# The Blessing and the Curse
## of the Multiplicative Updates

Manfred K. Warmuth

University of California, Santa Cruz

EE Biology Seminar, UCSC
Oct. 19, 2016

# Definition of multiplicative updates

- Algorithm maintains a weight vector
- Weights multiplied by non-negative factors
- Motivated by relative entropies as regularizer

# Machine learning vs nature

Typical multiplicative update in ML

- Set of experts predict something on a trial by trial basis
- $s_i$ is belief in of $i$th expert at trial $t$
- Update: [V,LW]

$$s_i := \frac{s_i\, e^{-\eta \text{loss}_i}}{Z} \qquad \text{where } \eta \text{ learning rate and } Z \text{ normalizer}$$

- Bayes rule is special case

# Machine learning vs nature

Typical multiplicative update in ML

- Set of experts predict something on a trial by trial basis
- $s_i$ is belief in of $i$th expert at trial $t$
- Update: [V,LW]

$$s_i := \frac{s_i\, e^{-\eta \text{loss}_i}}{Z} \qquad \text{where } \eta \text{ learning rate and } Z \text{ normalizer}$$

- Bayes rule is special case

Nature

- Weights are concentrations / shares in a population

# Multiplicative updates

Blessing

- *Speed*

# Multiplicative updates

Blessing

- *Speed*

Curse

- *Loss of diversity*

# Multiplicative updates

Blessing

- *Speed*

Curse

- *Loss of diversity*

Will give 4 machine learning methods
for preventing the curse

and discuss related methods used in nature

# Multiplicative updates

Blessing

- *Speed*

Curse

- *Loss of diversity*

Will give 4 machine learning methods
for preventing the curse

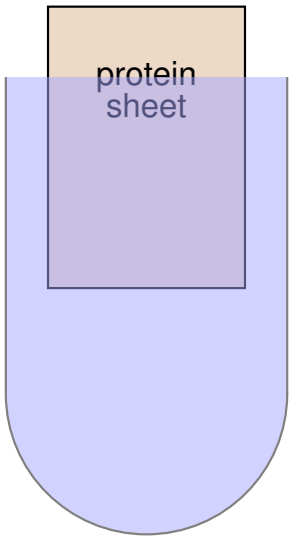
and discuss related methods used in nature

- Next: implementation of Bayes in the tube

# *In-vitro* selection algorithm

- for finding RNA strand that binds to protein

Make tube of *random* RNA strands



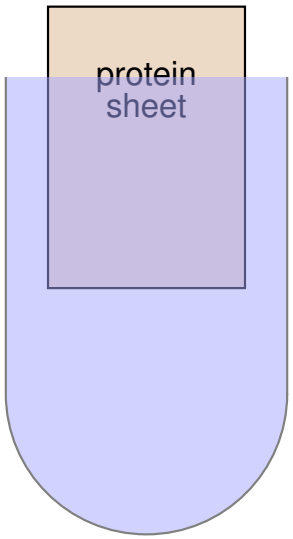

for t=1 to 20 do

1. Dip sheet coated with protein into tube
2. Pull out and wash off RNA that stuck
3. Multiply washed off RNA back to original amount (normalization)

# *In-vitro* selection algorithm

- for finding RNA strand that binds to protein

Make tube of *random* RNA strands




for t=1 to 20 do

1. Dip sheet coated with protein into tube
2. Pull out and wash off RNA that stuck
3. Multiply washed off RNA back to original amount (normalization)

Can't be done with computer
because $10^{15}$ RNA strands per liter of soup

# Basic scheme

Start with unit amount of random RNA

Loop

1. Functional separation into
   good RNA and bad RNA
2. Amplify good RNA to unit amount

# Duplicating DNA with PCR

## needed for normalization step

- Invented by Kary Mullis 1985
- Heat - double stranded DNA comes apart
- Cool - Short primers hybridize at the ends
- Taq Polymerase runs along DNA strand and complements bases
- Back to double stranded DNA

# Duplicating DNA with PCR

## needed for normalization step

- Invented by Kary Mullis 1985
- Heat - double stranded DNA comes apart
- Cool - Short primers hybridize at the ends
- Taq Polymerase runs along DNA strand and complements bases
- Back to double stranded DNA

- Ideally all DNA strands multiplied by factor of 2

# The caveat

- Not many interesting/specific functional separations found
- Need high throughput for functional separation

# Mathematical description of in-vitro selection

- Number the unknown RNA strands as $1, 2, \ldots n$ where $n \approx 10^{15}$ (# of strands in 1 liter)
- $s_i \geq 0$ is share of RNA $i$

# Mathematical description of in-vitro selection

- Number the unknown RNA strands as $1, 2, \ldots n$ where $n \approx 10^{15}$ (# of strands in 1 liter)
- $s_i \geq 0$ is share of RNA $i$
- Contents of tube represented as share vector $\boldsymbol{s} = (s_1, s_2, \ldots, s_n)$
- When tube has unit amount, then $s_1 + s_2 + \cdots + s_n = 1$

# Mathematical description of in-vitro selection

- Number the unknown RNA strands as $1, 2, \ldots n$ where $n \approx 10^{15}$ (# of strands in 1 liter)
- $s_i \geq 0$ is share of RNA $i$
- Contents of tube represented as share vector $\boldsymbol{s} = (s_1, s_2, \ldots, s_n)$
- When tube has unit amount, then $s_1 + s_2 + \cdots + s_n = 1$
- $W_i \in [0, 1]$ is the fraction of one unit of RNA $i$ that is good
- $W_i$ is fitness of RNA strand $i$ for attaching to protein
- Protein represented as fitness vector $\boldsymbol{W} = (W_1, W_2, \ldots, W_n)$

# Mathematical description of in-vitro selection

- Number the unknown RNA strands as $1, 2, \ldots n$ where $n \approx 10^{15}$ (# of strands in 1 liter)
- $s_i \geq 0$ is share of RNA $i$
- Contents of tube represented as share vector $\boldsymbol{s} = (s_1, s_2, \ldots, s_n)$
- When tube has unit amount, then $s_1 + s_2 + \cdots + s_n = 1$
- $W_i \in [0, 1]$ is the fraction of one unit of RNA $i$ that is good
- $W_i$ is fitness of RNA strand $i$ for attaching to protein
- Protein represented as fitness vector $\boldsymbol{W} = (W_1, W_2, \ldots, W_n)$
- Vectors $\boldsymbol{s}$, $\boldsymbol{W}$ unknown: Blind Computation!

# Mathematical description of in-vitro selection

- Number the unknown RNA strands as $1, 2, \ldots n$ where $n \approx 10^{15}$ (# of strands in 1 liter)
- $s_i \geq 0$ is share of RNA $i$
- Contents of tube represented as share vector $\boldsymbol{s} = (s_1, s_2, \ldots, s_n)$
- When tube has unit amount, then $s_1 + s_2 + \cdots + s_n = 1$
- $W_i \in [0, 1]$ is the fraction of one unit of RNA $i$ that is good
- $W_i$ is fitness of RNA strand $i$ for attaching to protein
- Protein represented as fitness vector $\boldsymbol{W} = (W_1, W_2, \ldots, W_n)$
- Vectors $\boldsymbol{s}$, $\boldsymbol{W}$ unknown: Blind Computation!

Strong assumption:

- Fitness $W_i$ independent of share vector $\boldsymbol{s}$

# Update

Good RNA in tube $\boldsymbol{s}$:

$$s_1 W_1 + s_2 W_2 + \cdots + s_n W_n = \boldsymbol{s} \cdot \boldsymbol{W}$$

Bad RNA:

$$s_1(1 - W_1) + s_2(1 - W_2) + \cdots + s_n(1 - W_n) = 1 - \boldsymbol{s} \cdot \boldsymbol{W}$$

- Amplification:
  - Good share of RNA $i$ is $s_i W_i$ - multiplied by factor $F$
  - If precise, then all good RNA multiplied by same factor $F$
  - Final tube at end of loop

$$F\ \boldsymbol{s} \cdot \boldsymbol{W}$$

  - Since final tube has unit amount of RNA

$$F\ \boldsymbol{s} \cdot \boldsymbol{W} = 1 \text{ and } F = \frac{1}{\boldsymbol{s} \cdot \boldsymbol{W}}$$

  - Update in each loop

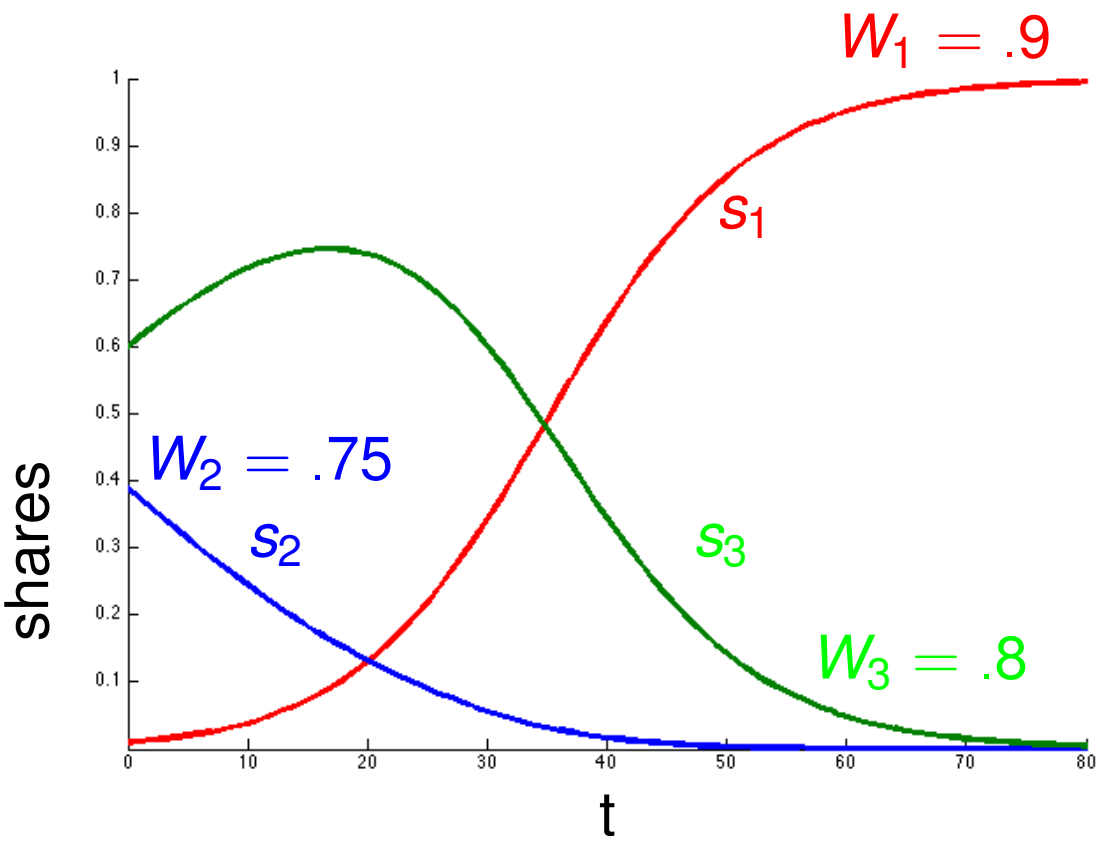
$$s_i := \frac{s_i W_i}{\boldsymbol{s} \cdot \boldsymbol{W}}$$

# Implementing Bayes rule in RNA

- $s_i$ is prior $P(i)$
- $W_i \in [0..1]$ is probability $P(Good|i)$
- $\boldsymbol{s} \cdot \boldsymbol{W}$ is probability $P(Good)$
-

$$s_i := \frac{s_i W_i}{\boldsymbol{s} \cdot \boldsymbol{W}} \quad \text{is Bayes rule} \quad \underbrace{P(i|Good)}_{\text{posterior}} = \frac{\overbrace{P(i)}^{\text{prior}} P(Good|i)}{P(Good)}$$

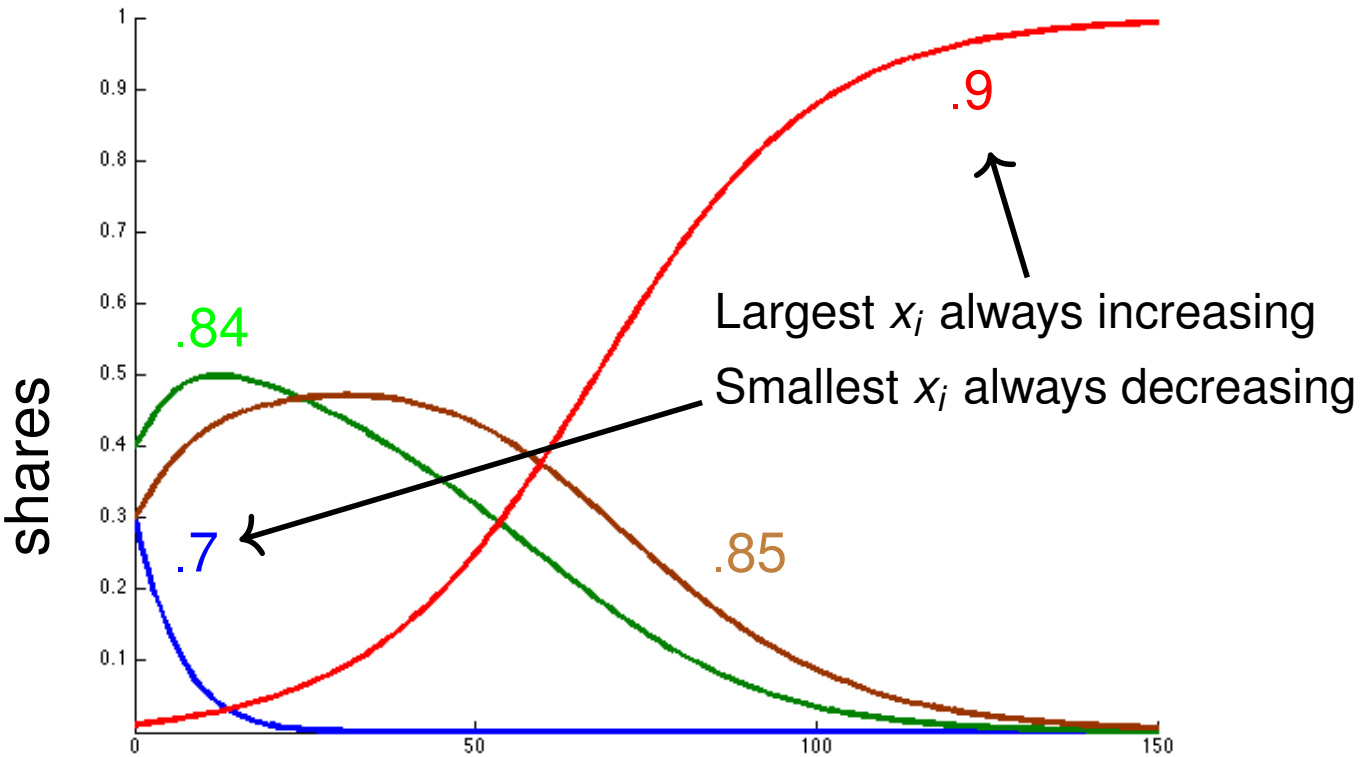
# Iterating Bayes Rule with same data likelihoods




Initial $\boldsymbol{s} = (.01, .39, .6)$

$\boldsymbol{W} = (.9, .75, .8)$

# $\max_i W_i$ eventually wins




$$s_{t,i} = \frac{s_{0,i} W_i^t}{\text{normalization}}$$

$t$ is speed parameter

# The best are *too* good

- Multiplicative update

$$s_i \sim s_i \underbrace{\text{fitness factor}_i}_{\geq 0}$$

# The best are *too* good

- Multiplicative update

$$s_i \sim s_i \underbrace{\text{fitness factor}_i}_{\geq 0}$$

- Blessing

  Best get amplified exponentially fast
  Can do this with $10^{15}$ variables

# The best are *too* good

- Multiplicative update

$$s_i \sim s_i \underbrace{\text{fitness factor}_i}_{\geq 0}$$

- Blessing

  Best get amplified exponentially fast
  Can do this with $10^{15}$ variables

- Curse

  The best wipe out the others
  Loss of diversity

# View of evolution

Simple view

- Inheritance
  - mutation
  - selection for the fittest with multiplicative update

# View of evolution

Simple view

- Inheritance
  - mutation
  - selection for the fittest with multiplicative update

Different view

- Multiplicative updates are brittle
- A mechanism is needed to ameliorate the curse of the multiplicative update
- Machine Learning and Nature uses various mechanisms

# Another tube experiment

Tube with nutrient solution

- Add 100 different bacteria and wait
- How many survive?

# Another tube experiment

Tube with nutrient solution

- Add 100 different bacteria and wait
- How many survive?
- Three survive:
  - one in the muck on the bottom,
  - one on the side and
  - one at the surface

# Another tube experiment

Tube with nutrient solution

- Add 100 different bacteria and wait
- How many survive?
- Three survive:
  - one in the muck on the bottom,
  - one on the side and
  - one at the surface

Same experiment but agitate tube

- How many survive

# Another tube experiment

Tube with nutrient solution

- Add 100 different bacteria and wait
- How many survive?
- Three survive:
  - one in the muck on the bottom,
  - one on the side and
  - one at the surface

Same experiment but agitate tube

- How many survive
- One - because only one niche left

# Nature 1: Boundaries prevent the curse

# Nature 1: Boundaries prevent the curse

Globalization

- Humans introduce ever more powerful roads
- Roads cause mixing
- Loss of diversity

# Nature 1: Boundaries prevent the curse

Globalization

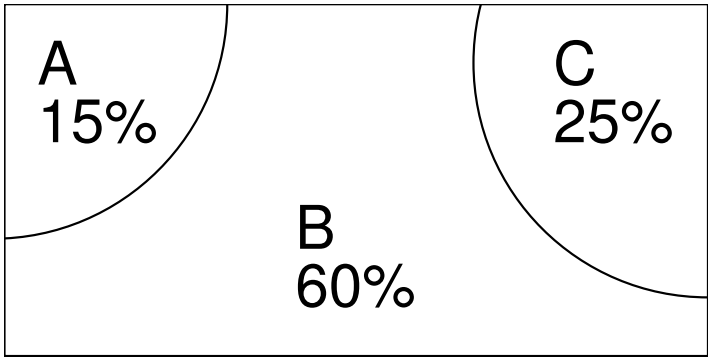
- Humans introduce ever more powerful roads
- Roads cause mixing
- Loss of diversity

Why is this bad?

# A first key problem

## that cannot be solved by multiplicative update alone

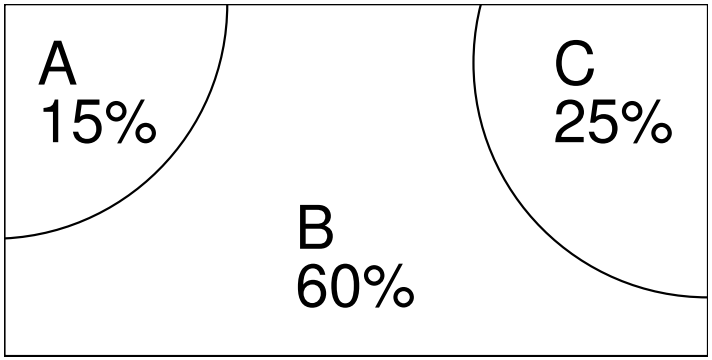
- 3 strands of RNA
- Want to amplify mixture while keeping percentages unchanged



# A first key problem

## that cannot be solved by multiplicative update alone

- 3 strands of RNA
- Want to amplify mixture while keeping percentages unchanged

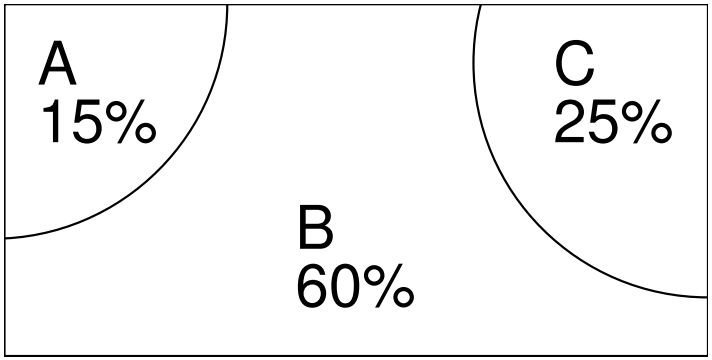



Iterative amplification will favor one!

# A first key problem

## that cannot be solved by multiplicative update alone

- 3 strands of RNA
- Want to amplify mixture while keeping percentages unchanged



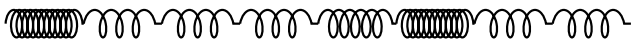

Iterative amplification will favor one!

Fitness factor $W_i$
- ideally all equal 2
- not the same to high precision
- dependant on the concentrations

# Solution

- Make one long strand with roughly right percentages of A,B,C

  A B B B C A B B

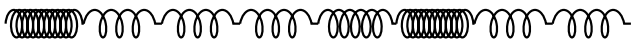


- Amplify same long strand many times
- Chop into constituents
- Relative frequencies fixed

# Solution

- Make one long strand with roughly right percentages of A,B,C

  A B B B C A B B



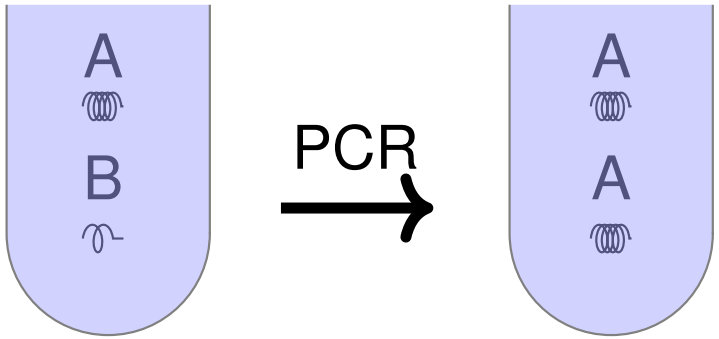
- Amplify same long strand many times
- Chop into constituents
- Relative frequencies fixed

Big picture

- Long strand functions as "chromosome"
  - Free floating genes in the nucleus would compete
  - Loss of diversity

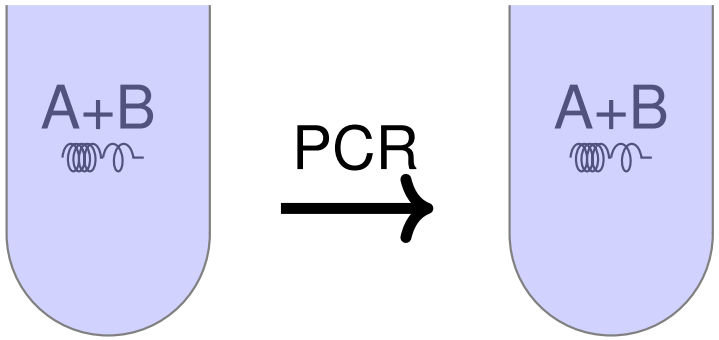
# Nature 2: Coupling preserves diversity

Un-coupled




A and B compete

Coupled




A and B should cooperate

# Many questions already about in-vitro selection

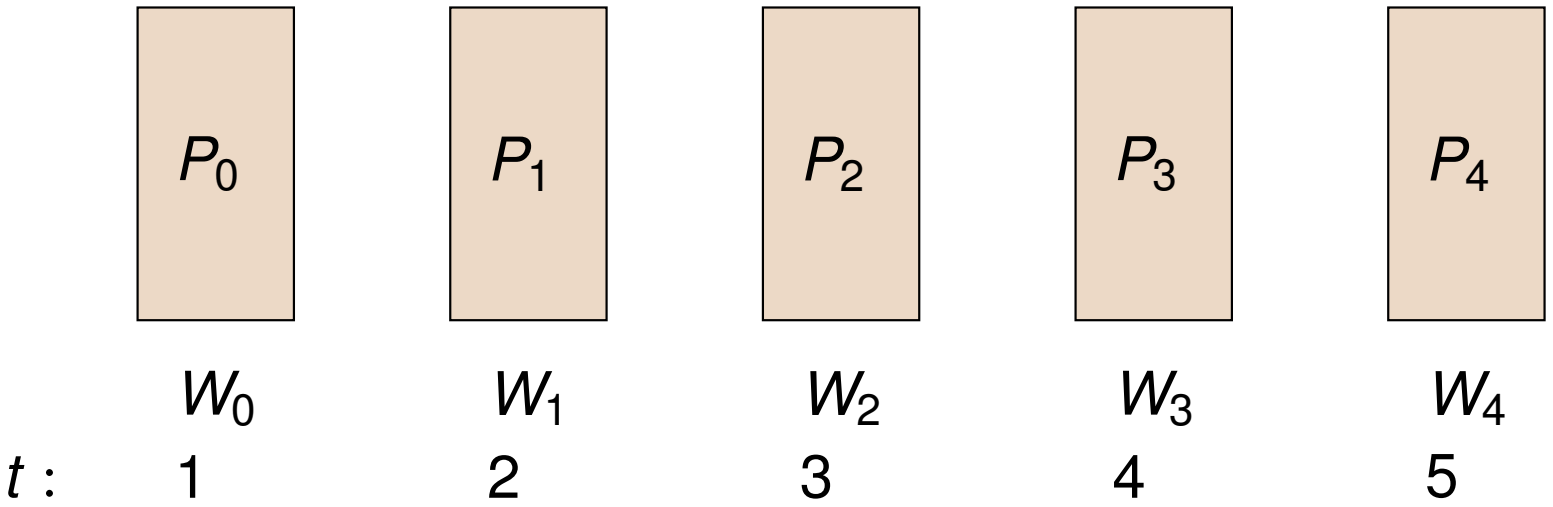
- a bare-bones evolutionary process

- What updates to $\boldsymbol{s}$ represent an iteration of an in-vitro selection update?
- What updates are possible with blind computation?
- Must maintain non-negative weights

# A second problem

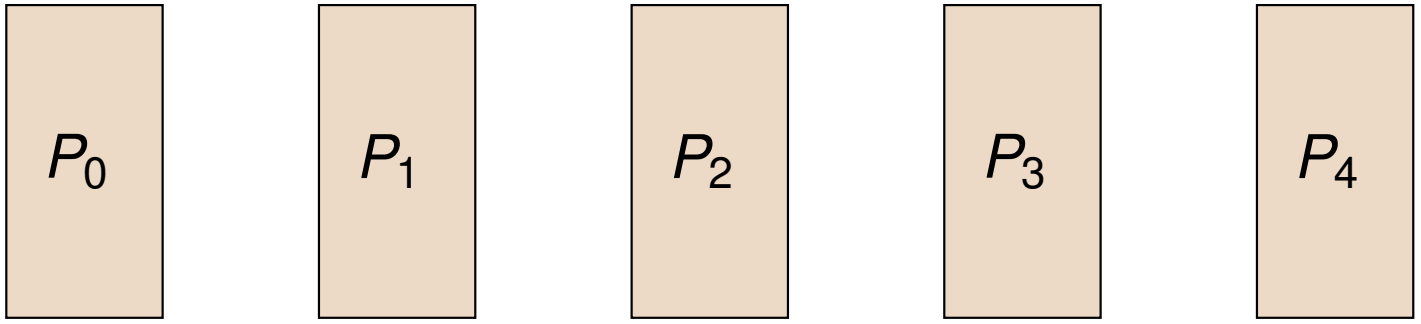
- that also cannot solved by just a multiplicative update

Find set of RNA strands that binds to a $q$ different proteins $W_j$




- In trial $t$, functional separation based on fitness vector $\boldsymbol{W}_{t \mod q}$
- So far all $\boldsymbol{W}_j$ the same

# In-vitro selection algorithm




| | | | | | |
|---|---|---|---|---|---|
| $W_{j,1}$ | 0.9 | 0.8 | 0.96 | 0.2 | 0.04 |
| $W_{j,2}$ | 0.1 | 0.01 | 0.8 | 0.9 | 0.8 |

# In-vitro selection algorithm

| | $P_0$ | $P_1$ | $P_2$ | $P_3$ | $P_4$ |
|---|---|---|---|---|---|
| $W_{j,1}$ | 0.9 | 0.8 | 0.96 | 0.2 | 0.04 |
| $W_{j,2}$ | 0.1 | 0.01 | 0.8 | 0.9 | 0.8 |
| $\boldsymbol{u} \cdot \boldsymbol{W}_j$ | 0.5 | 0.405 | 0.88 | 0.55 | 0.42 |

## Goal Tube

Tube: $\boldsymbol{u} = (0.5, 0.5, 0, \ldots, 0)$

The two strands cooperatively solve the problem
Related to disjunction of two variables

# Key problem 2

## Goal

Starting with 1 liter tube in which all strands appear with frequency $\approx 10^{-15}$, use PCR and . . . , to arrive at tube :

$$(\approx 0.5, \approx 0.5, \approx 0, \ldots, \approx 0)$$

- Problem
  - Over train with $P_0$ and $P_1 \Rightarrow s_1 \approx 1, s_2 \approx 0$
  - Over train with $P_3$ and $P_4 \Rightarrow s_1 \approx 0, s_2 \approx 1$
- Want blind computation!
  - $\boldsymbol{W}_j$ and initial $\boldsymbol{s}$ unknown
- Need some kind of feedback in each trial

# Related machine learning problem

- Tube $\equiv$ share vector $\boldsymbol{s} \in$ probability simplex$^N$
- Proteins $\equiv$ Example vectors $\boldsymbol{W}_t \in \{0,1\}^N$

## Assumption

$\exists \boldsymbol{u} = (0, 0, \frac{1}{k}, 0, 0, \frac{1}{k}, 0, 0, \frac{1}{k}, 0, 0, 0)$
with $k$ non-zero components of value $\frac{1}{k}$
s.t. $\forall t : \boldsymbol{u} \cdot \boldsymbol{W}_t \geq \frac{1}{k}$

## Goal

Find $\boldsymbol{s} : \boldsymbol{s} \cdot \boldsymbol{W}_t \geq \frac{1}{2k}$

# Normalized Winnow Algorithm (N. Littlestone)

Do passes over examples $\boldsymbol{W}_t$

if $\underbrace{\boldsymbol{s}_{t-1} \cdot \boldsymbol{W}_t}_{\text{good side}} \geq \frac{1}{2k}$

$$\text{then} \quad \boldsymbol{s}_t := \boldsymbol{s}_{t-1} \qquad \text{conservative update}$$

$$\text{else} \quad s_{t,i} := \begin{cases} s_{t-1,i} & \text{if} \quad W_{t,i} = 0 \\ \alpha s_{t-1,i} & \text{if} \quad W_{t,i} = 1, \quad \text{where } \alpha > 1 \end{cases}$$

and re-normalize

- $O(k \log \frac{N}{k})$ updates needed
  when labels consistent with $k$-literal disjunction
- Logarithmic in $N$
- Optimal to within constant factors

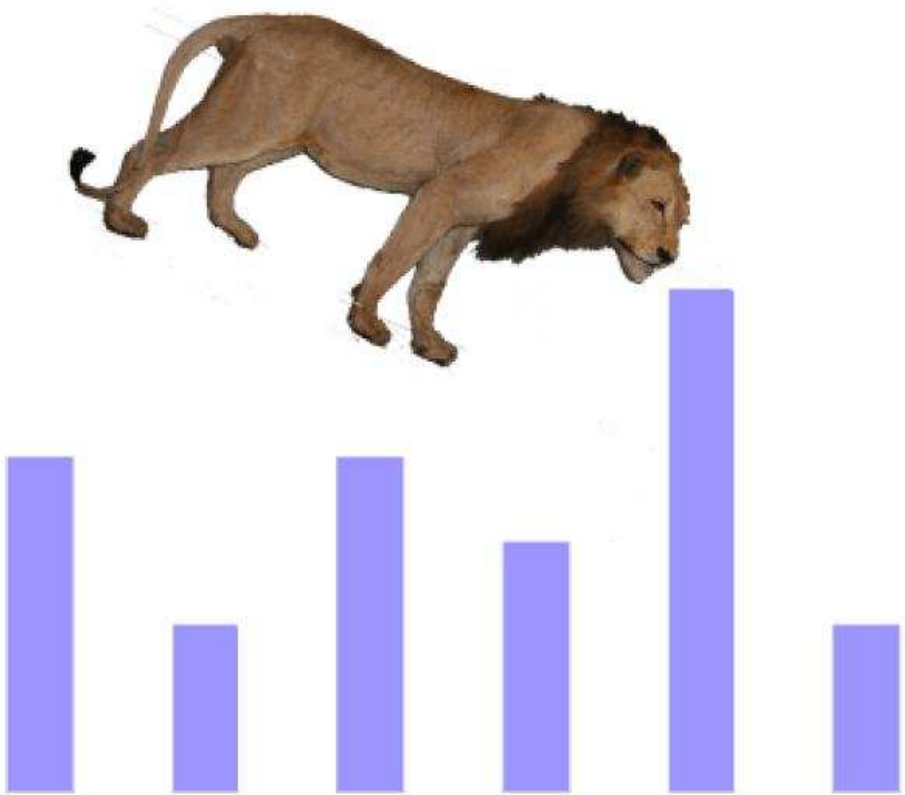
# ML 1: prevent overtraining with conservative update

- Assume we can measure the dot product $\boldsymbol{s}_{t-1} \cdot \boldsymbol{W}_t$ i.e. a concentration of RNA strands,
- then normalized Winnow can be implemented with Blind Computation
- Algorithms uses $n$ weights to learn $\binom{n}{k}$ concepts
- Coupling is replaced by thresholding

## Next trick

Cap weights
- Start with nature
- How we used same trick in Machine learning

# Nature 3: Super predator



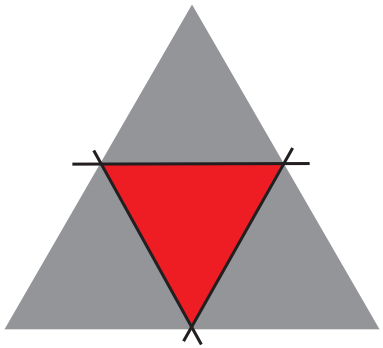
Preserves diversity

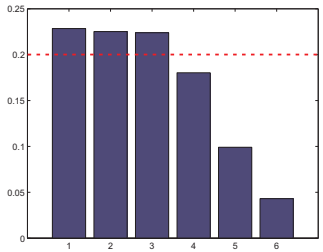
©Lion copyrights belong to Andy Warhol

# ML 2: cap weights

$$s'_i = \frac{s_i \; e^{-\eta \; Loss_i}}{Z}$$

$$\boldsymbol{s} = \text{capped and rescaled version of } \boldsymbol{s}'$$

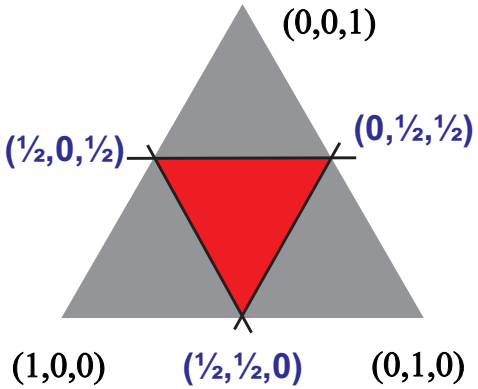


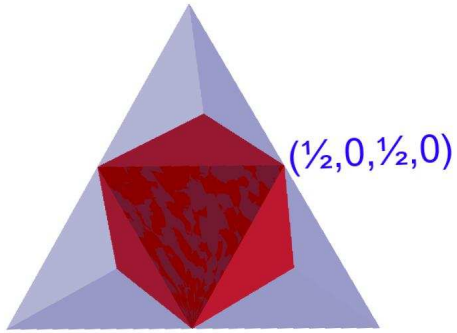



Cap and re-scale rest

# Why capping?

- $m$ sets encoded as probability vectors $(0, \frac{1}{m}, 0, 0, \frac{1}{m}, 0, \frac{1}{m})$ called $m$-corners $\binom{n}{m}$ of them
- The convex hull of the $m$-corners = capped probability simplex




- Combined update converges to best corner of capped simplex - best set of size $m$
- Capping lets us learn multiple goals - learn the best set of experts
- The matrix version of this algorithm leads to the best on-line algorithm for Principal Component Analysis
- How to cap in in-vitro selection setting?

# What if environment changes over time (M. Herbster)

- Initial tube contains $10^{15}$ different strands
- After a while —
  - Loss of diversity
  - Some selfish strand manages to dominate without doing the wanted function

## Fixes

- Mix in a little bit of initial soup for enrichment
- Or do sloppy PCR that introduces mutations

# Disk spindown problem

- Experts a set of $n$ fixed timeouts : $\tau_1, \tau_2, \ldots, \tau_n$
- Master Alg.
  - maintains a set of weights : $s_1, s_2, \ldots, s_n$
  - predicts with weighted average
- Multiplicative update

$$s_{t+1,i} = \frac{s_{t,i}\ \boldsymbol{e}^{-\eta \text{ energy usage of timeout i}}}{Z}$$

- Problem

# Disk spindown problem

- ML 3: Mix in little bit of uniform vector

$$\begin{aligned} \boldsymbol{s}' &= \text{Multiplicative Update} \\ \boldsymbol{s} &= (1-\alpha)\,\boldsymbol{s}' + \alpha\,(\frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N}) \\ & \quad \text{where } \alpha \text{ is small} \end{aligned}$$

# Disk spindown problem

- ML 3: Mix in little bit of uniform vector

$$\begin{aligned} \boldsymbol{s}' &= \text{Multiplicative Update} \\ \boldsymbol{s} &= (1-\alpha)\,\boldsymbol{s}' + \alpha\,(\frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N}) \\ & \quad \text{where } \alpha \text{ is small} \end{aligned}$$

- Nature 4: use mutation for same purpose

# ML 4: sleeping

- Keep track of past average share vector $\boldsymbol{r}$

$$\begin{aligned} \boldsymbol{s}' &= \text{Multiplicative Update} \\ \boldsymbol{s} &= (1-\alpha)\,\boldsymbol{s}' + \alpha\,\boldsymbol{r} \end{aligned}$$

- facilitates switch to previously useful vector
- long-term memory

Sleeping interpretation - “Putting Bayes to sleep” [KAW]

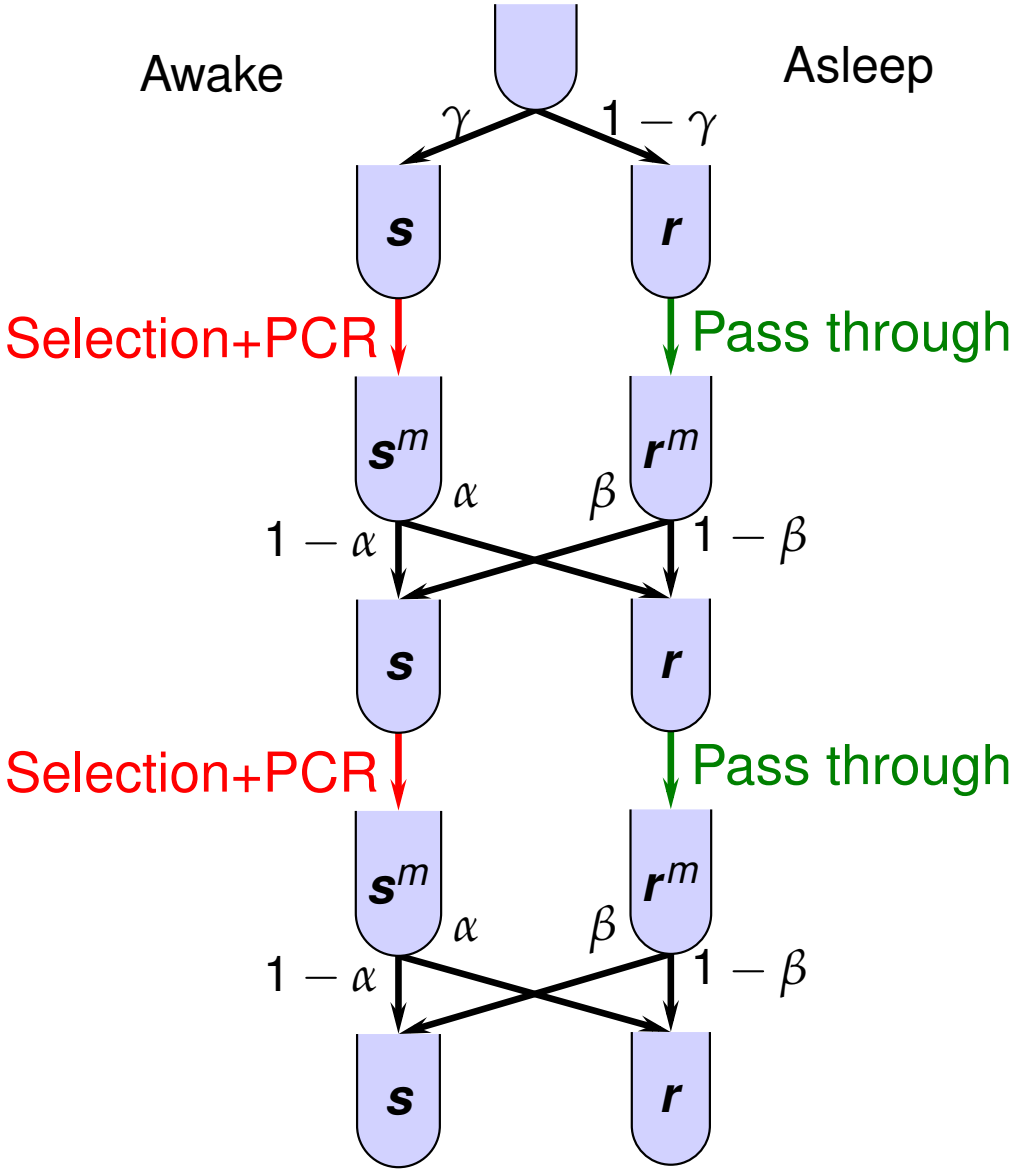
- Some models predict with predictive distribution
- Weights of those models fixed - Bayes rule is vacuous

# Two-track in-vitro implementation of sleeping

Multiplicative update + small soup exchange gives long-term memory

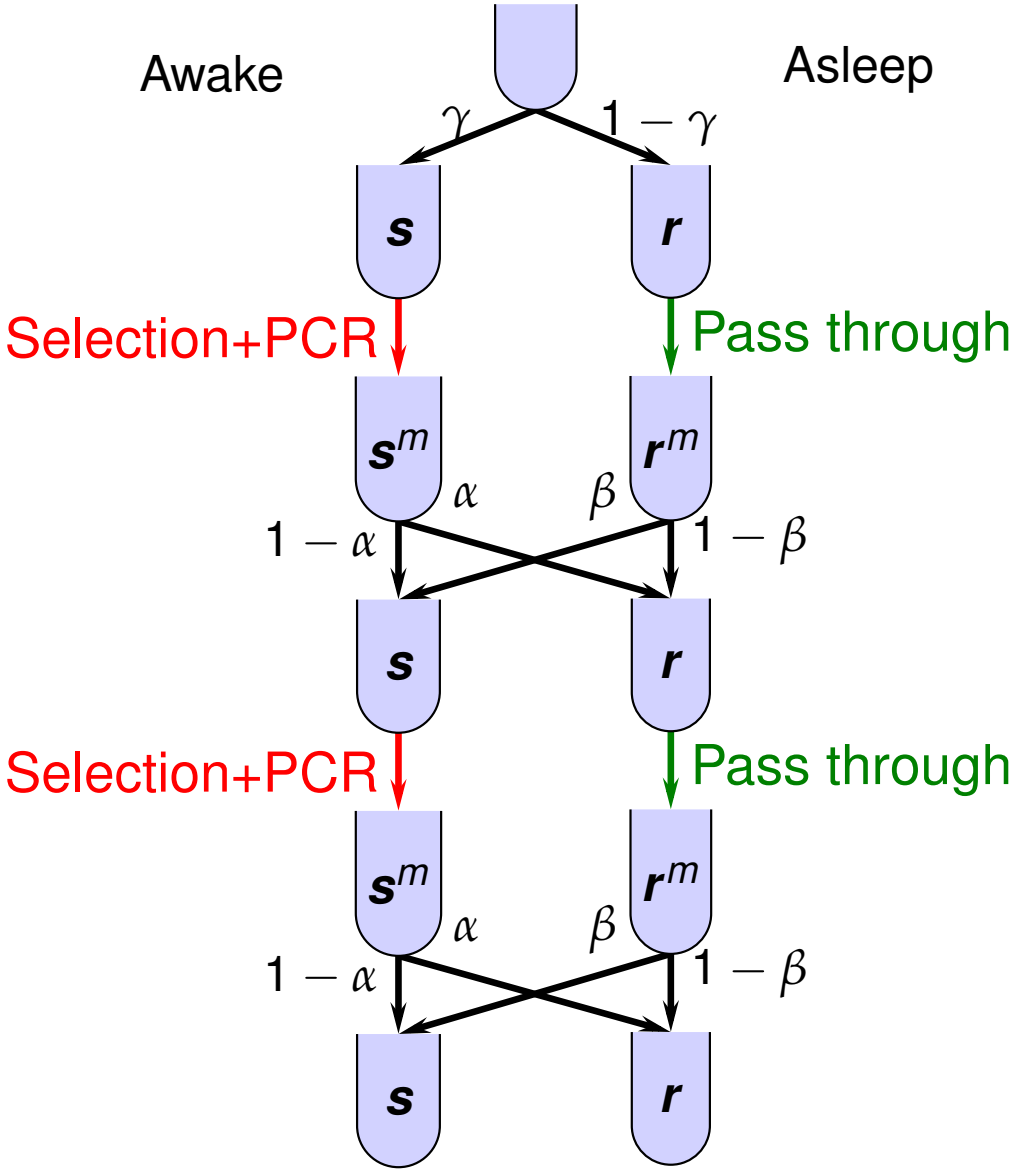
# Two-track in-vitro implementation of sleeping

Multiplicative update + small soup exchange gives long-term memory



# Two-track in-vitro implementation of sleeping

Multiplicative update + small soup exchange gives long-term memory




Initially:

$\boldsymbol{s} = \gamma$ "initial tube"

$\boldsymbol{r} = (1-\gamma)$ "initial tube"

$\boldsymbol{s}^m =$ "mult.update"$(\boldsymbol{s})$

$\boldsymbol{r}^m = \boldsymbol{r}$

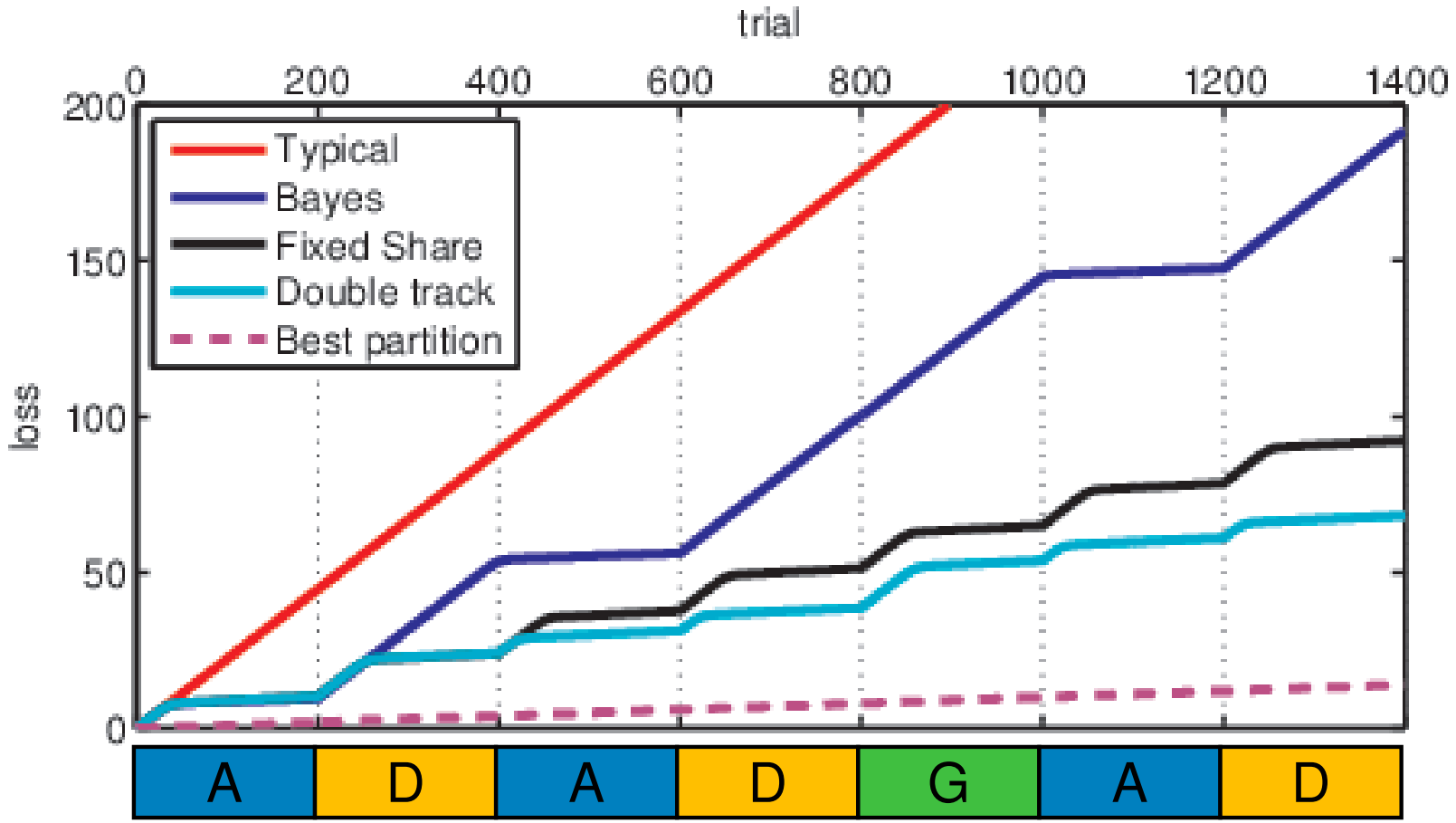
Soup exchange:

$\tilde{\boldsymbol{s}} = (1-\alpha)\boldsymbol{s}^m + \beta \boldsymbol{r}^m$

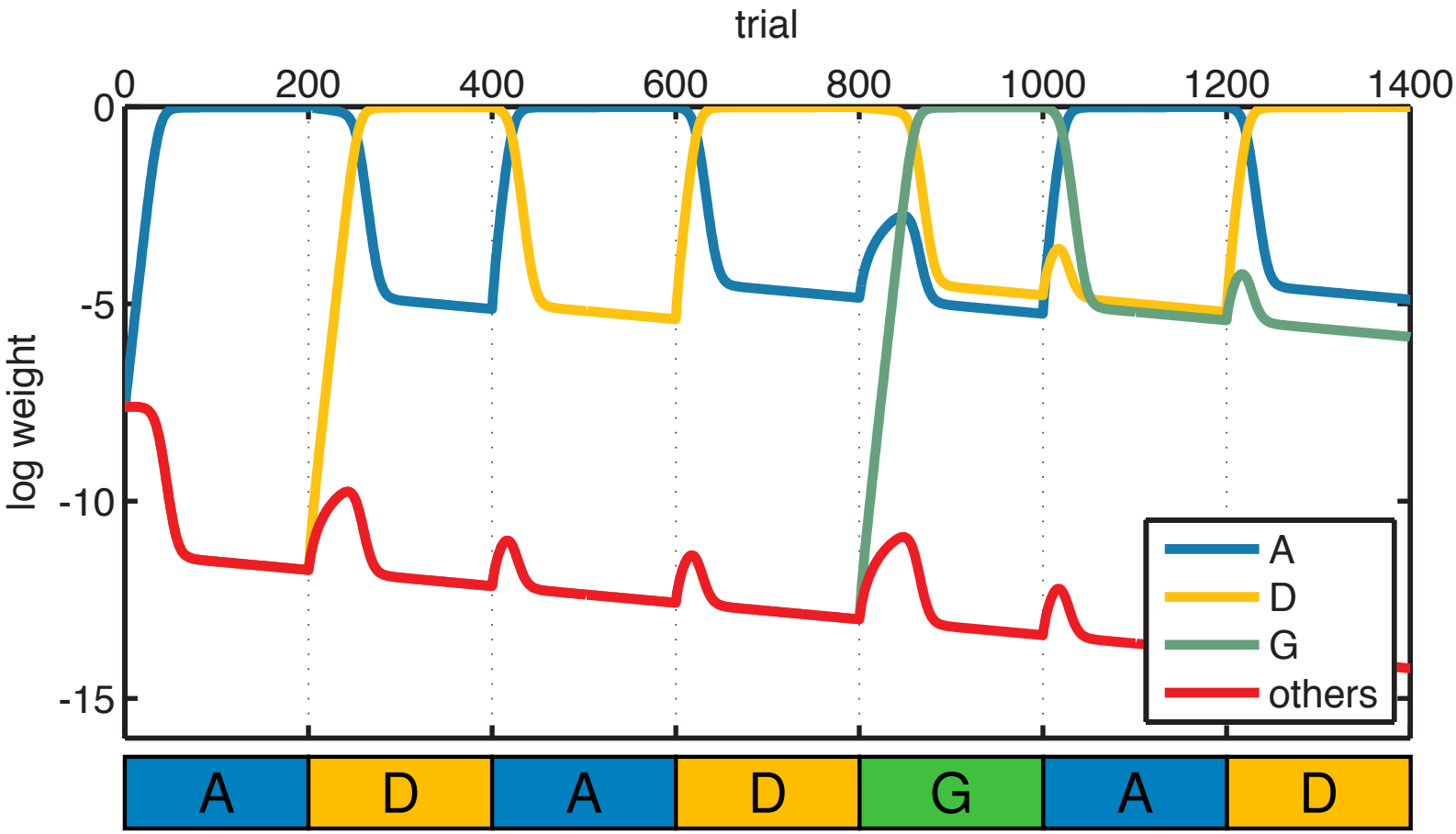
$\tilde{\boldsymbol{r}} = \alpha \boldsymbol{s}^m + (1-\beta)\boldsymbol{r}^m$

$\gamma = \frac{\beta}{\alpha+\beta}$

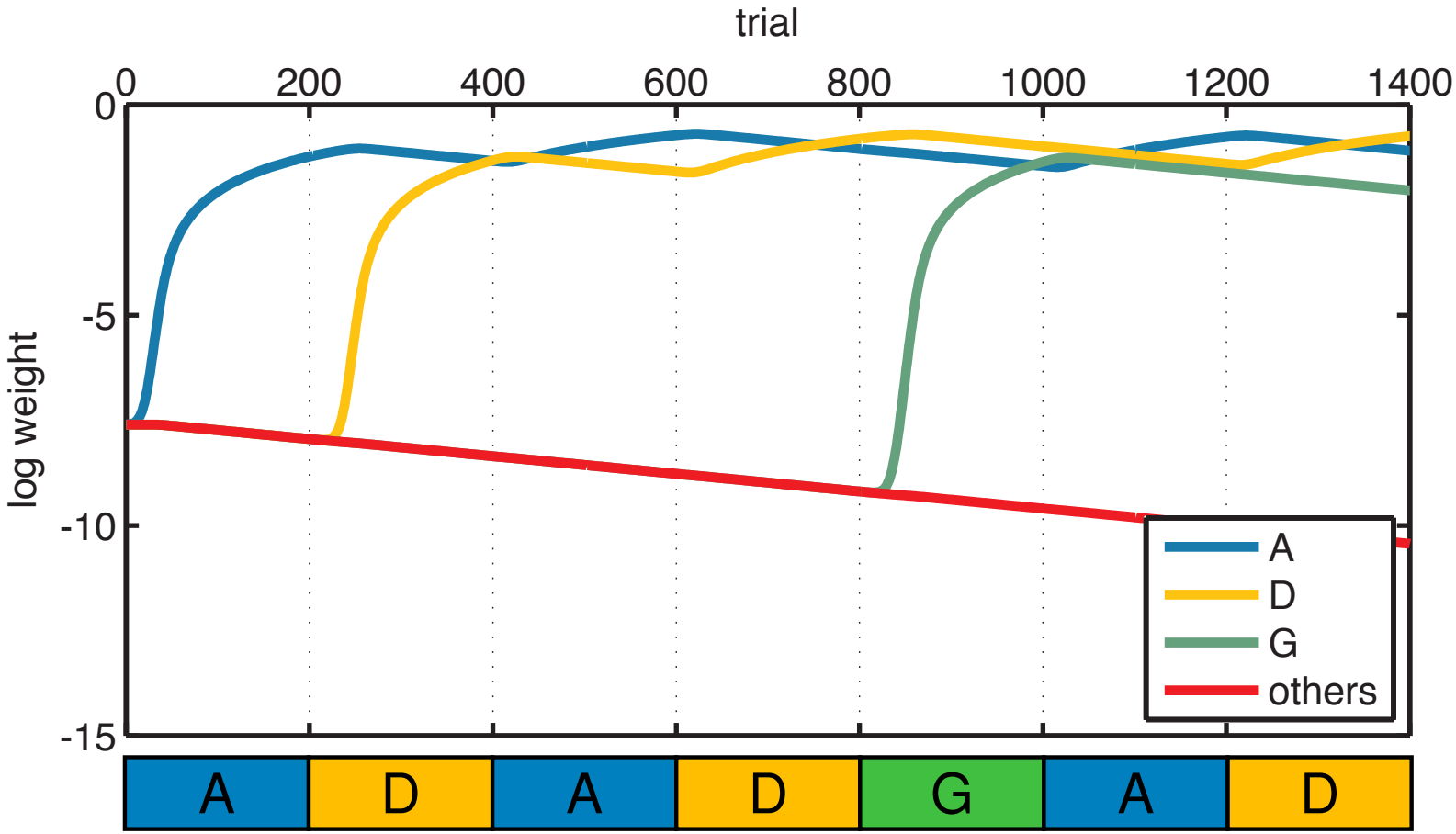
# Total loss plots



# Weights $\boldsymbol{s}$ on the awake track



# Weights *r* on the sleeping track



# Question

- How long-term memory realized in nature?
  - Junk DNA?
  - Sex?

## ML methods

ML 1: Conservative update - learn multiple goals
ML 2: Upper bounding weights - ”
ML 3: Lower bounding weights - robust against change
ML 4: Mixing in past average/sleeping - longterm memory

## Nature’s methods

Nature 1: Boundaries
Nature 2: Coupling
Nature 3: Super-predators
Nature 4: Mutations

# Summary

- Multiplicative updates converge quickly - their blessing but wipe out diversity - their curse
- Changing conditions require reuse of previously learned knowledge/alternatives
- Diversity is a requirement for success
- A mechanism is need to ameliorate the curse
  - ML and Nature have different tricks

# Final question 1

Would three tracks imbue a layered long-term memory

# Final question 2

- Does nature use the matrix version of the multiplicative updates?

# Final question 2

- Does nature use the matrix version of the multiplicative updates?
- Parameter is density matrix
  - maintains uncertainty over directions as symmetric positive matrix $\boldsymbol{S}$

$$\widetilde{\boldsymbol{S}} := \frac{\exp(\log \boldsymbol{S} - \eta \nabla L(\boldsymbol{S}))}{\text{trace of above}}$$

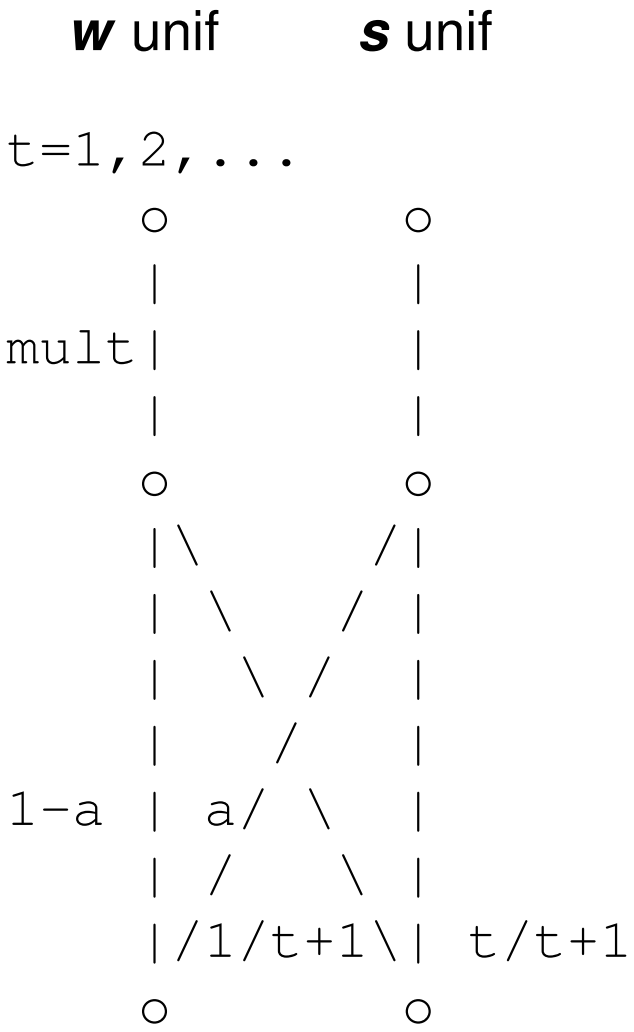
# Final question 2

- Does nature use the matrix version of the multiplicative updates?
- Parameter is density matrix
  - maintains uncertainty over directions as symmetric positive matrix $\boldsymbol{S}$

$$\widetilde{\boldsymbol{S}} := \frac{\mathbf{exp}(\mathbf{log}\boldsymbol{S} - \eta \nabla L(\boldsymbol{S}))}{\text{trace of above}}$$

- Quantum relative entropy as regularizer

# Fixed Share to Uniform Past

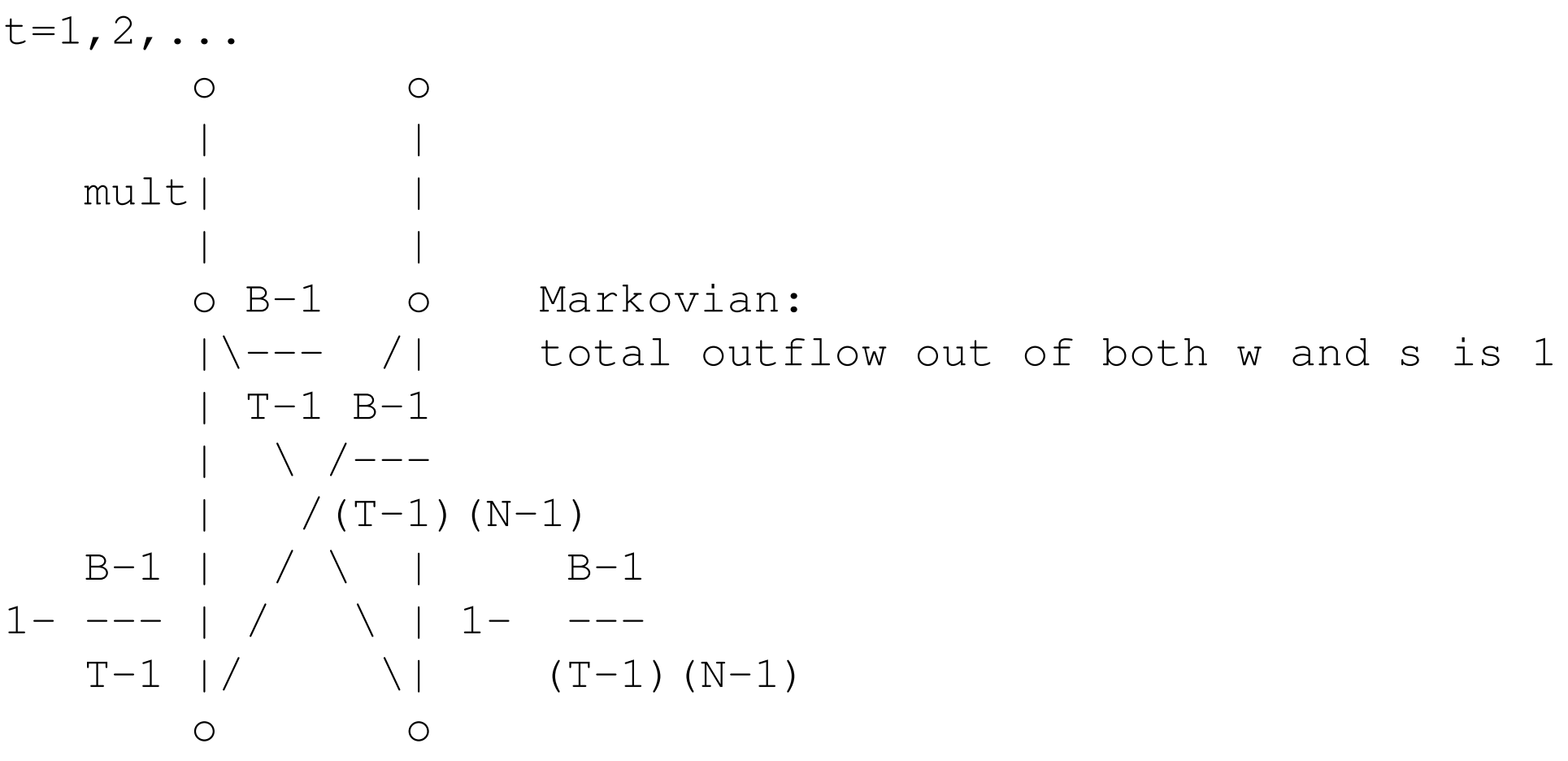



total inflow into w and s is 1

***w*** and ***s*** remain probability vectors

# Fixed Share to Exponentially Decaying Past

Tuned for $B$ shifts in $T$ trials
Shift among small pool of $N$ in large pool of $M$ $\quad \boldsymbol{w} = (\frac{1}{M}, \ldots, \frac{1}{M})\frac{1}{N}$
$\boldsymbol{s} = (\frac{1}{M}, \ldots, \frac{1}{M})(1 - \frac{1}{N})$

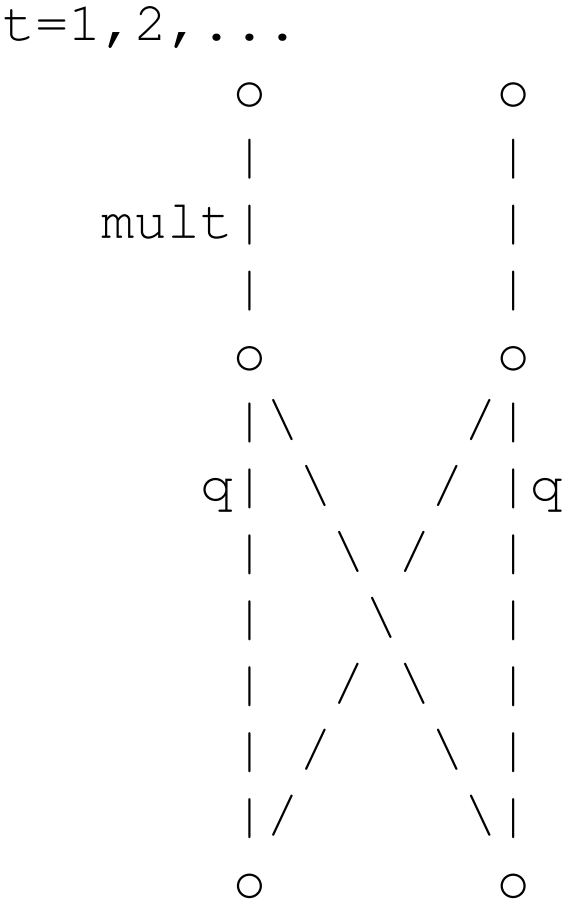



$\boldsymbol{w}$ and $\boldsymbol{s}$ remain at total weight $\frac{1}{N}$ and $1 - \frac{1}{N}$, resp.

# Fixed Share to Exponentially Decaying Past

Number of segments $B$ and horizon $T$ unknown

$\boldsymbol{w} = \frac{\text{unif}}{2}$ $\quad$ $\boldsymbol{s} = \frac{\text{unif}}{2}$




Markovian:
total outflow out of both w and s is 1

where $q = exp(-\frac{1}{t \ln^2(t+1)})$

$\boldsymbol{w}$ and $\boldsymbol{s}$ remain at total weight $\frac{1}{2}$