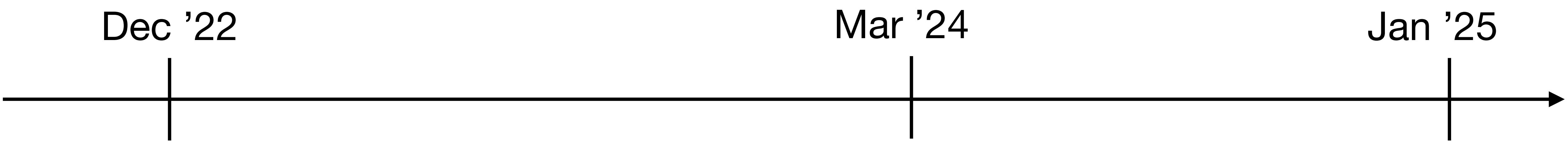
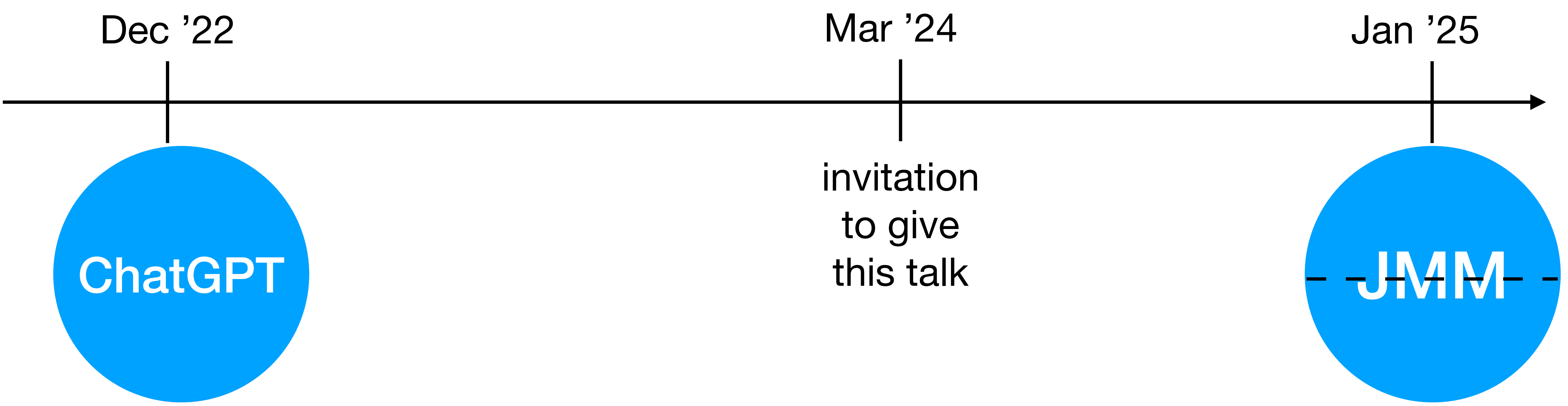
Problems in AI and Machine Learning for Mathematicians

AMS Special Session on AI for the Working Mathematician, JMM 2025

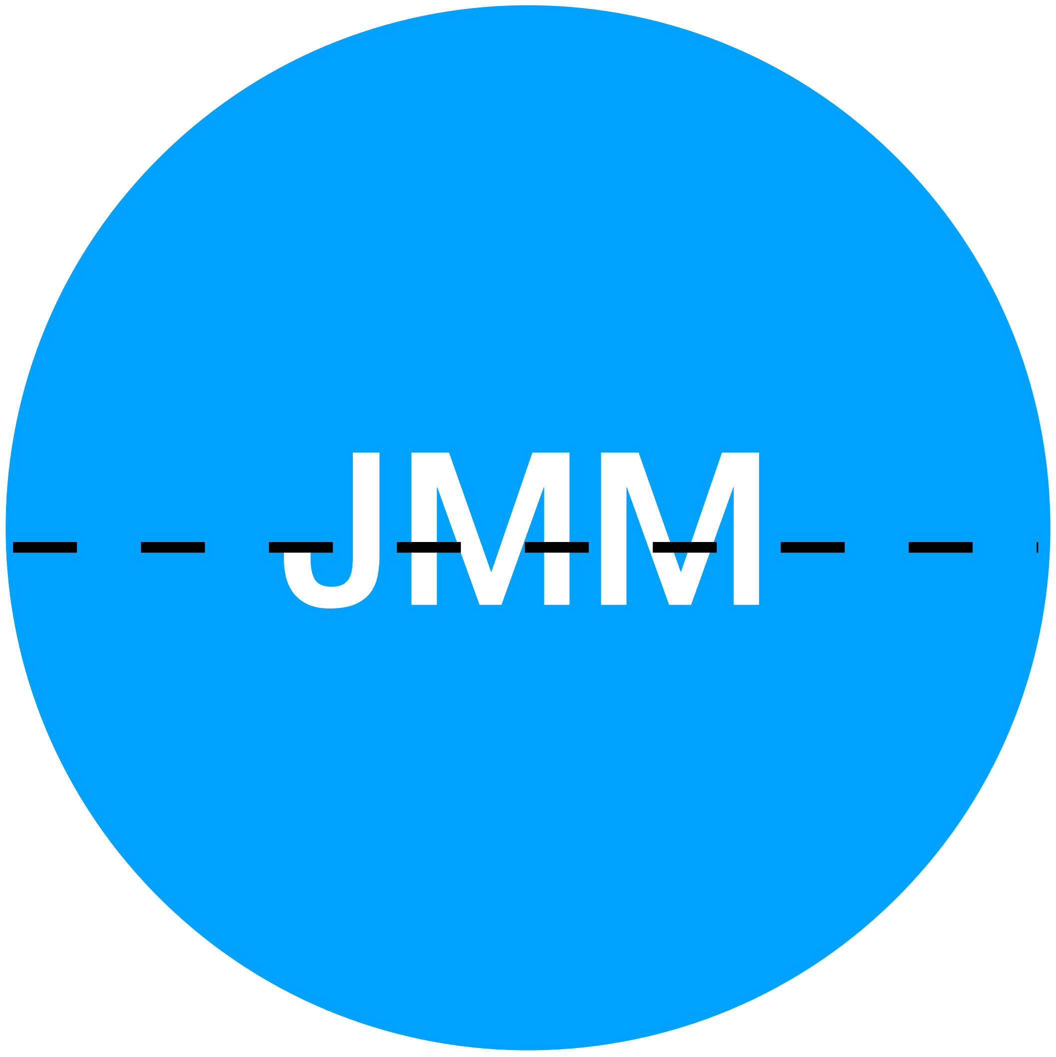
Lev Reyzin
Department of Mathematics
University of Illinois Chicago


Dec ’22
Mar ’24
Jan ’25


Dec ’22
Mar ’24
Jan ’25
ChatGPT
invitation
to give
this talk
JMM

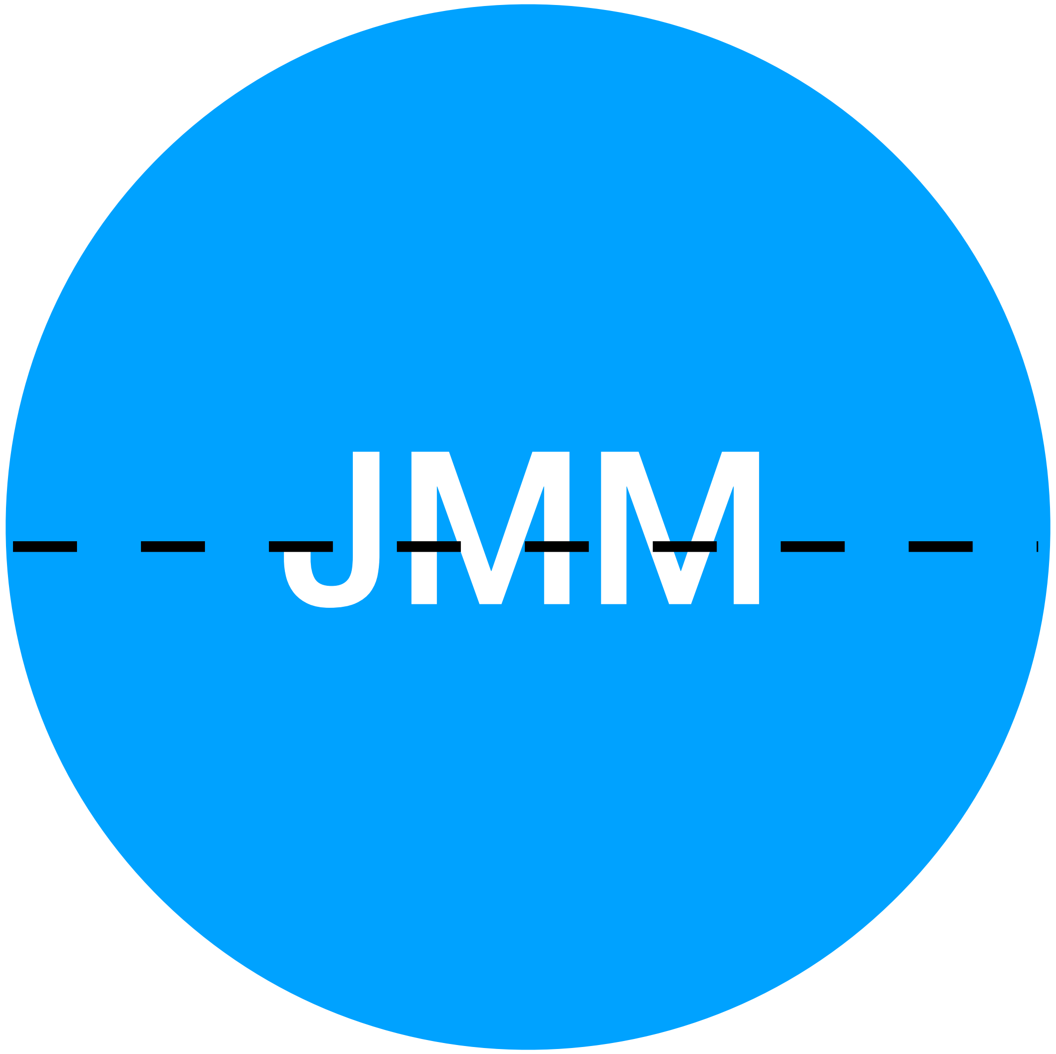
“Consider all measurable subsets of the real interval…”

“Let f be an analytic sequence that converges uniformly…”



“Consider all measurable subsets of the real interval…”

“Let f be an analytic sequence that converges uniformly…”



“How can we automate theorem proving in lean?”

“How can I take advantage of current AI technologies?”

“When will I lose my job?”

AI for Mathematicians

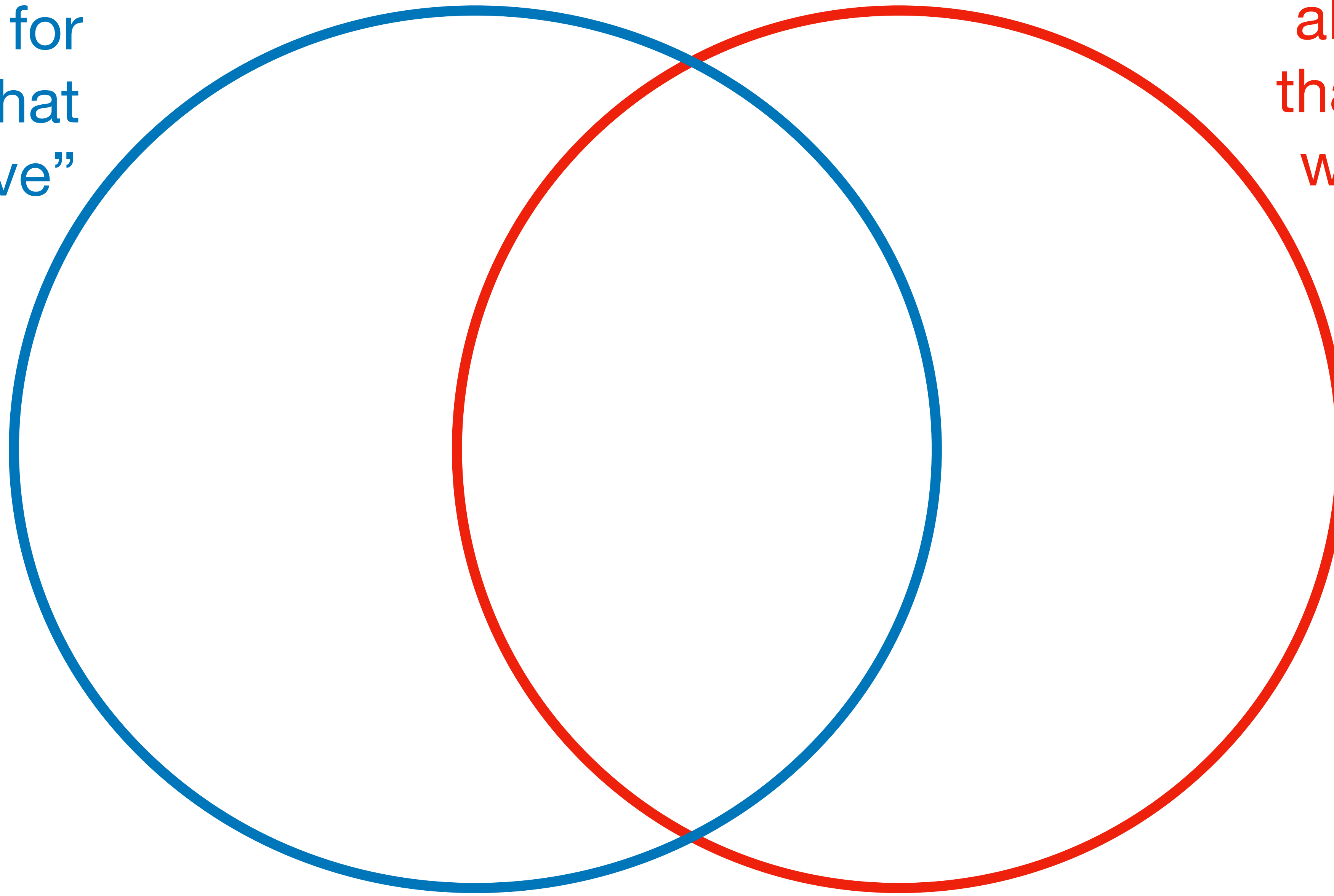
How have mathematicians used AI? How can they?*

*This wasn't a planned part of the talk, but the state of the art has quickly changed!

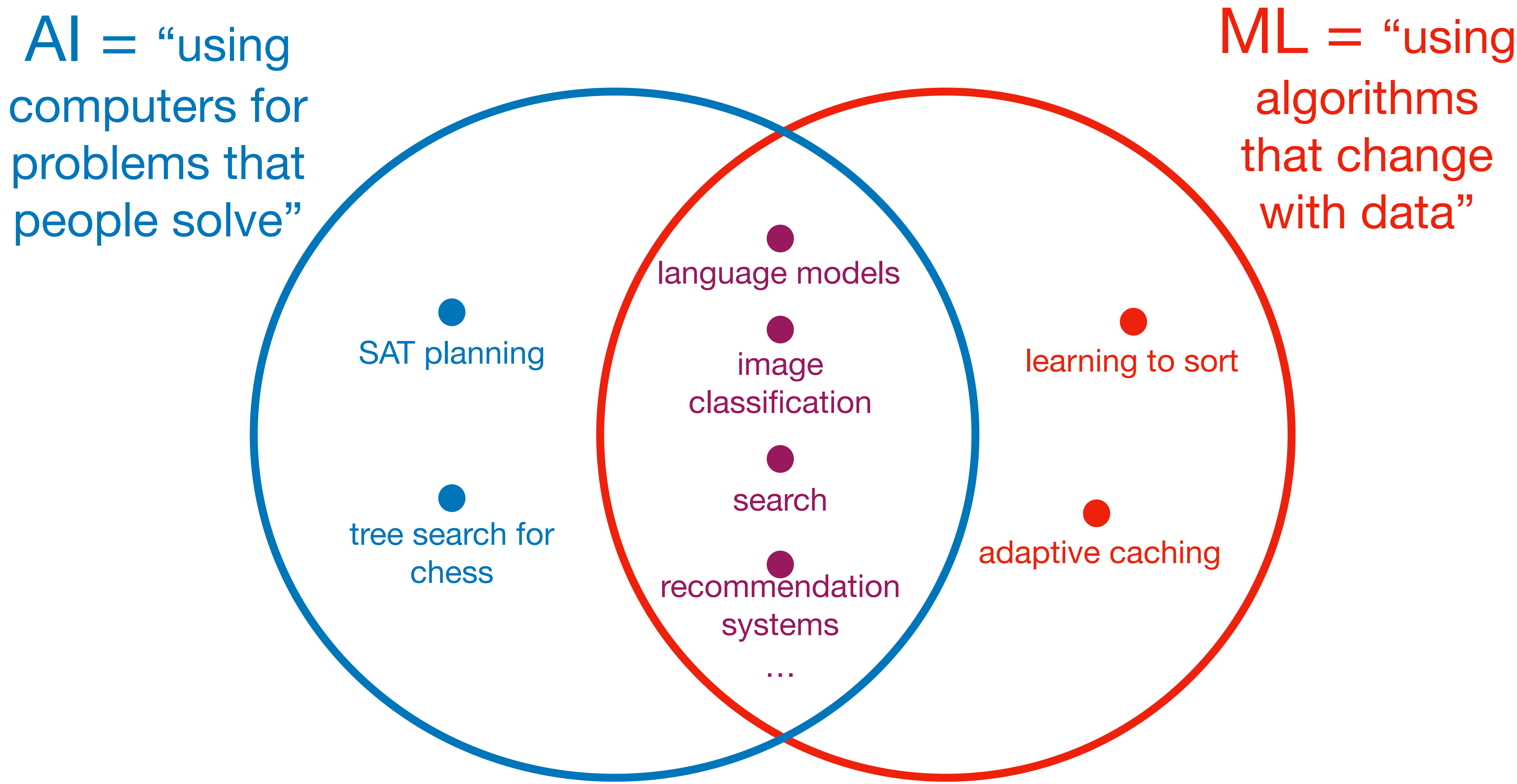
Two definitions

AI = “using computers for problems that people solve”

ML = “using algorithms that change with data”



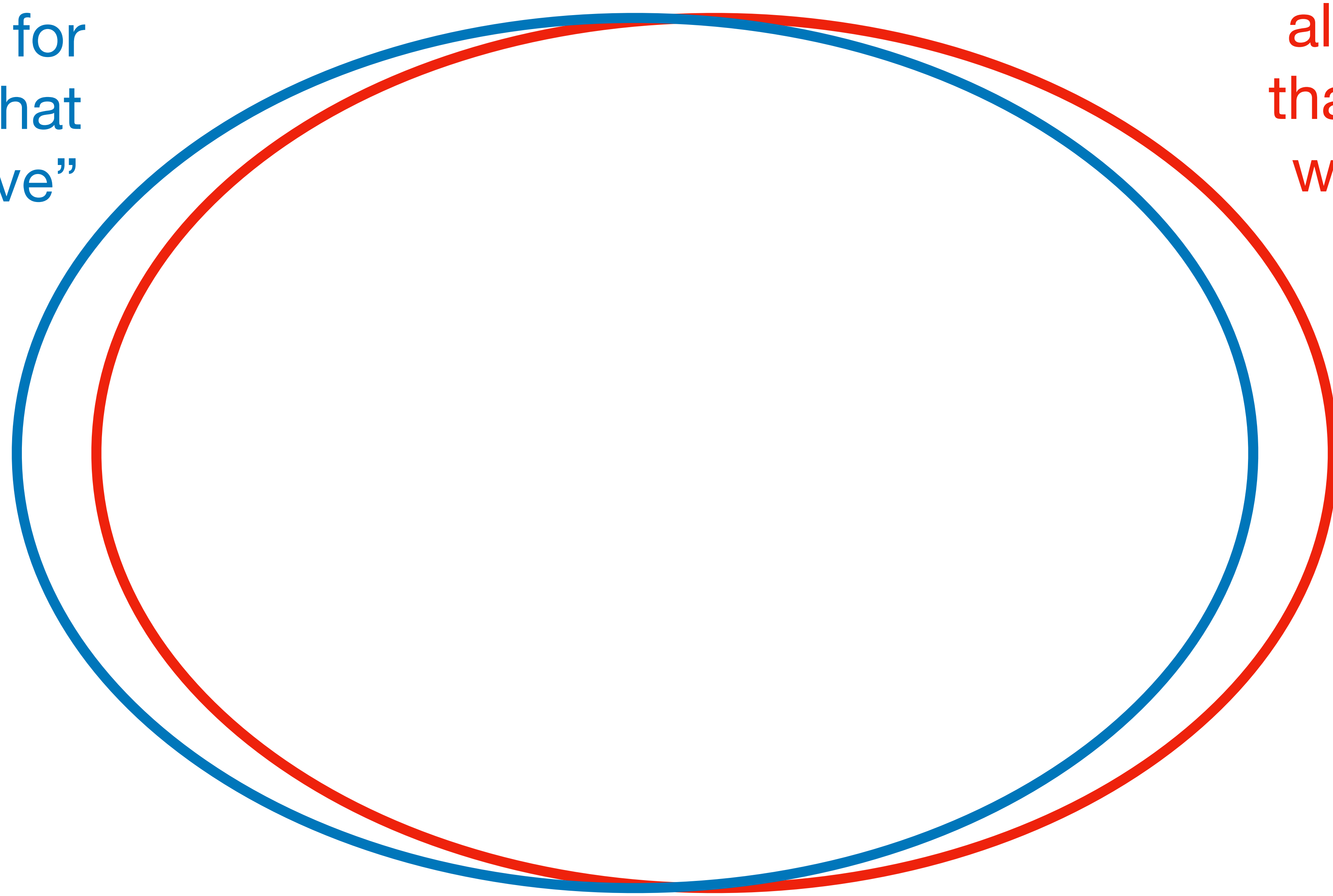
Two definitions



Over time…

AI = "using computers for problems that people solve"

ML = "using algorithms that change with data"



SAT solvers

SAT solvers, from cstheory.stackexchange.com:

The following question arises out of Ramsey Theory.

Consider a k-coloring of the n-by-m grid graph. A `monochromatic rectangle` exists whenever four cells with the same color are arranged as the corners of some rectangle. For example, $(0,0),(0,1),(1,1)$, and $(1,0)$ form a monochromatic rectangle if they have the same color. Similarly, $(2,2),(2,6),(3,6)$, and $(3,2)$ form a monochromatic rectangle, if colored with the same color.

Question: Does there exist a 4-coloring of the 17-by-17 grid graph that does not contain a monochromatic rectangle? If so, provide the explicit coloring.

Some known facts:

- 16-by-17 *is* 4-colorable without a monochromatic rectangle, but the known coloring scheme does not appear to extend to the 17-by-17 case. (I'm omitting the known 16-by-17 coloring because it would very likely be a red herring for deciding 17-by-17.)
- 18-by-19 is *NOT* 4-colorable without a monochromatic rectangle.
- 17-by-18 and 18-by-18 are also unknown cases; an answer to these would be interesting as well.

SAT solvers

SAT solvers, from cstheory.stackexchange.com:

13

This is not really an answer to the question, but I've encoded the 17x17 4-coloring problem as a 4-CNF (in the standard DIMACS format for SAT-solvers) and uploaded it here. If anyone has access to a good SAT solver (and a supercomputer!) maybe we can make some progress.

Note: in my encoding, if gridpoint (i, j) *is assigned color* $c \in \{0, 1, 2, 3\}$, *then the variable* $(17i + j + 289c + 1)$ *takes the value* 1, *and* 0 *otherwise.*

Mod Share Cite Edit Delete Flag

answered Sep 27, 2010 at 13:38

Lev Reyzin ♦
12.1k ● 13 ● 65 ● 103

SAT solvers

SAT solvers, from cstheory.stackexchange.com:

13

This is not really an answer to the question, but I've encoded the 17x17 4-coloring problem as a 4-CNF (in the standard DIMACS format for SAT-solvers) and uploaded it **here**. If anyone has access to a good SAT solver (and a supercomputer!) maybe we can make some progress.

Note: in my encoding, if gridpoint (i, j) *is assigned color* $c \in \{0, 1, 2, 3\}$*, then the variable* $(17i + j + 289c + 1)$ *takes the value* 1*, and* 0 *otherwise.*

Mod Share Cite Edit Delete Flag

answered Sep 27, 2010 at 13:38

Lev Reyzin

12.1k 13 65 103

4 @Lev, just a random update: it appears the runtime of the 17x17, even using the best possible supercomputer and a really fast SAT solver, is still astronomical. Plus side: it appears within the realm of reason to

SAT solvers

SAT solvers, from cstheory.stackexchange.com:

13

This is not really an answer to the question, but I've encoded the 17x17 4-coloring problem as a 4-CNF (in the standard DIMACS format for SAT-solvers) and uploaded it **here**. If anyone has access to a good SAT solver (and a supercomputer!) maybe we can make some progress.

Note: in my encoding, if gridpoint (i, j) is assigned color $c \in \{0, 1, 2, 3\}$, then the variable $(17i + j + 289c + 1)$ takes the value 1, and 0 otherwise.

Mod Share Cite Edit Delete Flag

answered Sep 27, 2010 at 13:38

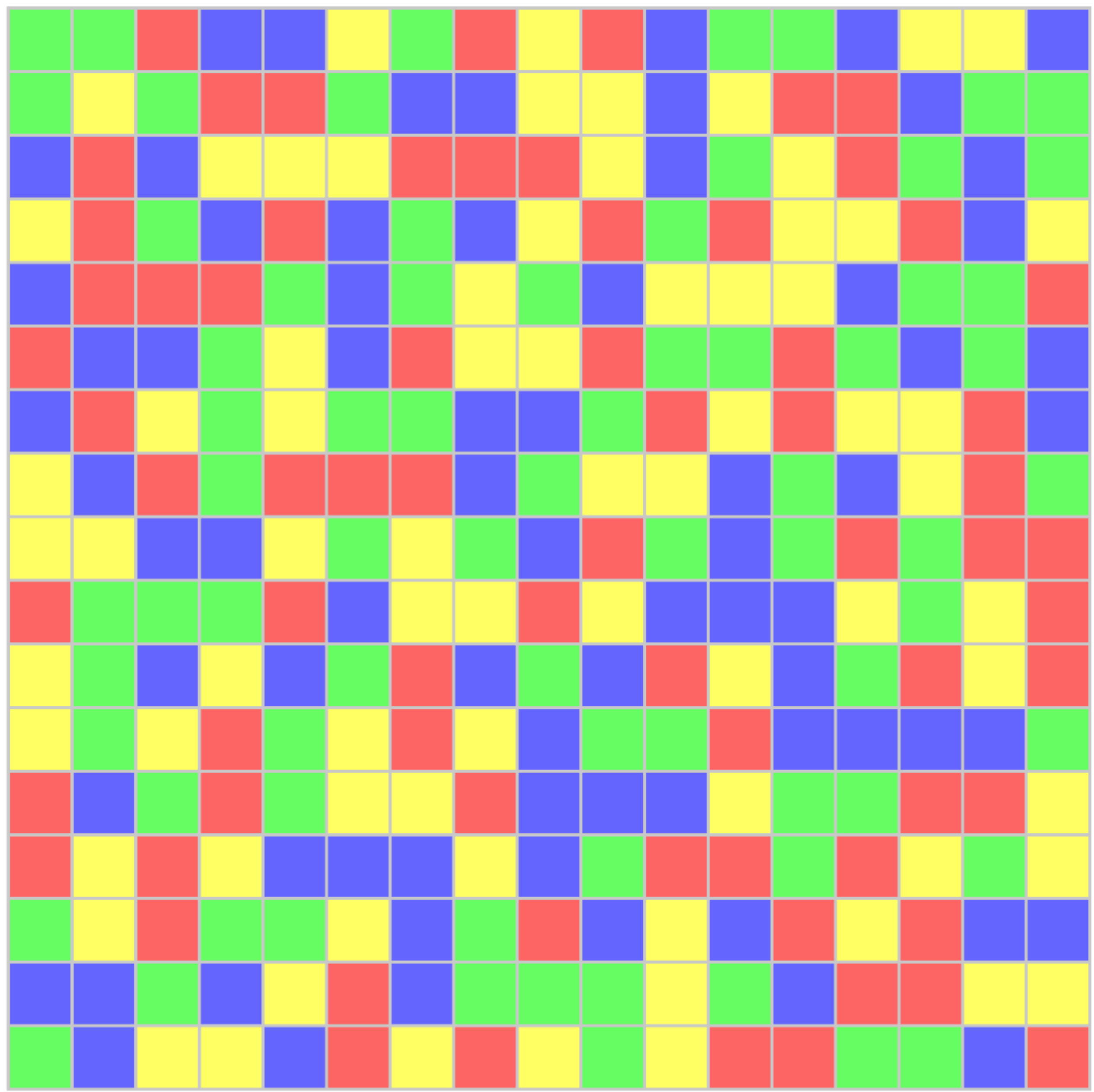
Lev Reyzin

12.1k 13 65 103

4 @Lev, just a random update: it appears the runtime of the 17x17, even using the best possible supercomputer and a really fast SAT solver, is still astronomical. Plus side: it appears within the realm of reason to

problem. Personally, after spending a large amount of time marinating my brain in this problem, I'm willing to hypothesize that there is no legal 4-coloring of the 17x17 grid. As a result, all of the approaches I'm personally investigating involve the ability to prove the hypothesis by (essentially) clever brute-force search. Down side: No approximate

SAT solvers



Steinbach and Posthoff ’12

SAT solver sucesses

SAT solvers (list from Bernardo Subercaseaux's talk on Wedneday):

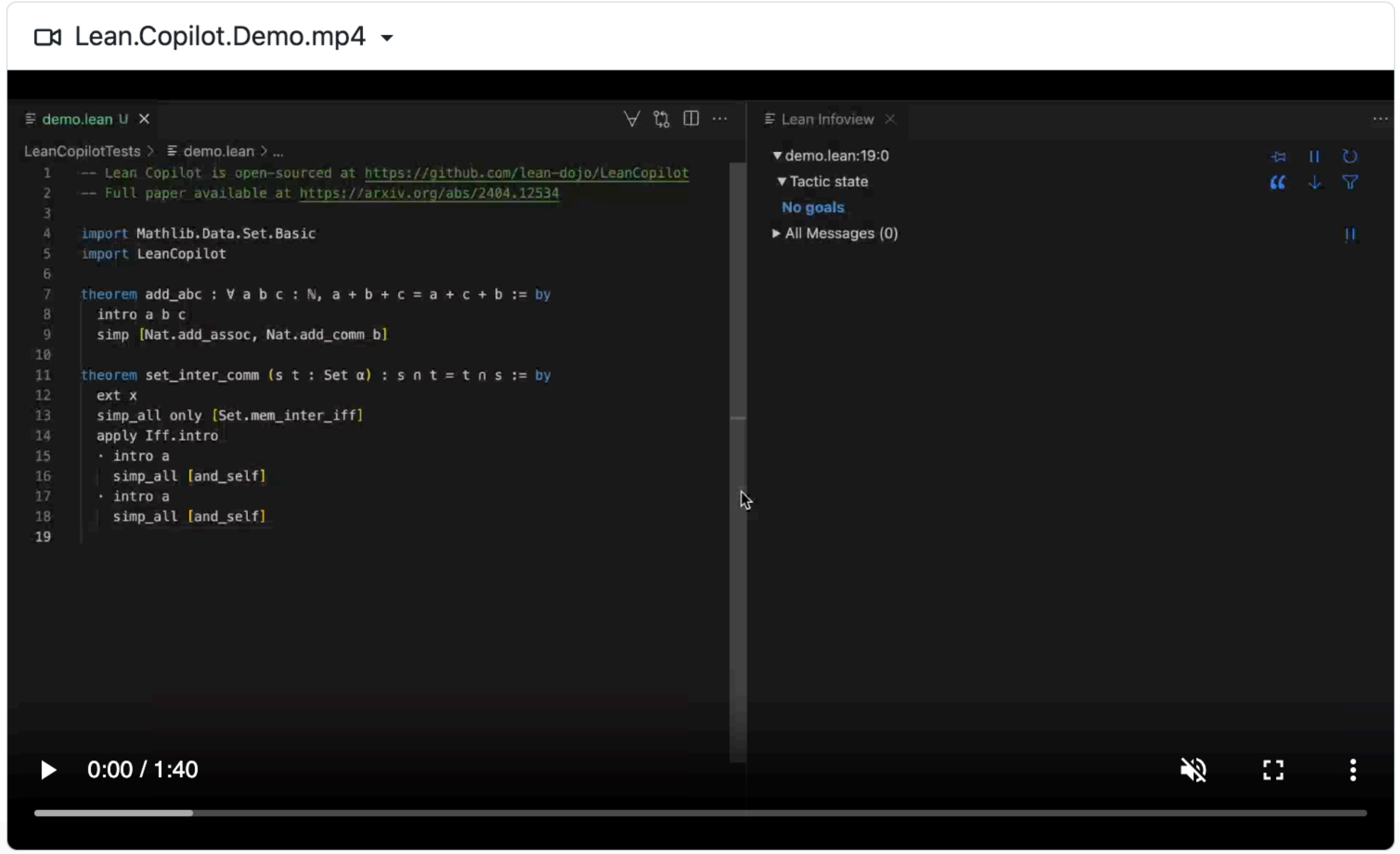
- (2014) Boolean Erdős Discrepancy Problem
- (2016) Boolean Pythagorean Triples
- (2018) Schur Number 5
- (2019) Keller's Conjecture
- (2023) Packing Chromatic Number of Z^2
- (2024) Empty Hexagon every 30 Points

Lean with copilot

AI assistants for lean

Lean Copilot: LLMs as Copilots for Theorem Proving in Lean

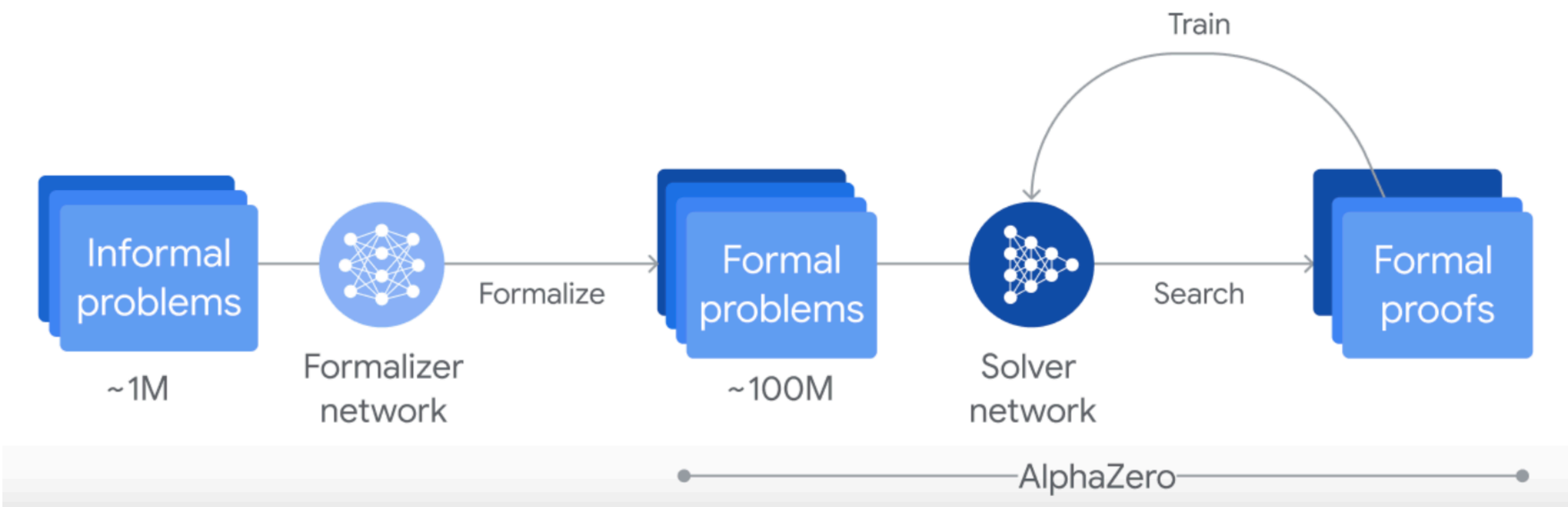
Lean Copilot allows large language models (LLMs) to be used in Lean for proof automation, e.g., suggesting tactics/premises and searching for proofs. You can use our built-in models from LeanDojo or bring your own models that run either locally (w/ or w/o GPUs) or on the cloud.



Deep learning + Tree search

AlphaProof

AI achieves silver-medal standard solving International Mathematical Olympiad problems

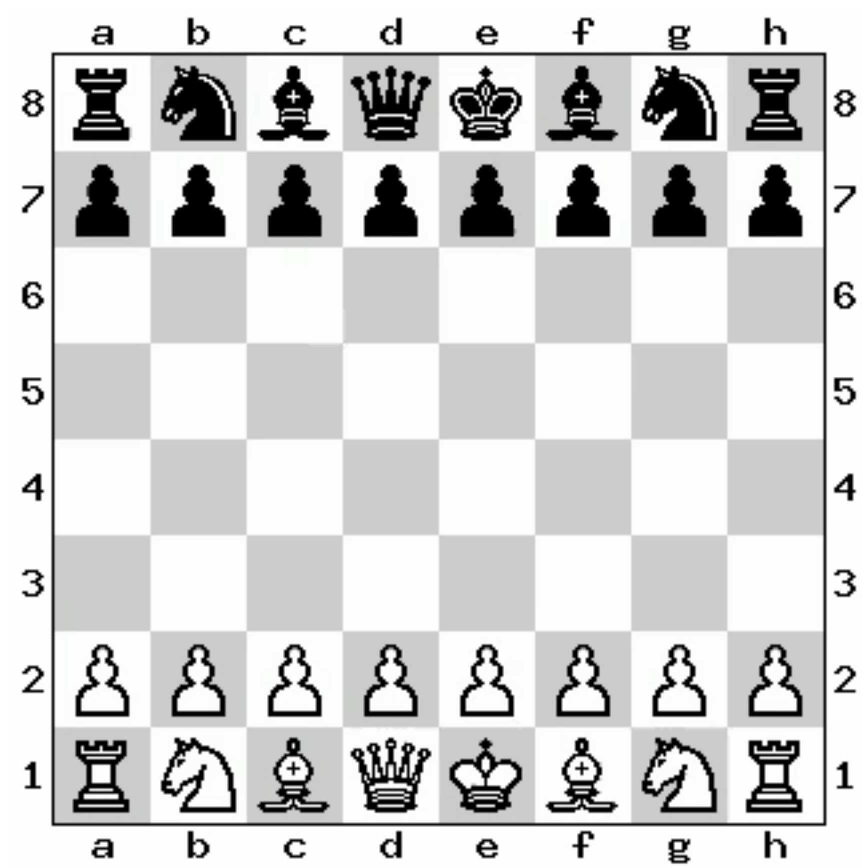


LLMs

On the other hand, there are many examples of LLMs like ChatGPT failing at proofs and other more formal taks, including chess.

Example from Alex Kontorovich's Wednesday talk:

ChatGPT



Stockfish

LLM failures at proof

Can you prove that P is contained in SPACE(n)?

Proof Sketch: $\mathbf{P} \subseteq \mathbf{SPACE}(n)$

- Consider a deterministic Turing machine M that decides a language in $O(n^c)$ time, where $c \geq 1$.
- At any step, the machine's configuration can be represented by:
  - The current state (q),
  - The position of the tape head ($O(n^c)$ differer possible positions over time),
  - The tape contents (limited to the portion of th input accessed, which is at most $O(n^c)$).
- Instead of storing the entire tape history, the machine can recompute prior configurations usinc small, constant amount of workspace, effectively using only $O(n)$ space to simulate the computation.

On the other hand...

ChatGPT launched just ~2 years ago, now according to Sam Altman…

> “We are now confident we know how to build AGI as we have traditionally understood it. We believe that, in 2025, we may see the first AI agents “join the workforce” and materially change the output of companies. We continue to believe that iteratively putting great tools in the hands of people leads to great, broadly-distributed outcomes.”

Sebastien Bubeck
@SebastienBubeck

This year I turn 40. Would be nice to celebrate half-life with AGI ...

5:30 PM · Jan 6, 2025 · **38.8K** Views

A positive story

max affine regression uses the maximum of k affine functions in a d-dimensional space to represent an unknown function.

- known to be NP-hard (to find such functions from data) in high dimensions, even for k = 2.
- for some research project, I needed it to be NP-hard in low dimensions for large k.

Asking ChatGPT

I could probably prove it myself, but I decided to ask ChatGPT.

> I want to argue that the maximum of k affine functions in R^d is computationally hard to compute when k is high, even for low/constant d when there's noise. Do you have any suggestions on how to tackle this problem?

Your problem is intriguing, as it touches on computational complexity and learning theory, especially in the presence of noise. To argue that the computation is hard, you can use a combination of theoretical reductions, hardness results from optimization, and examples of noise-influenced decision boundaries. Here are some concrete suggestions:

⋮

Guiding ChatGPT

can you help me come up with a reduction from k-means to my problem?

A reduction from k-means clustering to your problem of computing the maximum of k affine functions in $\mathbb{R}^d$ is a strong approach because k-means is known to be NP-hard in the general case. Here's how you might construct such a reduction:

⋮

Guiding ChatGPT

Eventually it gets to the correct proof…

Transform k-Means to k-Affine Functions

- The squared Euclidean distance can be rewritten in terms of inner products:

$$\|x - \mu_j\|^2 = \|x\|^2 - 2\langle x, \mu_j \rangle + \|\mu_j\|^2.$$

  - Here, $\|x\|^2$ is independent of j, so for a fixed x, minimizing the distance is equivalent to maximizing:

$$\langle x, \mu_j \rangle - \frac{1}{2}\|\mu_j\|^2.$$

- Define affine functions $f_j(x) = \langle x, \mu_j \rangle - \frac{1}{2}\|\mu_j\|^2$. Then the cluster assignment becomes:

$$i = \arg\max_j f_j(x).$$

- Thus, the k-means clustering problem reduces to identifying the maximum of k affine functions for each data point x.

Testing ChatGPT's "understanding"

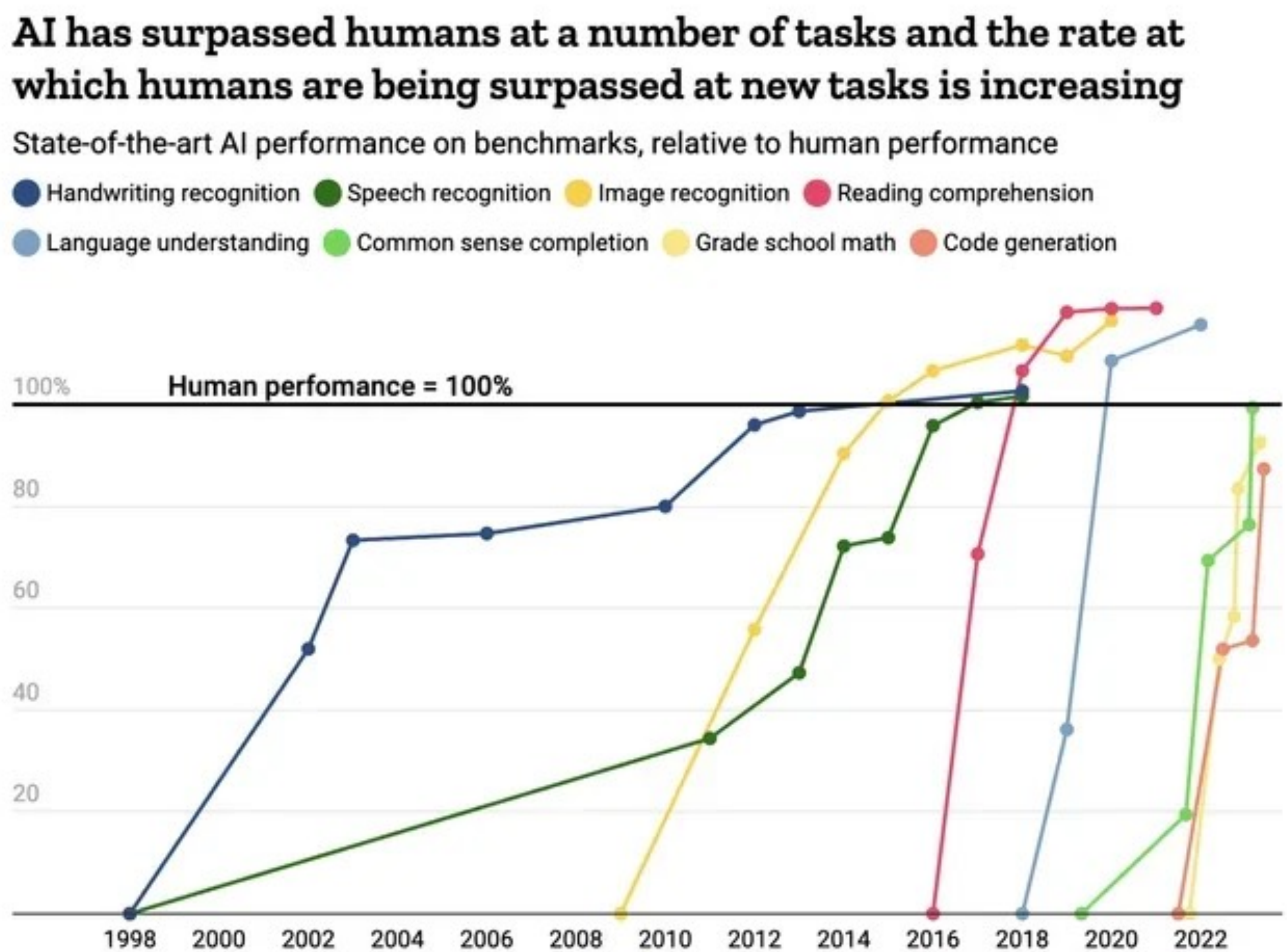
are the f_j's really affine? isn't the term on the \mu's quadratic?

You're absolutely correct! The functions $f_j(x) = \langle x, \mu_j \rangle - \frac{1}{2}\|\mu_j\|^2$ are indeed **affine in x**, but the term $-\frac{1}{2}\|\mu_j\|^2$ is a constant with respect to x. Let's break it down clearly to confirm why $f_j(x)$ is affine:

We have reached the point where generic LLMs can become useful research tools. Try ChatGPT-o1 if you don't believe me!

What makes math different?

Matus Telgarsky (from his talk on Thursday), proofs are amenable to Chess techniques. “Our days are numbered”



Mathematicians for AI

How can mathematicians improve AI? (Should we?)

How can we help understand AI?

LLMs

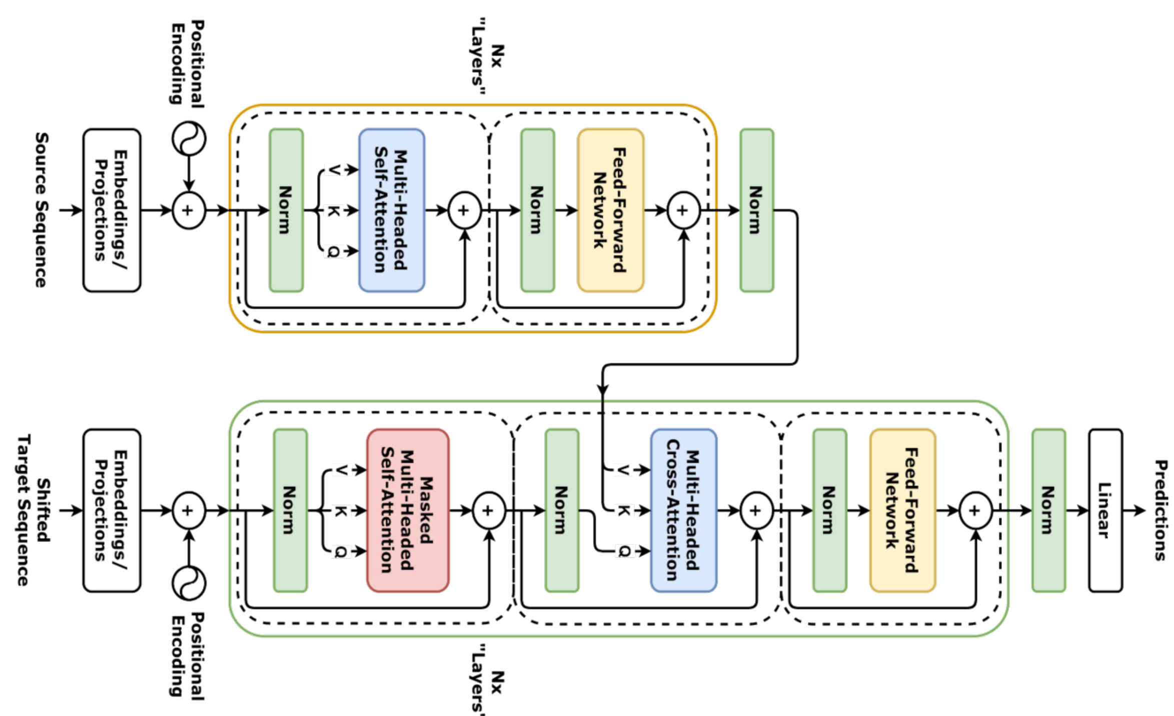
LLMs, like ChatGPT, are next-token predicters, given all previous tokens, which include the user input, (plus some internal network states).

e.g.
me: Why is 2+2 = 4?
ChatGPT: In formal mathematics, natural nu_

To achieve they use a transformer, which is a deep neural network with certain properties (e.g. self-attention, parallel processing, encoder/decoder).

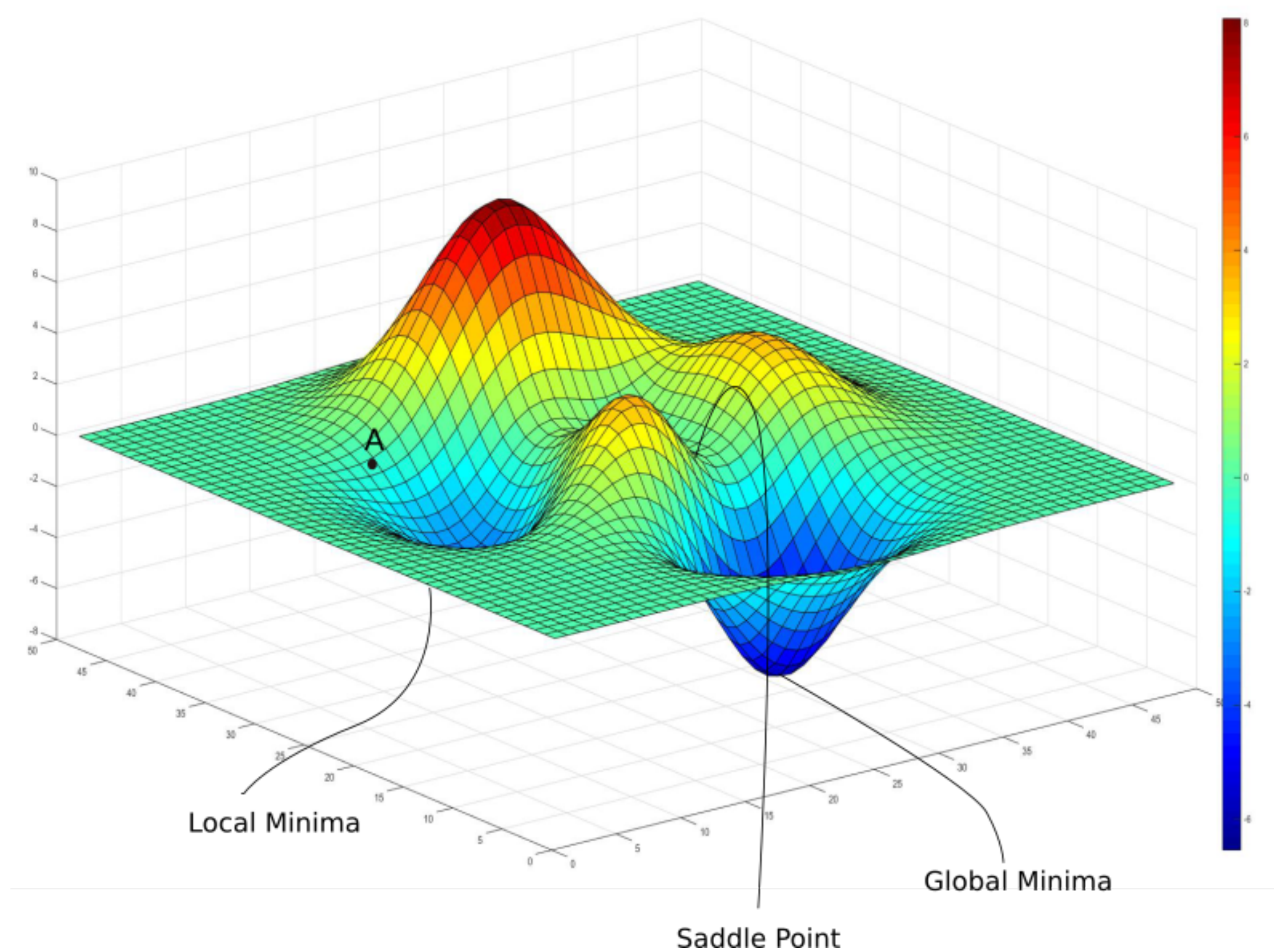
"Attention Is All You Need"
Vaswani et al. NIPS 2017

148334 citations
0 theorems



How can we help understand Deep Learning / LLMs?

Why does gradient descent work so well? Why do solutions reached by gradient descent generalize?



How can we help understand Deep Learning / LLMs?

Why don't deep neural networks overfit? GPT4 has hundreds of billions of parameters!

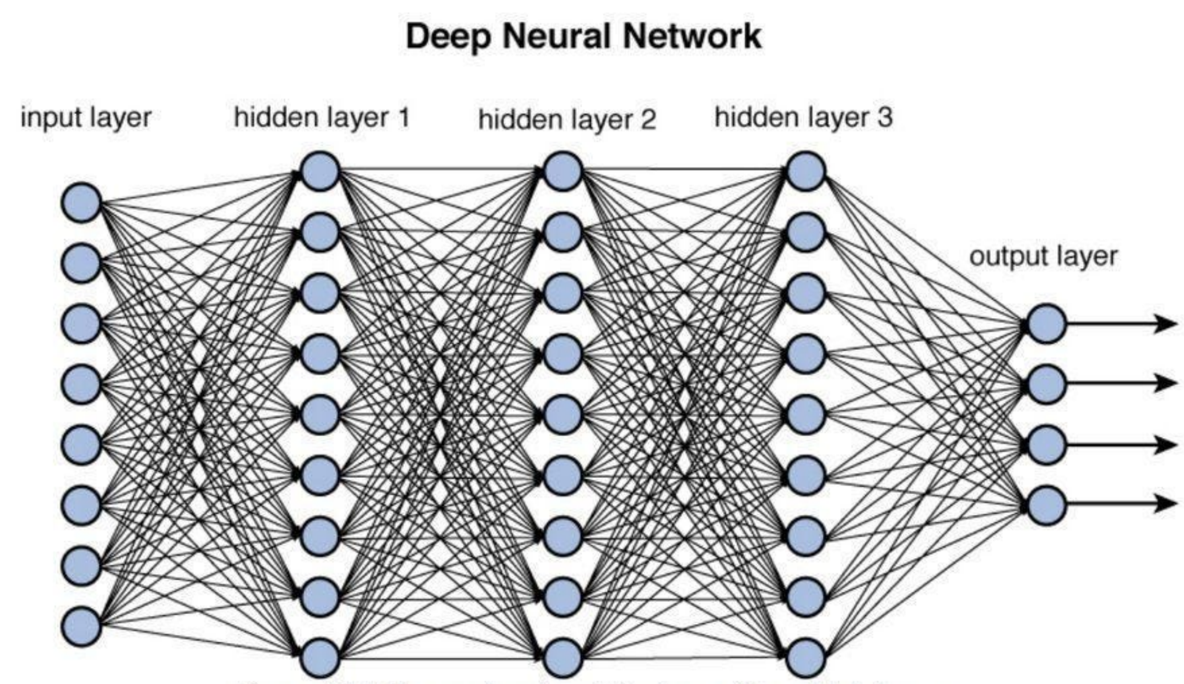


Figure 12.2 Deep network architecture with multiple layers.

We have some compelling theories, but not nearly as convincing as eg for more traditional methods like SVM and boosting.

Understanding + improving LLMs

- Speeding up training.
- Explainability.
- A mathematical theory of LLM architecture.
- Better optimization techniques?

Non-LLM example problem 1: Data reuse

Problem: data sets are often reused, which leads to overfitting.

"false discovery," "the garden of forking paths," "p-hacking"

Open access, freely available online

Essay

Why Most Published Research Findings Are False

John P. A. Ioannidis

Summary

There is increasing concern that most current published research findings are false. The probability that a research claim is true may depend on study power and bias, the number of other studies on the same question, and, importantly, the ratio of true to no relationships among the relationships probed in each scientific field. In this framework, a research finding is less likely to be true when the studies conducted in a field are smaller; when effect sizes are smaller; when there is a greater number and lesser preselection of tested relationships; where there is greater flexibility in designs, definitions, outcomes, and analytical modes; when there is greater financial and other interest and prejudice; and when more teams are involved in a scientific field in chase of statistical significance. Simulations show that for most study designs and settings, it is more likely for

factors that influence this problem and some corollaries thereof.

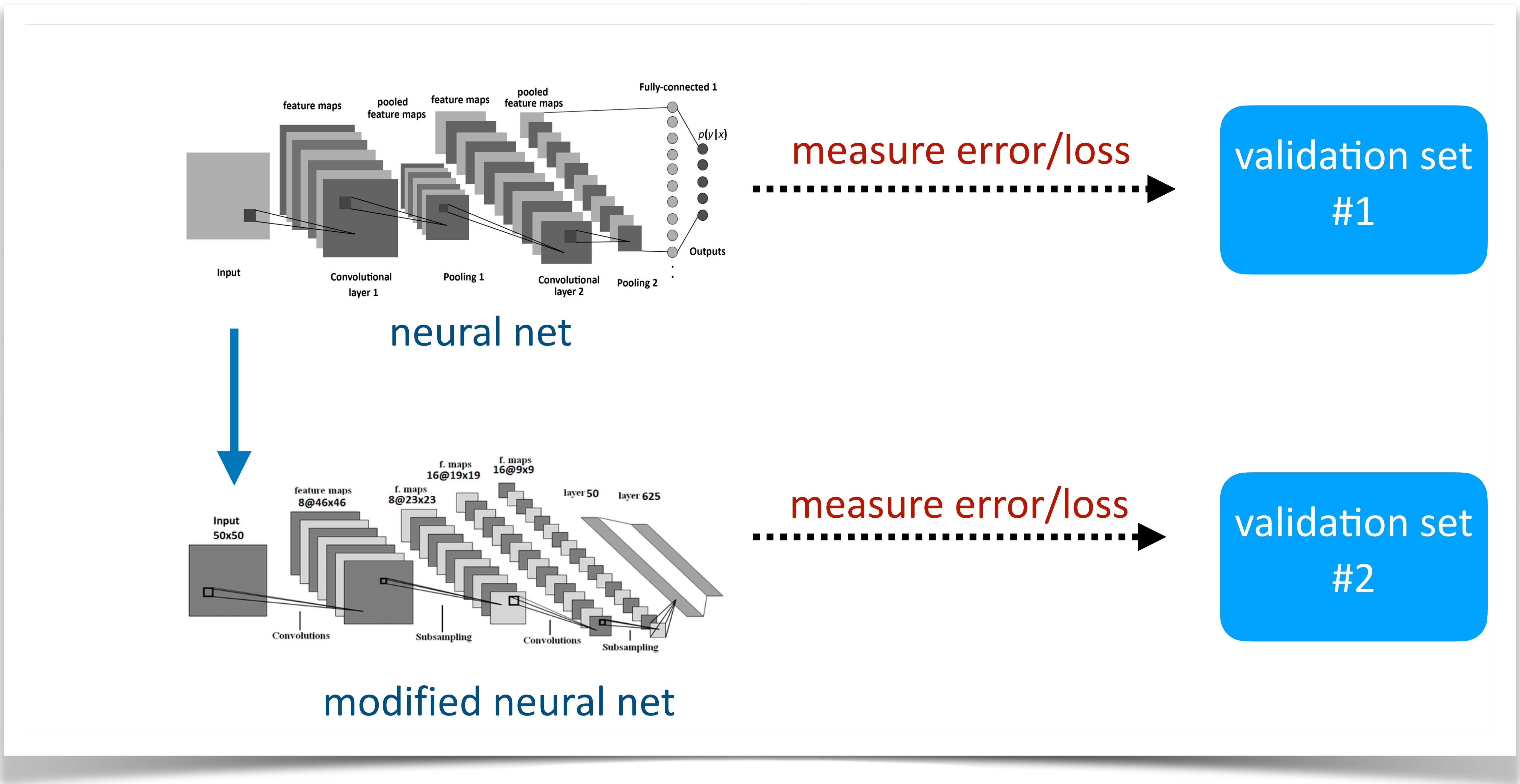
Modeling the Framework for False Positive Findings

Several methodologists have pointed out [9–11] that the high rate of nonreplication (lack of confirmation) of research discoveries is a consequence of the convenient, yet ill-founded strategy of claiming conclusive research findings solely on the basis of a single study assessed by formal statistical significance, typically for a *p*-value less than 0.05. Research is not most appropriately represented and summarized by *p*-values, but, unfortunately, there is a widespread notion that medical research articles

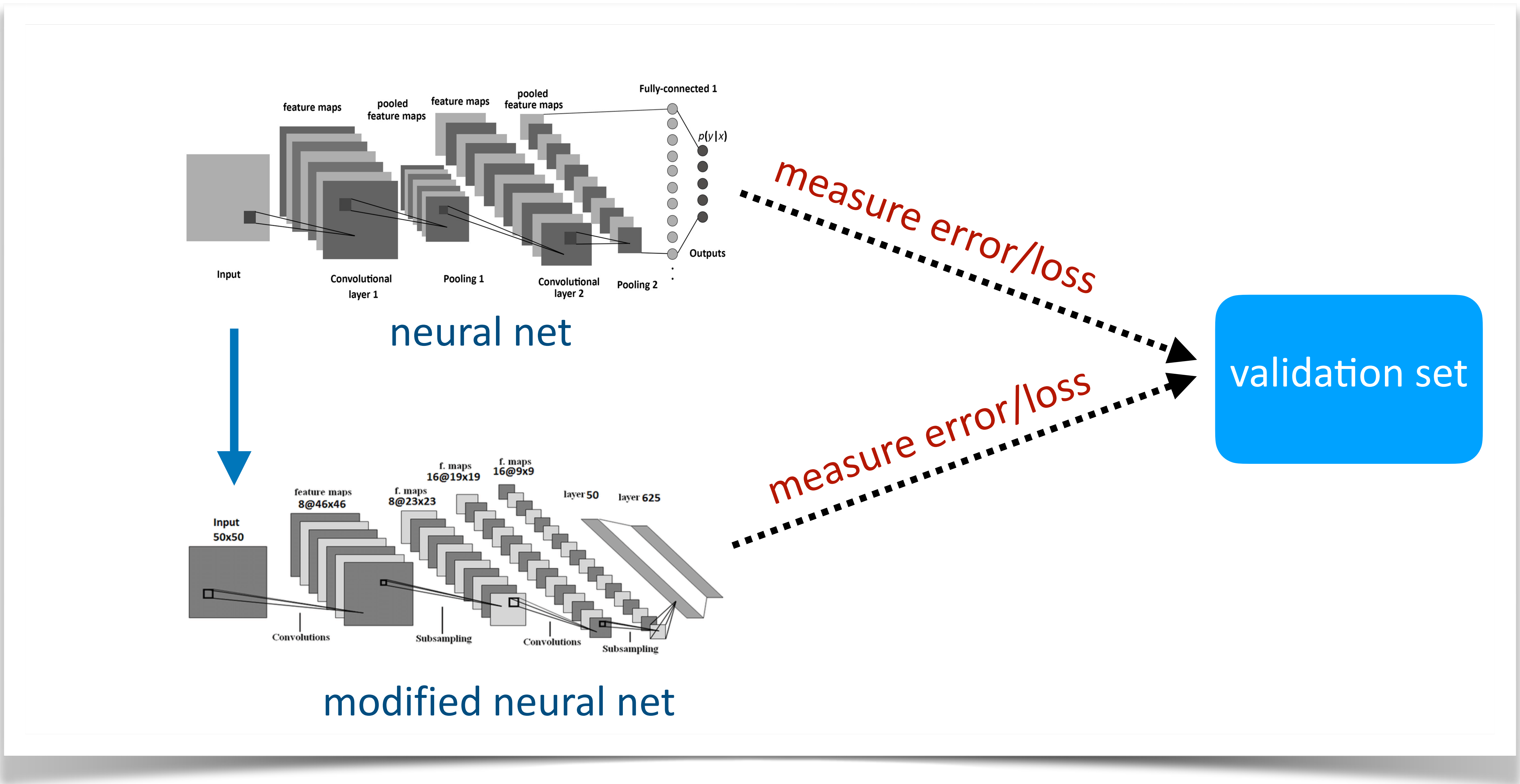
It can be proven that most claimed research findings are false.

is characteristic of the field and can vary a lot depending on whether the field targets highly likely relationships or searches for only one or a few true relationships among thousands and millions of hypotheses that may be postulated. Let us also consider, for computational simplicity, circumscribed fields where either there is only one true relationship (among many that can be hypothesized) or the power is similar to find any of the several existing true relationships. The pre-study probability of a relationship being true is $R/(R+1)$. The probability of a study finding a true relationship reflects the power $1-\beta$ (one minus the Type II error rate). The probability of claiming a relationship when none truly exists reflects the Type I error rate, α. Assuming that c relationships are being probed in the field, the expected values of the 2×2 table are given in Table 1. After a research finding has been claimed based on

The theory


feature maps
pooled feature maps
feature maps
pooled feature maps
Fully-connected 1
Input
Convolutional layer 1
Pooling 1
Convolutional layer 2
Pooling 2
Outputs
neural net
measure error/loss
validation set #1
Input 50x50
feature maps 8@46x46
f. maps 8@23x23
f. maps 16@19x19
f. maps 16@9x9
layer 50
layer 625
Convolutions
Subsampling
Convolutions
Subsampling
modified neural net
measure error/loss
validation set #2

The practice



After modification, we have no guarantee that loss will generalize!

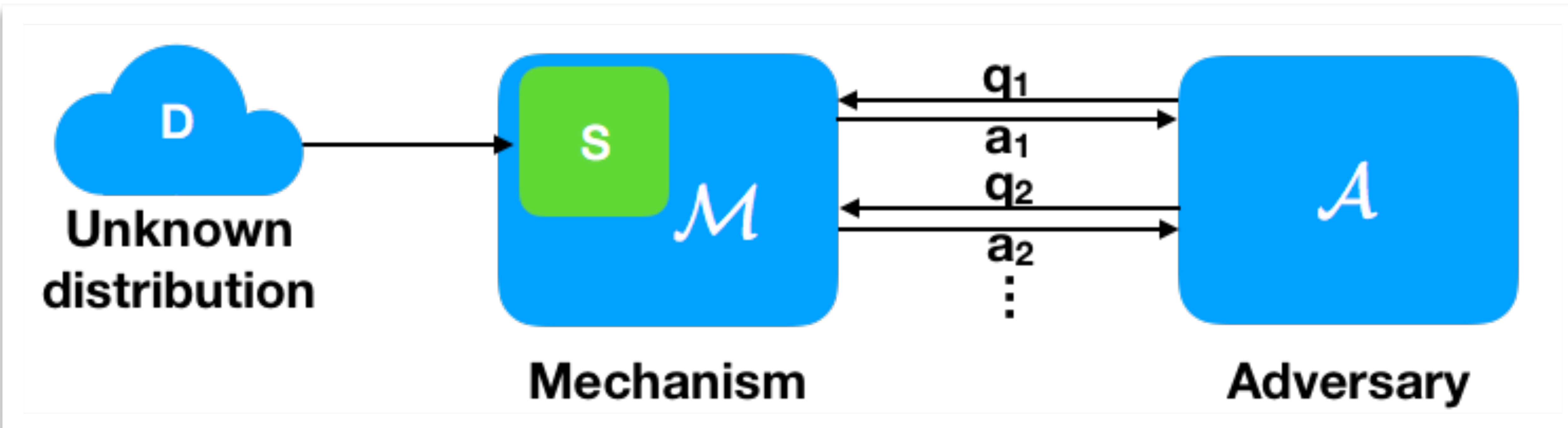
The goal

We want a way to answer queries adaptively *on a fixed dataset* but without having to assume:

1. anything about the nature of the adaptivity
2. that the queries come from a class of bounded complexity (e.g. bounded VC dimension)

Adaptive data analysis (ADA)

[Dwork et al. '15]:



How many queries can we answer, and how long does it need to take?

Low-sensitivity queries

A *query* is just a function $q : D \to \mathbb{R}$, on which the mechanism wants to return a value close to $q(D)$.

Importantly *low-sensitivity queries* ***are*** specified by a function $q : X^n \to \mathbb{R}$, where for all samples $S, S' \in X^n$ that differ on only one element, $|q(S) - q(S')| \leq 1/n$. Then define $q(D) := E_{S \sim D^n}[q(S)]$. [Dwork et al. '06]

Adaptive accuracy

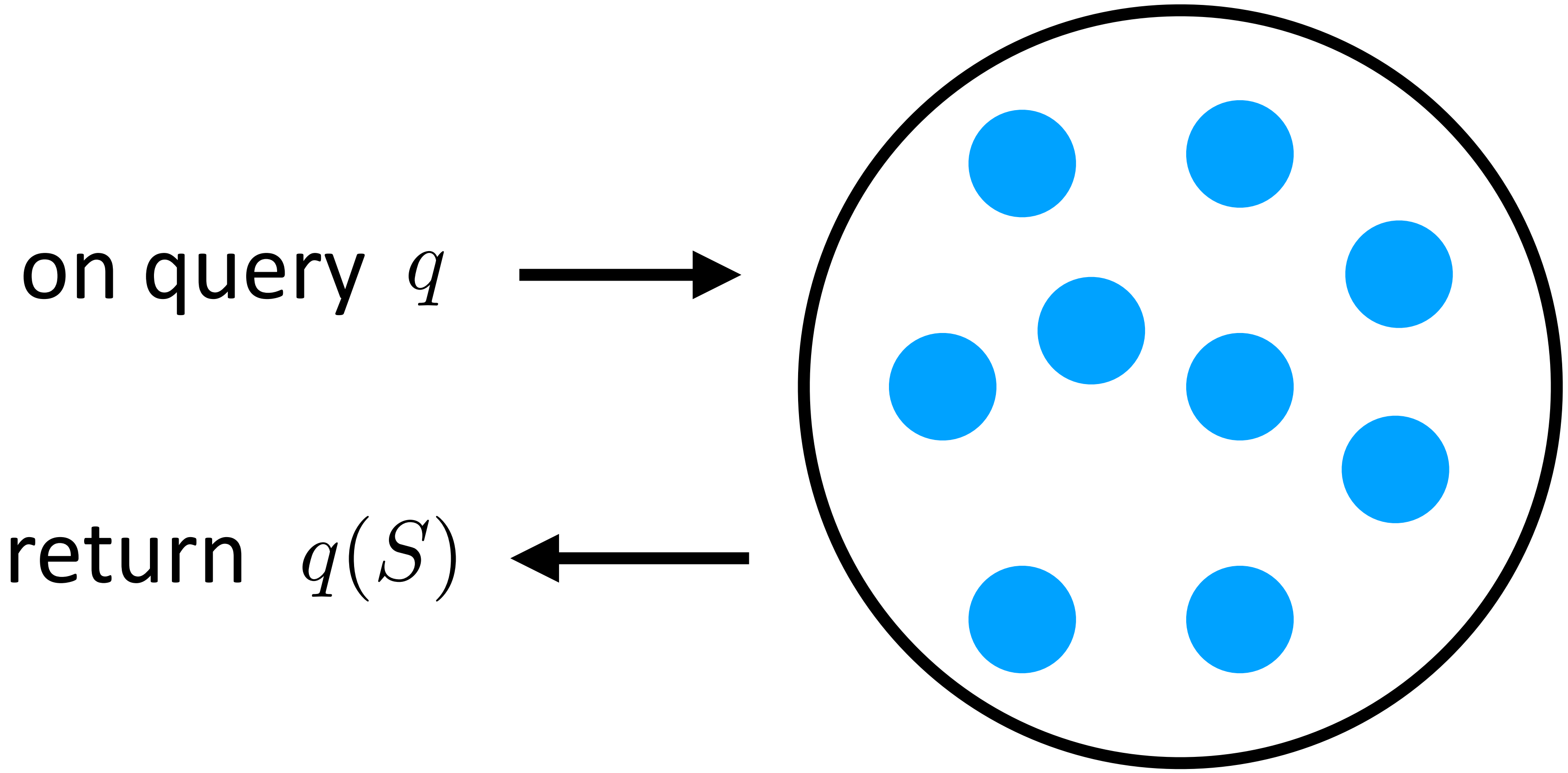
A mechanism M is (α, β)-accurate on the distribution *D* if for all queries q_i,

$$P_{\mathcal{M},\mathcal{A}}[\max_i |q_i(D) - a_i| \leq \alpha] \geq 1 - \beta$$

How many samples does it take to answer k adaptive queries efficiently with (α, β)-accuracy?

How fast can we accurately answer such queries?

Standard machine learning



But returning the empirical estimate turns out to be suboptimal for adaptive data analysis!

Exploiting adaptivity

The leaderboard takes a hold-out set of n points with labels $y \in \{0,1\}^n$ and for any label prediction $u \in \{0,1\}^n$ returns the average 0-1 loss $\mathcal{L}(u)$.

#	Team Name	Notebook	Team Members	Score	Entries	Last
1	Sirish Somanchi			0.9931	276	5d
2	Yuval & nosound			0.9926	402	5d
3	Bestoverfitting			0.9767	245	7d
4	yosef huang			0.9766	46	5d
5	Overfitting?? Nah.			0.9754	319	5d
6	Marwa_Fatto			0.9748	123	5d
7	prd			0.9738	273	5d

Exploiting adaptivity

The leaderboard takes a hold-out set of n points with labels $y \in \{0,1\}^n$ and for any label prediction $u \in \{0,1\}^n$ returns the average 0-1 loss $\mathcal{L}(u)$.

The boosting attack [Blum-Hardt '15]:

1. Pick k vectors $u_1, \ldots, u_k$ uniformly at random, and receive losses $\mathcal{L}_1, \ldots, \mathcal{L}_k$ in response. Call the set $I = \{i : \mathcal{L}_i \leq 1/2\}$.
2. Output $u^* = \mathrm{maj}(\{u_i : i \in I\})$, applied coordinate-wise.

Boosting attack

	x_1	x_2	x_3	x_4	x_5	…	x_n
u_1	0	0	1	0	1	…	0
u_2	0	0	1	1	0	…	1
u_3	1	0	0	0	1	…	0
u_4	1	1	0	0	1	…	1
u_5	1	0	0	1	1	…	0
u_6	1	1	1	0	1	…	1
…	…	…	…	…	…	…	
u_k	0	1	0	0	0		1

Boosting attack

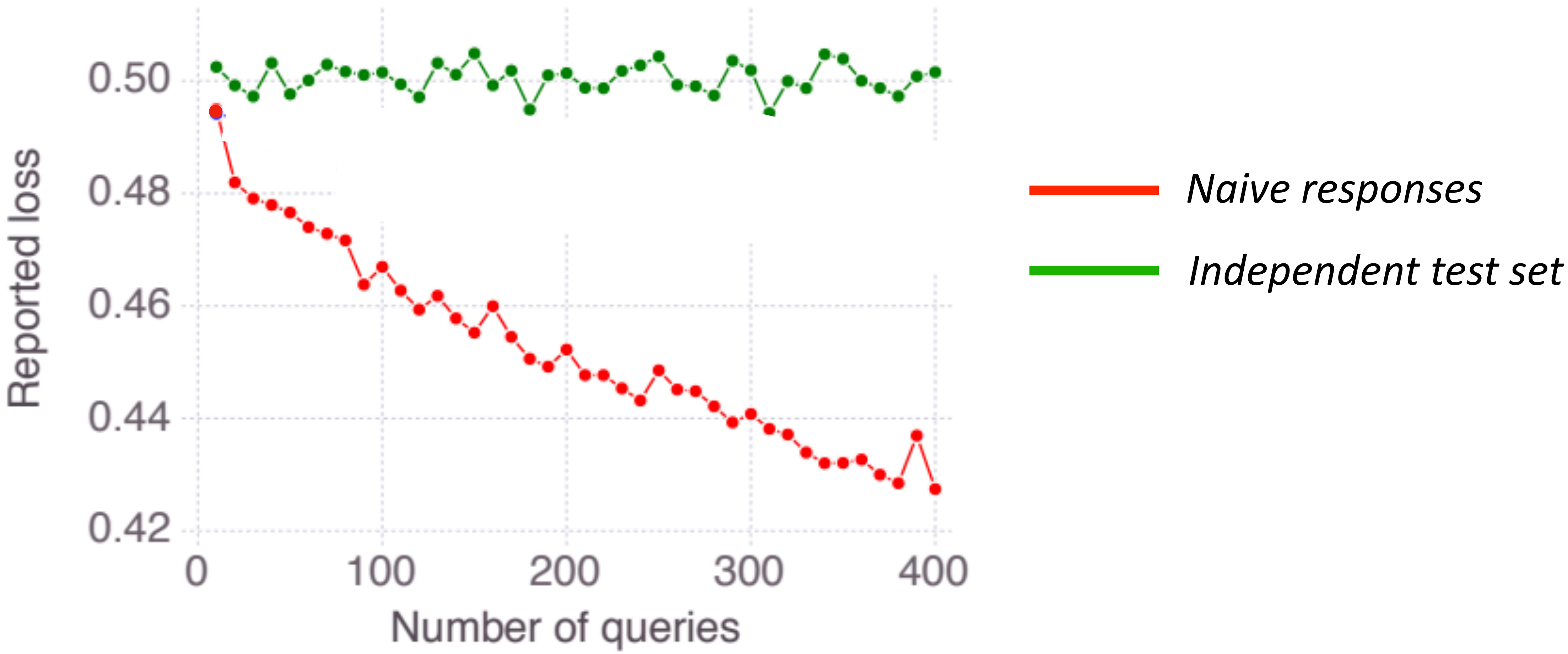
	x_1	x_2	x_3	x_4	x_5	…	x_n	
u_1	0	0	1	0	1	…	0	.55
u_2	0	0	1	1	0	…	1	.48
u_3	1	0	0	0	1	…	0	.53
u_4	1	1	0	0	1	…	1	.51
u_5	1	0	0	1	1	…	0	.49
u_6	1	1	1	0	1	…	1	.52
…	…	…	…	…	…	…		…
u_k	0	1	0	0	0		1	.47

Boosting attack

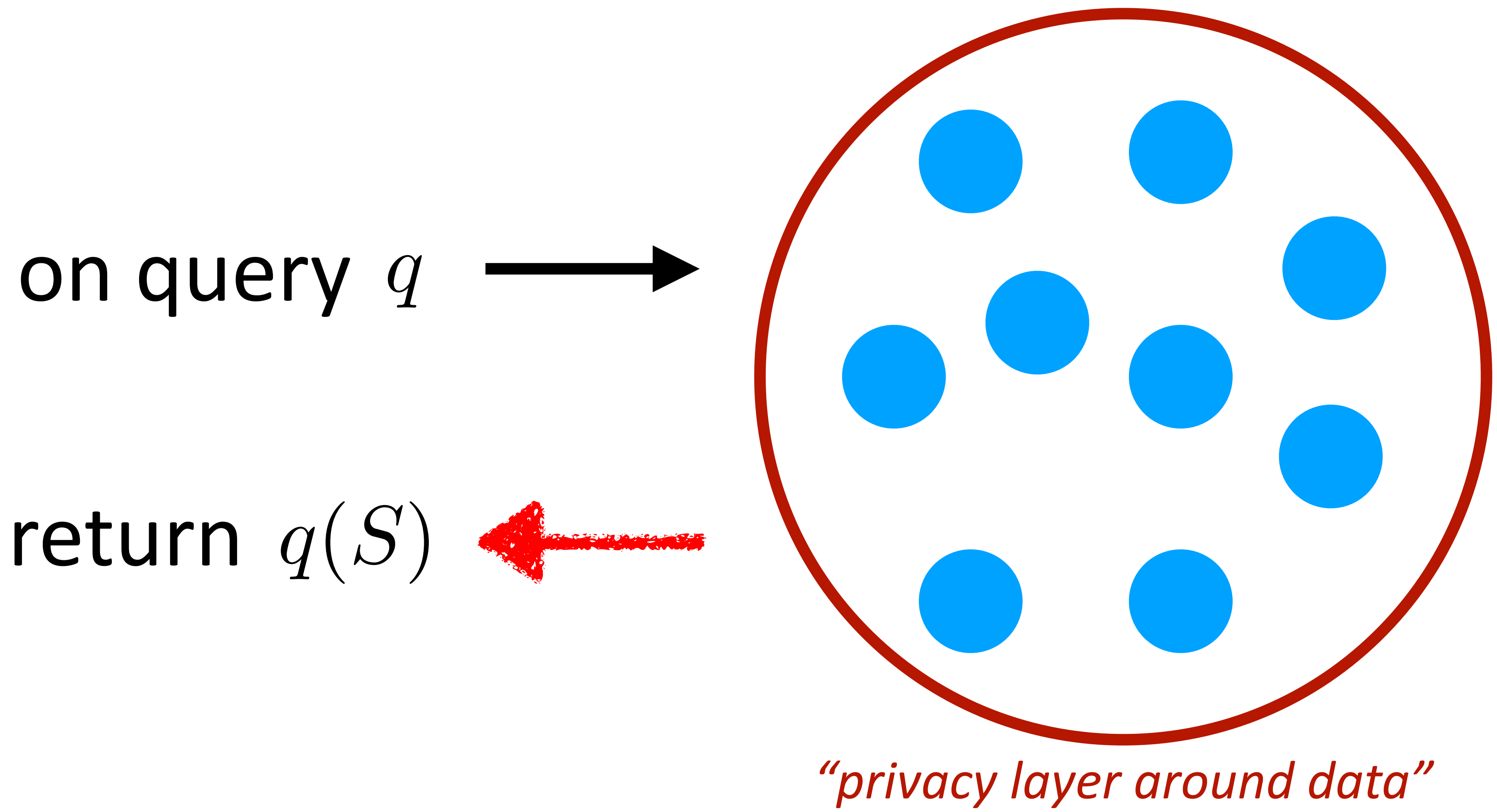
	x_1	x_2	x_3	x_4	x_5	…	x_n	
u_1	0	0	1	0	1	…	0	.55
u_2	0	0	1	1	0	…	1	.48
u_3	1	0	0	0	1	…	0	.53
u_4	1	1	0	0	1	…	1	.51
u_5	1	0	0	1	1	…	0	.49
u_6	1	1	1	0	1	…	1	.52
…	…	…	…	…	…	…		…
u_k	0	1	0	0	0		1	.47
u^*	**0**	**0**	**0**	**1**	**0**		**1**	

Exploiting without learning

Theorem (Blum and Hardt 2015). *With at least constant probability, the loss is* $\mathcal{L}(u) \leq 1/2 - \Omega\left(\sqrt{k/n}\right)$.



An idea



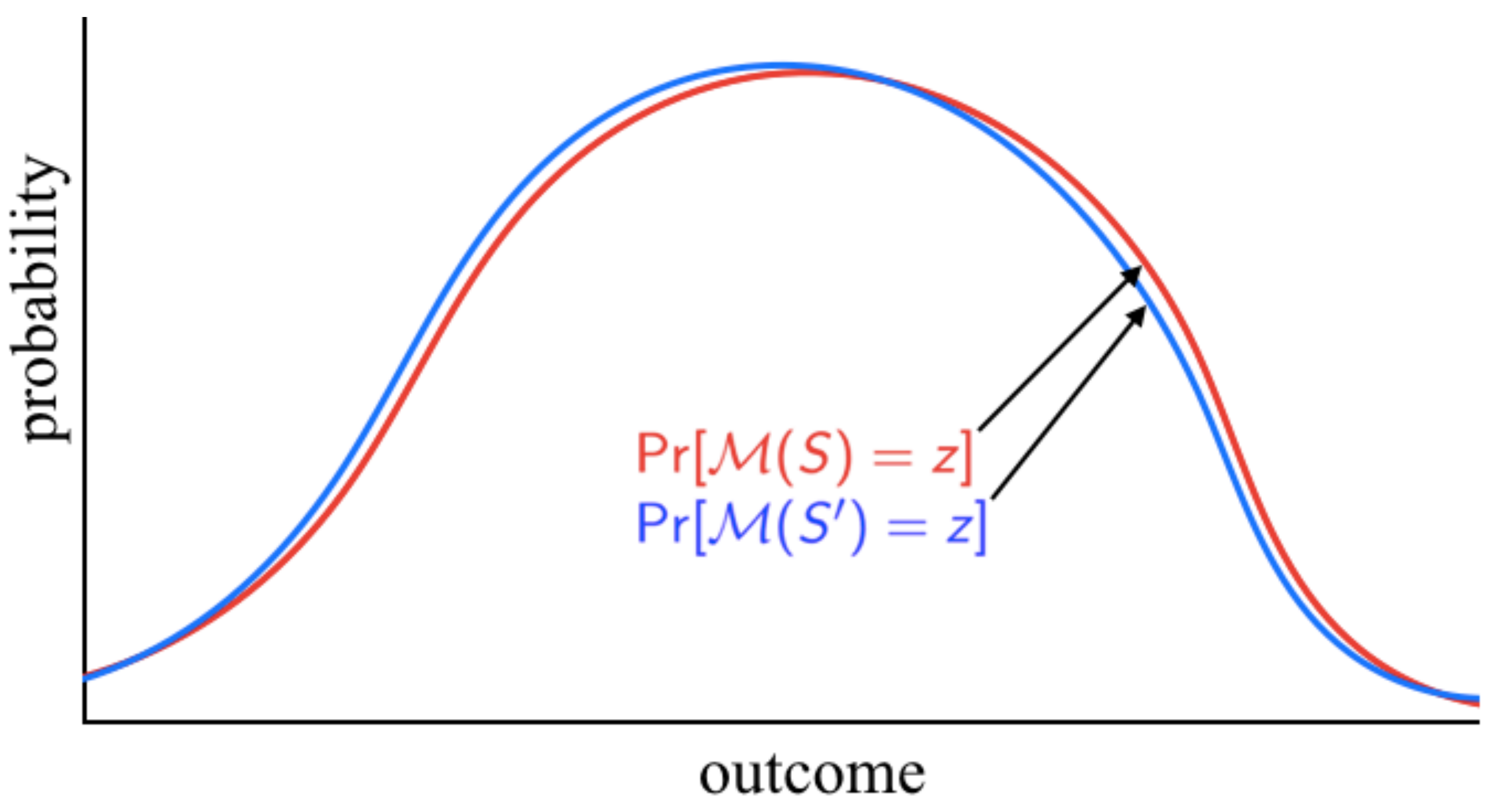
Intuition: an algorithm that cannot learn the data should not overfit!

Differential privacy (DP)

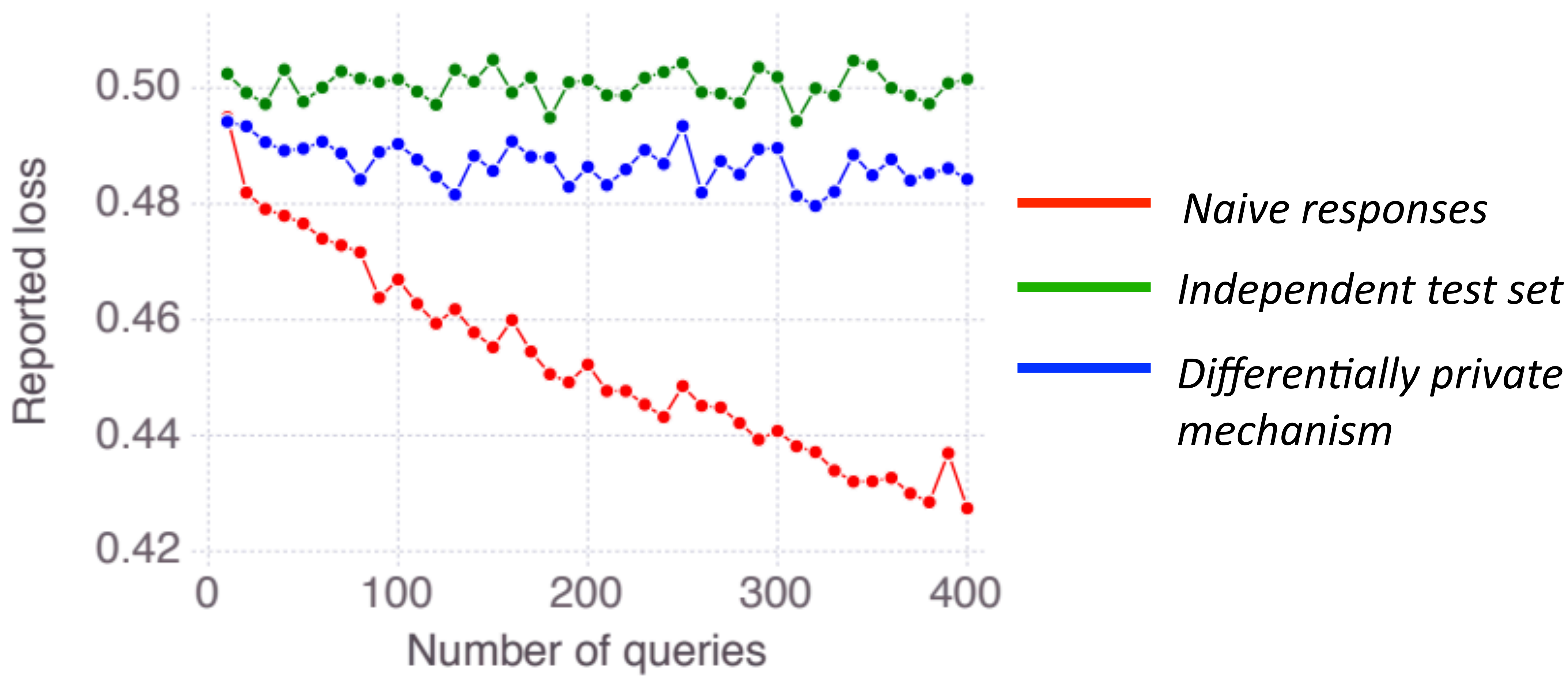
Differential privacy is one notion that works:

Definition (Dwork et al. 2006). *A mechanism* $\mathcal{M}$ *is* (ϵ, δ)-private *if for every two samples* $S, S' \in X^n$ *differing by at most one element and every out come* z,

$$P[\mathcal{M}(S) = z] \leq e^{\epsilon} \cdot P[\mathcal{M}(S') = z] + \delta$$



DP responses against boosting attack



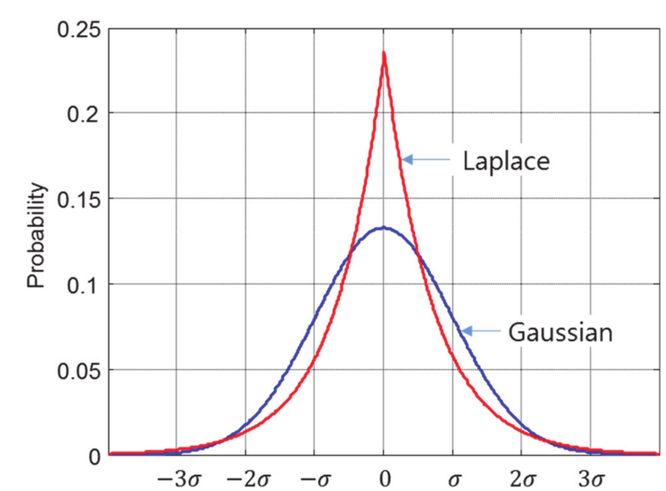
A mechanism for SQs

[Dwork et al. ’15; Bassily et al. ’16]

Given a data set S of size n and a query q, *M* will:

`return` $q(S) + \text{Lap}\left(\frac{1}{n\epsilon}\right)$ (Laplace noise)

$$Lap(z|b) = \frac{1}{2b}\exp(-\frac{|z|}{b})$$



A mechanism for SQs

[Dwork et al. '15; Bassily et al. '16]

Given a data set S of size n and a query q, *M* will:

`return` $q(S) + \mathrm{Lap}\left(\frac{1}{n\epsilon}\right)$ `(Laplace noise)`

1. This mechanism is: both accurate on the sample and also private.
2. That is sufficient to guarantee accuracy on the distribution (*transfer*).
3. But it is slow (linear time per query) and it doesn't reflect what practitioners sometimes do on real datasets (subsampling).

Example result in this field from my work: a faster mechanism

Theorem [Fish-R-Rubinstein '20]: There is a mechanism for answering statistical queries with:

- *sample complexity* $n = \tilde{O}\left(\frac{\sqrt{k}}{\alpha^2}\right)$ ← same as "Laplace" mechanism
- $\ell = \tilde{O}\left(\frac{\log(k)}{\alpha^2}\right)$ *samples per query*
- $\tilde{O}\left(\frac{\log^2(k)}{\alpha^2}\right)$ *time per query.*

much faster! (samples per query and time per query)

Fast mechanism for SQs

Given a data set S of size n and a query q, *M* will:

1. Sample ℓ points uniformly at random (with or without replacement) and call this sample S_ℓ
2. Return $q(S_\ell) + \mathrm{Lap}\left(\frac{1}{\ell\epsilon}\right)$ (Laplace noise)

This corresponds to bootstrapping: on every new query we re-subsample!

Why does this work?

Our main idea: while subsampling means we have a worse estimator, sampling also increases the amount of privacy we have.

An straightforward calculation shows that giving an ℓ subsample to an ε-private algorithm yields a $\log(1+(1+(1-1/n)^{\ell})(e^{\varepsilon}-1))$-private algorithm.

We can then prove that the two effects cancel out exactly for our choice of subsample size!

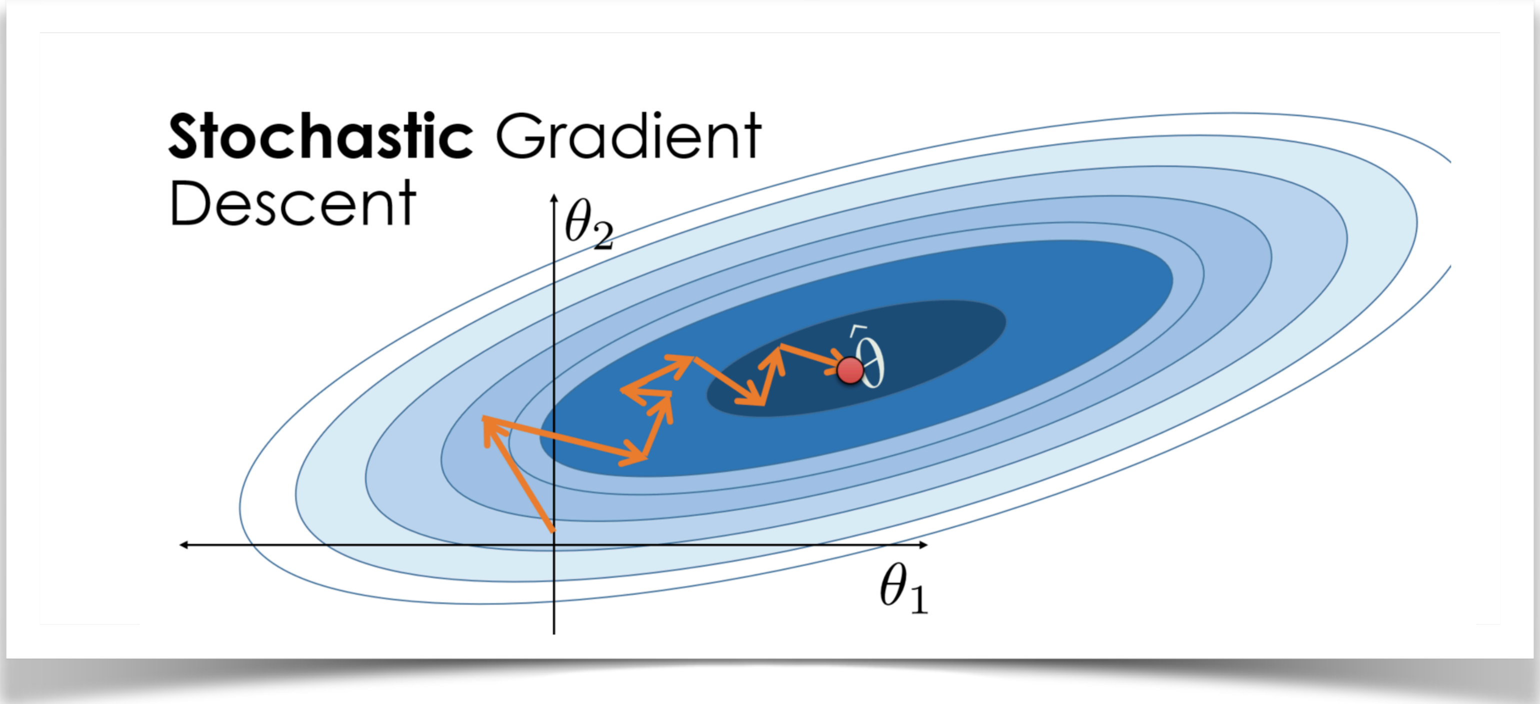
A new analysis of SGD

[Fish-R-Rubinstein '20]

1. Pick arbitrary $x_0 \in \Theta$
2. Repeat $x_t := x_{t-1} - \eta \tilde{\nabla} \mathcal{L}(S, x_{t-1})$

where each component of $\tilde{\nabla} \mathcal{L}(S, x_{t-1})$ is given by our mechanism $\mathcal{M}$ for statistical queries:

$$\tilde{\nabla} \mathcal{L}(S, x_{t-1})^{(i)} := \mathcal{M}(\nabla \mathcal{L}(S, x_{t-1})^{(i)}, S)$$



Even more interesting concerns

There are of course some technical challenges that remain in adaptive data analysis.

But one interesting recent question is actually why we can reuse large datasets in practice more than theory tells us we should!

Why do algorithms trained on ImageNet still generalize to the real world? Are our algorithms more resilient than you might expect? Is real-world data special?

Non-LLM example problem 2: Machine learning reductions

In most cases, engineers would rather make use of subroutines or libraries than code solutions from scratch.

Machine learning reductions

This often leads to unprincipled use…

MIT Technology Review

ARTIFICIAL INTELLIGENCE

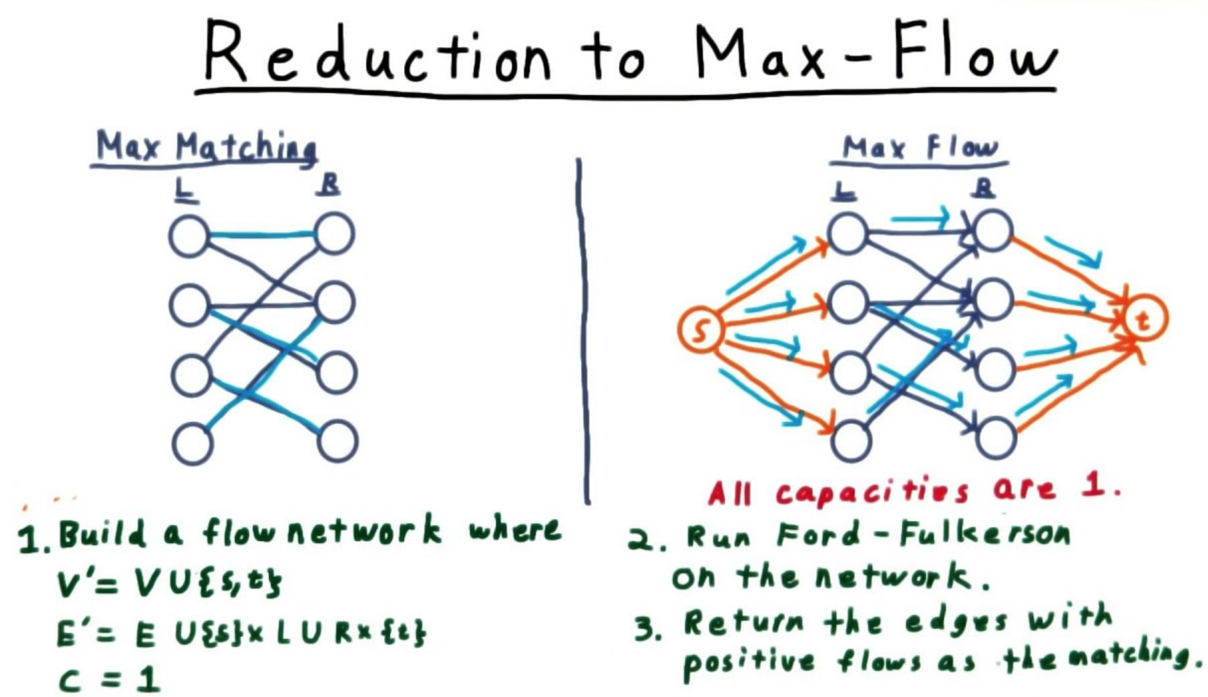
Hundreds of AI tools have been built to catch covid. None of them helped.

Some have been used in hospitals, despite not being properly tested. But the pandemic could help make medical AI better.

Machine learning reductions

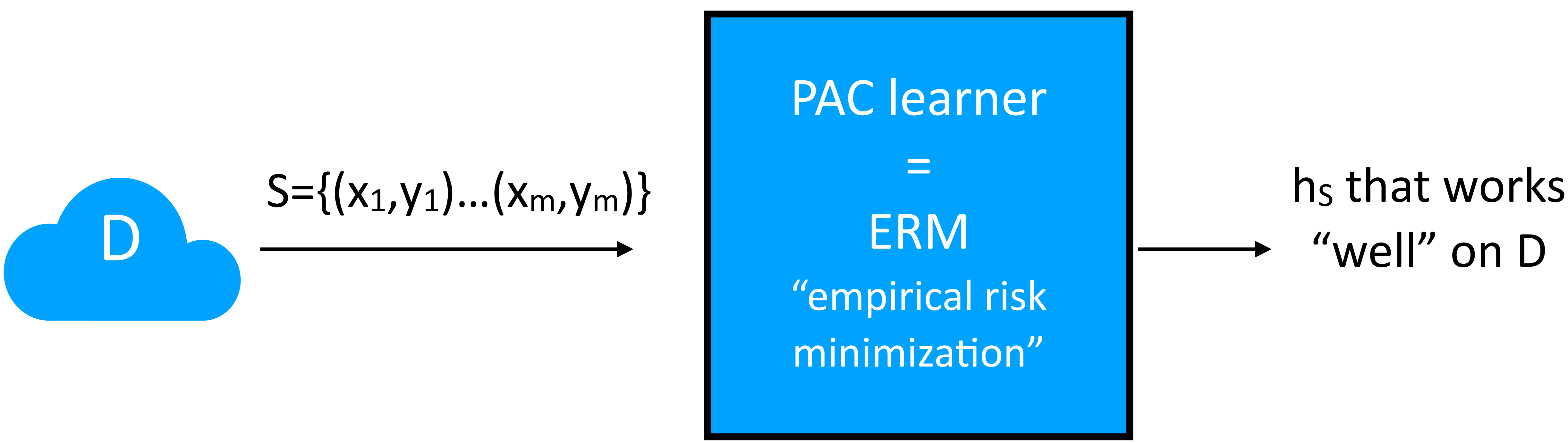
Traditional reductions solve one problem using an algorithm for another problem.

bipartite matching reduction to max-flow (image from GaTech's algorithms course) →



We can use the same idea for machine learning.

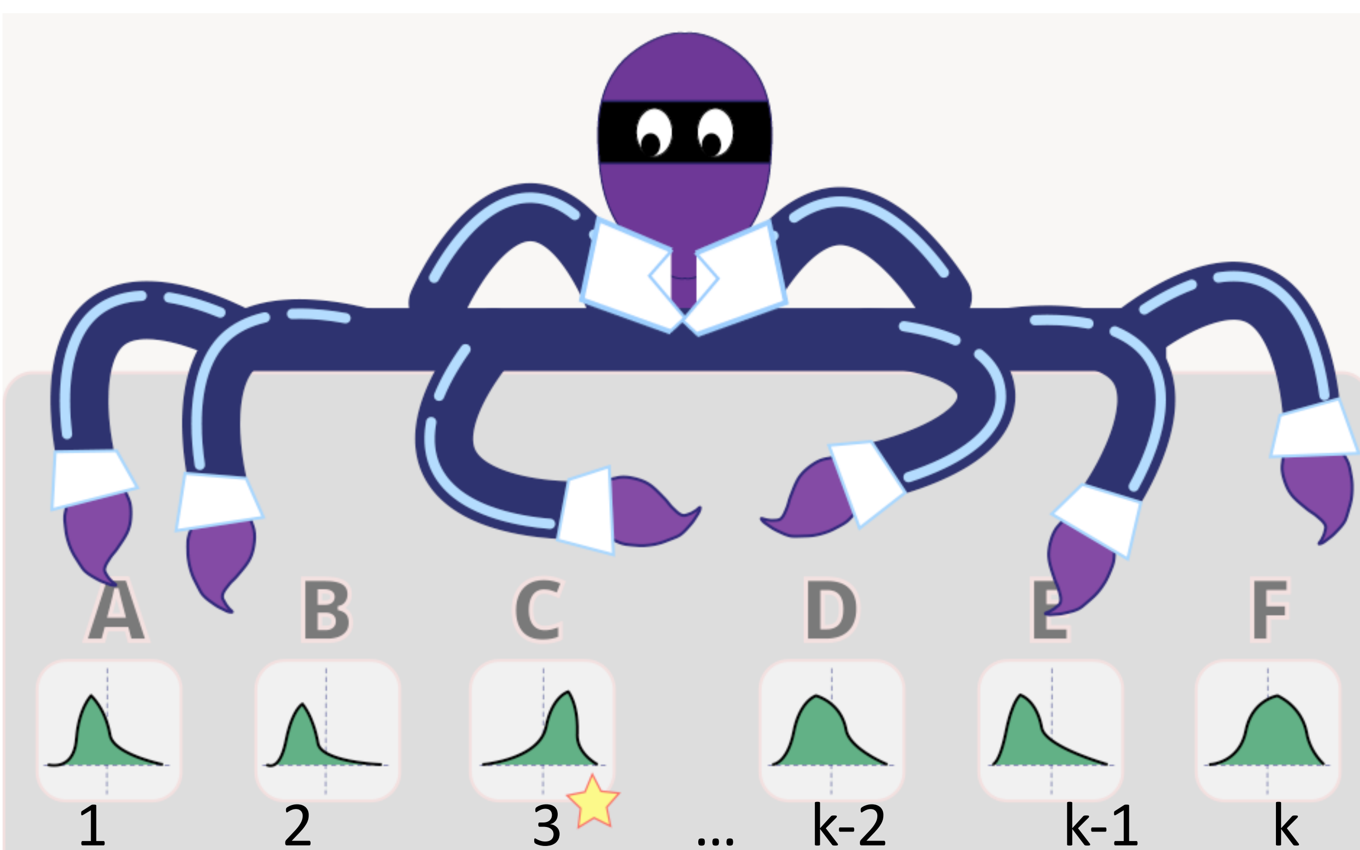
PAC Learning



Stochastic contextual bandits

for t = 1 to T 

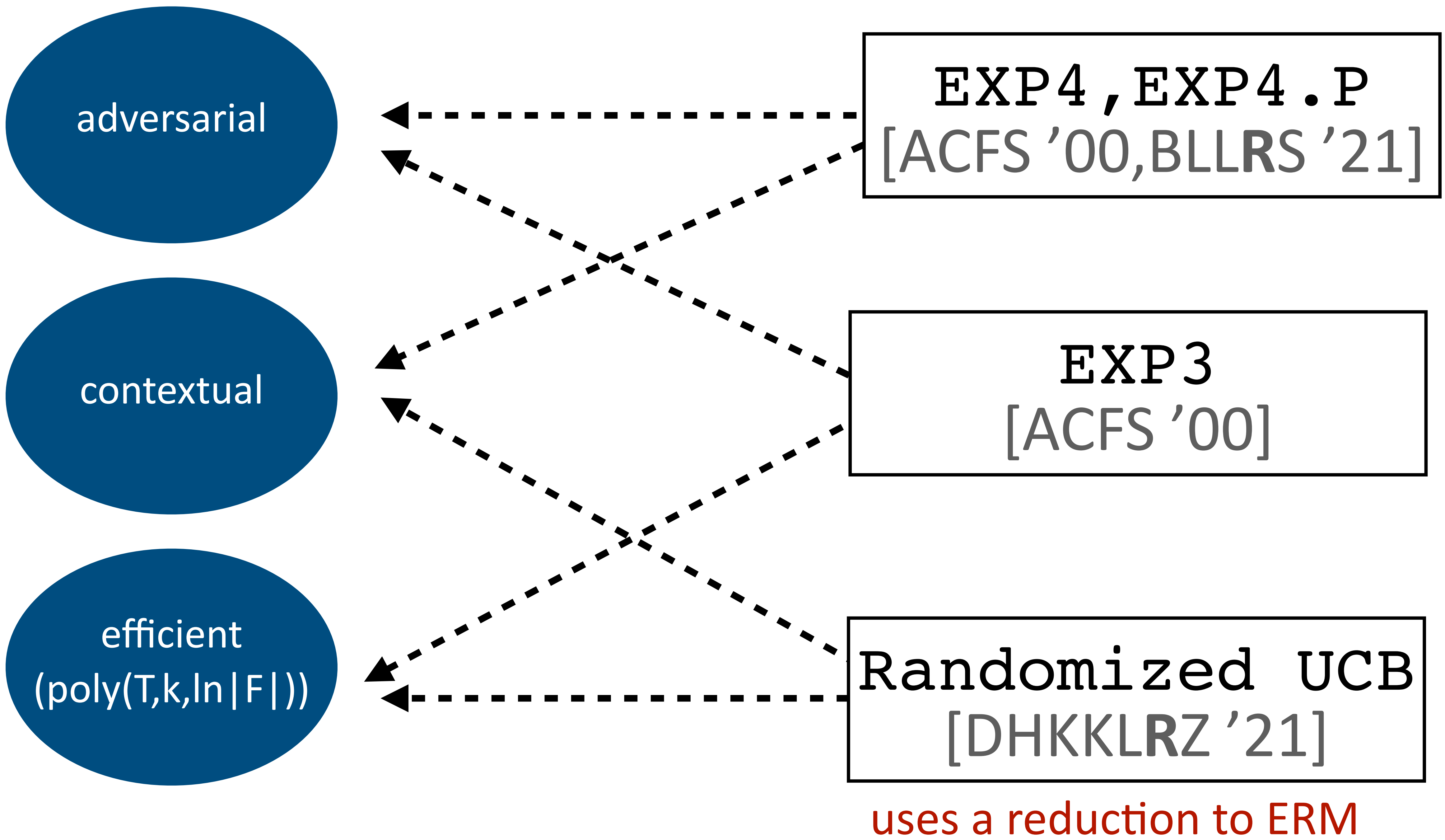
x_t



distributions of losses depend on x_t

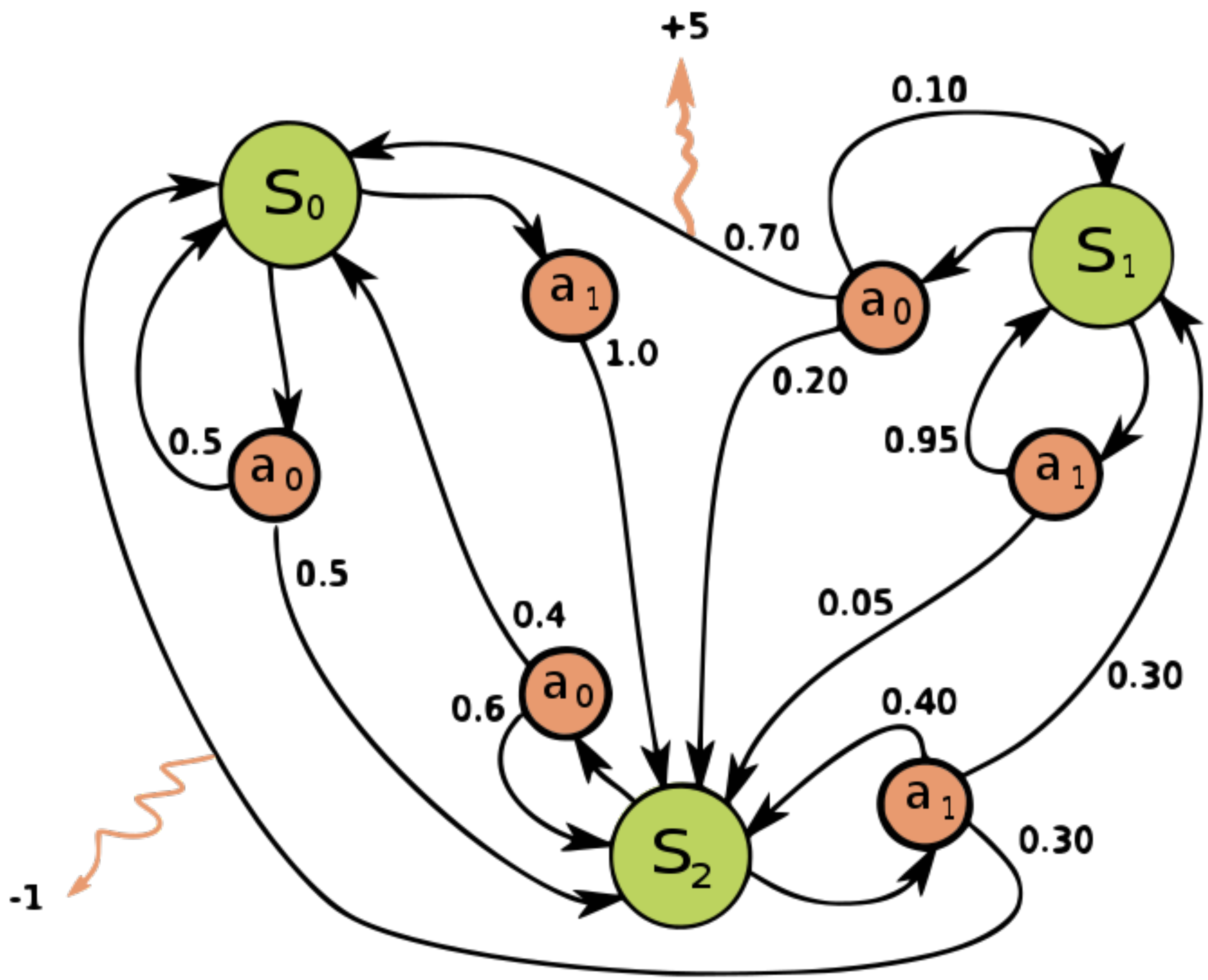
The learner must compete with the best function in *F: X—>{1…k}* in hindsight. Learner's regret should ideally scale as $O(T^{1/2})$ and logarithmically in |*F*|.

A bandits to PAC reduction



Reinforcement learning (RL)

Markov decision process (MDP)

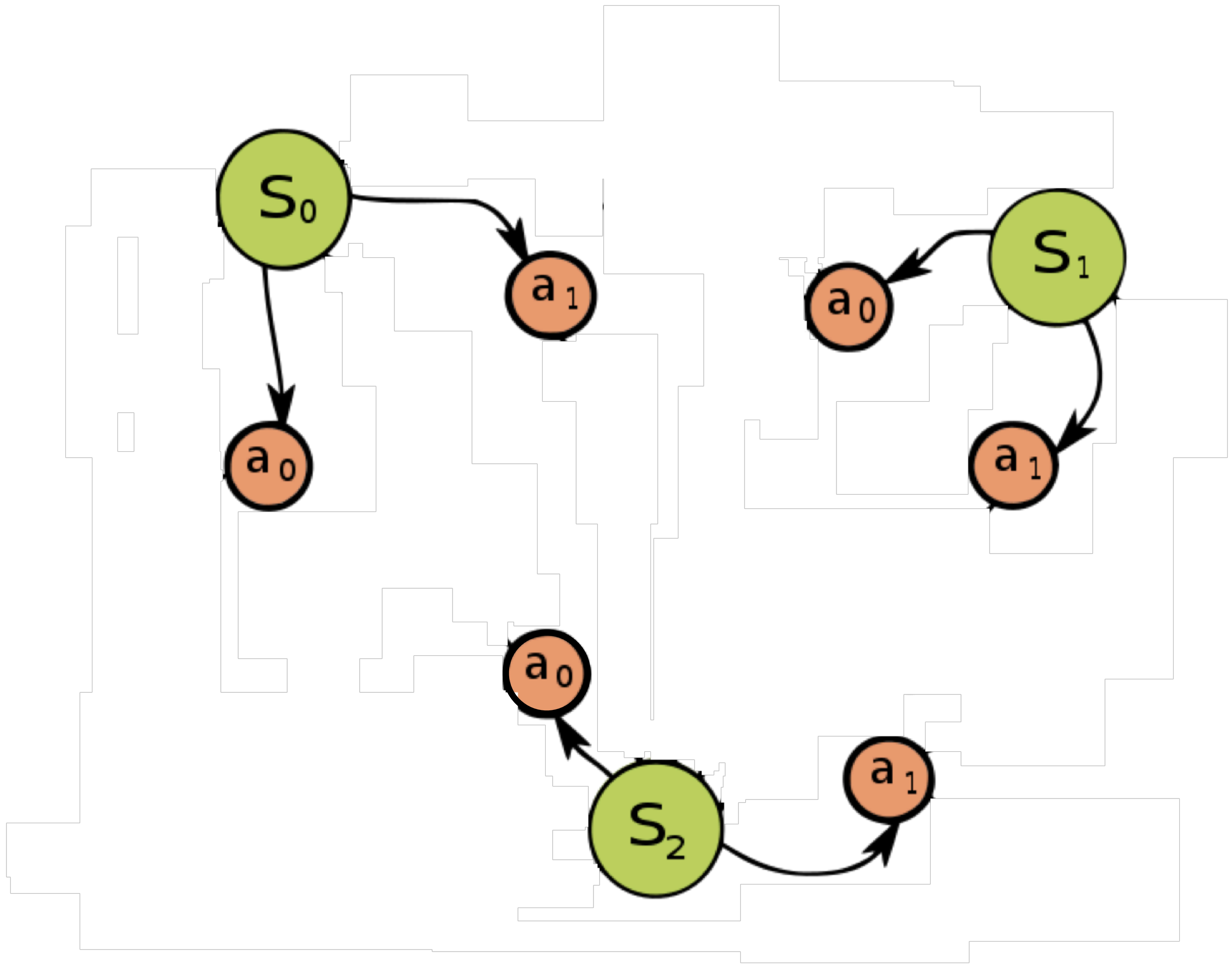


states are green, actions are orange, squiggly arrows are rewards

goal is to learn a good "policy" (what to do at each state)

If MDP is known, can solve for optimal policy

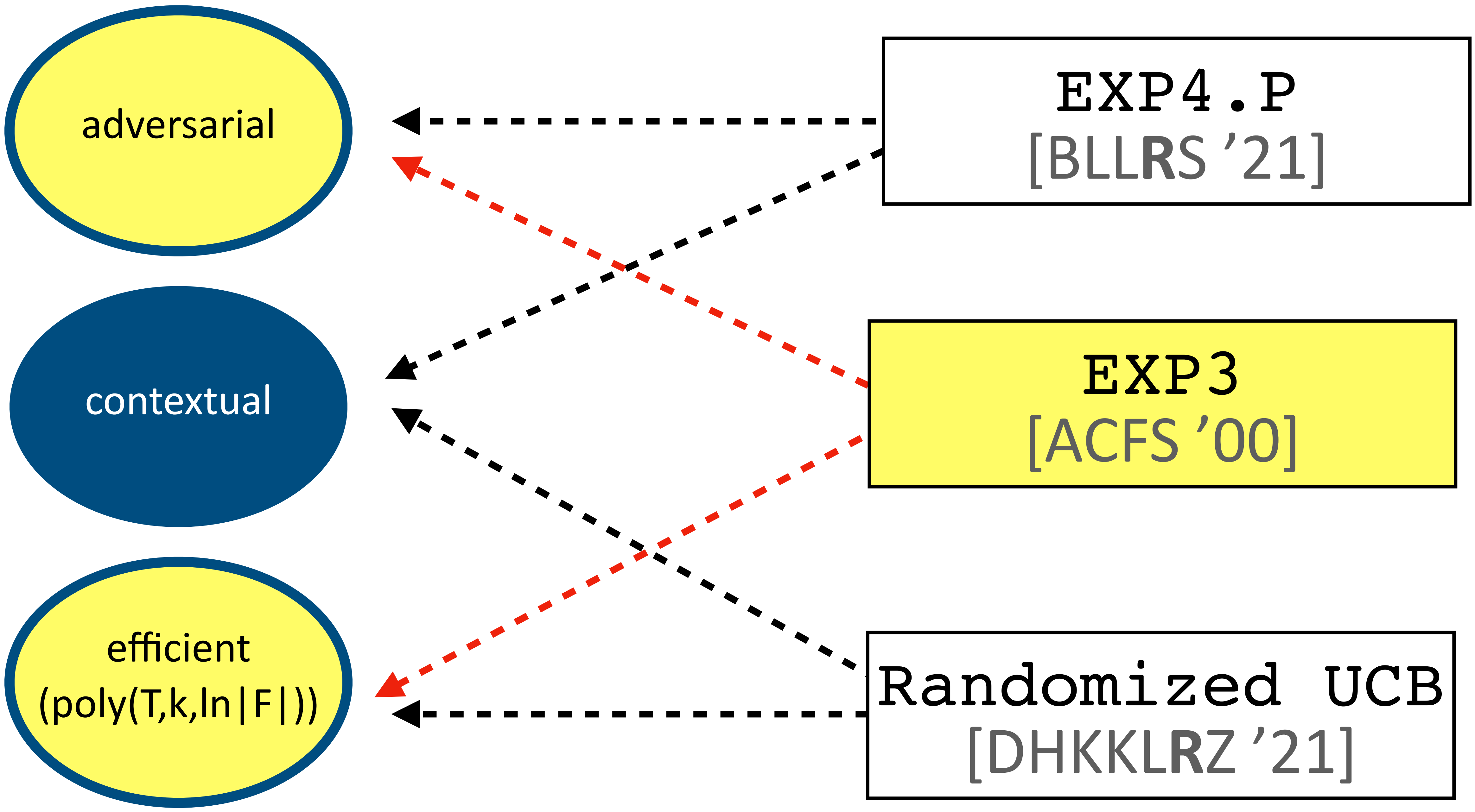
Reinforcement learning (RL)



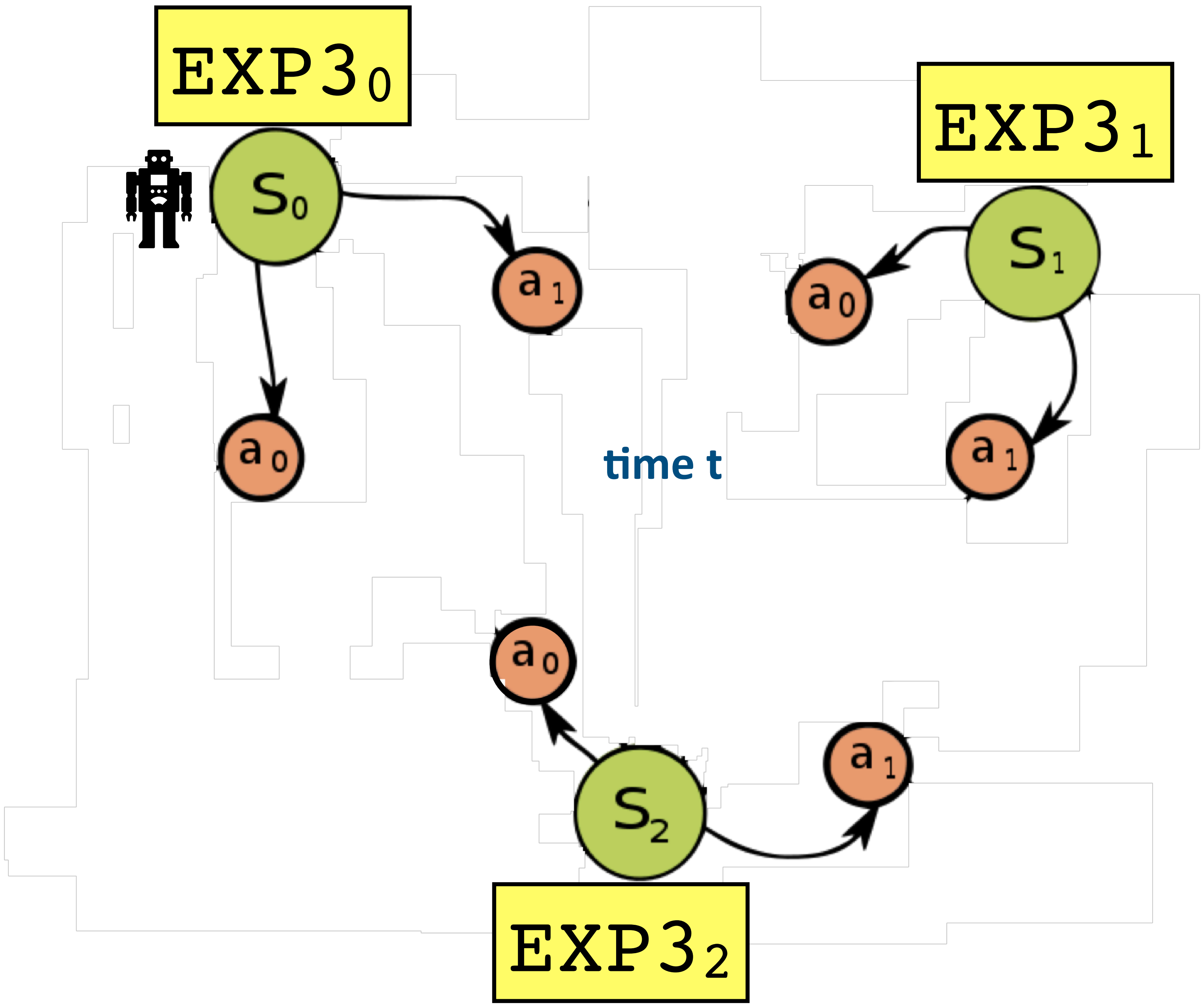
But the MDP is not known!

Problem is more general than bandits.

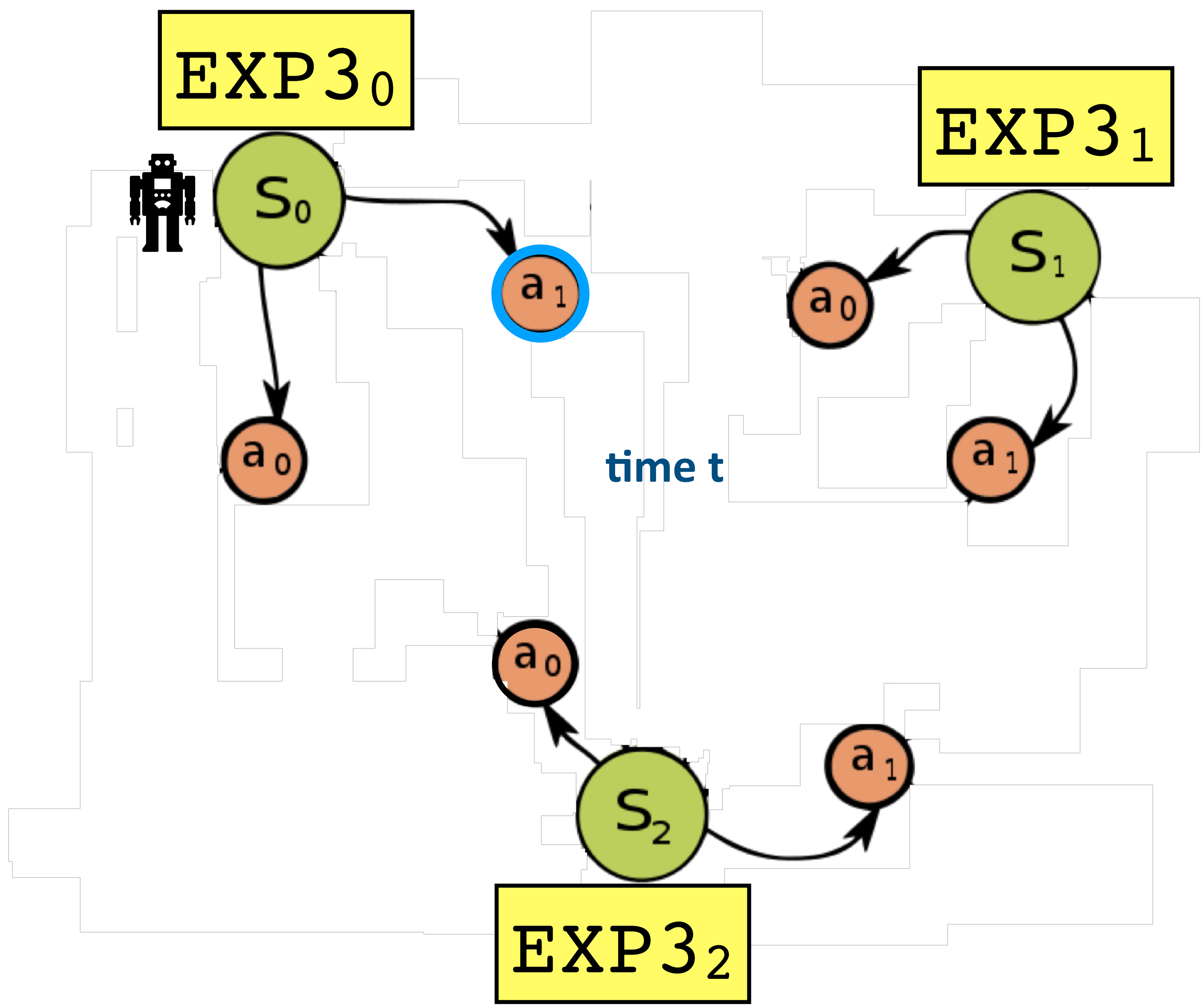
A bandit algorithm that fits



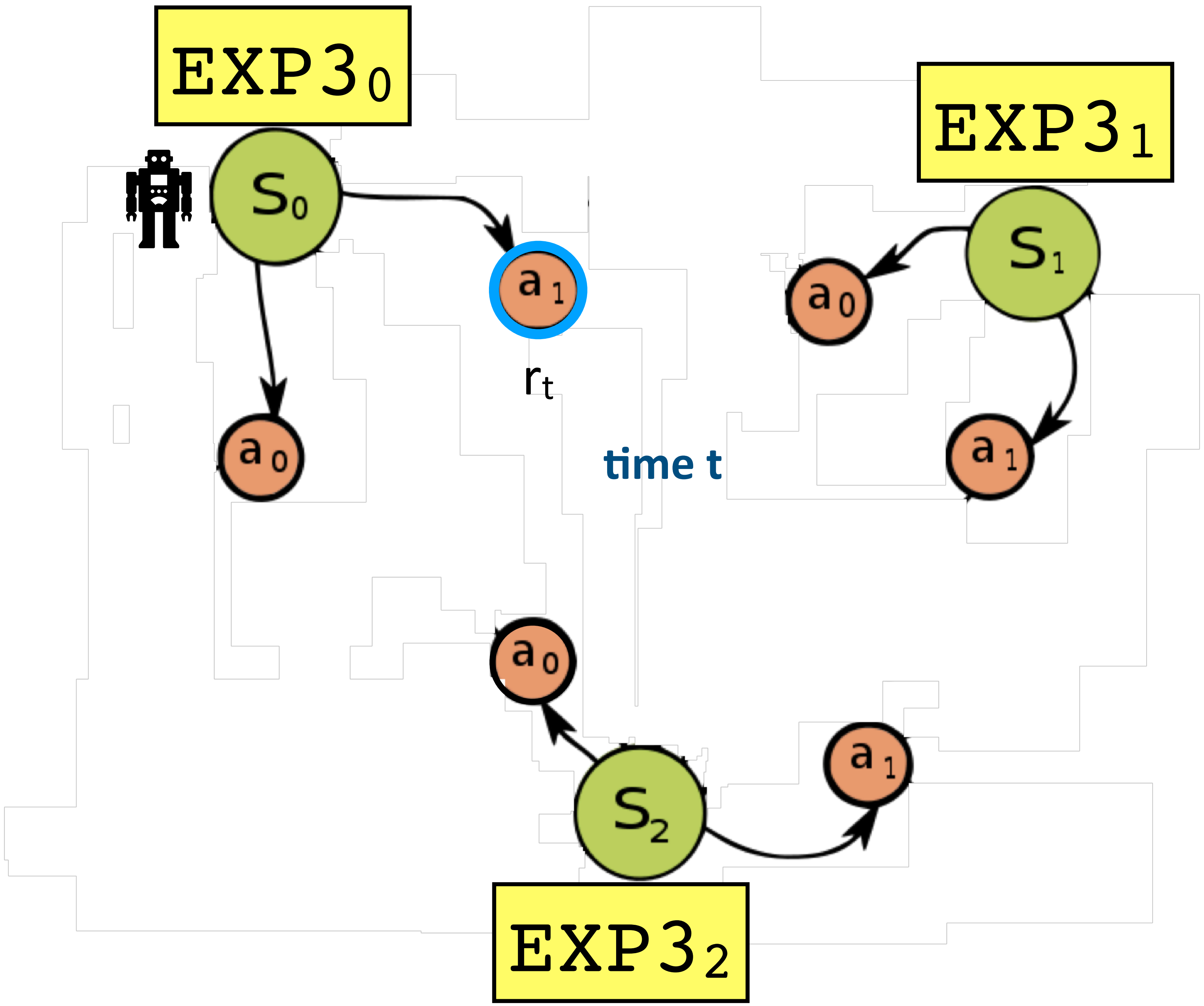
Reinforcement learning (RL)



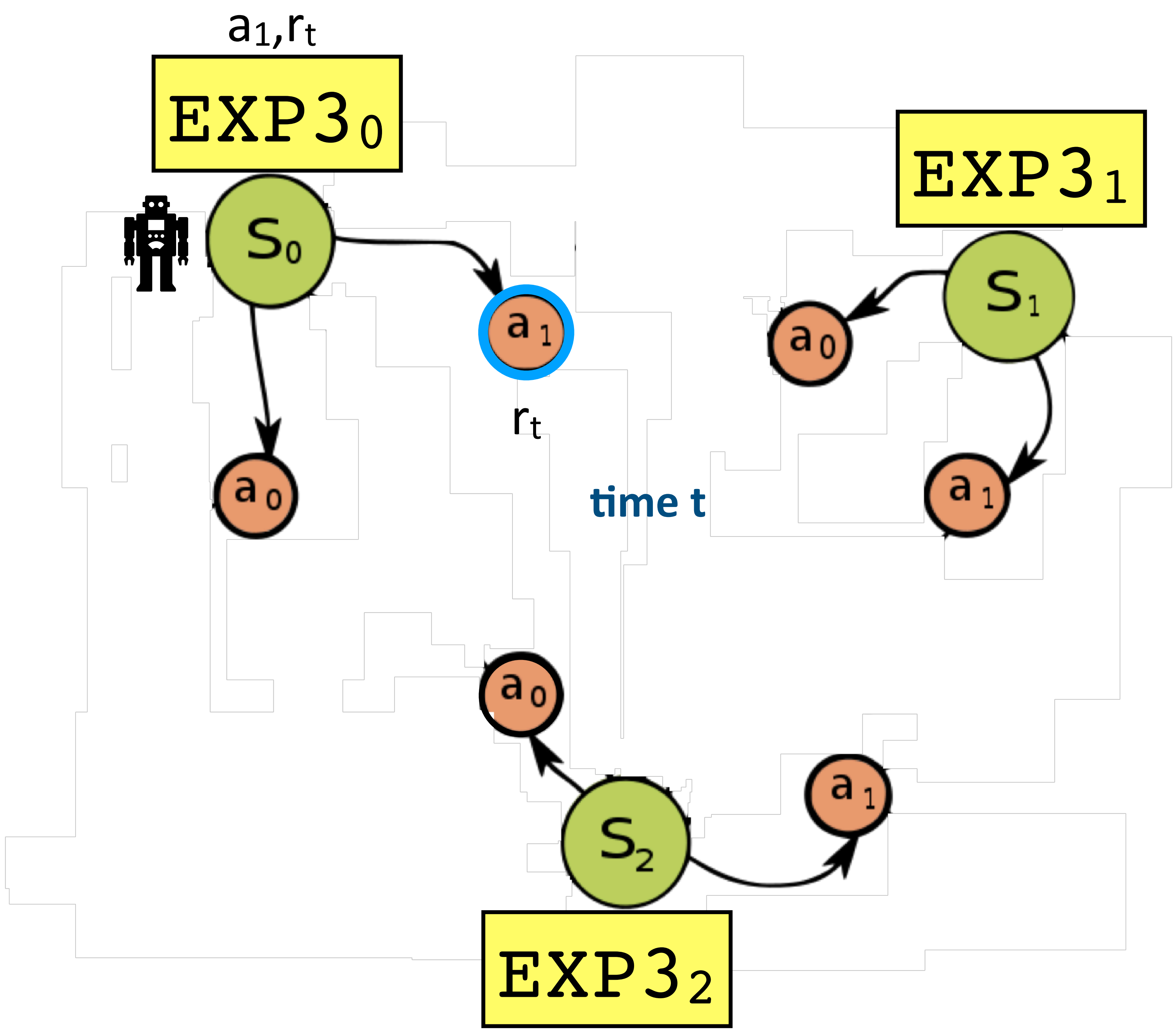
Reinforcement learning (RL)



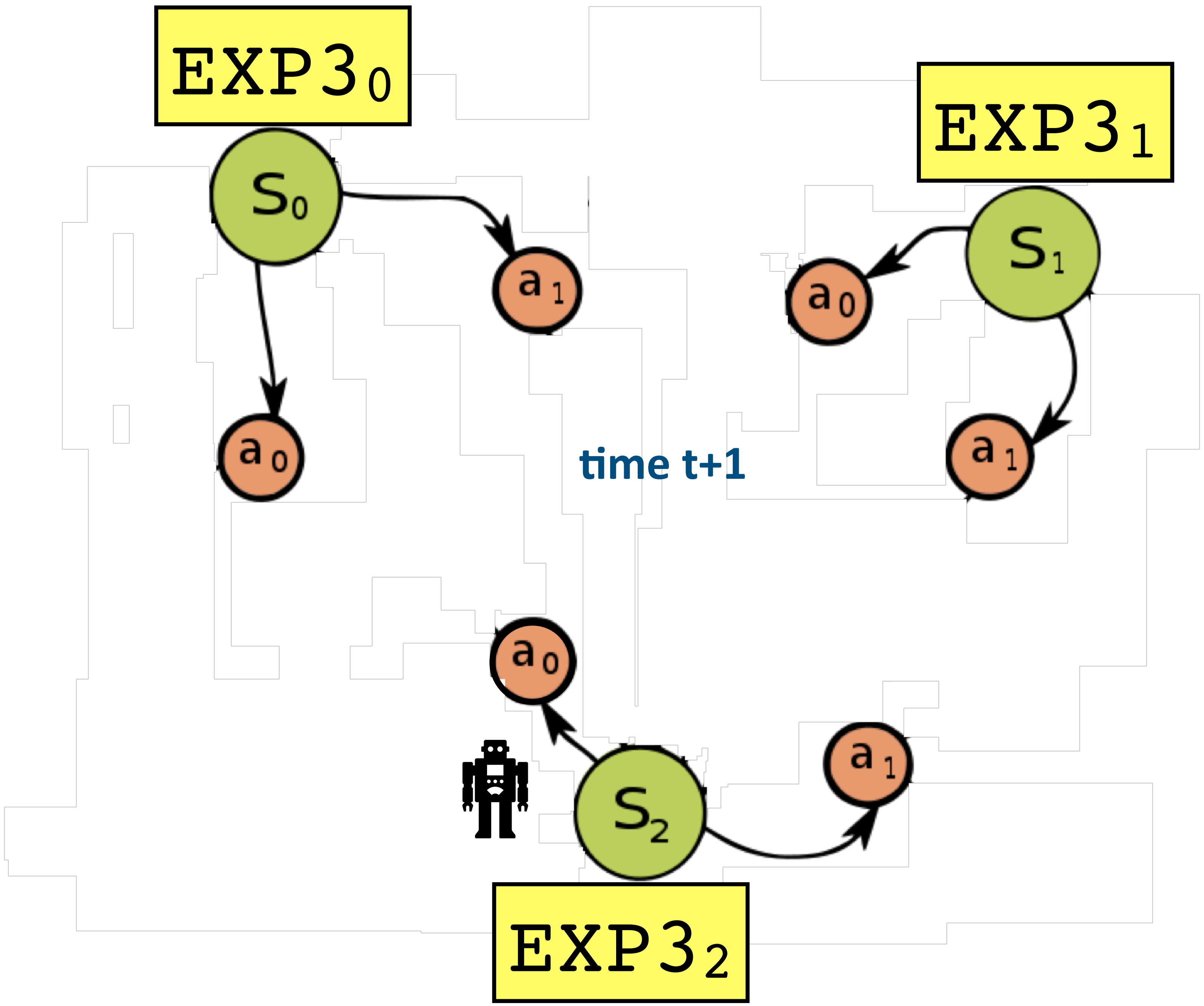
Reinforcement learning (RL)



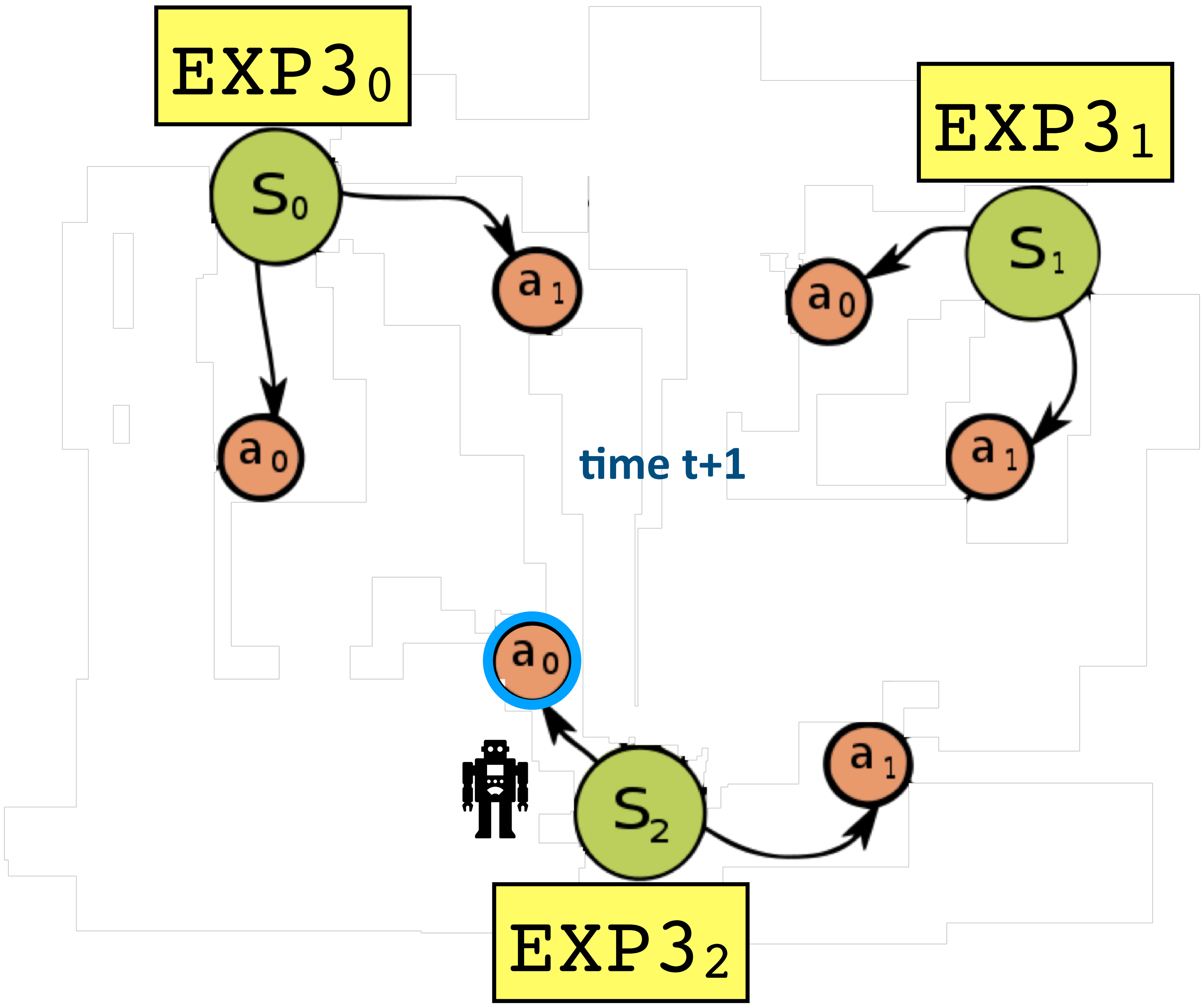
Reinforcement learning (RL)



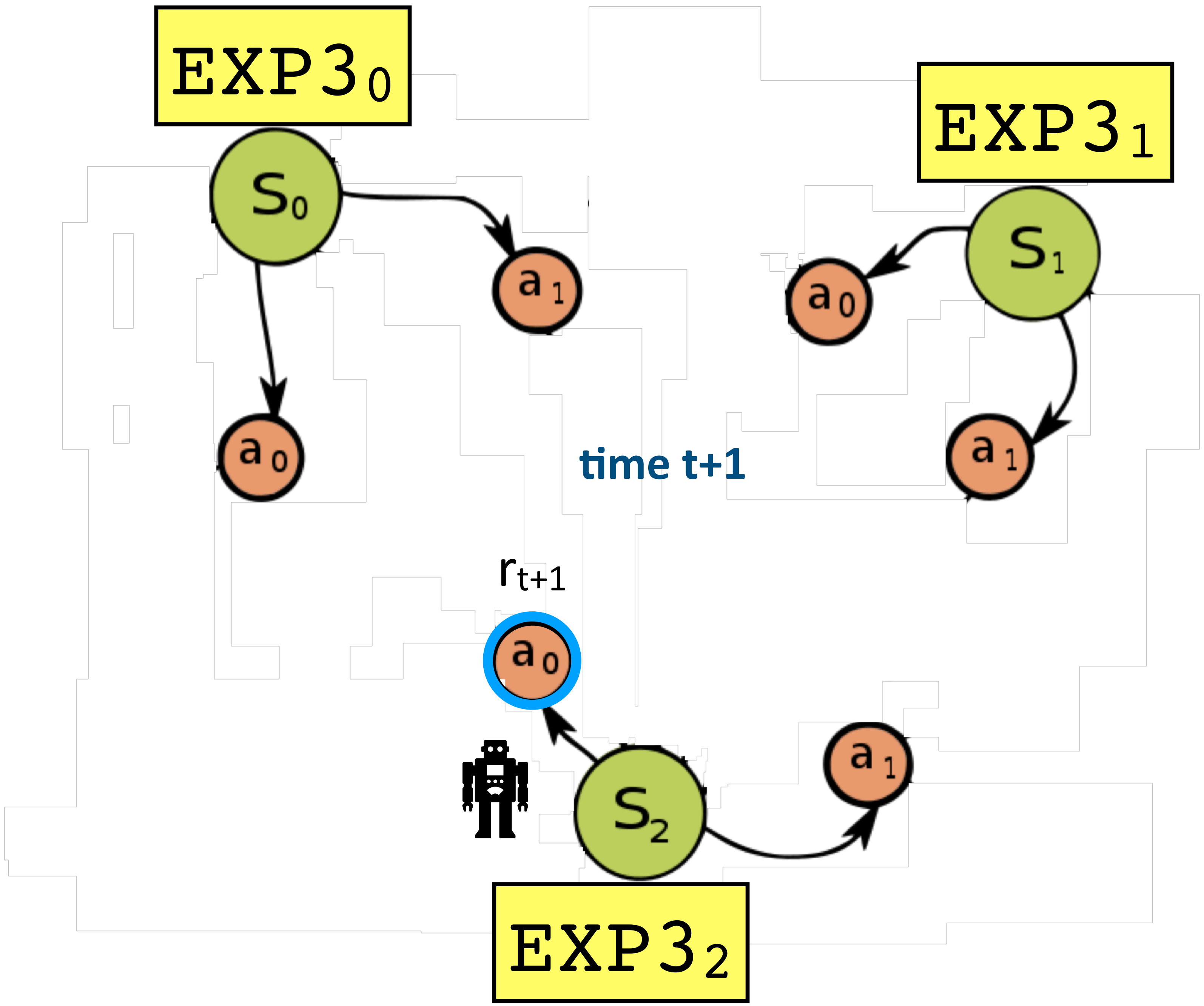
Reinforcement learning (RL)



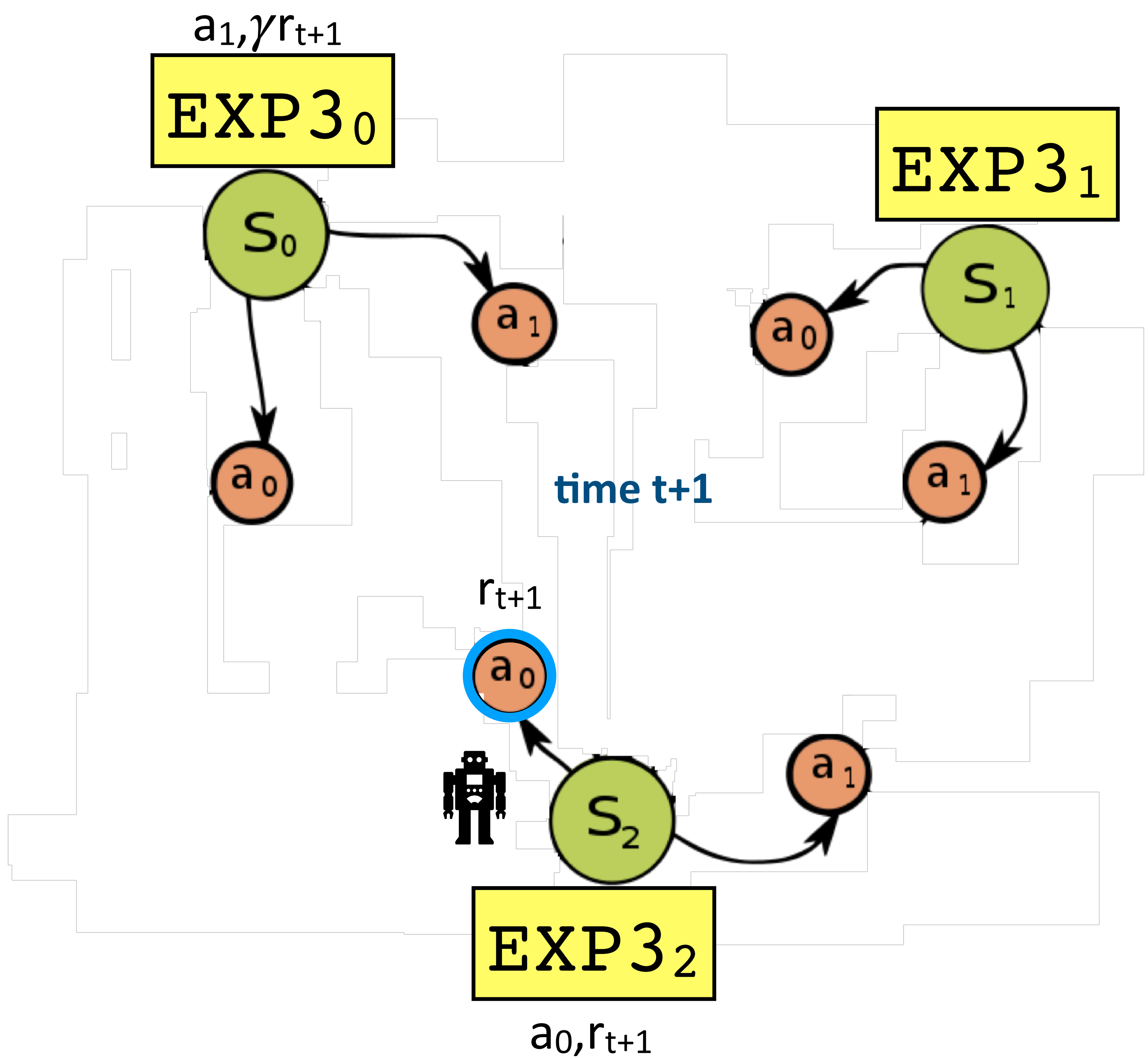
Reinforcement learning (RL)



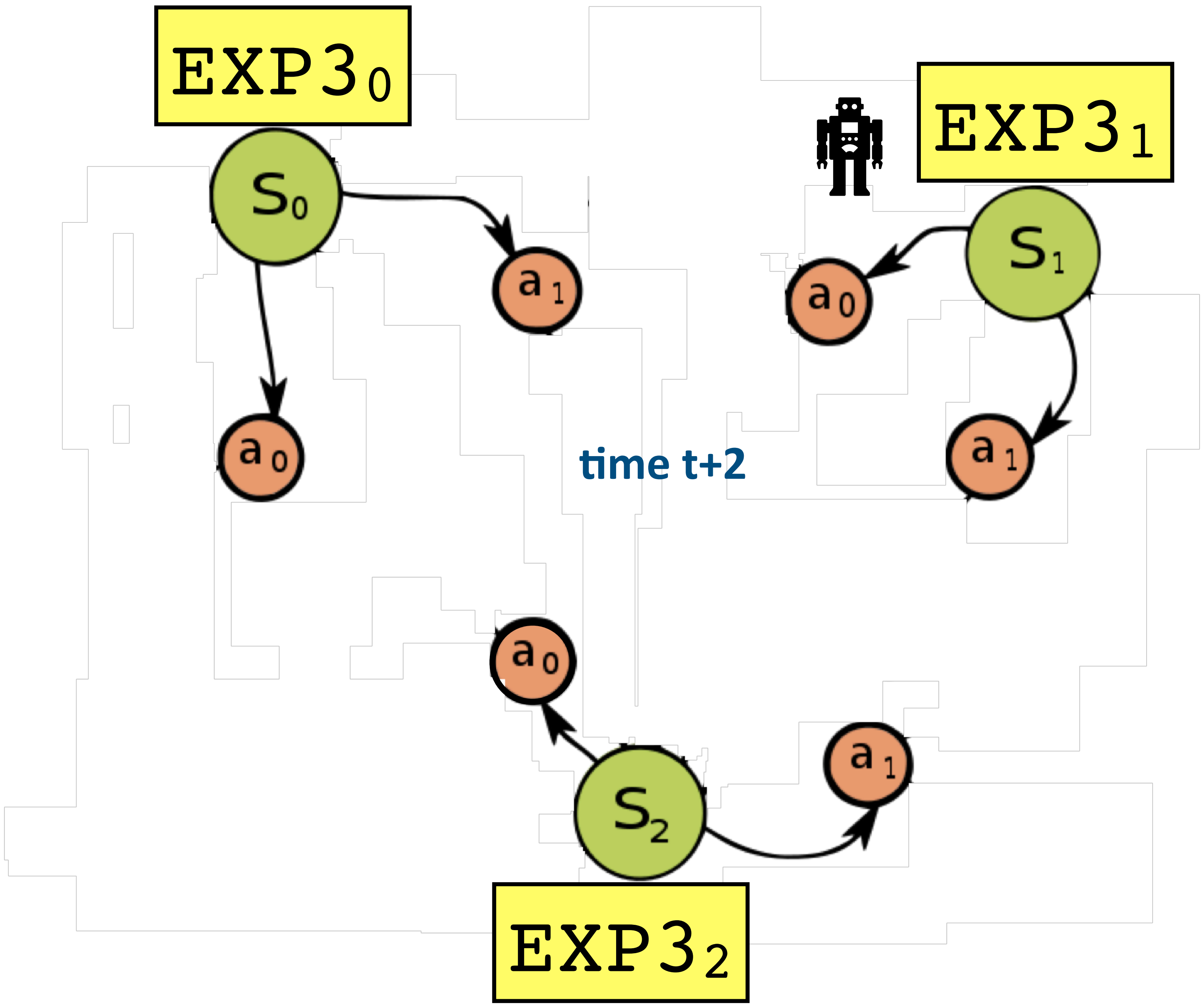
Reinforcement learning (RL)



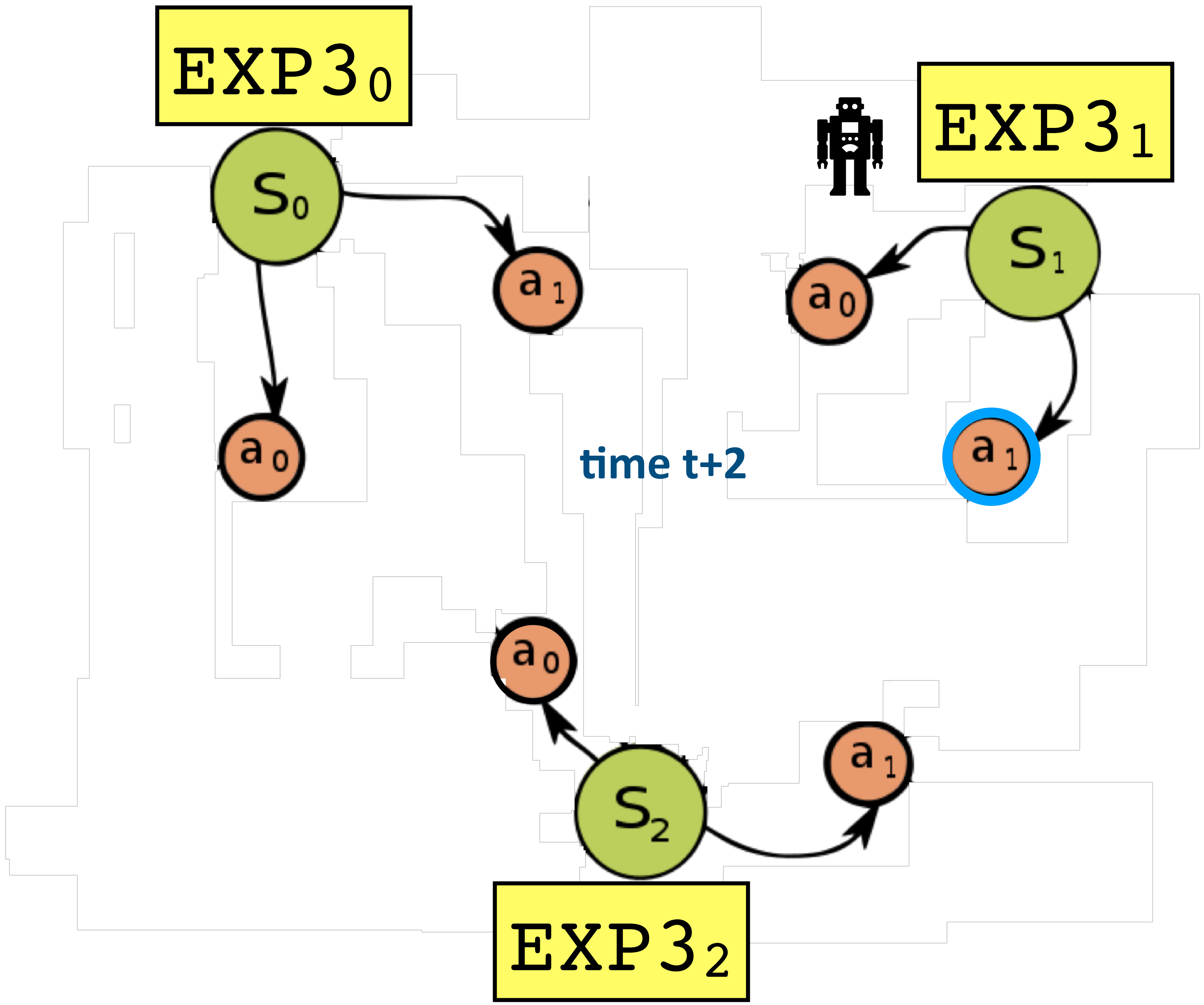
Reinforcement learning (RL)



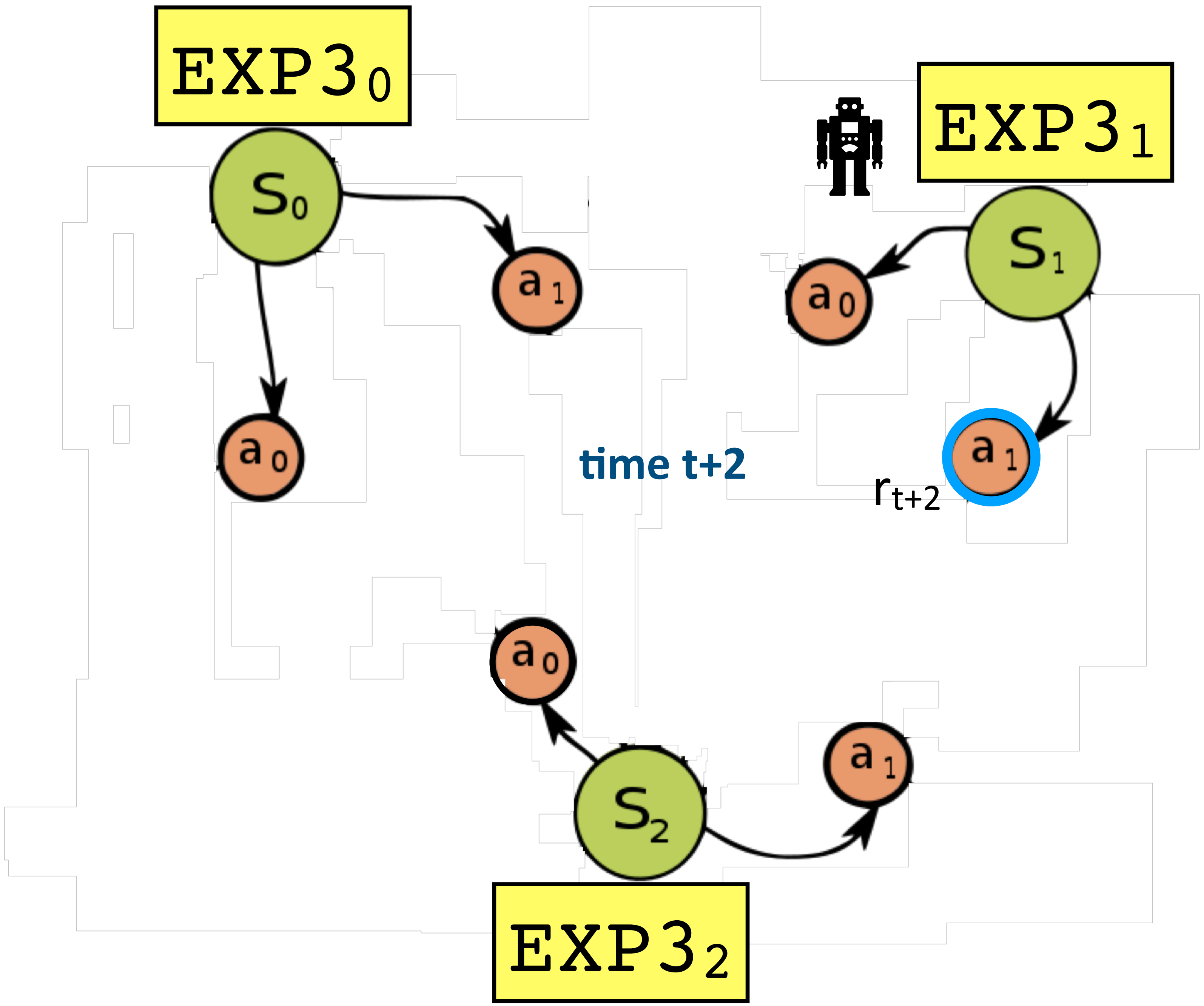
Reinforcement learning (RL)



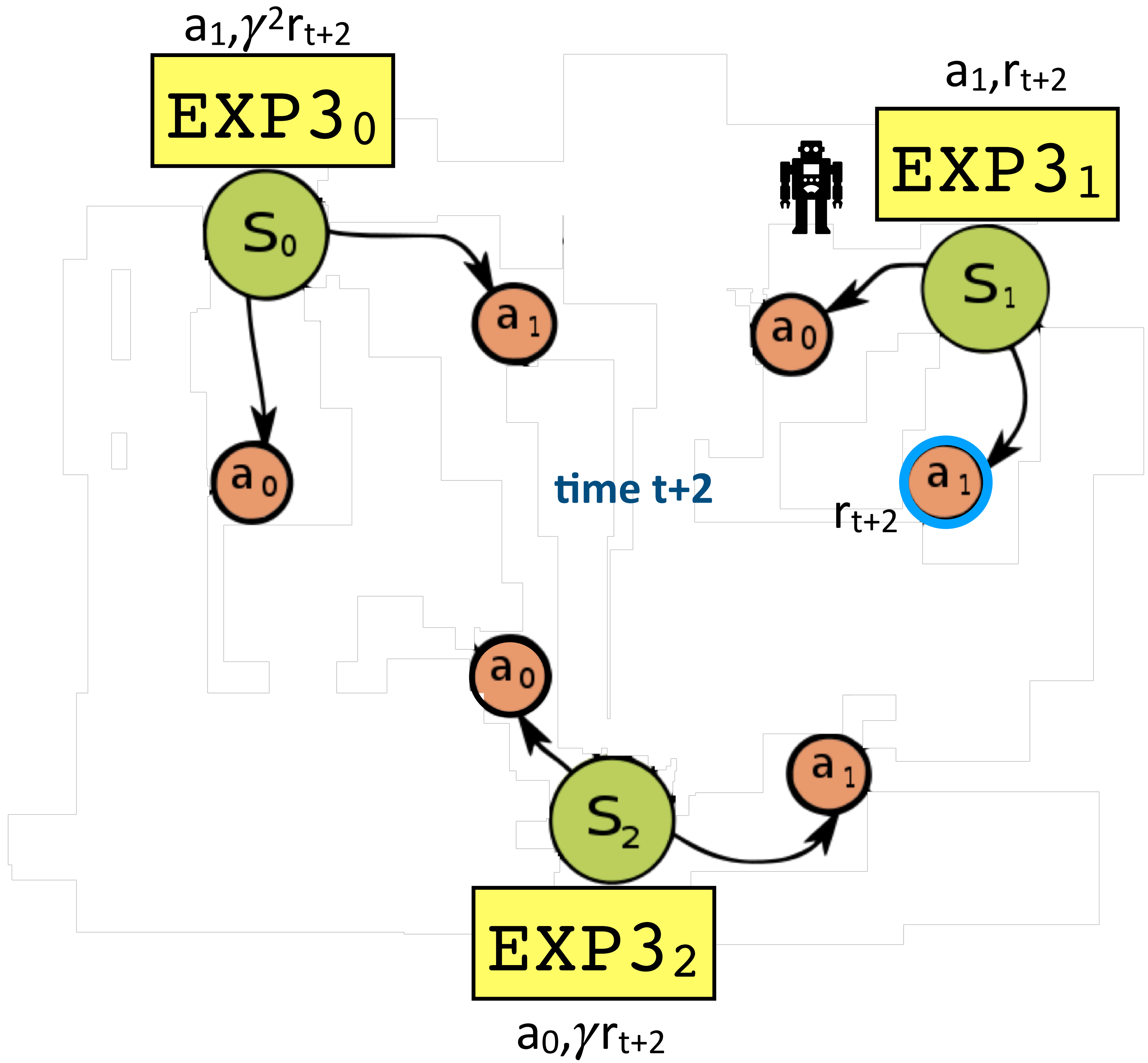
Reinforcement learning (RL)



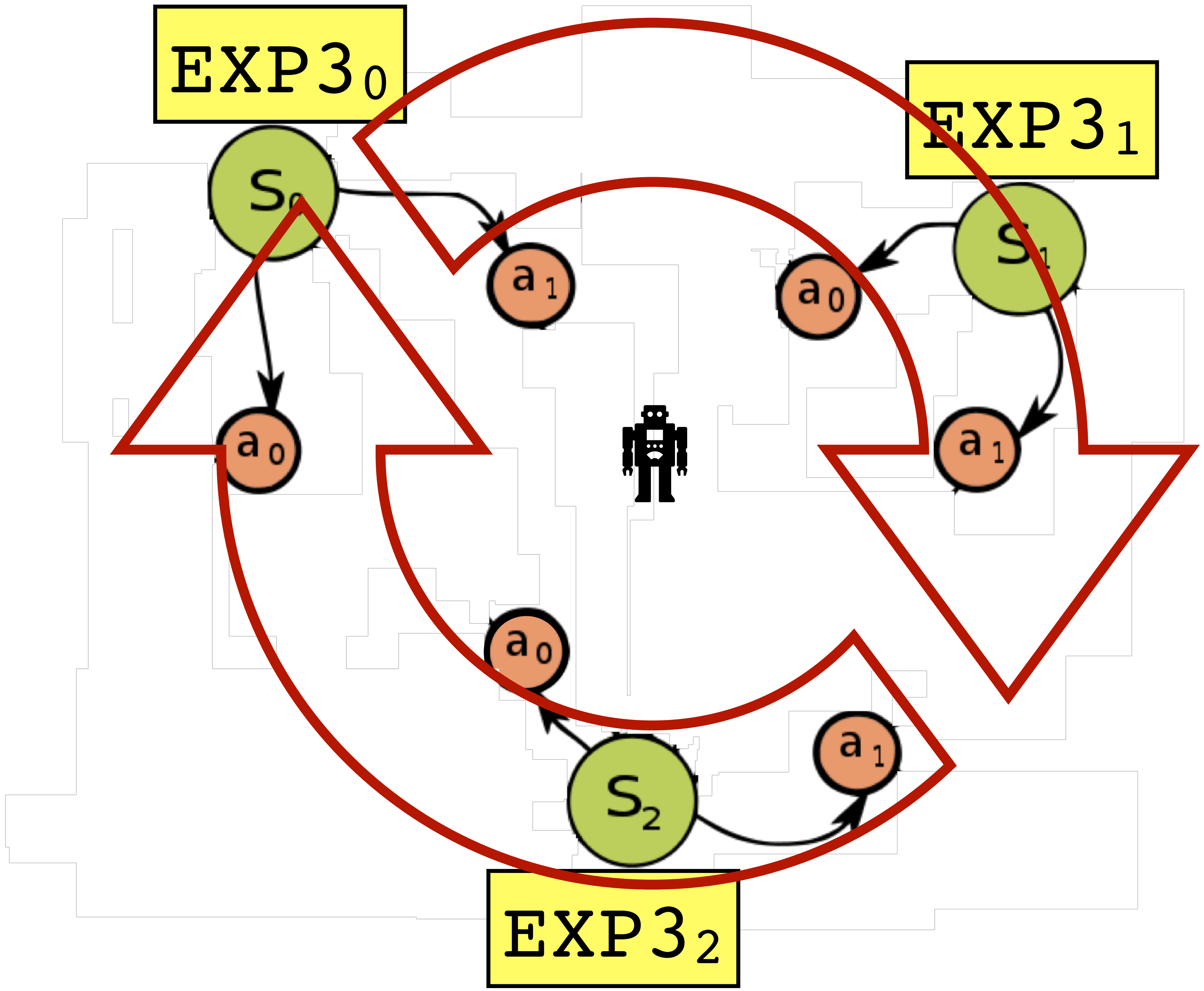
Reinforcement learning (RL)



Reinforcement learning (RL)



Reinforcement learning (RL)



Example result from my work in this area: an RL to bandits reduction

Theorem [Kash-R-Yu '24]: The algorithm `Bandits for MDPs`, when run using `EXP3` as the bandit learner, enjoys expected regret of

$$\tilde{\mathcal{O}}\left(\frac{\tau^2 S(S+A)}{\beta^3(1-\gamma)^2}\sqrt{T}\right),$$

where T = total time-steps, S = number of states, A = number of actions per state, γ = discount factor, τ = mixing time of MDP, and β = bounds $\pi_{\min}$.

Example result from my work in this area: an RL to bandits reduction

Theorem [Kash-R-Yu '24]: The algorithm `Bandits for MDPs`, when run using `EXP3` as the bandit learner, enjoys expected regret of

$$\tilde{\mathcal{O}}\left(\frac{\tau^2 S(S+A)}{\beta^3(1-\gamma)^2}\sqrt{T}\right),$$

where T = total time-steps, S = number of states, A = number of actions per state, γ = discount factor, τ = mixing time of MDP, and β = bounds $\pi_{\min}$.

By making reductions easier and improving their bounds, we could better deploy algorithms outside their designed scope!

Discussion

- AI, in the broader sense, has long been a tool in our toolkit. Tailored ML is now clearly useful, e.g. copilot for lean proofs and AlphaProof.
- LLMs are reaching the point where they can be useful too. This is all going in one direction, but how fast? How long will this “golden age” last? (Yes we are already in the Golden Age — take advantage of it now!)
- Deep learning practice is currently way past our understanding. LLMs, which use it, are even farther past. Mathematics can work to understand these models, possibly with their help! Theory will likely lag practice.
- Mathematics can make progress in other areas, e.g. understanding the properties of our data, helping deploy current algorithms in new areas, and of course others that I did not cover.