{aligned}(\Phi;\Psi)_{a\gamma c\gamma'} &= \sum_{\beta b} \Phi_{a\beta b\beta} \cdot \Psi_{b\gamma c\gamma'} \\ &= \sum_{\beta} \sum_{b} \Phi_{a\beta b\beta} \cdot \Psi_{b\gamma c\gamma'}\end{aligned}$$

Category of strategies

Composition

$$\frac{M_A \otimes S_B \xrightarrow{\Phi} M_B \otimes S_B \qquad\qquad M_B \otimes S_C \xrightarrow{\Psi} M_C \times S_C}{M_B \otimes S_C \xrightarrow{\Phi;\Psi} M_C \times S_C}$$

$$=$$

$$\mathrm{Tr}_{S_B}\Big(S_B \otimes M_A \otimes S_C \cong M_A \otimes S_B \otimes S_C \xrightarrow{\Phi\otimes S_C} M_B \otimes S_B \otimes S_C \cong$$
$$\cong S_B \otimes M_B \otimes S_C \xrightarrow{S_B\otimes\Psi} S_B \otimes M_C \times S_C\Big)$$

Games of perfect and complete information

$$S_A = S_B = (\mathbb{R} \times \mathbb{R})^{M_A \times M_B}$$

Best Response strategies

$$\langle b, \sigma^A\rangle \xrightarrow{\Sigma_A} \langle a, \sigma^A\rangle \quad \iff \quad \forall x \in M_A.\ \sigma^A_{xb} \leq \sigma^A_{ab}$$

$$\langle a, \sigma^B\rangle \xrightarrow{\Sigma_B} \langle b, \sigma^B\rangle \quad \iff \quad \forall y \in M_B.\ \sigma^B_{ay} \leq \sigma^B_{ab}$$

Nash equilibrium

$$\frac{A \xrightarrow{\Sigma_B} B \qquad\qquad B \xrightarrow{\Sigma_A} A}{A \xrightarrow{\Sigma_B;\Sigma_A} A \qquad\qquad B \xrightarrow{\Sigma_A;\Sigma_B} B}$$

$$\begin{array}{ccc} 1 & \xrightarrow{\quad NE \quad} & A \times B \\ & {\scriptstyle NE}\searrow \qquad \nearrow{\scriptstyle (\Sigma_B;\Sigma_A)\times(\Sigma_A;\Sigma_B)} & \\ & A \times B & \end{array}$$

Games of imperfect and complete information

$$S_A = P_A \times (\mathbb{R} \times \mathbb{R})^{M_A \times M_B}$$
$$S_B = P_B \times (\mathbb{R} \times \mathbb{R})^{M_A \times M_B}$$

Games of perfect and incomplete information

$$
\begin{aligned}
S_A &= \mathbb{R}^{M_A \times M_B} \times \Delta S_B \\
S_B &= \mathbb{R}^{M_A \times M_B} \times \Delta S_A
\end{aligned}
$$

Games of perfect and incomplete information

$$S_A = S_B = \prod_{i=0}^{\infty} \Delta^i \left(\mathbb{R}^{M_A \times M_B}\right)$$

Outsmarting

$$[A,B] = \left[M_A \times \Delta[B,A], M_B \times \Delta[B,A]\right]$$
$$[B,A] = \left[M_B \times \Delta[A,B], M_A \times \Delta[A,B]\right]$$

where $[X, Y]$ are Y's strategies against X

What is the math of outsmarting?

Outline

Dubious Statement 1

Dubious Statement 2

Dubious Statement 3

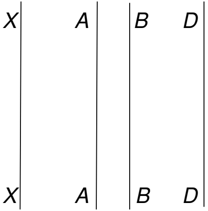
Dubious Statement 4

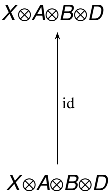
Dubious Statement 5

String diagrams: Types

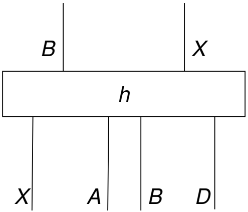
$X \otimes A \otimes B \otimes D$

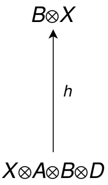
String diagrams: Identities



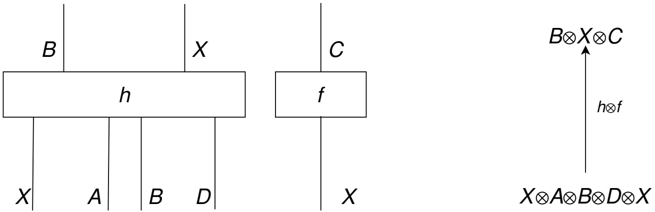


String diagrams: Operations

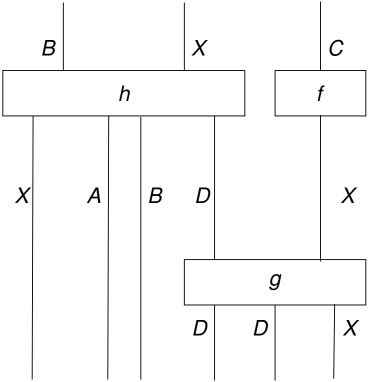


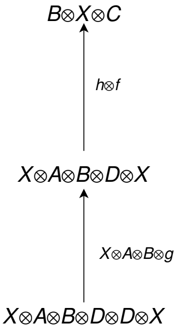


String diagrams: Parallel composition

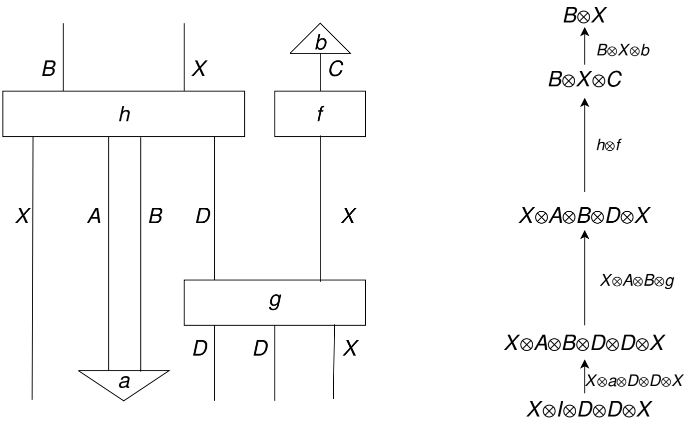


String diagrams: Sequential composition

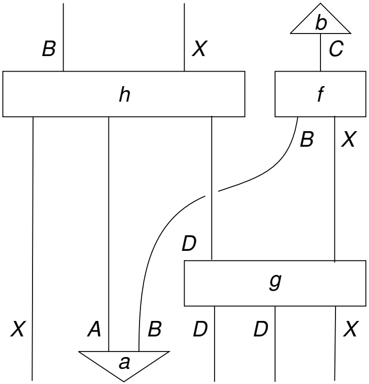




String diagrams: Values and deletion



String diagrams: Symmetry



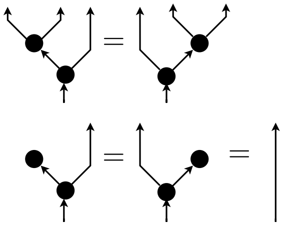
$$X \otimes I \otimes D \otimes D \otimes X \xrightarrow{X \otimes a \otimes D \otimes D \otimes X} X \otimes A \otimes B \otimes D \otimes D \otimes X \xrightarrow{X \otimes A \otimes B \otimes g} X \otimes A \otimes B \otimes D \otimes X \xrightarrow{X \otimes A \otimes c \otimes X} X \otimes A \otimes D \otimes B \otimes X \xrightarrow{h \otimes f} B \otimes X \otimes C \xrightarrow{B \otimes X \otimes b} B \otimes X$$

Data services: Copying and deletion

- the *copying* operation $A \xrightarrow{\delta} A \otimes A$, and
- the *deleting* operation $A \xrightarrow{\top} I$,

which together form a *comonoid*, i.e. satisfy the equations

$$\delta\,;(\delta \otimes A) = \delta\,;(A \otimes \delta)$$
$$\delta\,;(\top \otimes A) = \delta\,;(A \otimes \top) = \mathrm{id}_A$$



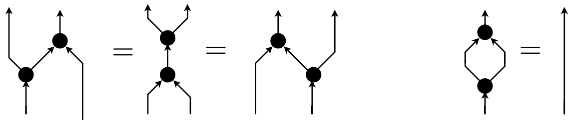
Data services: Comparison

- the *comparison* operation $A \otimes A \xrightarrow{\varrho} A$

which is required to be associative and thus makes A into a *semigroup*.

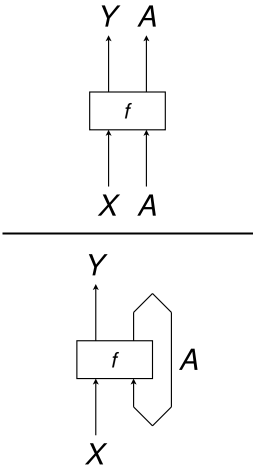
The copying and the comparison operations are further required to satisfy the *data distribution* conditions

$$(\delta \otimes A) ; (A \otimes \varrho) = \varrho ; \delta = (A \otimes \delta) ; (\varrho \otimes A) \qquad \delta ; \varrho = \mathrm{id}$$



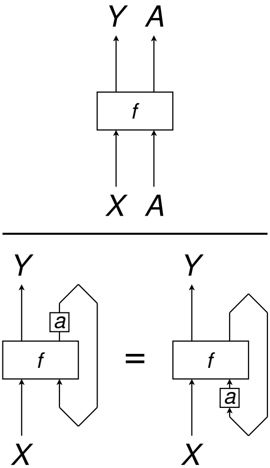
Trace

$$\frac{X \otimes A \xrightarrow{f} Y \otimes A}{X \xrightarrow{\mathrm{Tr}^A_{XY} f} Y}$$



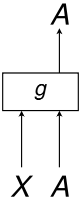
Trace

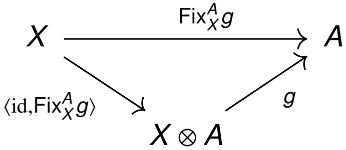
$$\frac{X \otimes A \xrightarrow{f} Y \otimes A}{X \xrightarrow{\mathrm{Tr}^A_{XY} f} Y}$$

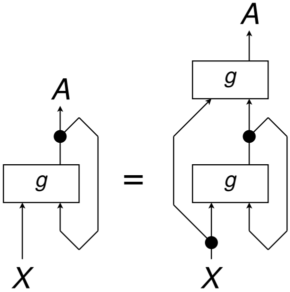


Fixpoint

$$X \otimes A \xrightarrow{g} A$$





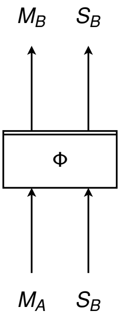


Strategies as morphisms

$$A \xrightarrow{\Phi} B$$

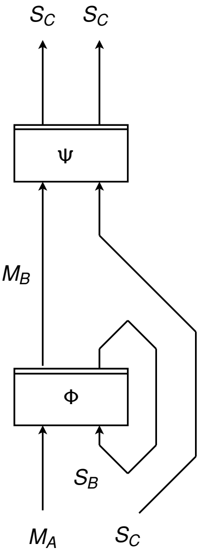
Strategies as morphisms

$$M_A \otimes S_B \xrightarrow{\Phi} \Delta(M_B \otimes S_B)$$



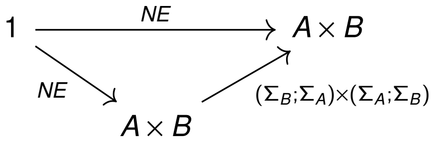
Composition of strategies

$$(\Phi;\Psi)_{a\gamma c\gamma'} = \sum_{\beta b \beta} \Phi_{a\beta b\beta} \cdot \Psi_{b\gamma c\gamma'}$$

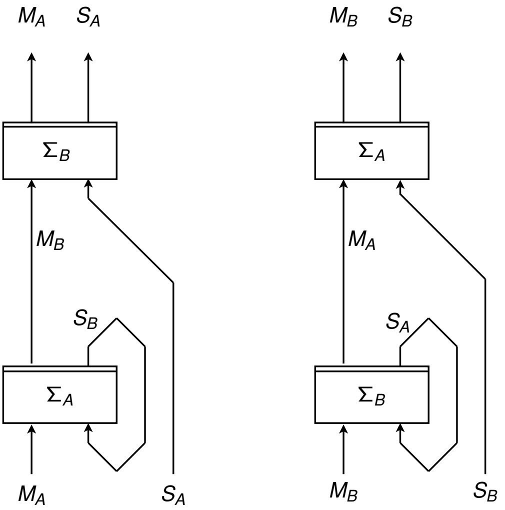


Nash equilibrium

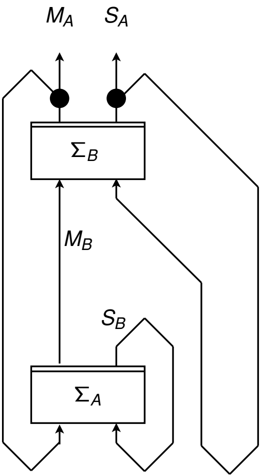
$$\dfrac{\dfrac{A \xrightarrow{\Sigma_B} B \qquad\qquad B \xrightarrow{\Sigma_A} A}{A \xrightarrow{\Sigma_B;\Sigma_A} A \qquad\qquad B \xrightarrow{\Sigma_A;\Sigma_B} B}}{}$$

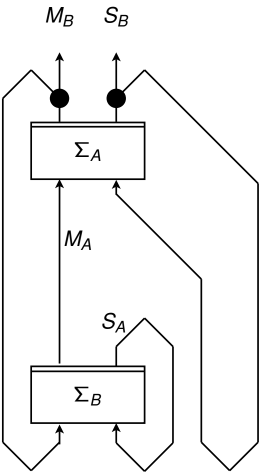


Nash equilibrium

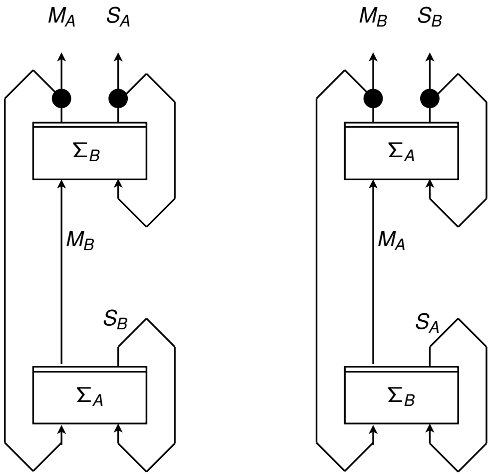


Nash equilibrium





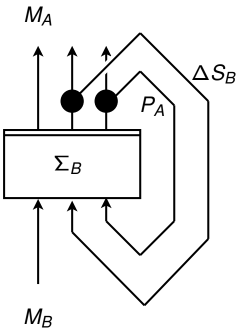
Nash equilibrium

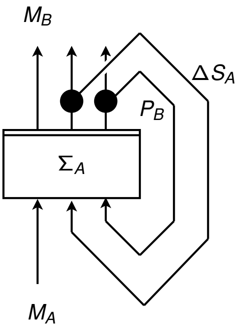


Explicit beliefs

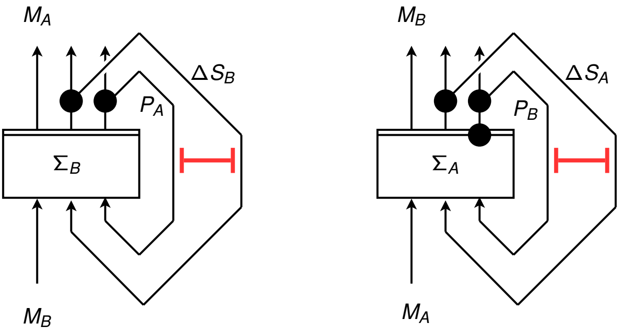
$$\begin{aligned} S_A &= P_A \times \Delta S_B \\ S_B &= P_B \times \Delta S_A \end{aligned}$$

Explicit beliefs equilibrium





Outsmarting



Outsmarting

- static types (Harsanyi)
  - play their preferences
  - observe the same
  - cannot agree to disagree (Aumann)
- dynamic types
  - must disagree
  - deceive, posture, mimic...
  - create false posteriors

Dubious Statement 3

Equilibrium (even coordinated) consists of individual fixed points and **can perpetrate false beliefs**.

Corollary

Outsmarting pumps complexity.

Outline

Dubious Statement 1

Dubious Statement 2

Dubious Statement 3

Dubious Statement 4

Dubious Statement 5

Outsmarting pumps complexity

- Simple games require complex computation.

Matching pennies

	0	1
0	-1 1	1 -1
1	1 -1	-1 1

Penalty Kick

Same game

Rules

- repeated infinitely
- loser has bounded expected payoff

Not losing is easy

Alice's reasoning

Suppose that I play

- 1 with a frequency $p \in [0, 1]$,
- 0 with a frequency $1 - p$.

Then

- $p < \frac{1}{2}$, Bob gets $(1 - p) - p = 1 - 2p > 0$ playing 1,
- $p > \frac{1}{2}$, Bob gets $p - (1 - p) = 2p - 1 > 0$ by playing 0,
- $p = \frac{1}{2}$, Bob's gets $1 - 2p = 2p - 1 = 0$.

Not losing is easy

Bob's reasoning

(same)

Not losing is easy

Both players must randomize

If either player's moves are predictable, then the other one can win.

But what does it mean to randomize?

- Suppose that Bob plays

  0101010101010101010101010101010101010101

But what does it mean to randomize?

- Suppose that Bob plays

  0101010101010101010101010101010101010101

- Alice predicts that Bob will play 0, and she plays 0.

But what does it mean to randomize?

- Suppose that Bob plays

  0101010101010101010101010101010101010101

- Alice predicts that Bob will play 0, and she plays 0.
  - But what if Bob plays 1?

But what does it mean to randomize?

- Suppose that Bob plays

  0101010101010101010101010101010101010101

- Alice predicts that Bob will play 0, and she plays 0.
  - But what if Bob plays 1?

- If Alice thinks probabilistically, she notices that a fair coin is as likely to generate the above string as e.g.

  1101000100110101001011100100000100000010

  or any other sequence of 40 bits.

Probability does not distinguish events

Probabilities

- talk about *ensembles*
- cannot tell apart *individual strings*

For probabilistic players mixing is obsolete

- If Bob believes that Alice thinks probabilistically, then he has no reason to randomize
  - (since Alice cannot tell)

For probabilistic players mixing is obsolete

- If Bob believes that Alice thinks probabilistically, then he has no reason to randomize
  - (since Alice cannot tell)
- If Alice believes that Bob thinks probabilistically, then she has no reason to randomize
  - (since Bob cannot tell)

Sleeping players

- If the players *agree* to use only
  - probability theory, and
  - knowledge logic

  then they will both commit fixed strategies, and go to sleep.

Alice wakes up

"Let me look for regularities in Bob's strategy!"

Trying to win

- Check if the frequency of 0 and 1 is $\frac{1}{2}$.

Trying to win

- Check if the frequency of 0 and 1 is $\frac{1}{2}$.
  - detect that 00000000 . . . is not random
  - not that 0101010101 . . . is not random

Trying to win

- Check if the frequency of 0 and 1 is $\frac{1}{2}$.
  - detect that 00000000 . . . is not random
  - not that 0101010101 . . . is not random
- Check if the frequencies of 00, 01, 10 and 11 are $\frac{1}{4}$
  - detect that 010101010101 . . . is not random
  - not that 0001101100011011 . . . is not random

Trying to win

- Check if the frequency of each $b_1 b_2 \cdots b_n$ is $\frac{1}{2^n}$.

Trying to win

- Check if the frequency of each $b_1 b_2 \cdots b_n$ is $\frac{1}{2^n}$.
  - The regularity within each $\mathbb{2}^n$ will be detected (after infinite amount of time)

Trying to win

- Check if the frequency of each $b_1 b_2 \cdots b_n$ is $\frac{1}{2^n}$.
  - The regularity within each $\mathbb{2}^n$ will be detected (after infinite amount of time)
  - The regularity of

    $$011011100101110111100010011010101111001101\ldots$$

    will remain undetected for every n

Trying to win

- Check if the frequency of each $b_1 b_2 \cdots b_n$ is $\frac{1}{2^n}$.
  - The regularity within each $\mathbb{2}^n$ will be detected (after infinite amount of time)
  - The regularity of

    $$011011100101110111100010011010101111001101\ldots$$

    will remain undetected for every n
    - 0,1,2,3. . . in the binary notation, concatenated

Trying to win

General problem: Regular substrings

Bob may

- randomize even bits, and use a rule for the odd bits
- randomize a substring and use a rule for the rest

Trying to win

General problem: Regular substrings

Bob may

- randomize even bits, and use a rule for the odd bits
- randomize a substring and use a rule for the rest

General task

Alice must

- check ***every substring*** for regularities

Question

But what is regularity?

Regularity is programmability

- Let $\mathcal{L}$ be a programming language, e.g.
  - finite state machines
  - Turing machines
  - Python, Scala, Java ...
- Let $h : \mathbb{2}^* \to \mathbb{2}^*$ be an $\mathcal{L}$-program interpreter.

Regularity is programmability

- Let $\mathcal{L}$ be a programming language, e.g.
  - finite state machines
  - Turing machines
  - Python, Scala, **Java**...
- Let $h : \mathbb{2}^* \to \mathbb{2}^*$ be an $\mathcal{L}$-program interpreter.

Regularity is compressibility

$$0101010101010101010101010101010101010101 = h\Big(for(i=0; i<20; i++)\{print\ 01\} \Big)$$

Regularity is compressibility

$$011011100101110111100010011010101111001101 \;=\; h\Big(\, for(i = 0;;i++)\{print\ i\}\,\Big)$$

Randomness is incompressibility

$$1101000100110101001011100100000100000010 \;=\;$$
$$h\Big(\,\mathit{print}\ 1101000100110101001011100100000100000010\,\Big)$$

(Monoidal computer)

Definition

Monoidal computer is a strict symmetric monoidal category $\mathbb{C}$ with

(i) data services

(ii) a distinguished type $\mathbb{P}$ ("of programs")

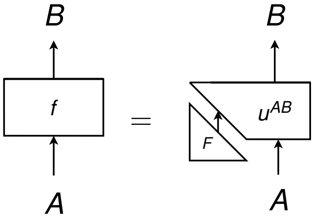
(Monoidal computer)

Definition

Monoidal computer is a strict symmetric monoidal category $\mathbb{C}$ with

(iii) *universal evaluators* u^{AB} such that

$\forall f \; \exists F$

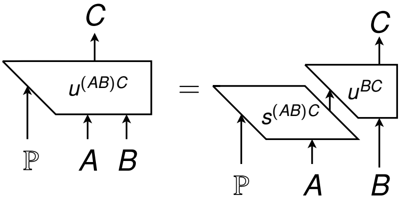


(Monoidal computer)

Definition

Monoidal computer is a strict symmetric monoidal category $\mathbb{C}$ with

(iv) *partial evaluators* $s^{(AB)C}$ such that



Strategy for matching pennies

Definition

An $\mathcal{L}$-*detector* is an $\mathcal{L}$-function $h : \mathcal{2}^* \to \mathcal{2}^*$ such that

$$h(\vec{x}) = \vec{y} \implies \ell(\vec{x}) < \ell(\vec{y})$$

Strategy for matching pennies

Definition

An $\mathcal{L}$-*detector* is an $\mathcal{L}$-function $h : 2^* \to 2^*$ such that

$$h(\vec{x}) = \vec{y} \implies \ell(\vec{x}) < \ell(\vec{y})$$

A bitstring $\vec{y} \in 2^*$ is h-regular at the level n if

$$\exists \vec{x}.\ h(\vec{x}) \sqsupseteq \vec{y} \ \wedge\ \ell(\vec{y}) - \ell(\vec{x}) > n$$

Strategy for matching pennies

Definition

An $\mathcal{L}$-*detector* is an $\mathcal{L}$-function $h : 2^* \to 2^*$ such that

$$h(\vec{x}) = \vec{y} \implies \ell(\vec{x}) < \ell(\vec{y})$$

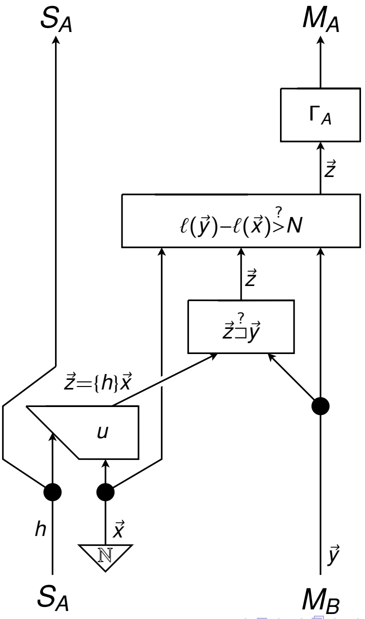
A bitstring $\vec{y} \in 2^*$ is h-regular at the level n if

$$\exists \vec{x}.\ h(\vec{x}) \sqsupseteq \vec{y} \ \wedge\ \ell(\vec{y}) - \ell(\vec{x}) > n$$

The h-regularity degree of $\vec{y}$ is

$$\sigma_h(\vec{y}) = \max\left\{n \mid \exists \vec{x}.\ h(\vec{x}) \sqsupseteq \vec{y} \ \wedge\ \ell(\vec{y}) - \ell(\vec{x}) \geq n\right\}$$

Strategy for matching pennies



Universal detector

Definition

An $\mathcal{L}$-detector $u : \mathbb{2}^* \to \mathbb{2}^*$ is *universal* if u-testing can detect all strings that are detected as h-regular with respect to any $\mathcal{L}$-detector h.

Universal detector

Definition

An $\mathcal{L}$-detector $u : \mathbb{2}^* \to \mathbb{2}^*$ is *universal* if u-testing can detect all strings that are detected as h-regular with respect to any $\mathcal{L}$-detector h.

More precisely, for every $\mathcal{L}$-detector $h : \mathbb{2}^* \to \mathbb{2}^*$ there is a constant c_h such that for every bitstring $\vec{y}$ holds

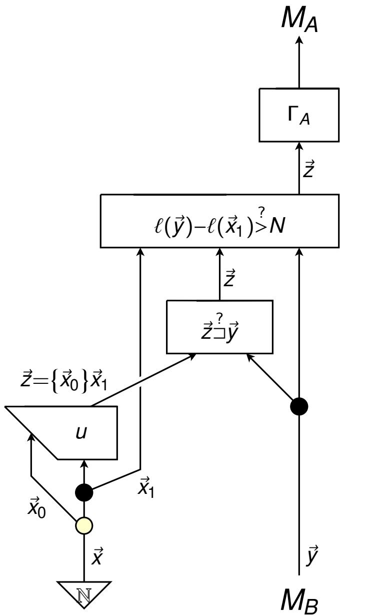
$$\sigma_h(\vec{y}) \quad \leq \quad c_h + \sigma_u(\vec{y})$$

Universal detector

Proposition

If $\mathcal{L}$ is a Turing complete language, then there is a universal $\mathcal{L}$-detector.

Universal strategy for matching pennies



Dubious Statement 4

Winning pennies can be arbitrarily hard.
(It requires a universal computer.)

Outline

Dubious Statement 1

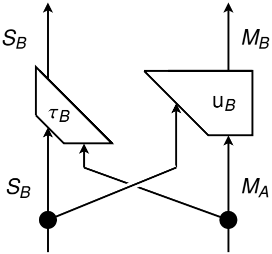
Dubious Statement 2

Dubious Statement 3

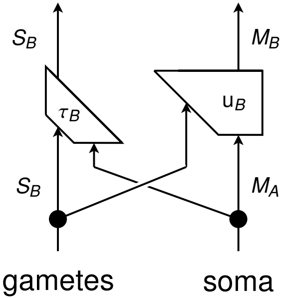
Dubious Statement 4

Dubious Statement 5

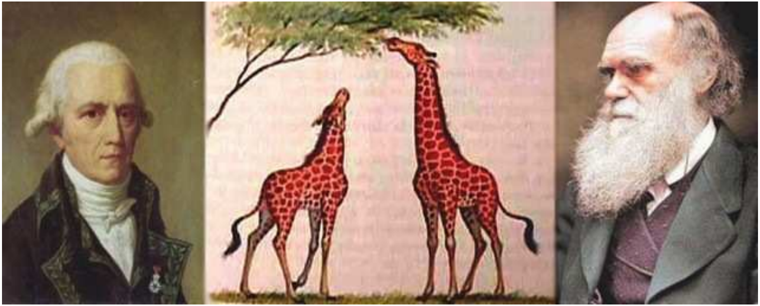
Learning strategy



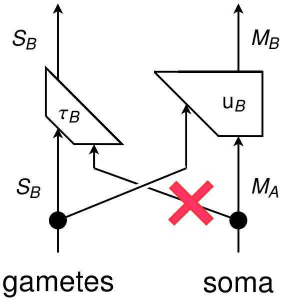
Adaptation strategy?



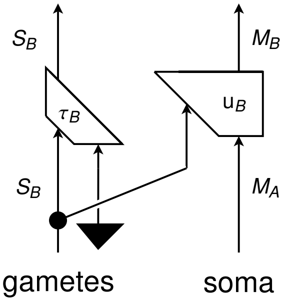
Hereditary adaptation vs Random mutation



Central Dogma of Molecular Biology



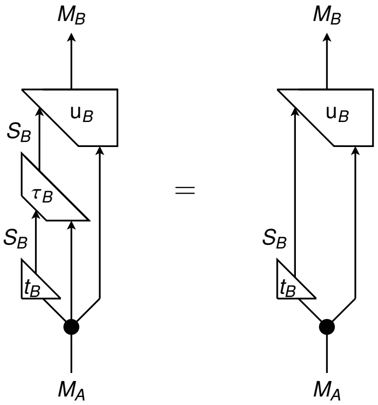
Central Dogma of Molecular Biology



Graduate strategy

Proposition

Every learning strategy τ_B has a fixed point, i.e. a state (type) t_B at which nothing more can be learned:



Learning equilibrium

Corollary

For every learning profile there is a graduate profile, where the players cannot improve their winnings by learning.

Dubious Statement 5

All algorithmic learning strategies reach equilibrium in one step.

Corollary

All ontogenetic adaptations that are inheritable can be expressed within one generation.

Corollary

All ontogenetic adaptations that are inheritable can be expressed within one generation.

- E.g., the epigenetic flows do not present filogenetic adoption of ontogenetic adaptations, but just triggers to express previously developed and stored filogenetic adaptations.

The Problem of Overfitting

- statistical overfitting: narrow statistic, eager training
- evolutionary overfitting: locked into an environment
- market model overfitting: windfall
- control overfitting: blind feedback

The Problem of Overfitting

- statistical overfitting: narrow statistic, eager training
- evolutionary overfitting: locked into an environment
- market model overfitting: windfall
- control overfitting: blind feedback
  - healthy heart is a chaotic system: interferring feedback loops
  - convergence is heart attack: **stablility preempts adaptation**

Dubious Statements 6–11

Too much reflection leads to overfitting.

Dubious Statements 6–11

Too much reflection leads to overfitting.

In the long run, ignorance is bliss.