

kinda in the direction of
A route towards quantum-enhanced artificial intelligence

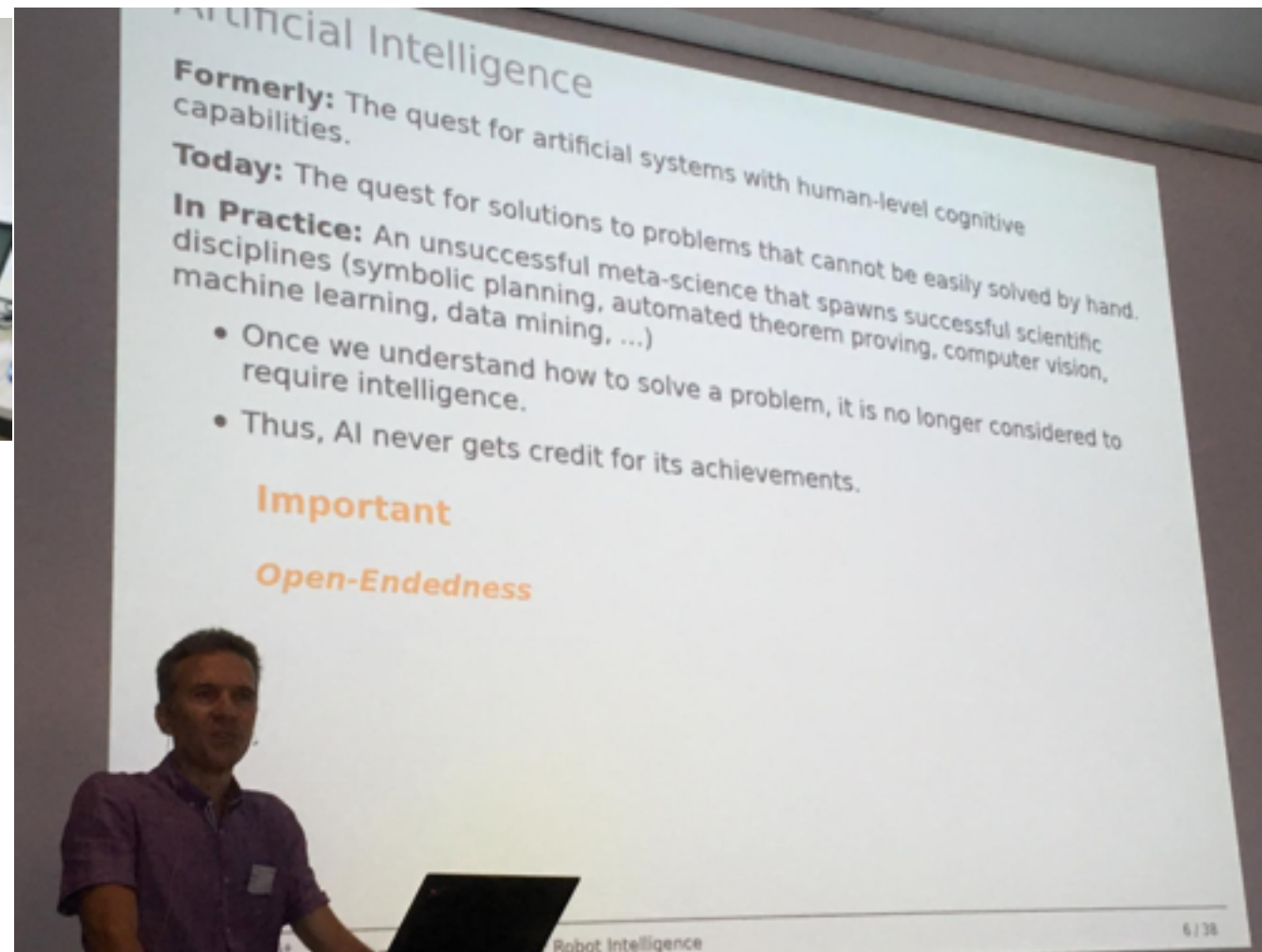
Vedran Dunjko

v.dunjko@liacs.leidenuniv.nl

What is AI

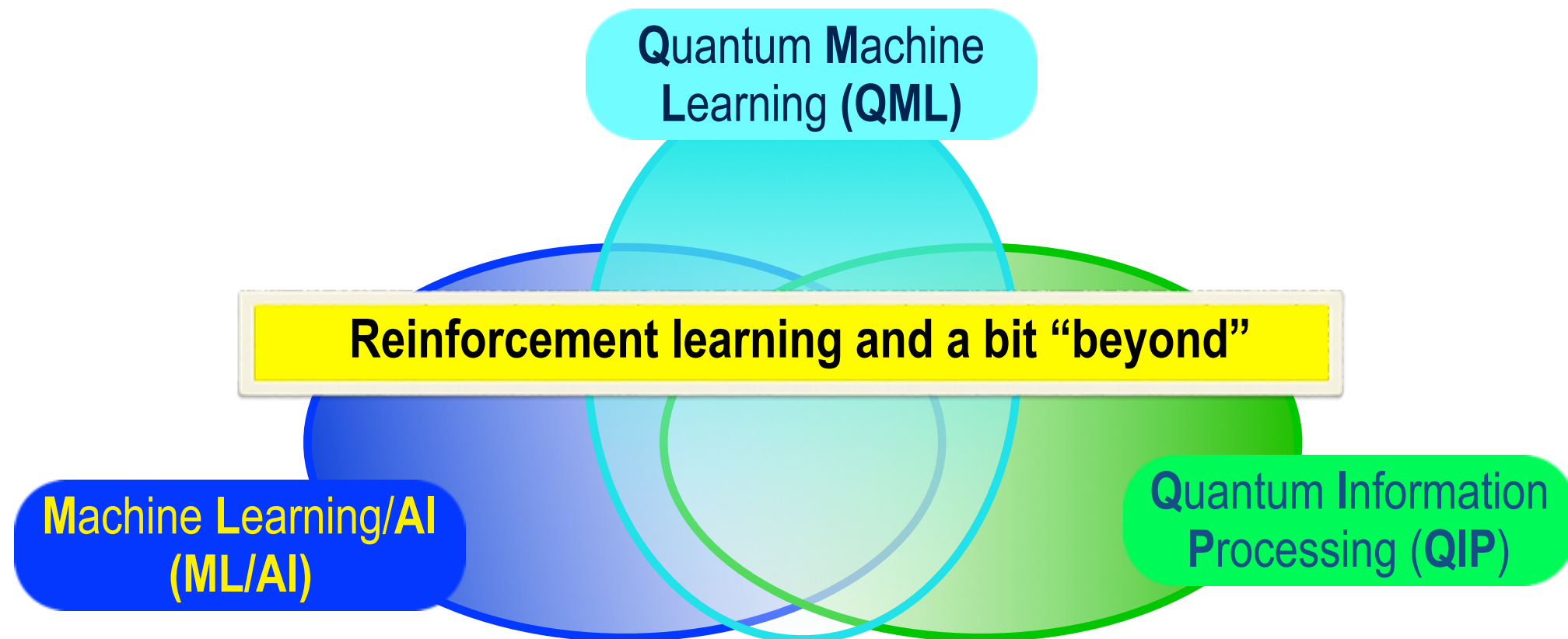


Justus Piater



Piater: “An *unsuccessful* meta-science that *spawns successful scientific disciplines*”
“**Catch-22**: once we understand how to solve a problem, it is no longer considered to require intelligence...”

What is this talk about? So what is AI? All? Nothing?



Outline

Part 1: “Ask not what Reinforcement Learning can do for you”

🌐 The theory, bottlenecks and applications

Part 2: “... ask what you can do for reinforcement learning...”

🌐 Quantum environments and model-based learning

Part 3: “... and for some aspects of planning on small QCs”

🌐 Learning and reasoning (actually...SAT solving)

But... what is Machine Learning?



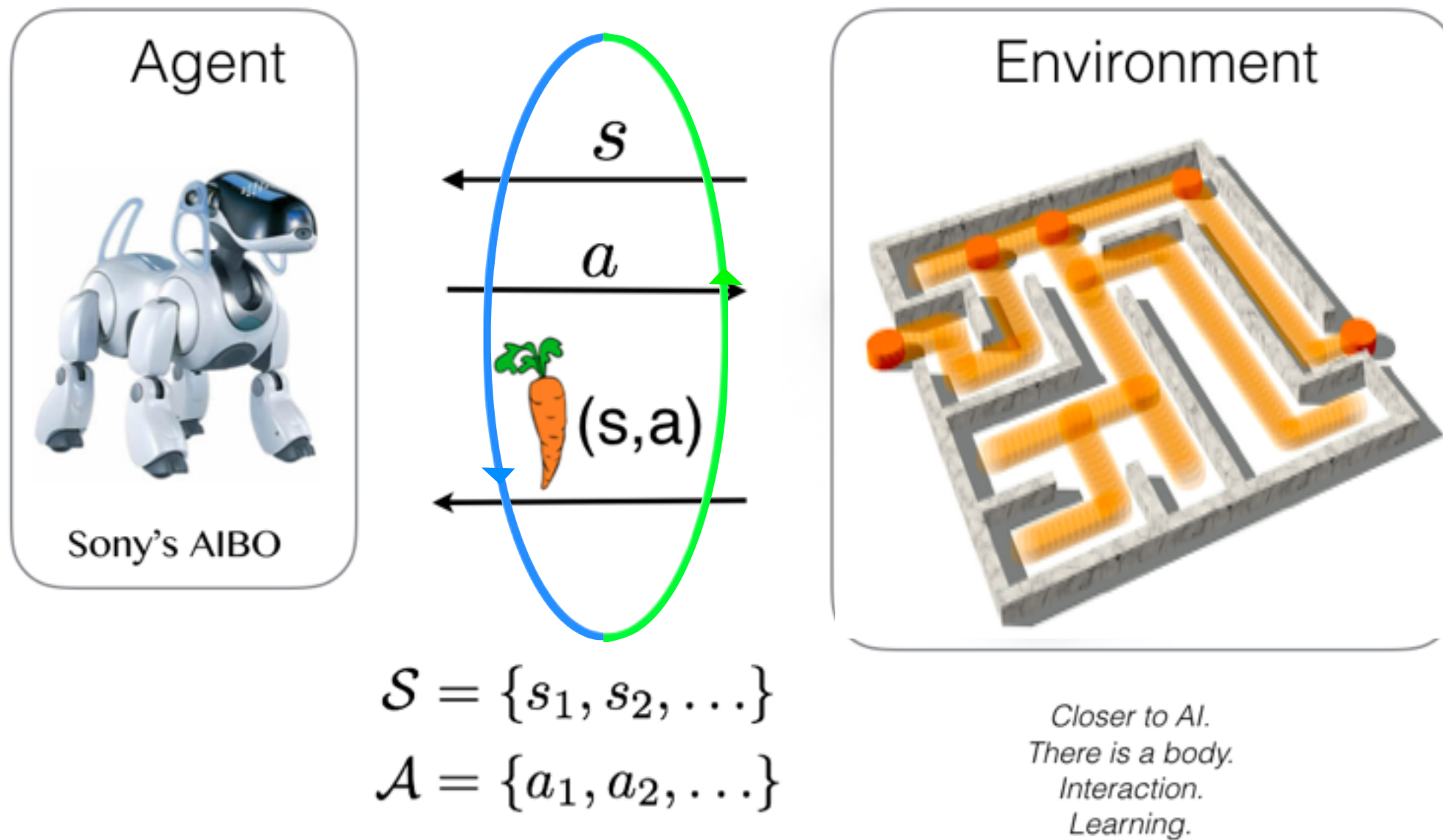
Learning $P(\text{labels}|\text{data})$ given
samples from $P(\text{data}, \text{labels})$

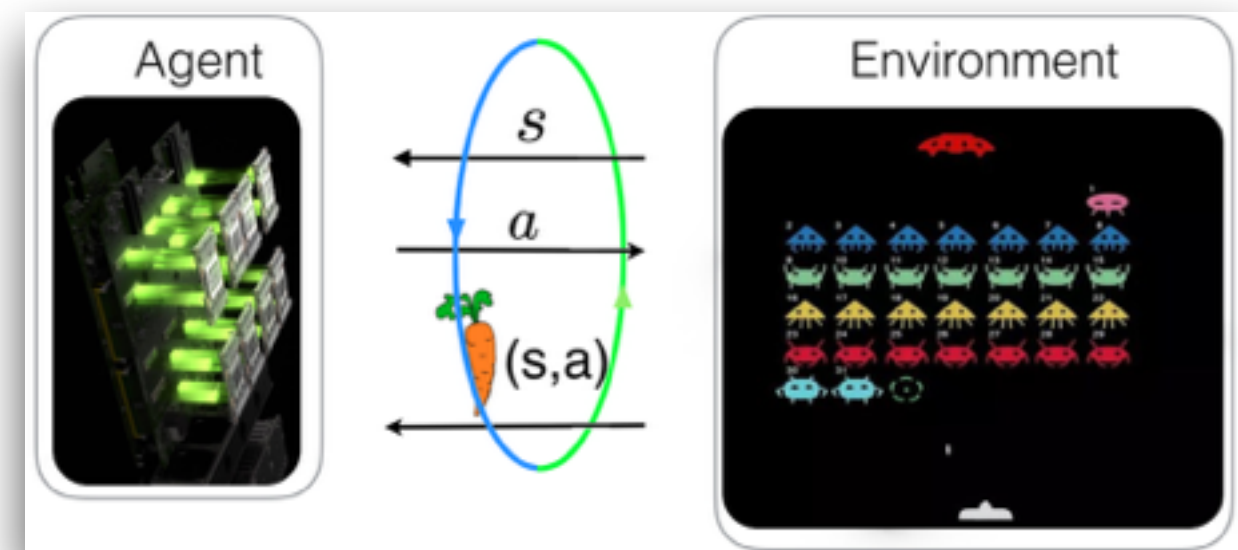
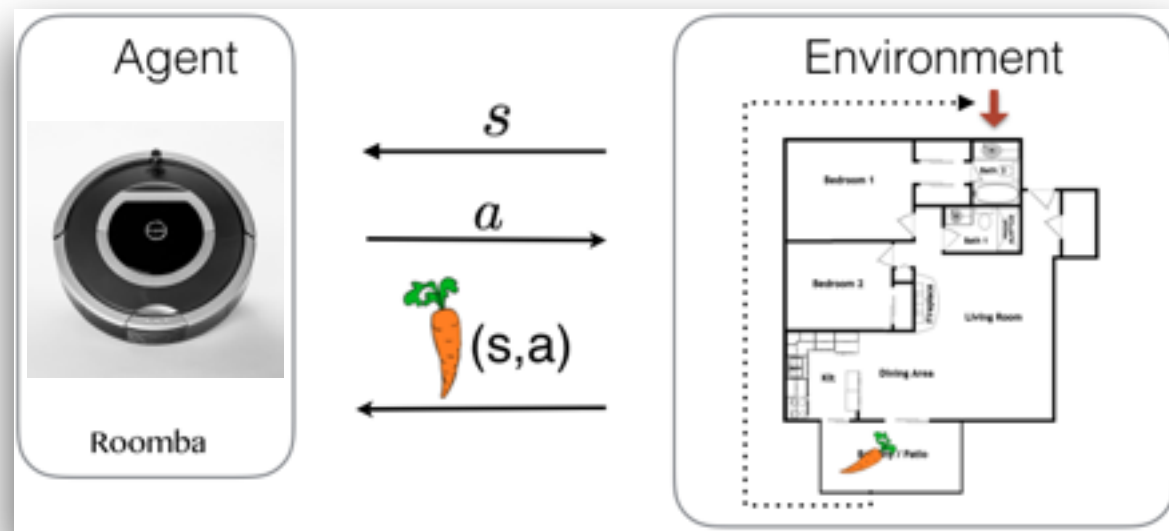
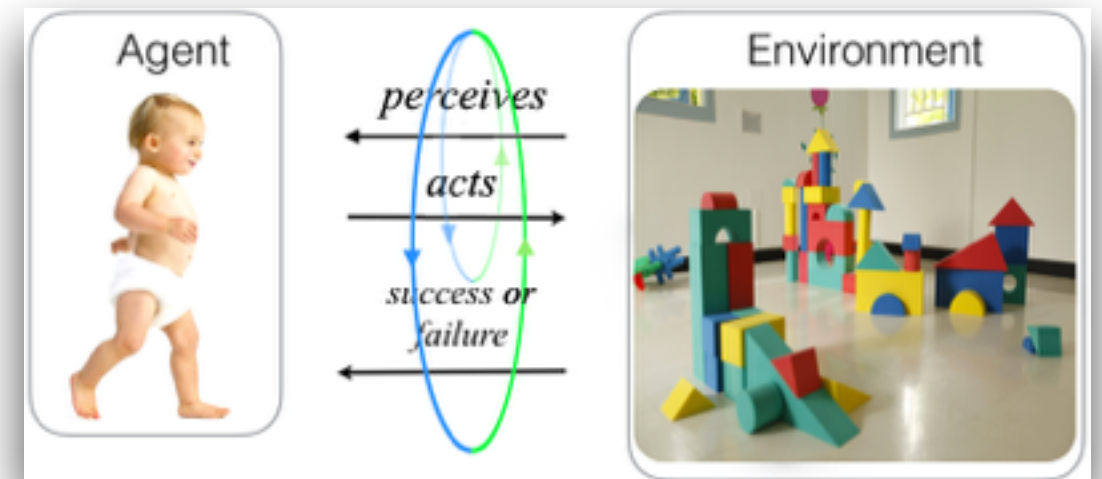
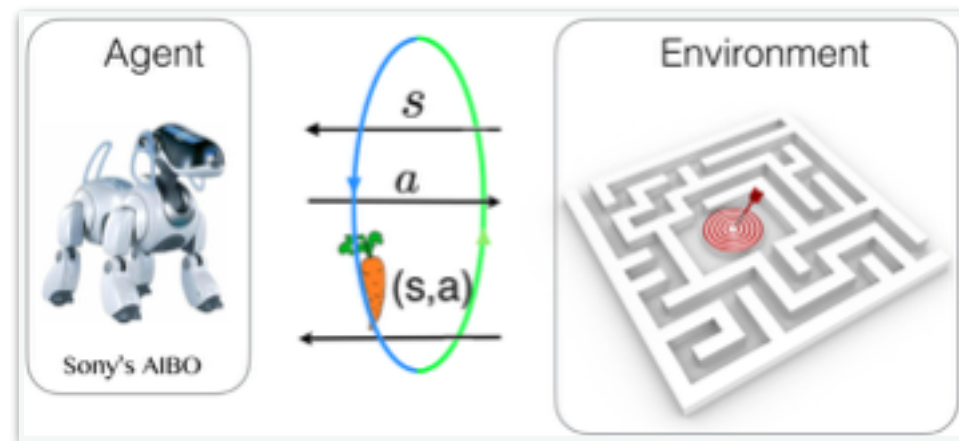
Learning structure in $P(\text{data})$
give samples from $P(\text{data})$

Generalize knowledge

Generate knowledge

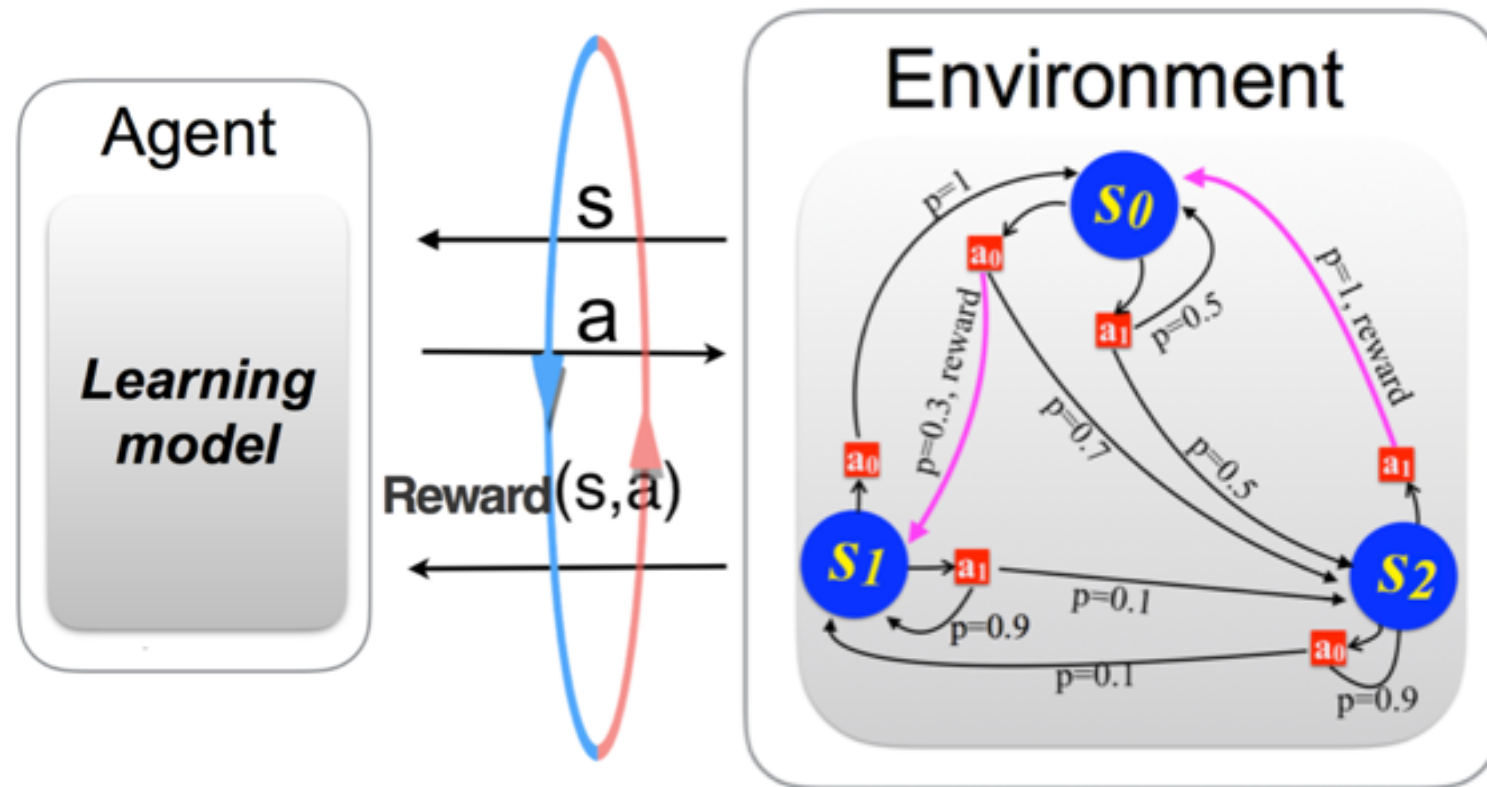
Reinforcement learning: *Agent - environment paradigm*





Also: MIT technology review breakthrough technology of 2017
[AlphaGo anyone?]

RL *more formal*



Basic concepts:

Environment:

Markov Decision Process

Policy: $\pi(a|s)$

Return: $\bar{R} = \bar{r}_1 + \bar{r}_2 + \dots + \bar{r}_k + \dots$

Figures of merit:

finite-horizon: $R_N = \sum_{t \leq N} \bar{r}_t$

infinite-horizon: $\bar{R} = \sum_k \gamma^k \bar{r}_k$

Optimality: $\pi_\gamma^*(a|s)$

Is that all?

- More complicated than it seems already in the simplest case; value iteration, policy search, value function approximation, model-free, model-based, actor-critic, *Projective Simulation*...
 - Infinite action/state spaces
 - Partially observable MDPs
 - Goal MDPs
- Knowledge transfer (and representation), Planning...
- ...AI?

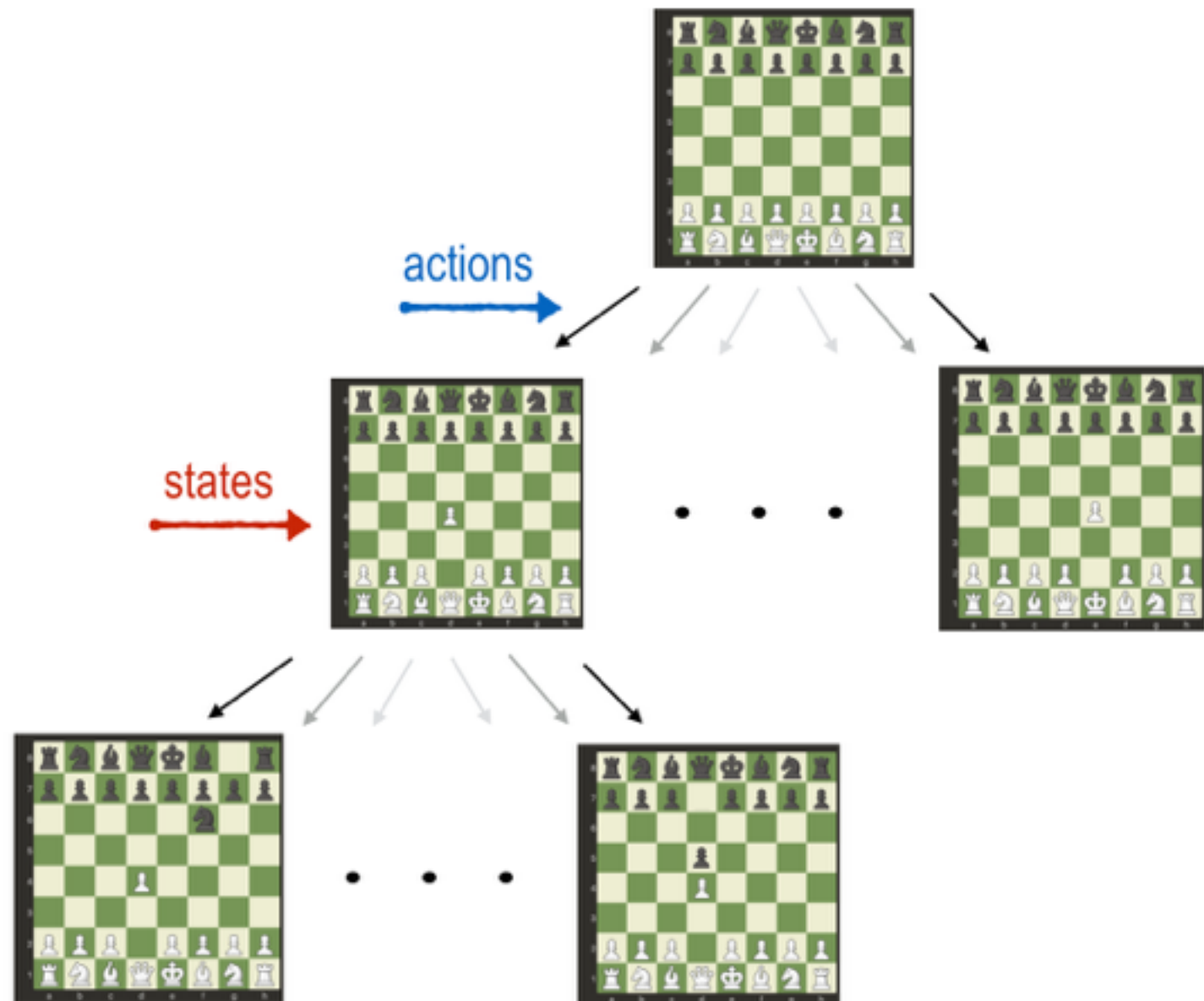
Reinforcement learning vs. supervised learning

- learning “*action*” - “*state*” associations similar to “*label*” - “*data*” association
- how data is *accessed*, and how it is *organized* is different
- not *i.i.d*, not learning a distribution, examples provided implicitly (delayed reward, credit assignment problems)

RL vs. SL

Example: learning chess

- MDP is tree-like

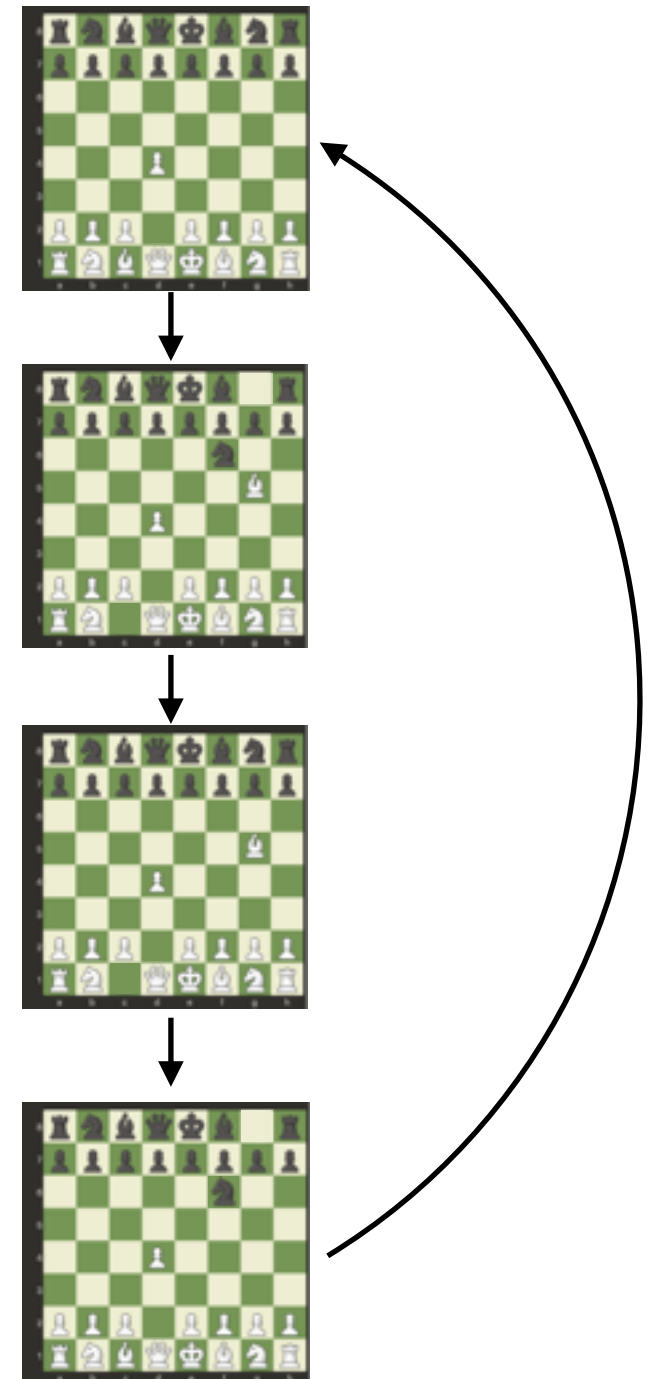


RL vs. SL

Example: learning chess

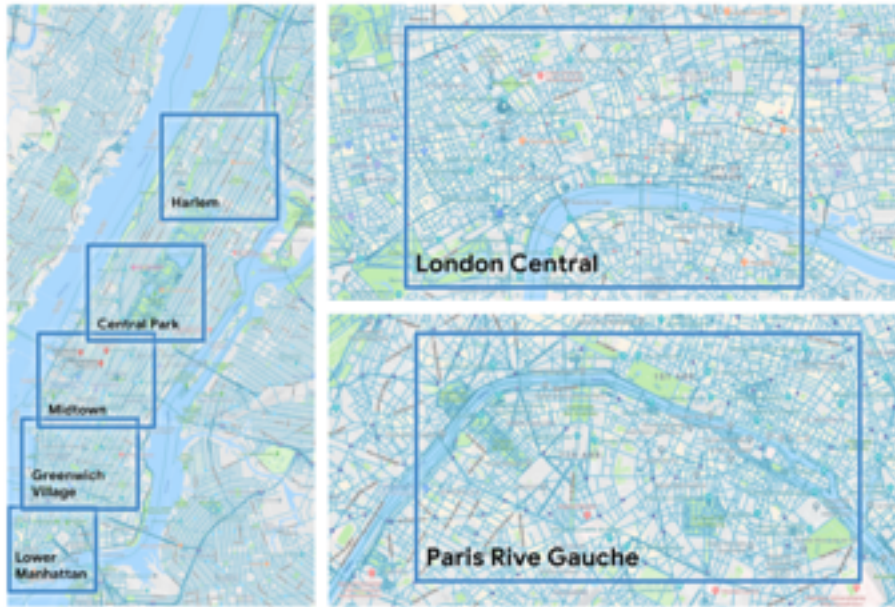
- MDP is tree-like, *but not a tree*
- *examples* given only indirectly: credit assignment (unless **immediate reward**)
- strong **causal & temporal** structure (agent's actions influence the environment)

NB: supervised learning, oracle identification, etc.
can be cast as (degenerate) MDP learning problems



From pretty MDPs ... to Using RL in Real Life

Navigating a city...



Stop-motion films of agent trained in Paris. The images are superposed with a map of the city, showing the goal location (in red) and the agent location and field of view (in green). Note that the agent does not see the map, only the lat/lon coordinates of the goal location.

<https://sites.google.com/view/streetlearn>

P. Mirowski et. al, *Learning to Navigate in Cities Without a Map*, arXiv:1804.00168




So how to do *RL* (real life) *RL*



- via *pure RL*: know only what to do in situations one encounters
- better: **generalize over personal experiences** — do similar in similar situations
(still, *unlike in big data*, “training set” is a near-negligible fraction...)
- what we actually do: **generate fictitious experiences**
 (“if I play X, my opponent plays Y, I play Z....”)

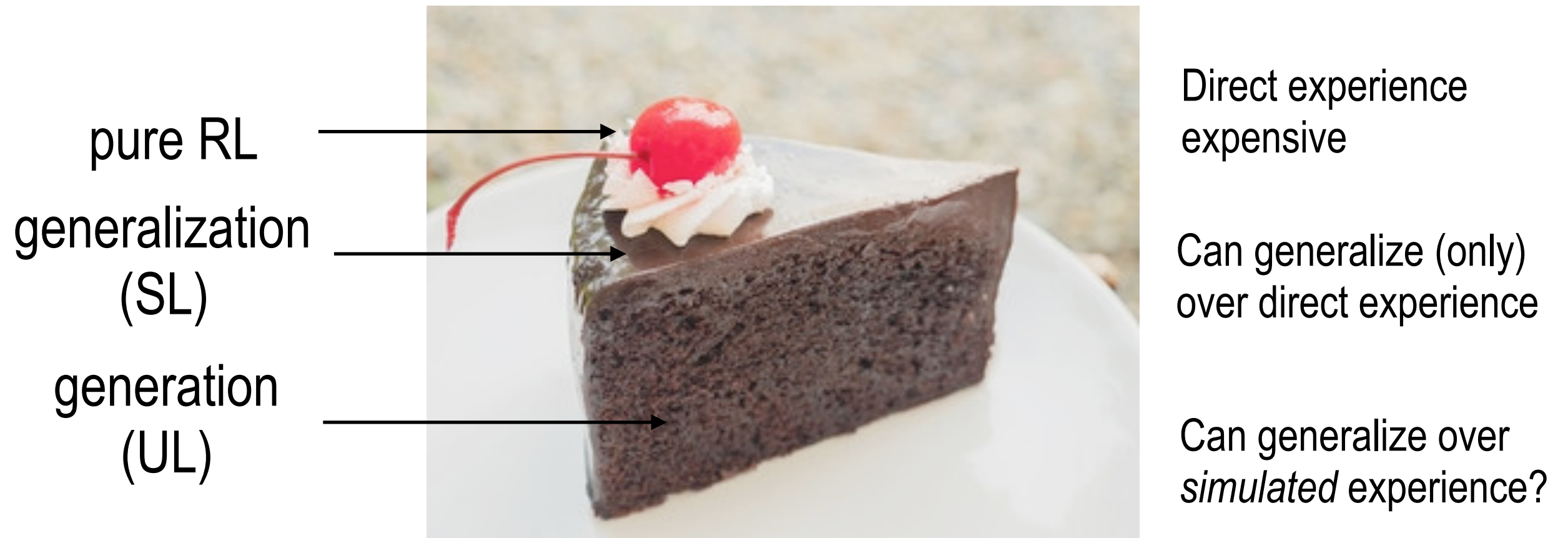
conjecture: most human experiences *are fictitious (tilted face problem)*

Learning unified

- via pure RL:  old-school RL *Slow*
- better: **generalize over personal experiences**  supervised learning-like Doing...ok
- further: **generate fictitious experiences**  unsupervised learning-like **Hard as heck**

conjecture: most human experiences are fictitious (*tilted face problem*)

“The cake picture” for general RL/AI: unifying ML



“If intelligence was a cake, unsupervised learning would be the cake, supervised learning would be the icing on the cake, and reinforcement learning would be the cherry on the cake.”

-Yann LeCun

even the cherry can be as complicated as you wish

Progress in RL (connecting RL, SL, and UL)

a) **generalization** (SL):

associating the correct actions, to previously unseen states

$$\pi(a|s) \xrightarrow{\text{function approximation}} \pi_{\theta}(a|s)$$

-linear models (Sutton, '88)

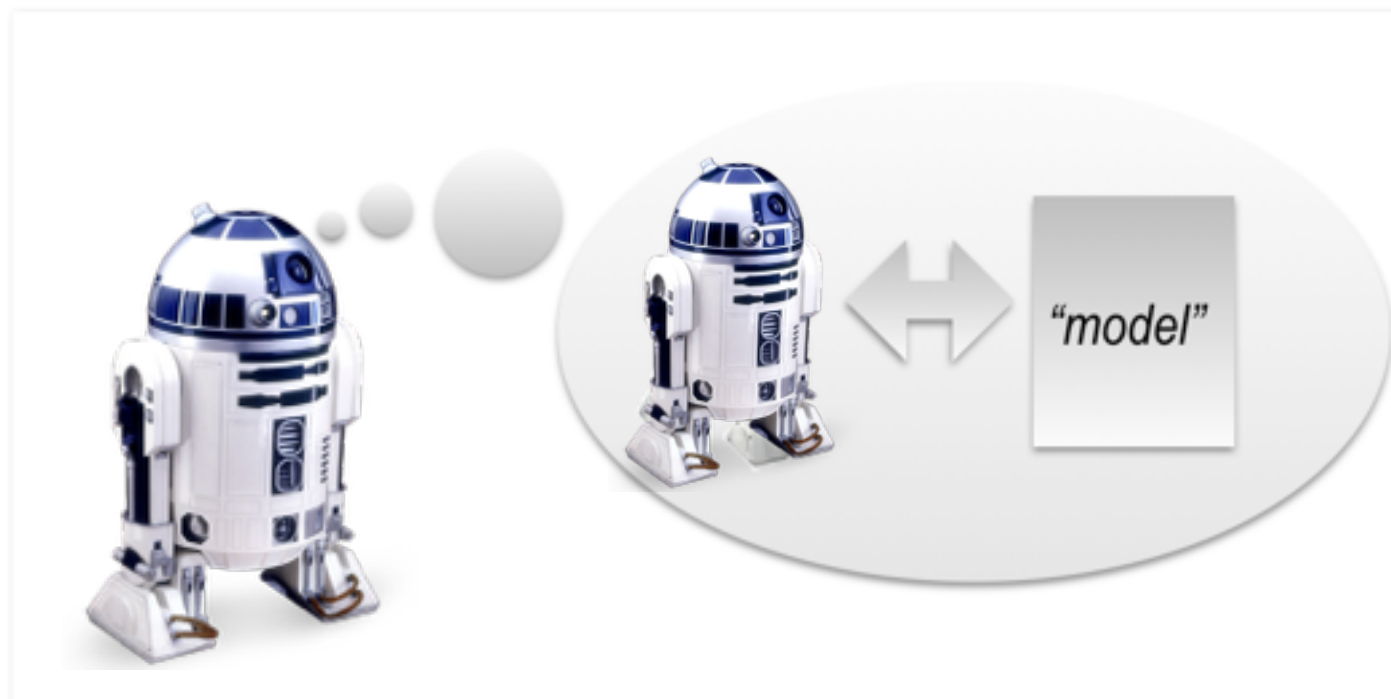
-neural networks (Lin, '92) $\xrightarrow[\text{(+ MCTS!)}]{\text{deep learning}}$ **AlphaGo**

-decision trees, etc...



?

b) **generation** (UL): *model-based learning*



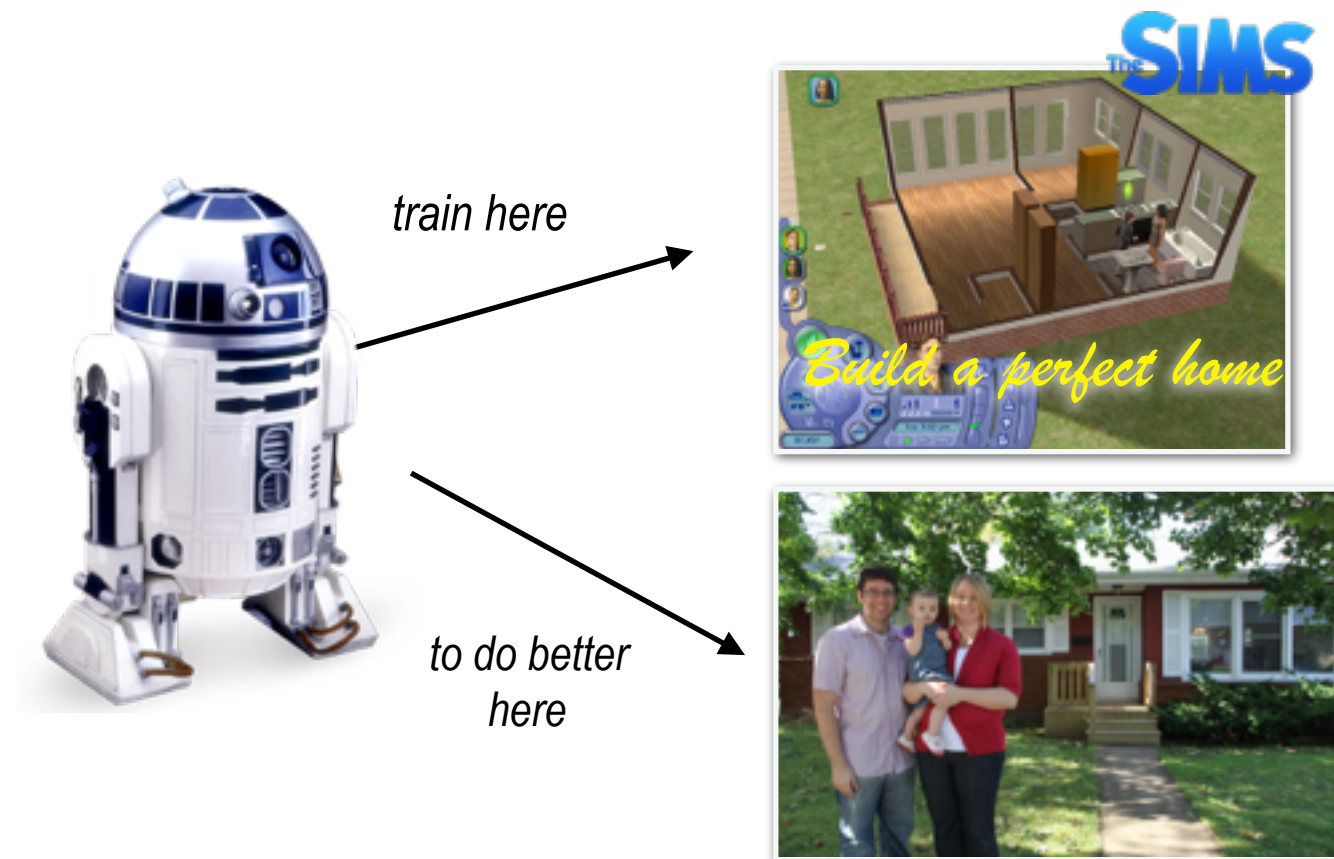
Another aspect:

2) generation as simulation



because real experiences can be painful
(and expensive)

What I want to do when I grow up



*good AI will
learn hierarchically
and transfer
the learned to a
new domain*

Pre-training will have at least two flavors...

- 1) reinforcement learning (slow, faster than real life)
- 2) optimization (find optimal patterns of behaviour)

Both are *computational bottlenecks*

Progress in RL (connecting RL, SL, and UL)

a) **generalization (SL):**

associating the correct actions, to previously unseen states

$$\pi(a|s) \xrightarrow{\text{function approximation}} \pi_{\theta}(a|s)$$

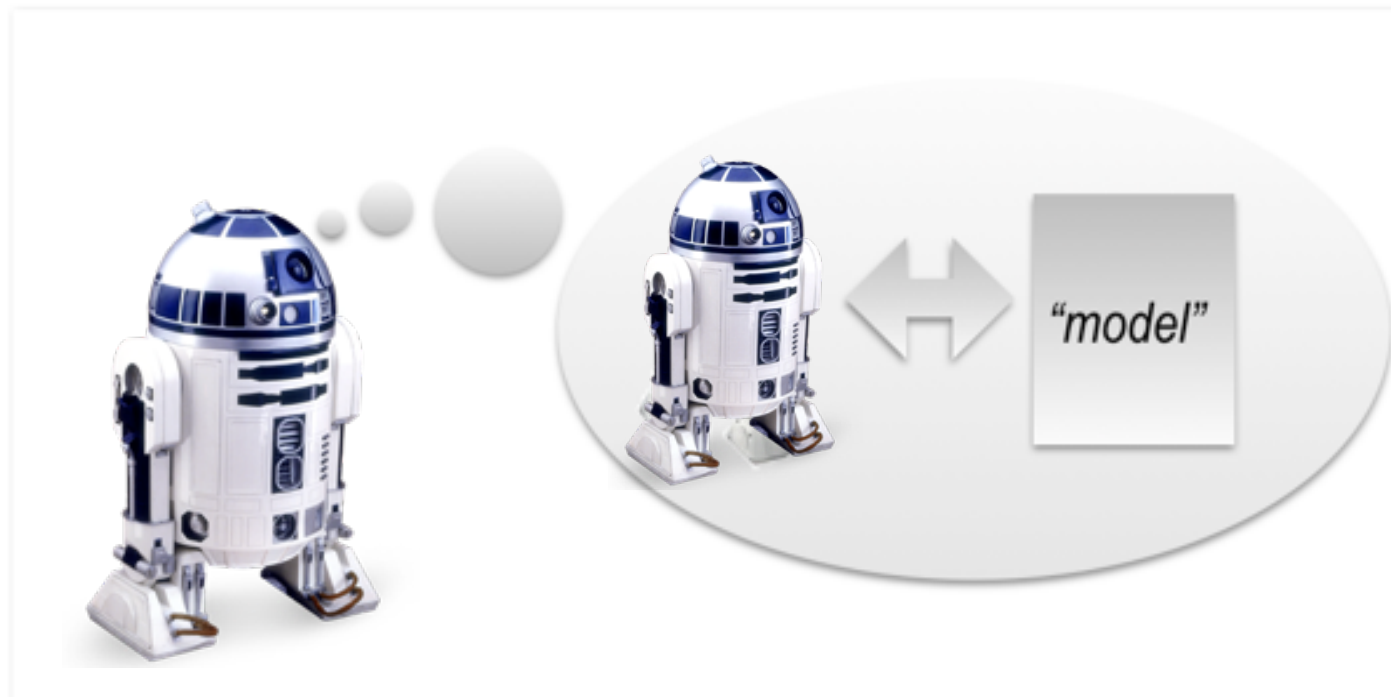
-linear models (Sutton, '88)

-neural networks (Lin, '92) $\xrightarrow[\text{(+ MCTS!)}]{\text{deep learning}}$ AlphaGo

-decision trees

Quantum enhancements have been considered for both problems. Here we focus on b)

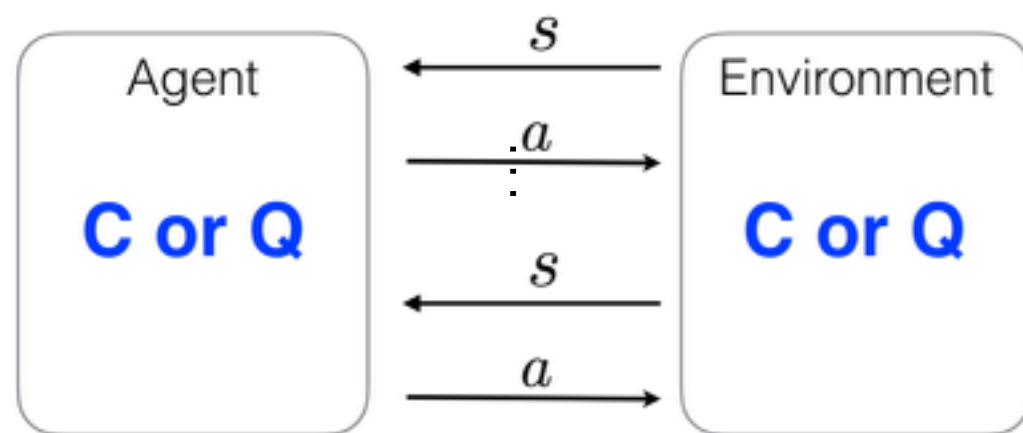
b) **generalization (SL): model-based learning**



Part 2: ... ask what you can do for reinforcement learning...

Can I RL *better* if the environment is *quantum*?
What are environments?

- Quantum Agent - Environment paradigm

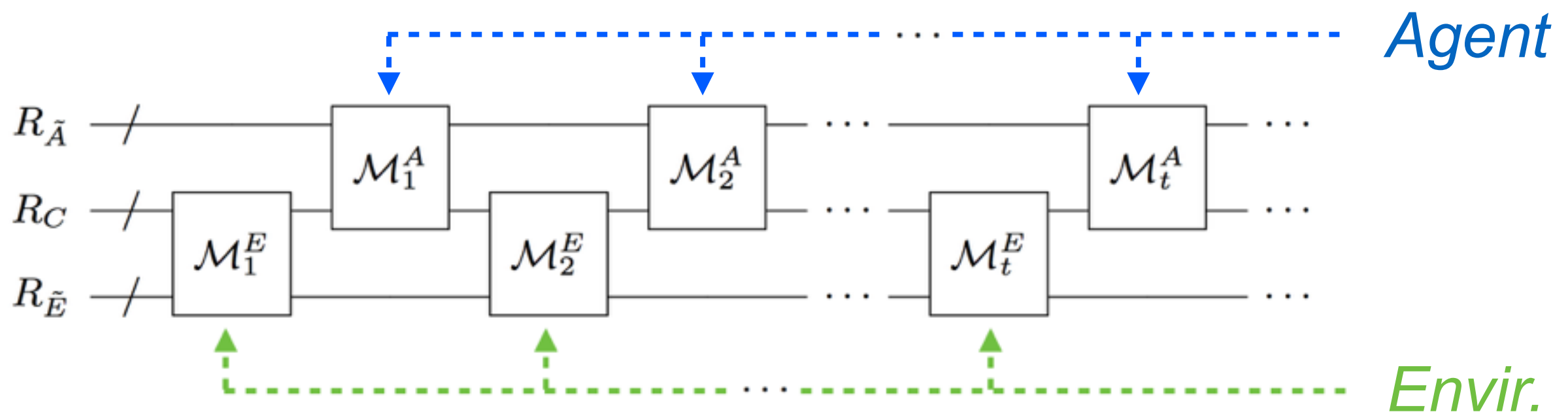


$$\mathcal{A} = \{a_1, a_2, \dots\} \quad \mathcal{S} = \{s_1, s_2, \dots\}$$

$$\Downarrow \quad \Downarrow$$

$$\mathcal{H}_{\mathcal{A}} = \text{span}\{|a_i\rangle\}, \quad \mathcal{H}_{\mathcal{S}} = \text{span}\{|s_i\rangle\}$$

is equivalent to



- Agents** (environments) are **sequences of CPTP maps**, acting on a private and a common register - the memory and the interface, respectively.
- Memory channels = combs = quantum strategies

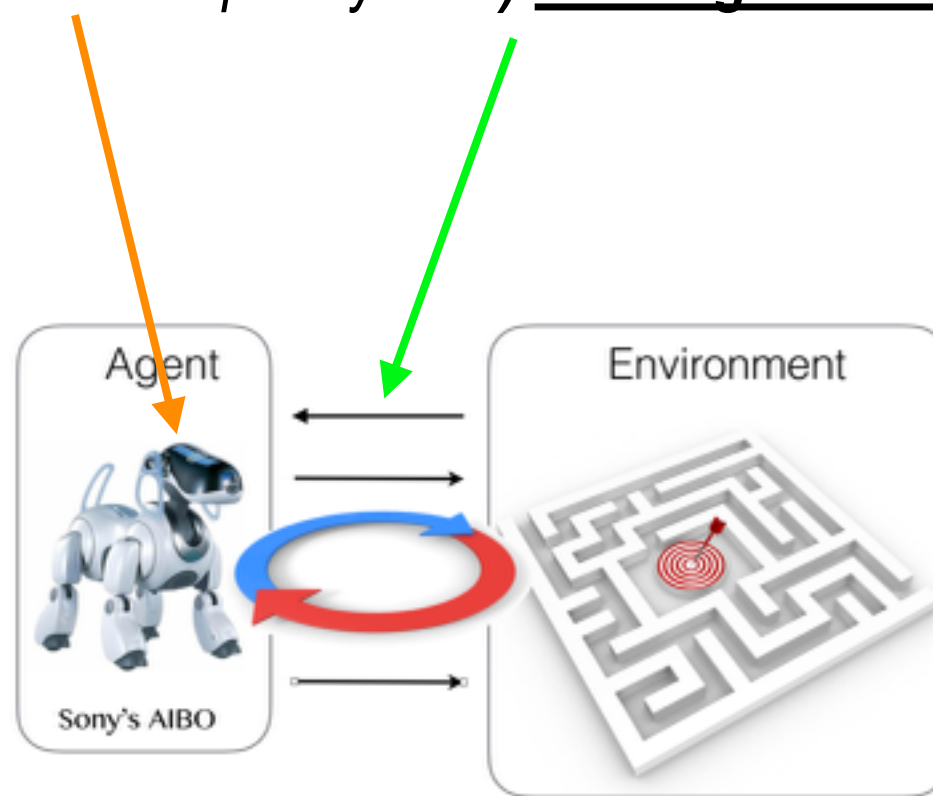
● *What is the motivation again?*

- 🌐 **Fundamental meaning of learning in the quantum world**

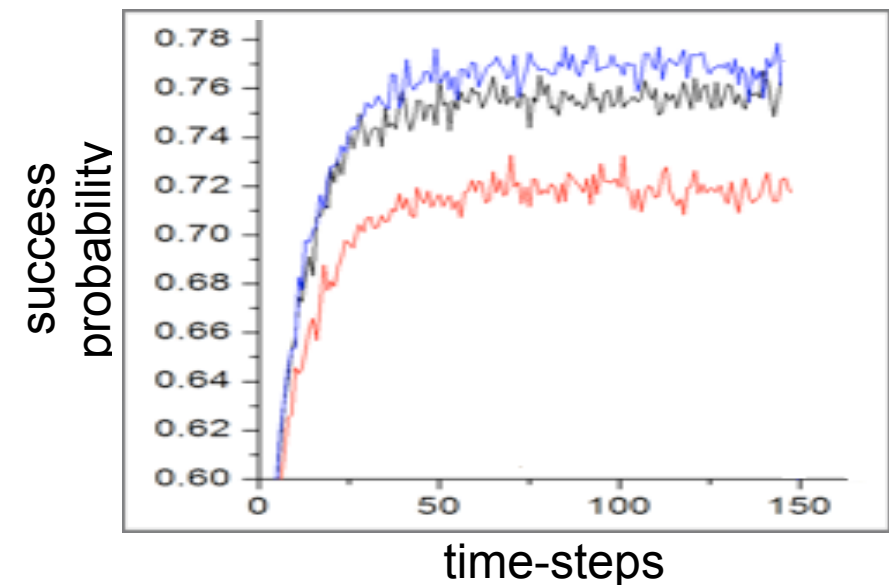
- 🌐 **Speed-ups! “faster”, “better” learning**

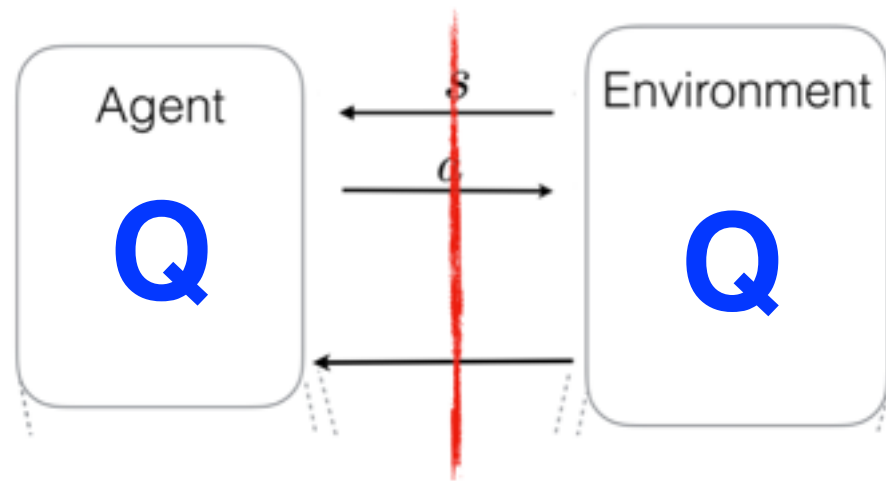
- 🌐 *What can we make better?*

a) computational complexity b) learning efficiency (“genuine learning-related figures of merit”)



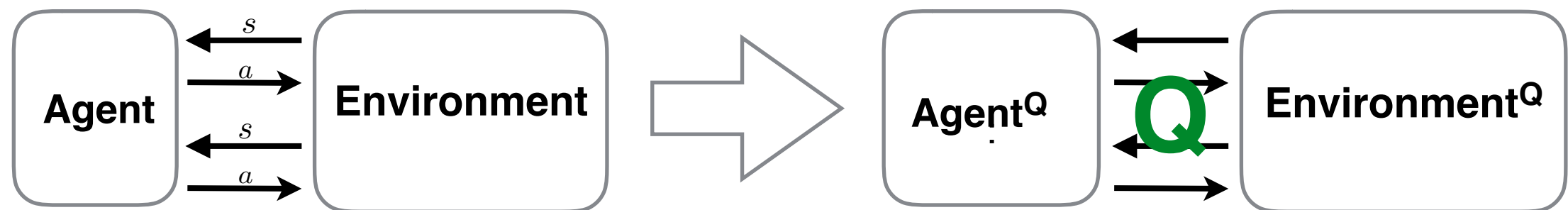
related to query complexity



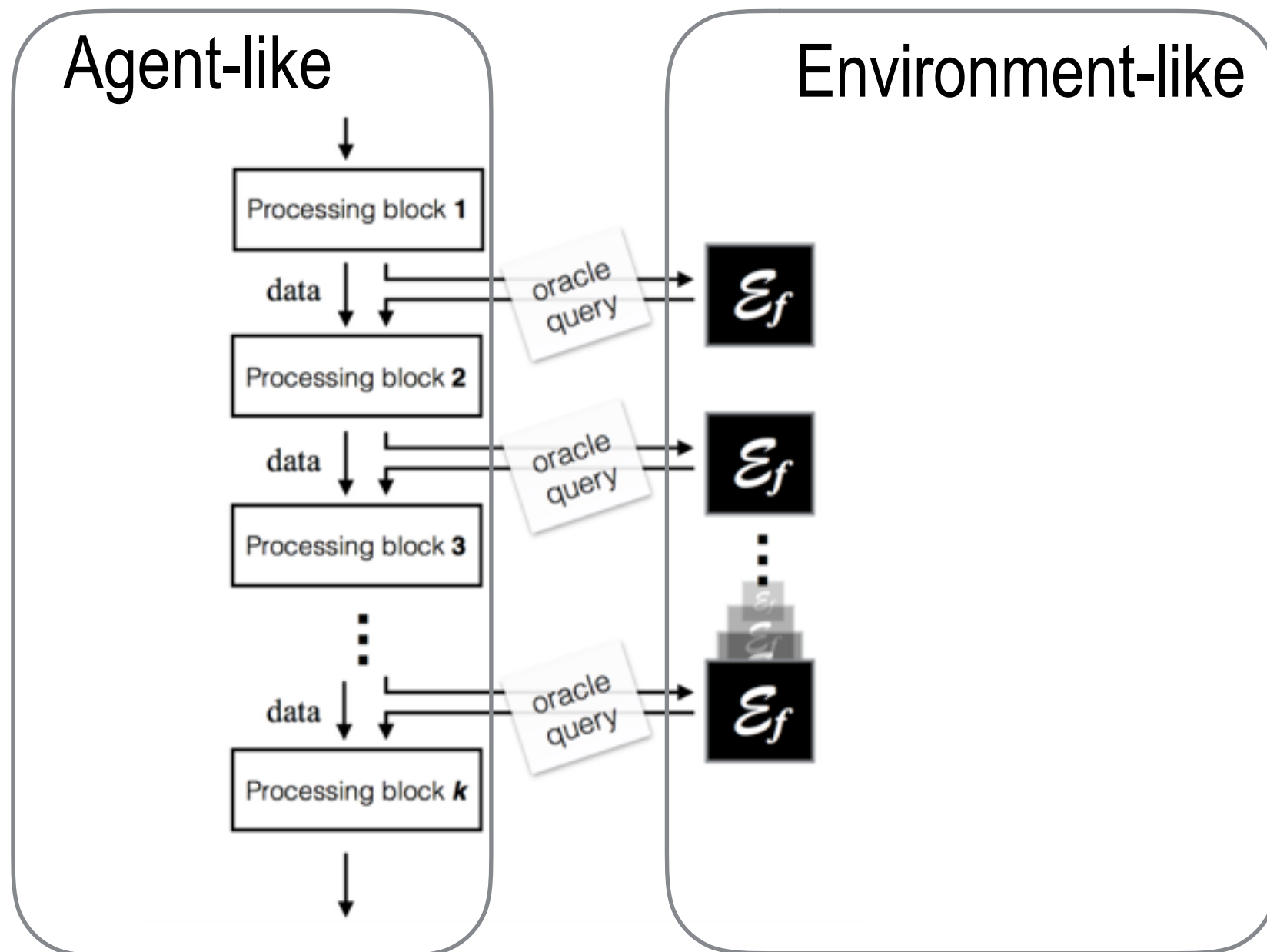


*speeding up classical interaction
is like Groverizing an old-school telephone book..*

Quantum-enhanced quantum-accessible RL

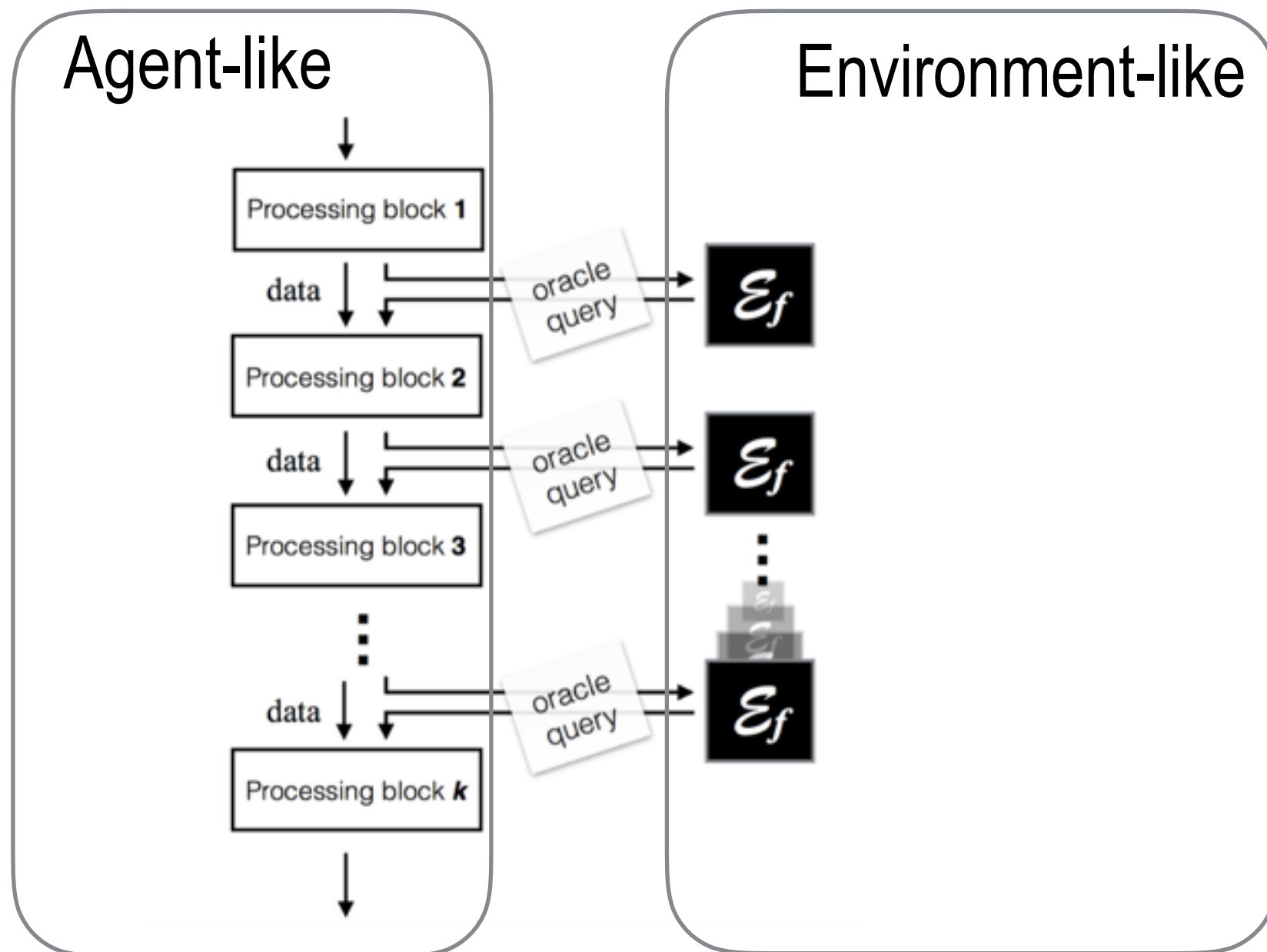


Quantum-enhanced access: *Inspiration from oracular quantum computation...*



🌐 think of Environment as Oracle

Quantum-enhanced access: *Inspiration from oracular quantum computation...*

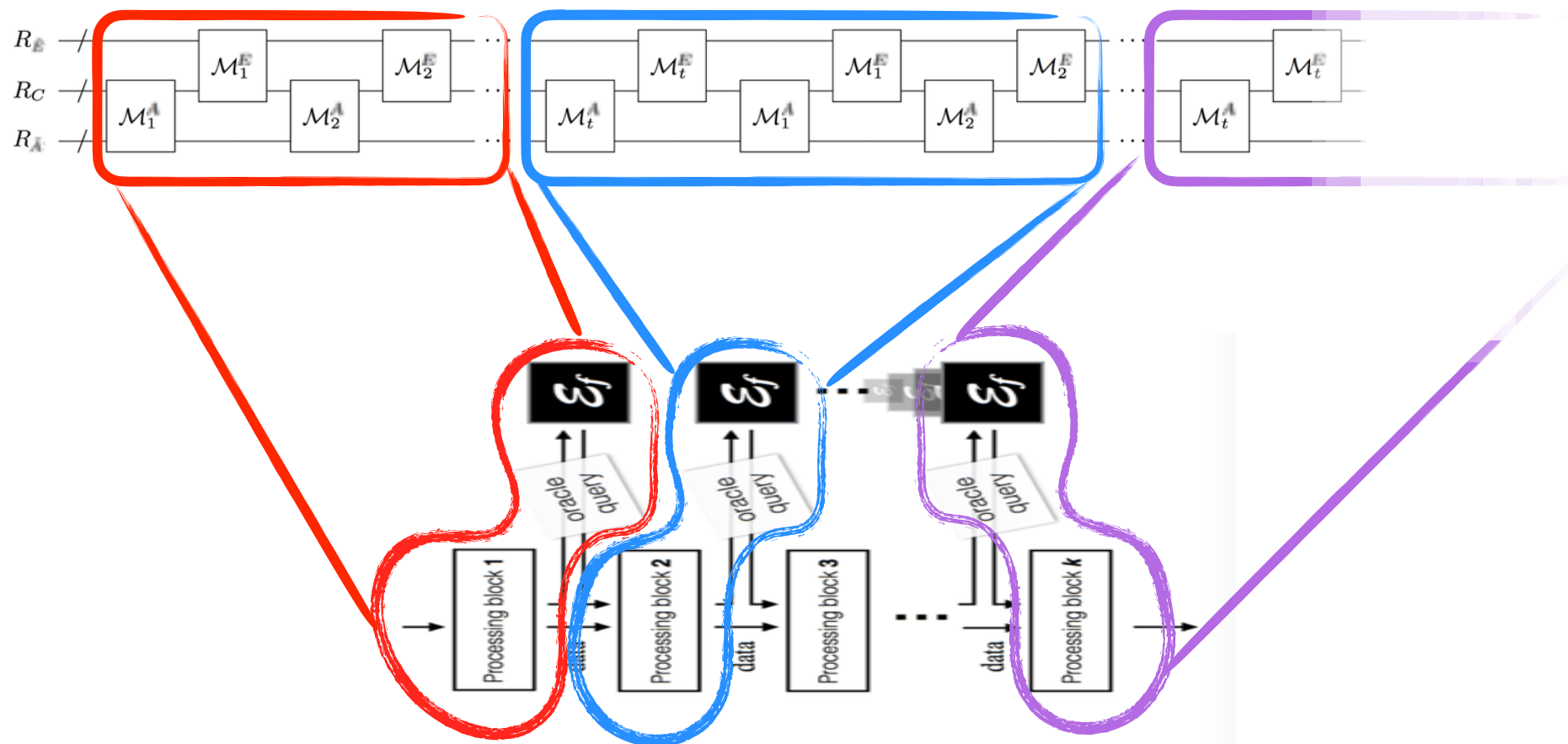


- 🌐 Use “quantum access” to oracle to learn useful information *faster*

But... environments are not like standard oracles...

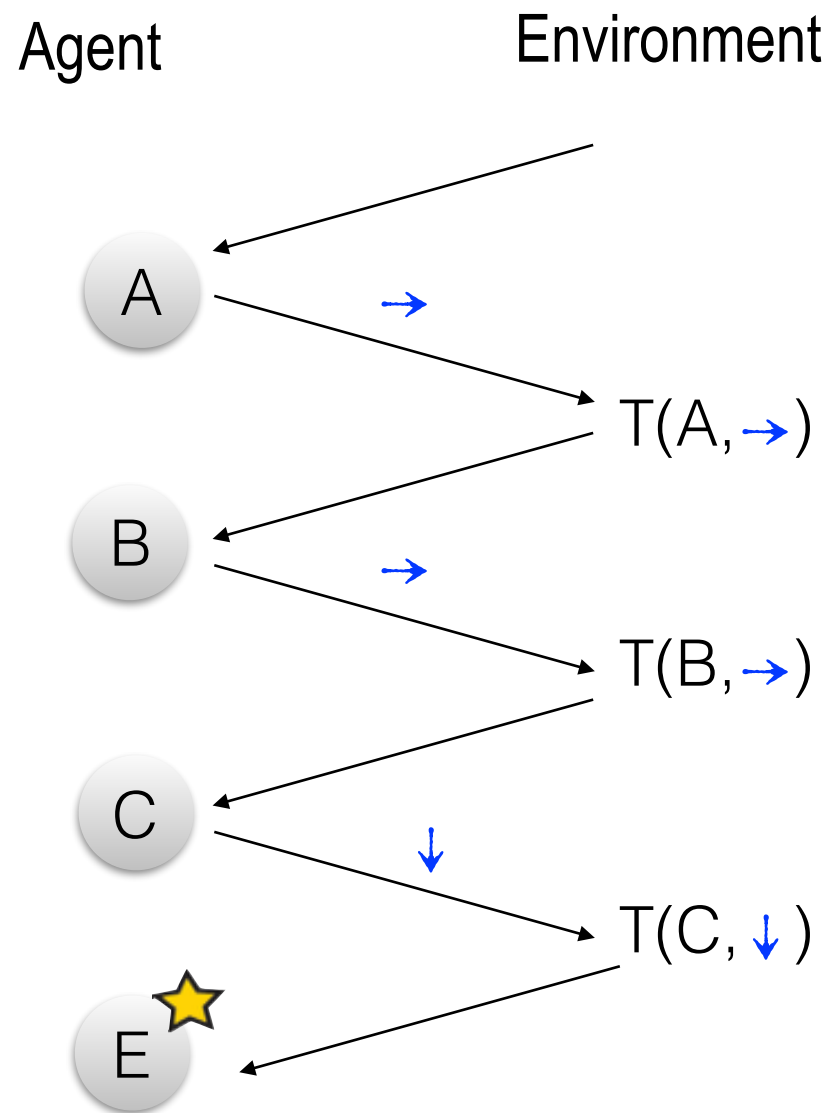
“Oraculization”
(taming the open environment)

(blocking, accessing purification and recycling)

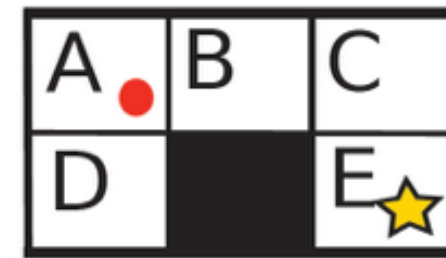


strict generalization

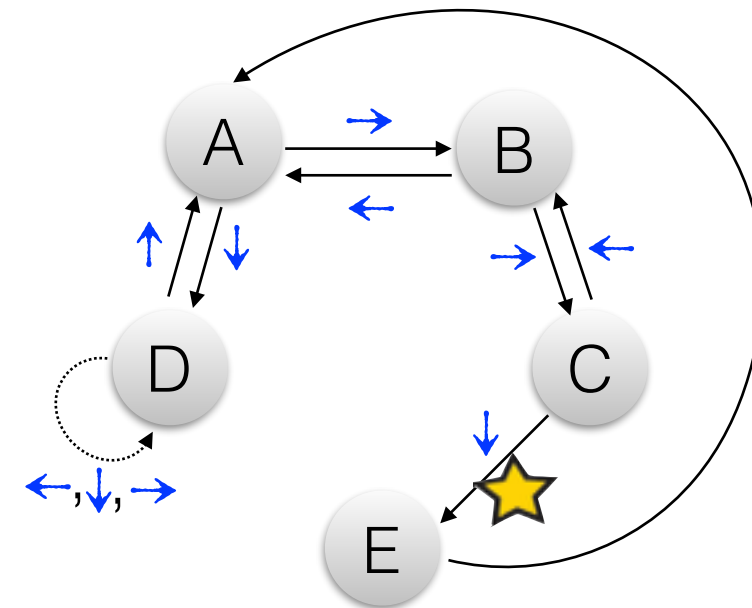
🍷 *Classical agent-environment*



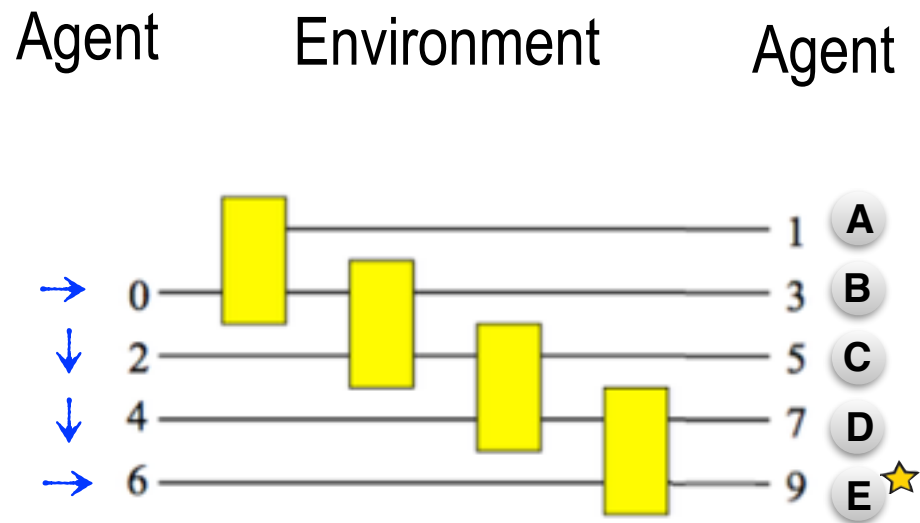
Maze:



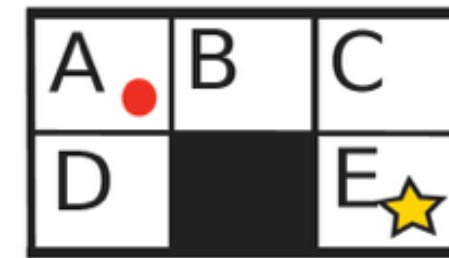
Markov Decision Process:



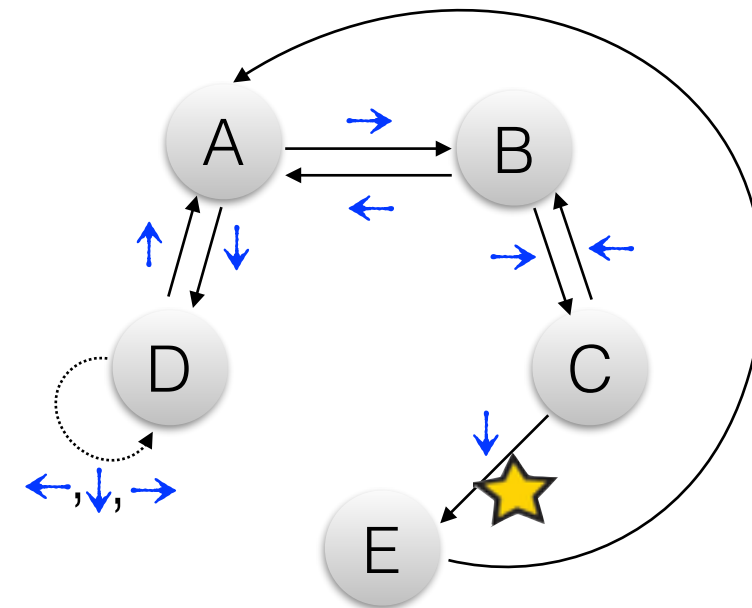
• *Classical agent-environment*



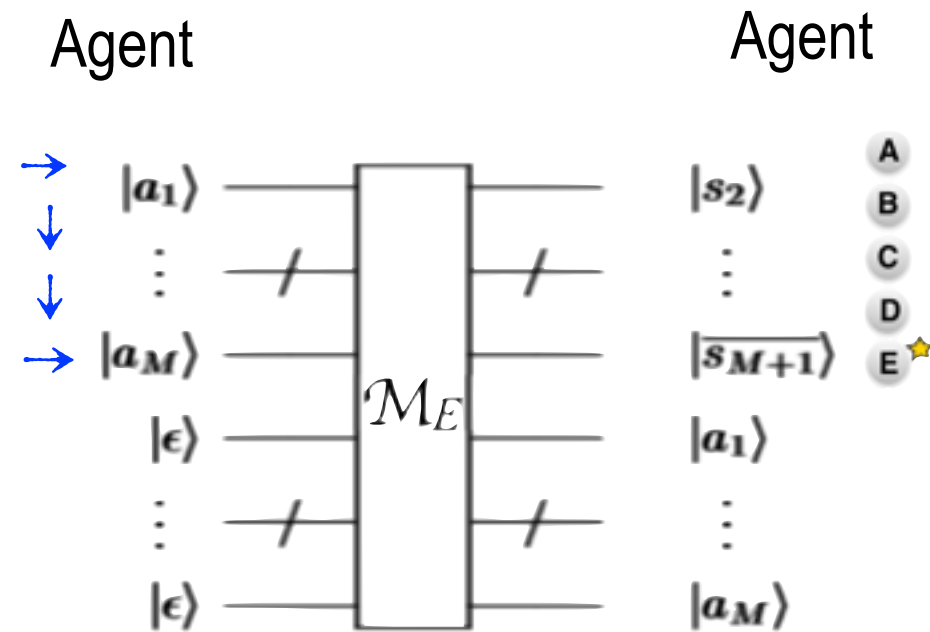
Maze:



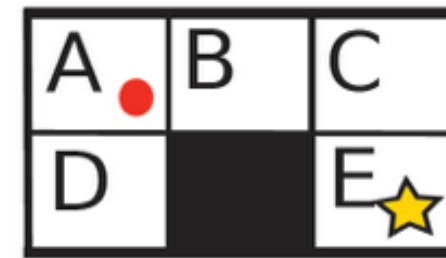
Markov Decision Process:



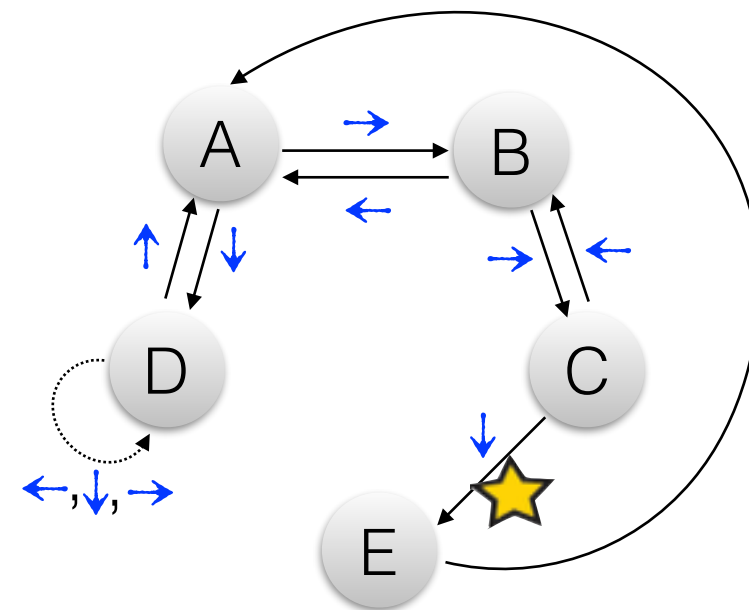
• (Semi-)classical agent-environment



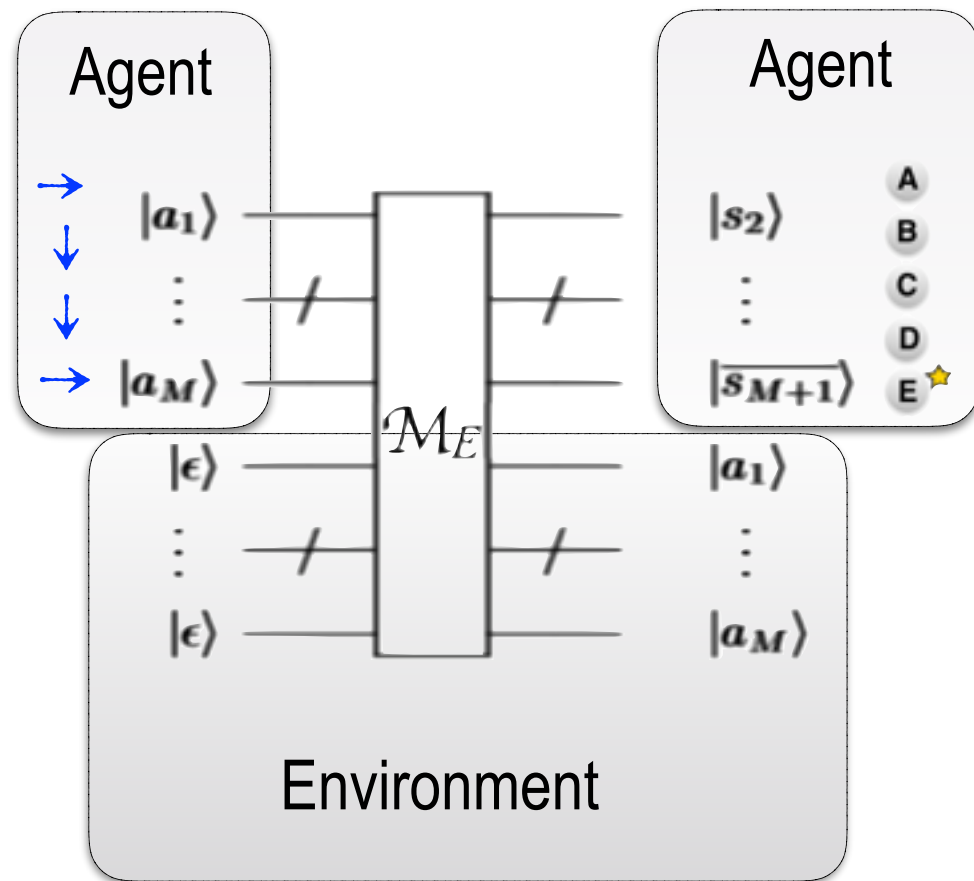
Maze:



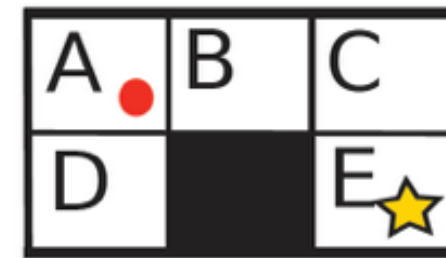
Markov Decision Process:



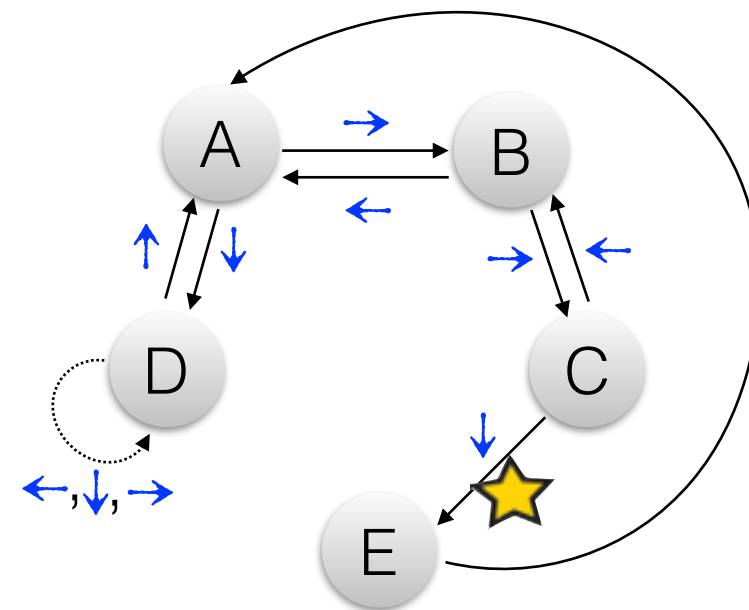
• (Semi-)classical agent-environment



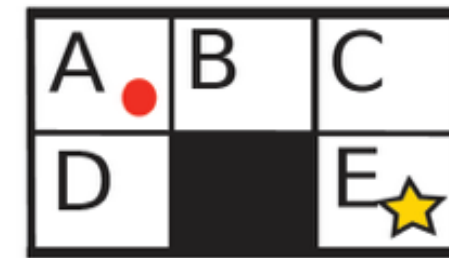
Maze:



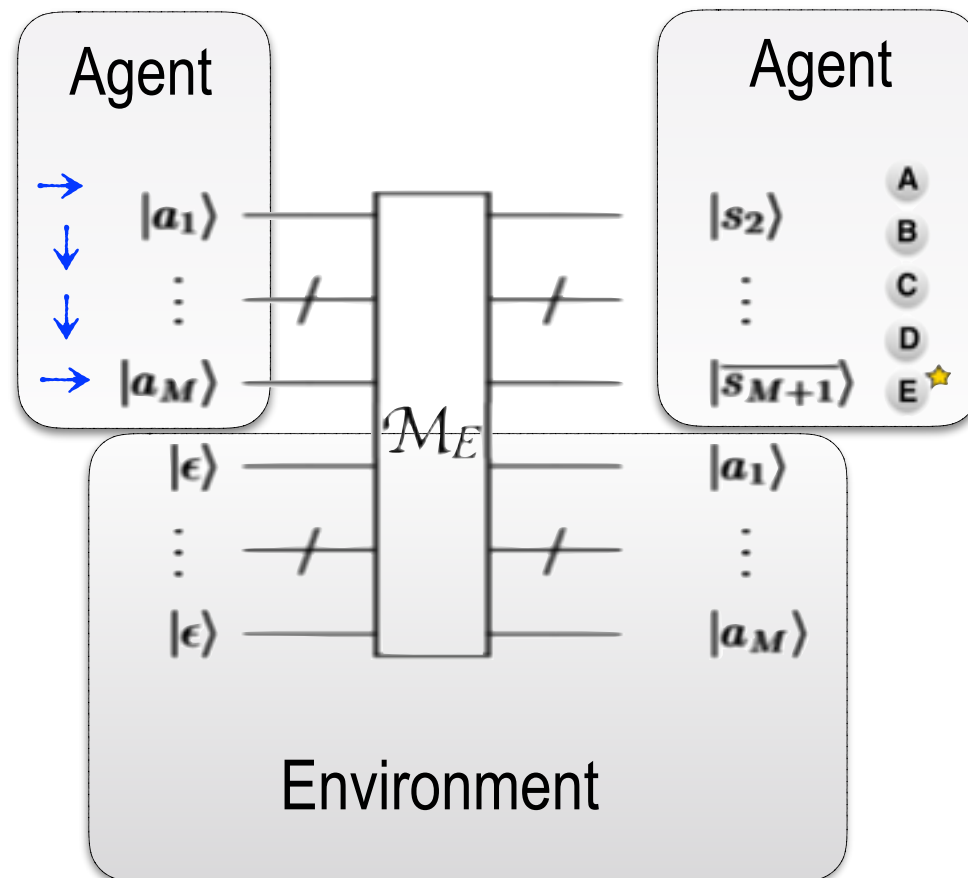
Markov Decision Process:



Maze:



• (Semi-)classical agent-environment



Have:

$$|a_1, \dots, a_M\rangle \rightarrow |s_1, \dots, \overline{s_{M+1}}\rangle_A |a_1, \dots, a_M\rangle_E$$

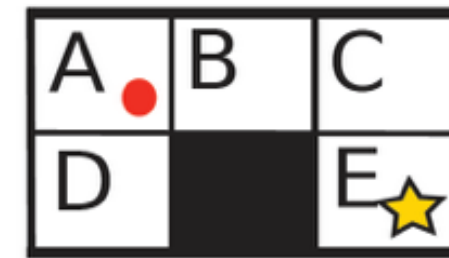
Want e.g.:

$$|a_1, \dots, a_M\rangle |0\rangle_A \rightarrow |a_1, \dots, a_M\rangle_A |?\star\rangle_A$$

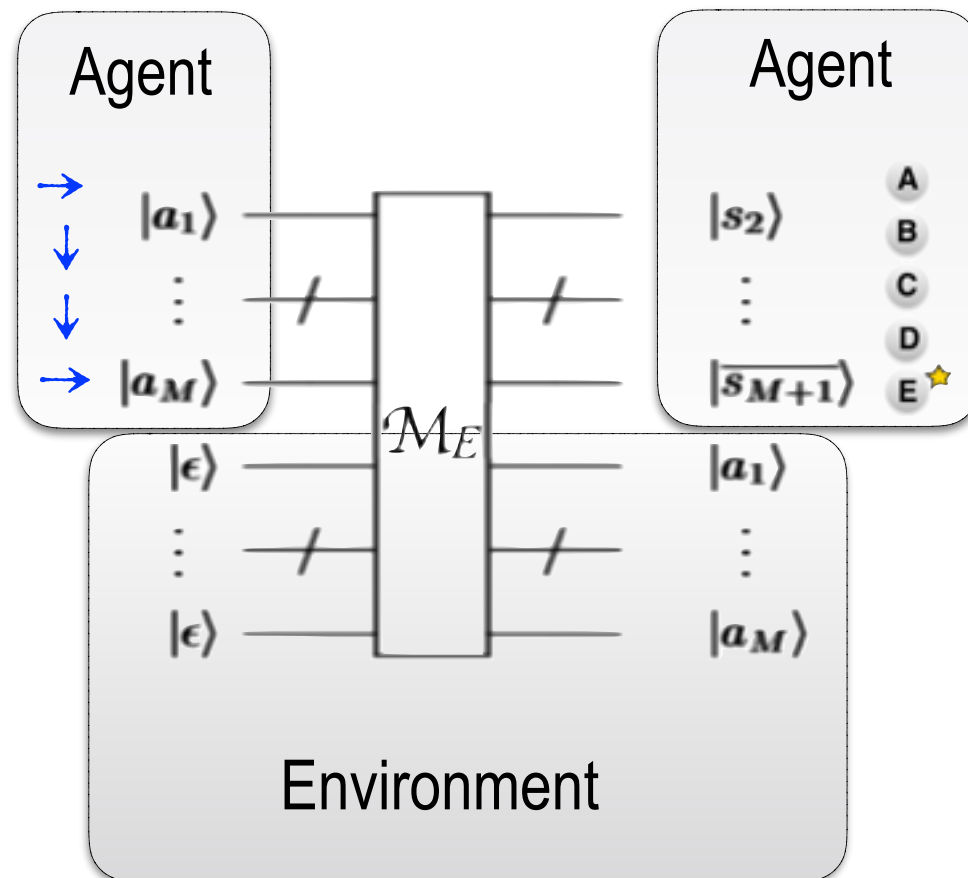
Why? Grover search for “best actions”

i.e.. convert environment to reflection about $|\rightarrow, \downarrow, \downarrow, \rightarrow\rangle$

Maze:



• (Semi-)classical agent-environment



Have:

$$|a_1, \dots, a_M\rangle \rightarrow |s_1, \dots, \overline{s_{M+1}}\rangle_A |a_1, \dots, a_M\rangle_E$$

Want e.g.:

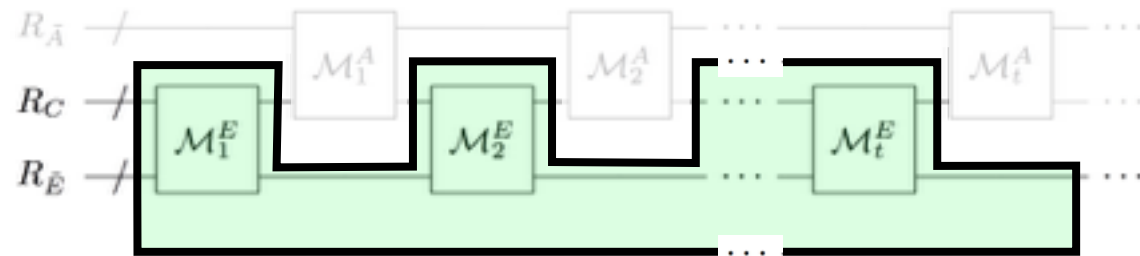
$$|a_1, \dots, a_M\rangle |0\rangle_A \rightarrow |a_1, \dots, a_M\rangle_A |?\star\rangle_A$$

How? Oraculization

Oracularization (blocking)

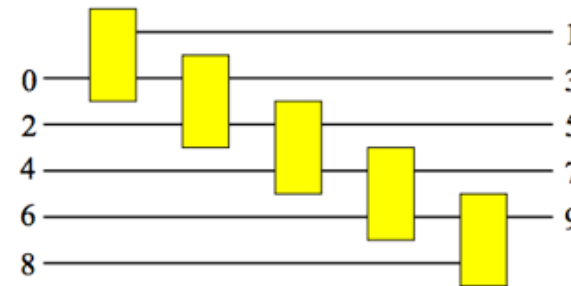
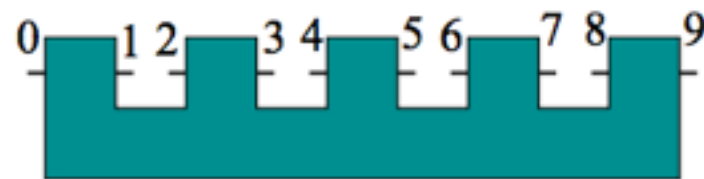
(taming the open environment)

1)



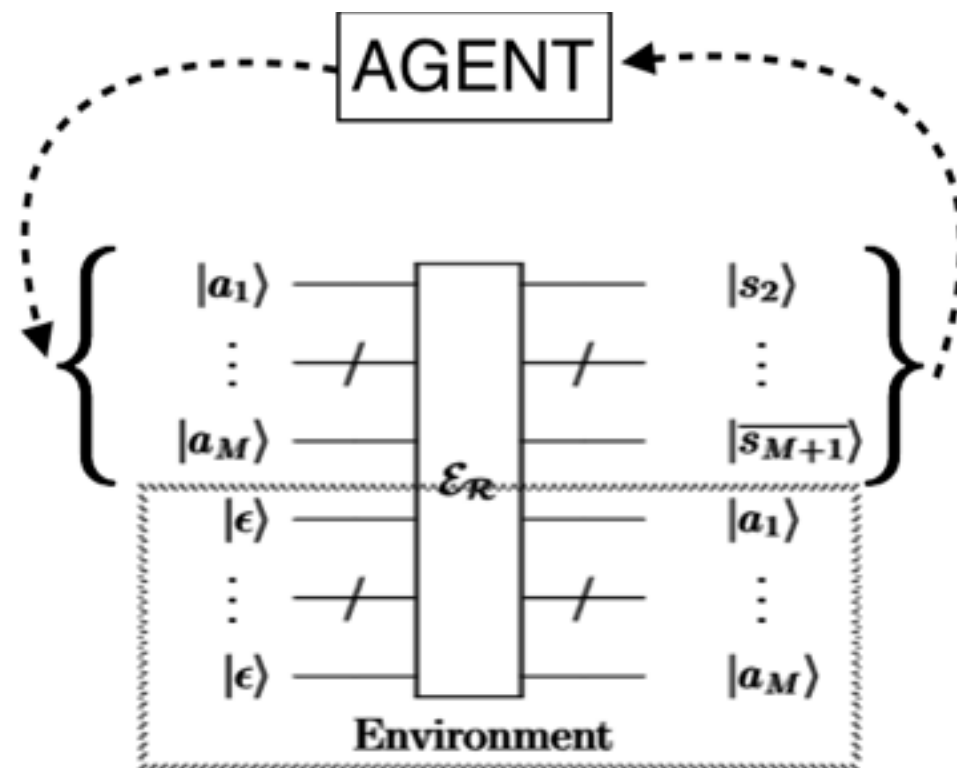
quantum comb

2)



causal network

3)

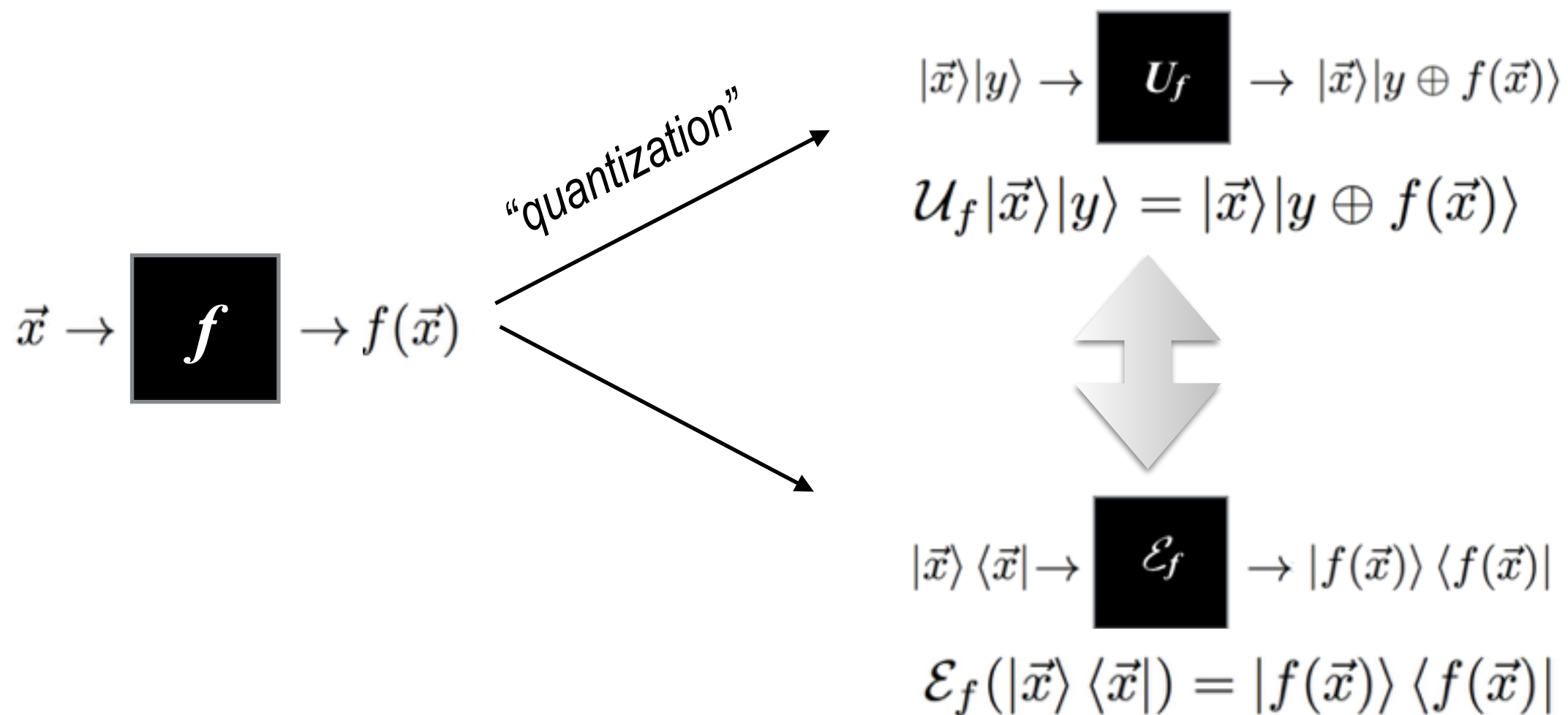


“blocking”

Oraculization (recovery and recycling)

(taming the open environment)

Classically specified oracle $\vec{x} = (x_1, x_2, \dots, x_n) \rightarrow f(\vec{x}) \in \{0, 1\}$

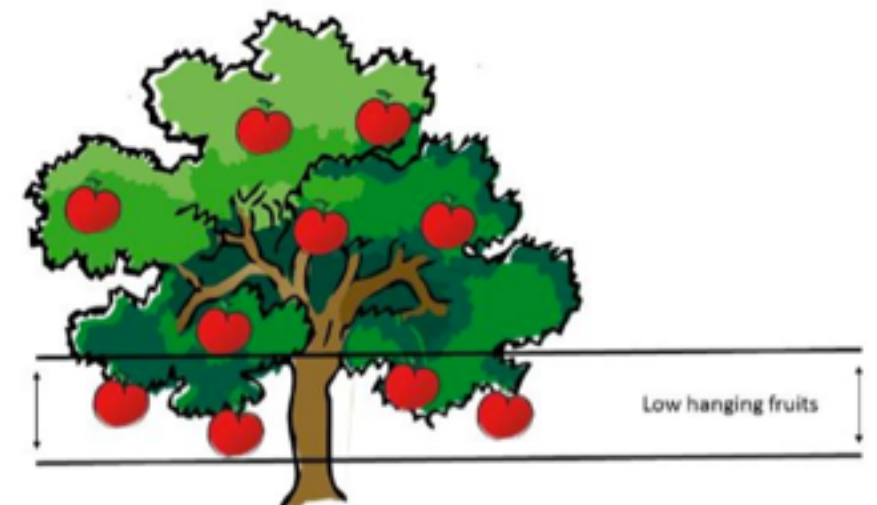
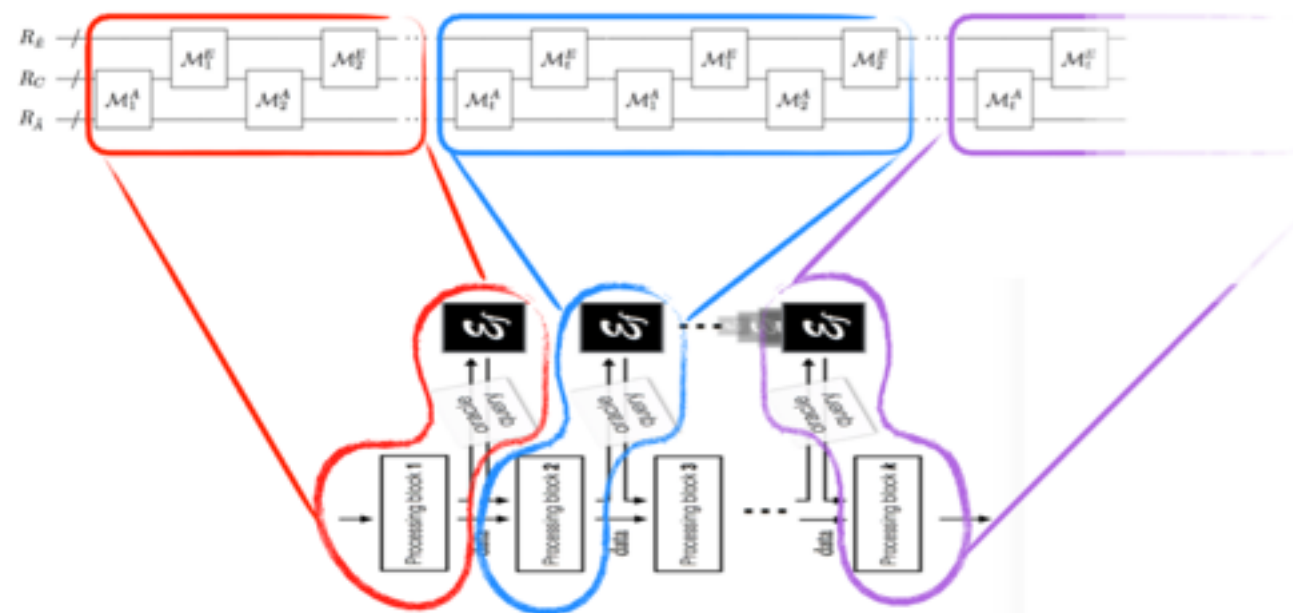
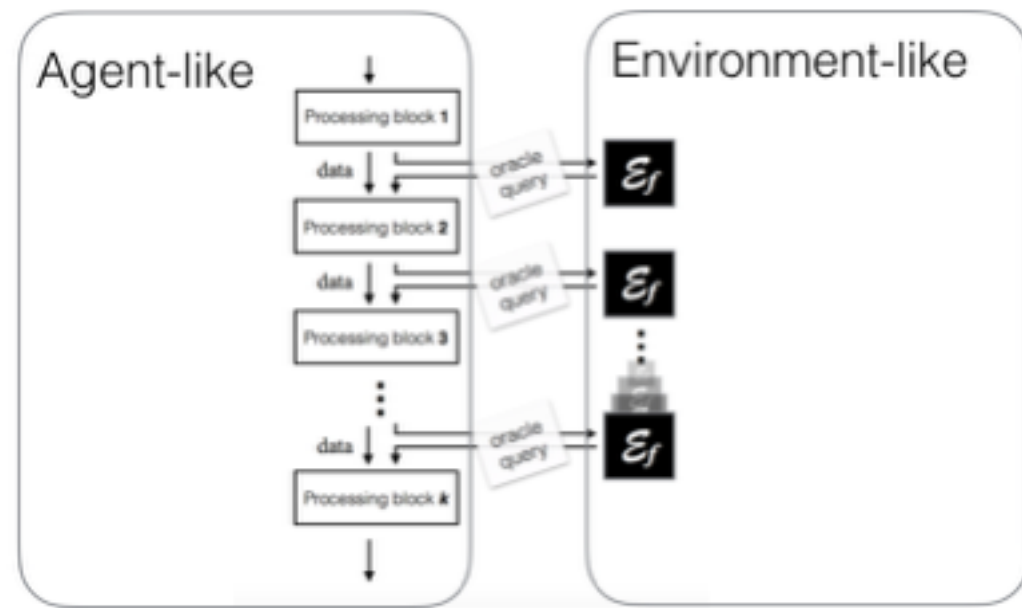


(A flavour of) *quantum-enhanced reinforcement learning*

A few results:

🌿 **Grover-like amplification for optima:**

- 🌿 *Learning speedup in luck-favoring environments*
- 🌿 *quadratic improvements in meta-learning*



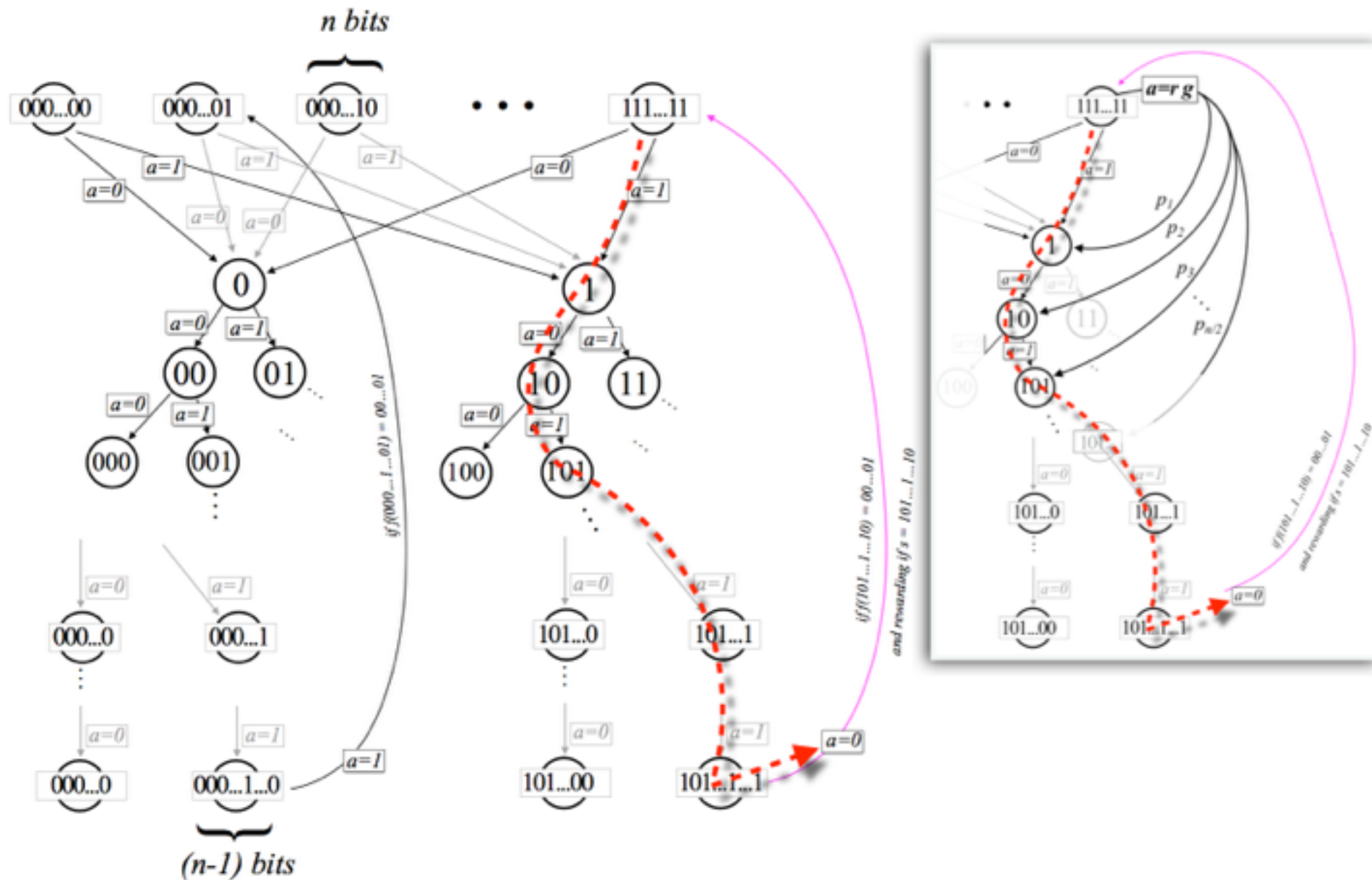
Quantum-enhanced machine learning
Vedran Dunjko, Jacob M. Taylor, Hans J. Briegel Phys. Rev. Lett 117, 130501 (2016)

Advances in quantum reinforcement learning
Vedran Dunjko, Jacob M. Taylor, Hans J. Briegel accepted to IEEE SMC 2017 (2017).

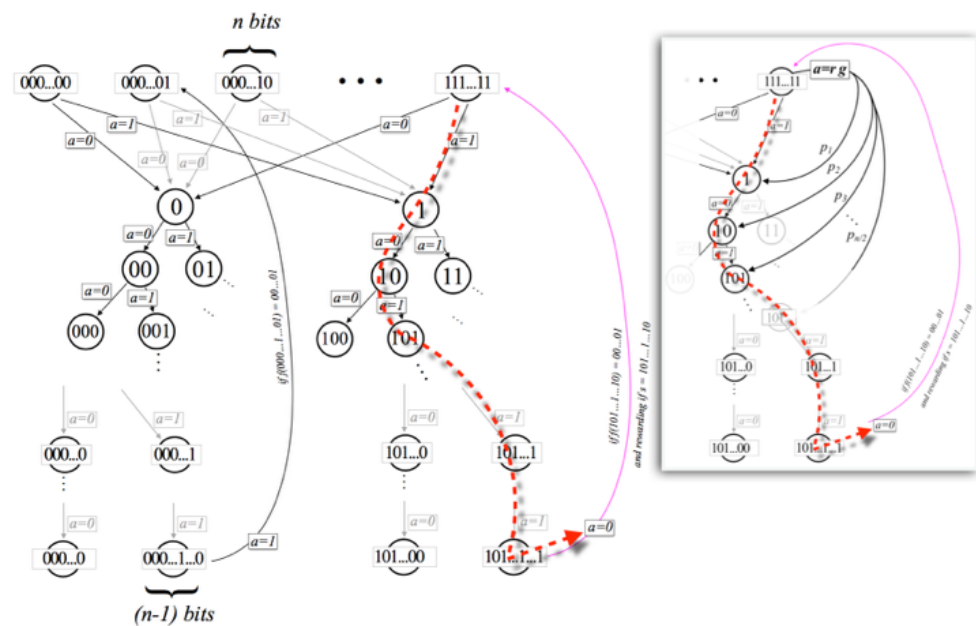
Just Grover-type speed-ups?
No... actually, most speedups are on the table...
in a booooooring way....

One step further: embedding oracles with exponential separation

Many oracular problems can be embedded into MDPs, while breaking some “degeneracies”



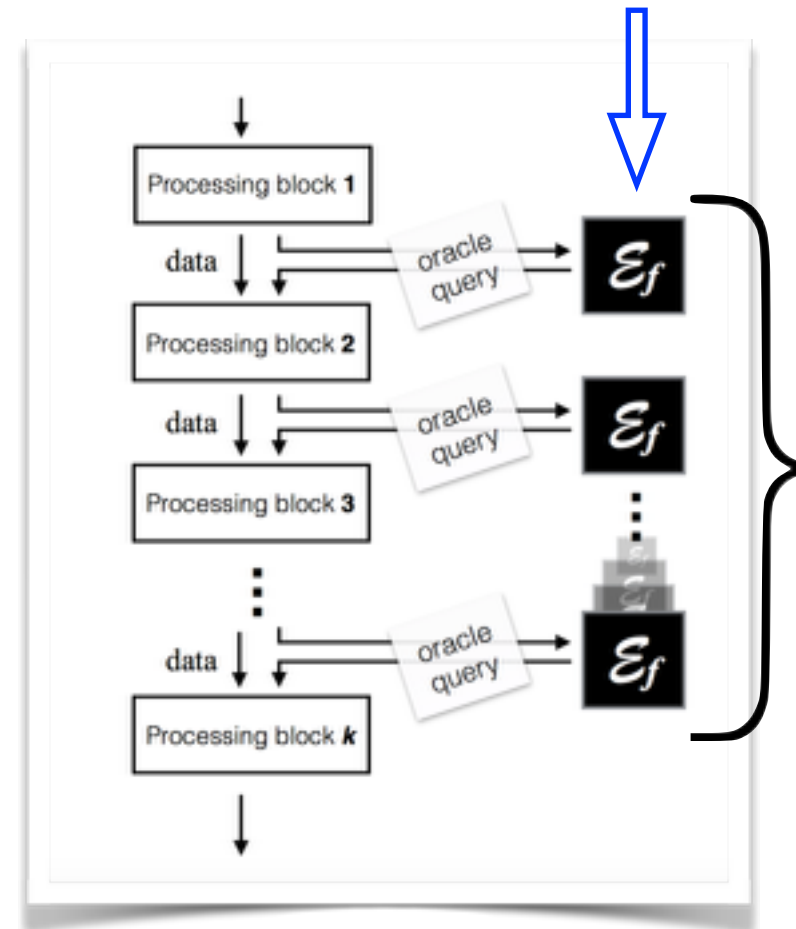
One step further: embedding oracles with exponential separation



oraculization
process



Oracle hiding a necessary “key”



Inherited
separations

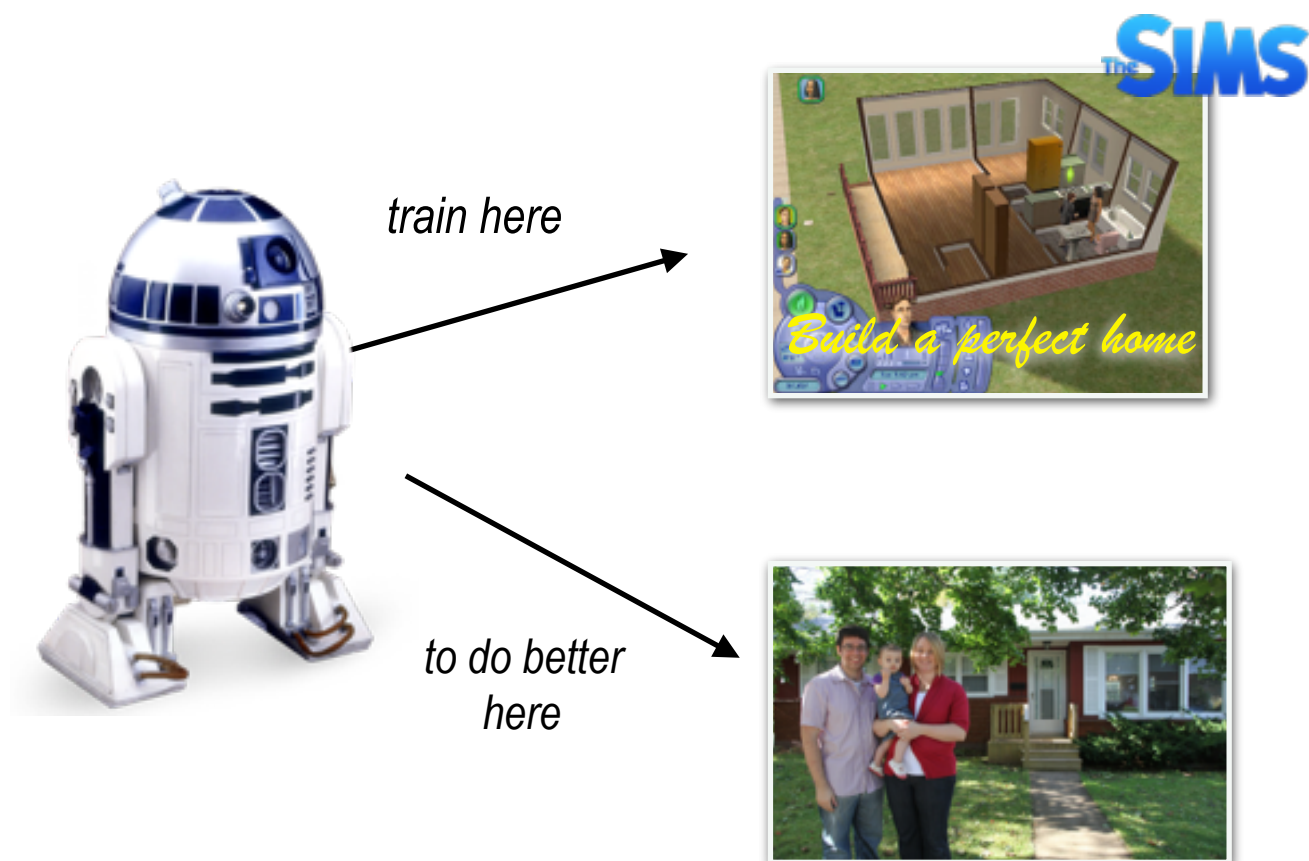
Few technical steps: make sure a) oraculization goes through; b) classical hardness is maintained.



Open problems:

- how far this can be pushed towards practically useful
- oraculization seems far fetched

Oraculization seems a stretch?
Think of it as intermediary step...



Summary:

- quantum-accessible environments can be “turned” into useful oracles
- these we can access using standard quantum tricks

Caveat: Speedups are relative to a black-box model



What if I want to **reason**
over my model

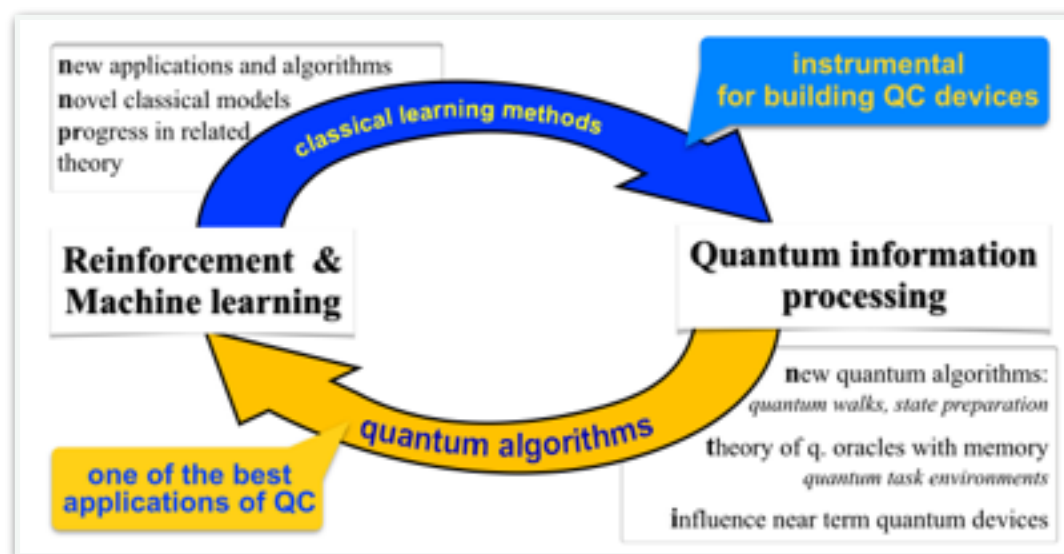
Pre-training will have at least two flavors...
1) reinforcement learning (slow, faster than real life)
2) optimization (find optimal patterns of behaviour)

Why ML/AI and QIP
make a perfect match



Why are ML/AI and QIP a perfect match

Both are *natural enhancers of other technologies*



There are **algorithmic conspiracies!**

Noise kills other algorithms...but
Noise is natural in ML!

Noise tolerance of problem

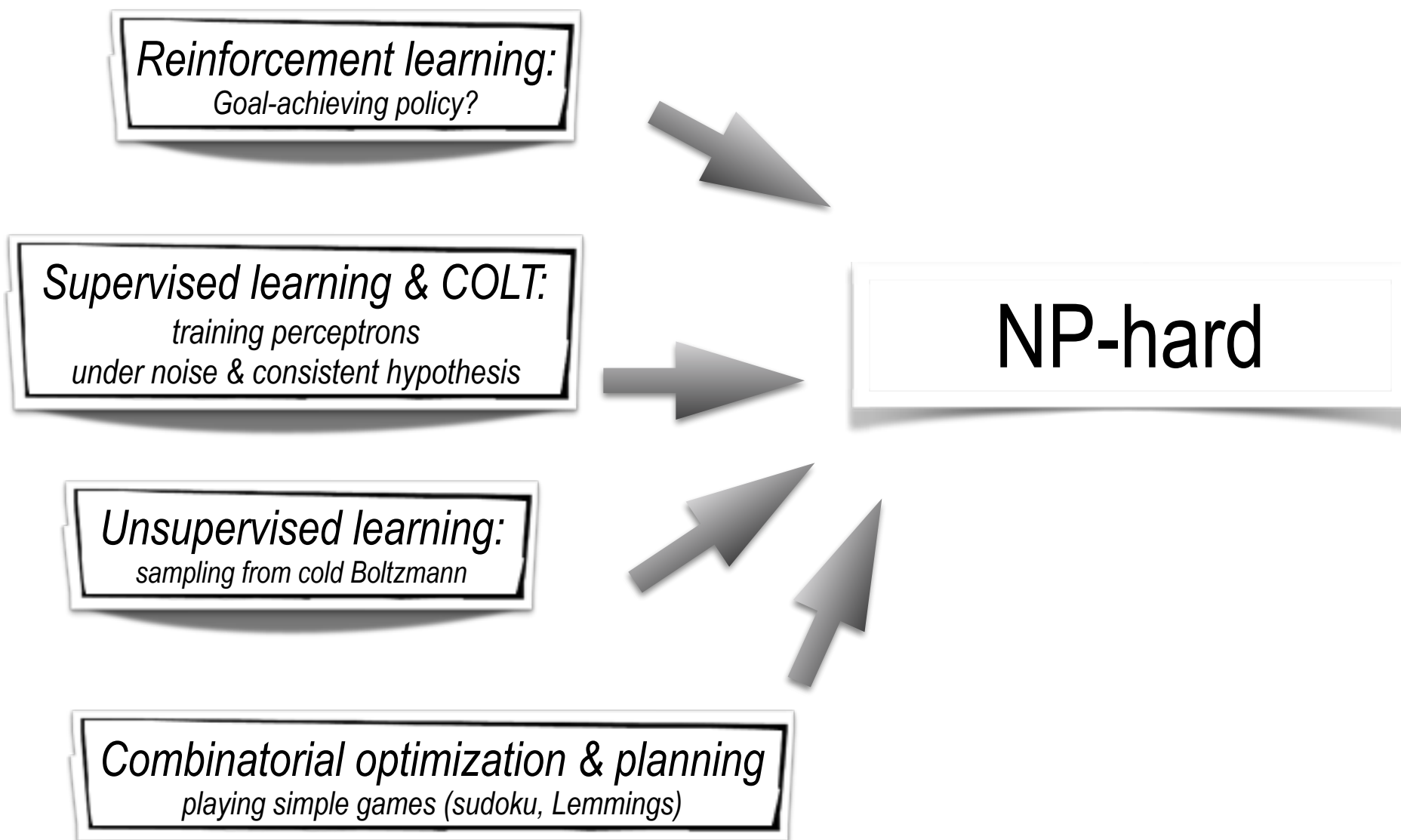
-better applicability to near term devices

-helps in database loading

Reasoning and planning is *hard*

Part 3: “... and for some aspects of planning on small QCs”

or: *Hard computational problems, AI,
and restricted quantum computers*



Many problems are *harder*: “do I win chess”, finding good policies in (PO)MDP are PSPACE, many games are EXPTIME, and verification of processes is *undecidable*...

Can quantum computers help here?

- fundamental, but...
- not believed to be in BQP - not elucidating power of quantum computing, less explored
- exponential run-times... in practice *heuristics*
- results studied continuously (Montanaro, Ambainis, Aaronson, etc...)
- a class of heuristics: annealers

QeML (quantum-enhanced learning)

- exponential separations...
- particularly well-matched class of applications, also for *near term*!
- plays well with noise, plays well with shallow computations...

NP-problems (quantum-enhanced *reasoning*)

- only poly-speed ups
- a-priori, unlikely to be well-suited for (near-term) quantum computing

Can quantum computers help here?

- fundamental, but...
- not believed to be in BQP - not elucidating power of quantum computing, less explored
- exponential run-times... in practice *heuristics*
- results studied continuously (Montanaro, Ambainis, Aaronson, etc...)
- a class of heuristics: annealers

QeML (quantum-enhanced learning)

- exponential separations...
- particularly well-matched class of applications, also for *near term*!
- plays well with noise, plays well with shallow computations...

NP-problems (quantum-enhanced *reasoning*)

- only poly-speed ups
- a-priori, unlikely to be well-suited for (near-term) quantum computing

remainder of talk is in here

A general question: *suppose you have a problem of size n ,
and quantum computer handling $m \ll n$ qubits.
What can you do?*

Could be... nothing!

Good algorithms exploit problem structure. Break it by “chunking”,
you loose (a lot of) speed. *Thresholds!*

An example: thresholds when quantum-enhancing a SAT solving algorithm.

3SAT

$$f : \{0, 1\}^n \rightarrow \{0, 1\}$$

$$f(x_1, \dots, x_n) = (x_1 \vee x_{10} \vee \bar{x}_{51}) \wedge (\bar{x}_3 \vee \bar{x}_{10} \vee \bar{x}_{11}) \wedge (\bar{x}_{11} \vee \bar{x}_{44} \vee \bar{x}_{51}) \cdots$$

↑
“or”

↑
“and”

⏟

clause or **constraint**

all constraints have to be satisfied

SAT problem: *Is there a choice (assignment) of the variables,
such that f evaluates to 1 (“true”)*

3SAT

$$f : \{0, 1\}^n \rightarrow \{0, 1\}$$

$$f(x_1, \dots, x_n) = (x_1 \vee x_{10} \vee \bar{x}_{51}) \wedge (\bar{x}_3 \vee \bar{x}_{10} \vee \bar{x}_{11}) \wedge (\bar{x}_{11} \vee \bar{x}_{44} \vee \bar{x}_{51}) \cdots$$

Schöning:

1. Pick assignment x_1, \dots, x_n randomly.
 2. Check if satisfying; output if is, and terminate
 3. Find first unsatisfied clause,
flip any variable of the clause in the assignment
- Do $3n$ times

A random, gently directed, walk in the space of assignments...

3SAT

Schöning (1999): if sat. exists, the walk finds it with probability $(3/4)^n$

Monte Carlo: $(4/3)^n = 2^{\gamma n}$, $\gamma = \log_2(4/3) \approx 0.415\dots$

3SAT

Schöning (1999): if sat. exists, the walk finds it with probability $(3/4)^n$

Monte Carlo: $(4/3)^n = 2^{\gamma n}$, $\gamma = \log_2(4/3) \approx 0.415\dots$

Quantum Schöning / any such sampling algorithm?

Instead of sampling, amplitude amplification (Grover):

Run-time: $O^*(2^{\gamma n}) \rightarrow O^*(2^{\frac{\gamma}{2} n}) = O^*(2^{\gamma_q n})$

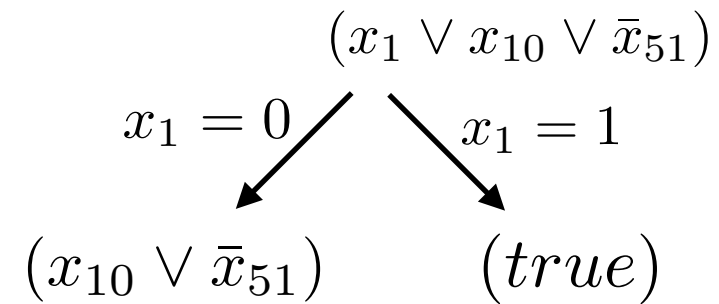
How many qubits needed? *Cca. $3n$ qubits just for purified randomness + evaluation*

What if I have only enough qubits for an m -sized formula?

What if I have only enough qubits for an m -sized formula?

Setting some variables shrinks the formula:

$\underbrace{x_1, x_2, x_3, x_4}_{\text{set}}, \underbrace{x_5, x_6, x_7, x_8 \dots}_{\text{free}}$



What could I do if I have only enough qubits for an m -sized formula?

Guess some variables:

$\underbrace{x_1, x_2, x_3, x_4}_{\text{set}}, \underbrace{x_5, x_6, x_7, x_8 \dots}_{\text{free}}$

- 1) Fix $x_V = x_{\sigma(1)}, \dots, x_{\sigma(n-m)}$
 - 2) $F(\vec{x}) \rightarrow \underbrace{F^{x_v}(\vec{x}|_{V^c})}_{\text{formula of size } m} \rightarrow \text{solve on QC!}$
- \leftarrow must do 2^{n-m} times

How fast is this?

$$\alpha = m/n$$

$$\underbrace{O^*(2^{((1-\alpha) \cdot 1 + \alpha \cdot \gamma_q)n})}_{\text{quantum}}$$

What could I do if I have only enough qubits for an m -sized formula?

Guess some variables:

$\underbrace{x_1, x_2, x_3, x_4}_{\text{set}}, \underbrace{x_5, x_6, x_7, x_8 \dots}_{\text{free}}$

- 1) Fix $x_V = x_{\sigma(1)}, \dots, x_{\sigma(n-m)}$
 - 2) $F(\vec{x}) \rightarrow \underbrace{F^{x_v}(\vec{x}|_{V^c})}_{\text{formula of size } m} \rightarrow \text{solve on QC!}$
- must do 2^{n-m} times*

How fast is this?

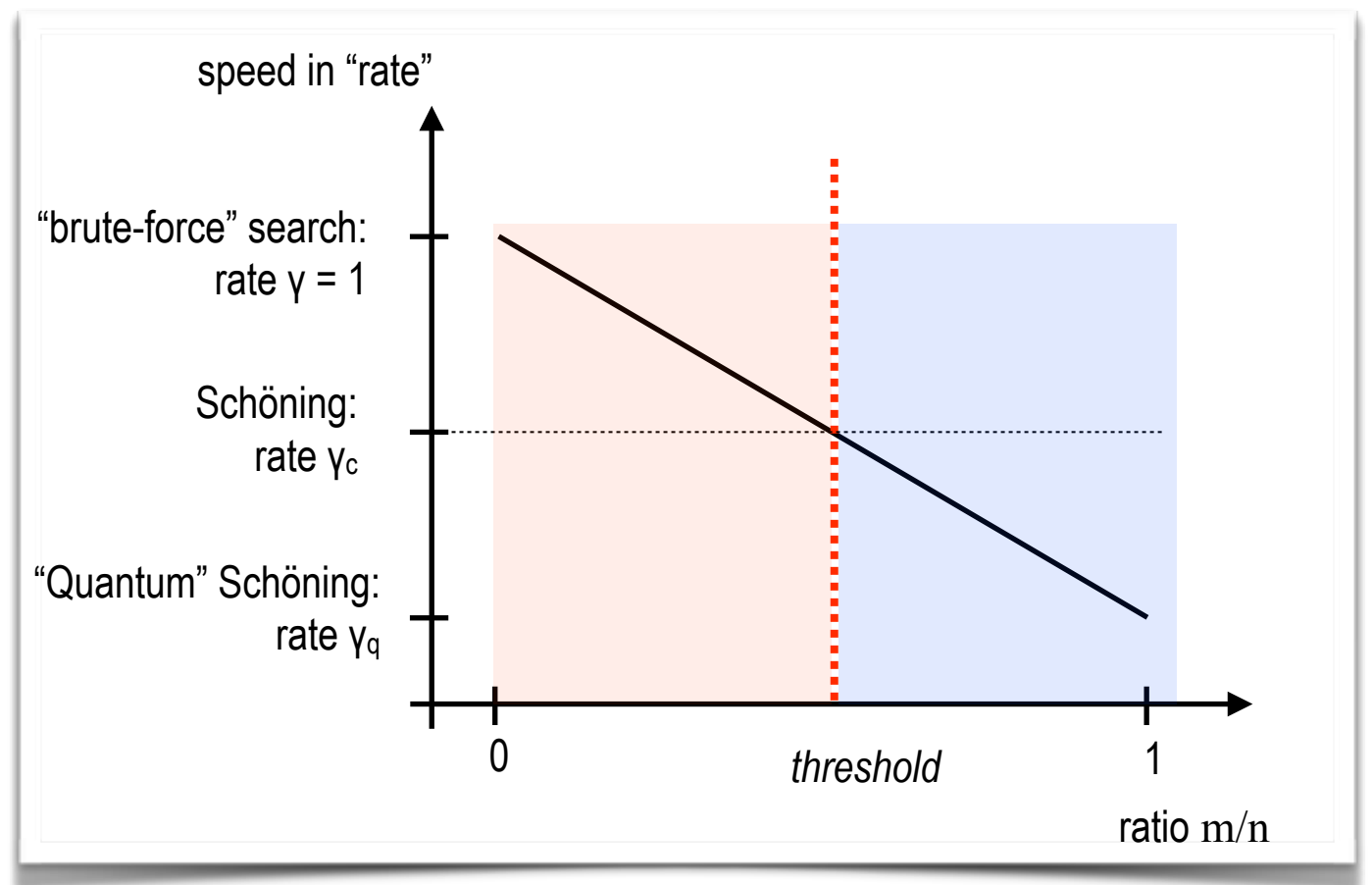
$$\alpha = m/n$$

$$\underbrace{O^*(2^{((1-\alpha) \cdot 1 + \alpha \cdot \gamma_q)n})}_{\text{quantum}} \quad \text{v.s.} \quad \underbrace{O^*(2^{\gamma n})}_{\text{classical}}$$

Naïve solution - did we win?

$$O^*(2^{((1-\alpha) \cdot 1 + \alpha \cdot \gamma_q)n}) \quad \Leftrightarrow \quad O^*(2^{\gamma n})$$

$$\alpha \Leftrightarrow \frac{1-\gamma}{1-\frac{\gamma}{2}} \approx 0.73 \quad m > 0.73n$$



threshold effect

other thresholds: speedup kicks in too late, e.g.

$$10^{15} \times n \in O(n) \text{ v.s. } n^2 \in O(n^2)$$

*Why? Problems have structure (except unstructured search)
How do you chop it up into chunks?*

Can be avoided for some for certain classes of problems

- if the algorithm does not use (too much) randomness*
- If the algorithm recursively calls itself or other sub-routines (like in **dynamical programming**)*
- If the subroutines do not depend on the original problem size*

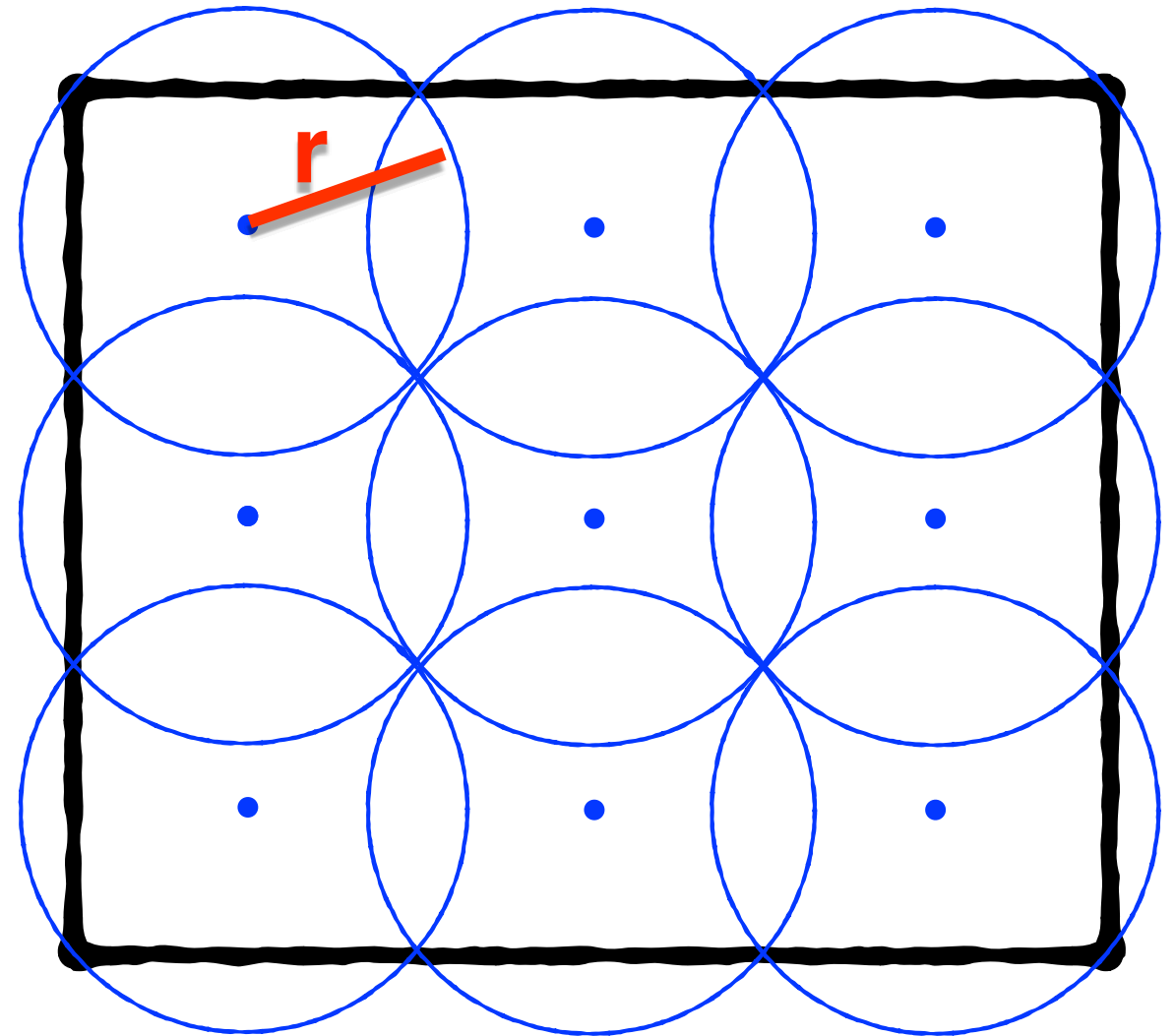
then we can use a “*hybrid approach*”:
use classical calls, until instance small enough!

SAT solving a-la Schöning

1) derandomized Schöning

- partition assignment space into r -balls
- solve PromiseBallSat for each

PromiseBallSat(\vec{x}, r)



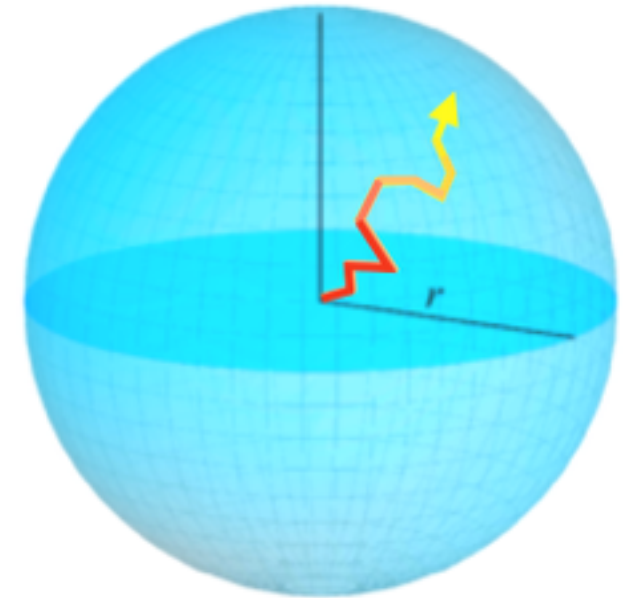
NB: r will be a fraction of n

SAT solving a-la Schönning...

- 1) derandomized Schönning...
- 2) ...reduces to PromiseBallSAT

PromiseBallSat(\vec{x}, r)

1. Start from \mathbf{x}
2. Find first unsatisfied clause (or done!)
3. Recurse algorithm on flipping each of the three possibilities, calling induced smaller formula



Non-recursive version

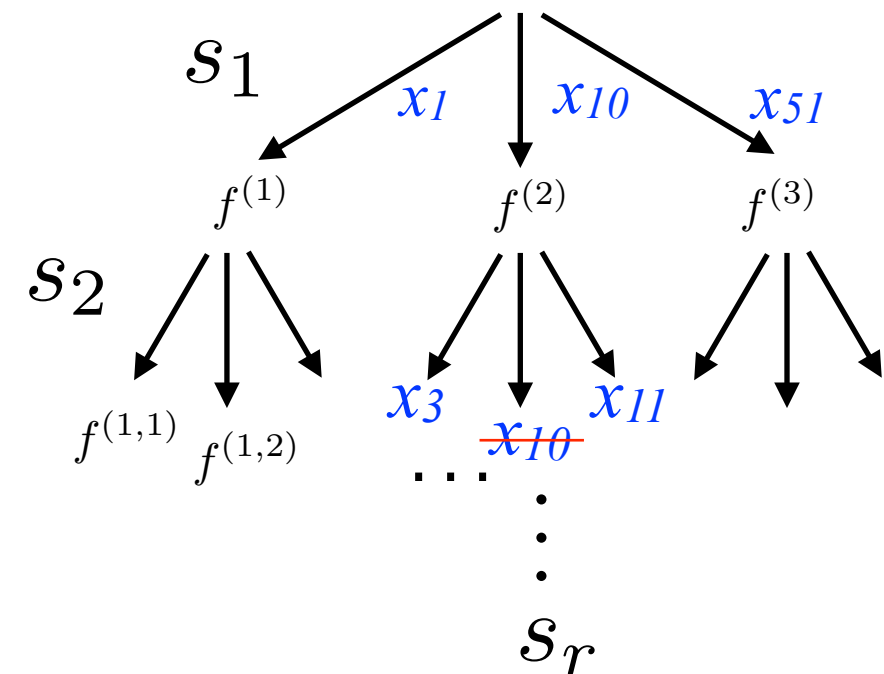
select s_1, s_2, \dots, s_r

Check every substring

Only flip ones not flipped previously

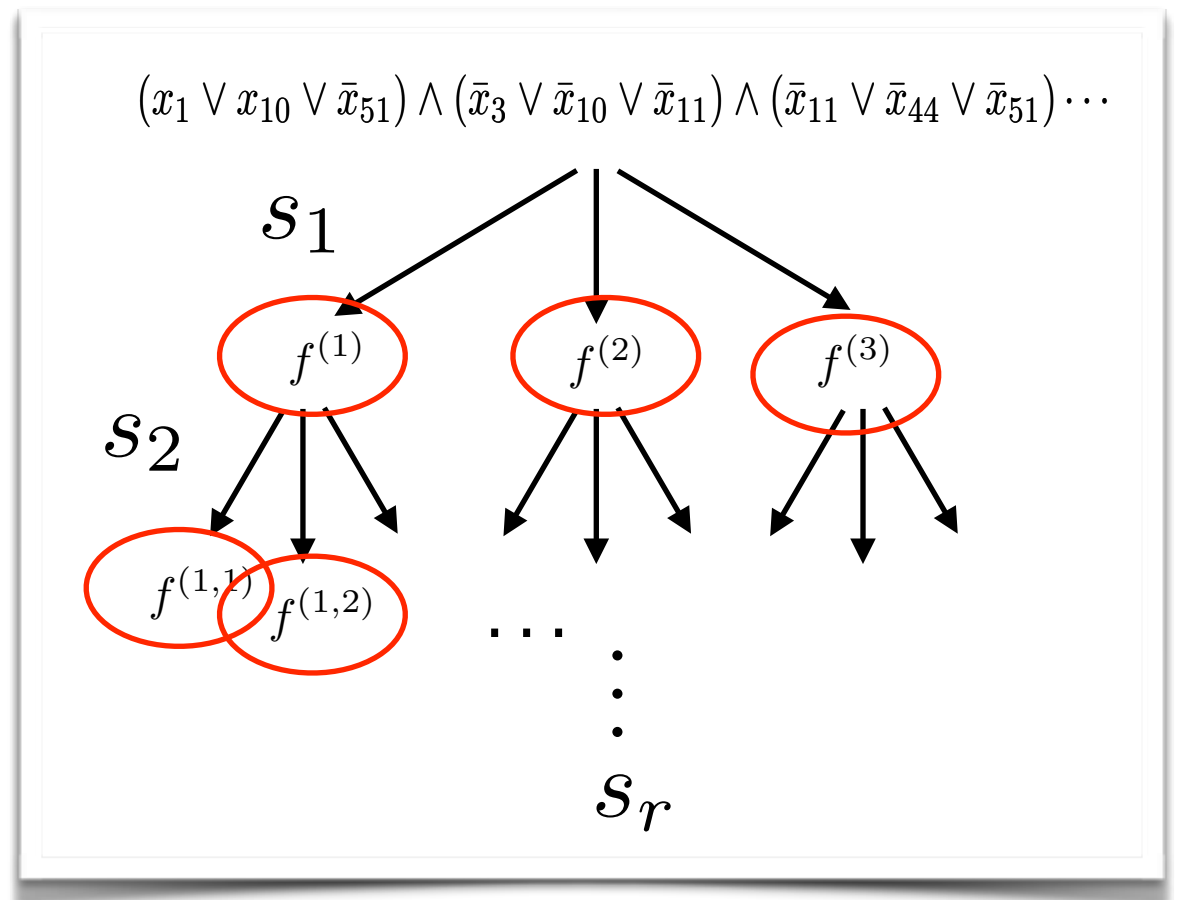
$$O(3^r)$$

$$(x_1 \vee x_{10} \vee \bar{x}_{51}) \wedge (\bar{x}_3 \vee \bar{x}_{10} \vee \bar{x}_{11}) \wedge (\bar{x}_{11} \vee \bar{x}_{44} \vee \bar{x}_{51}) \dots$$



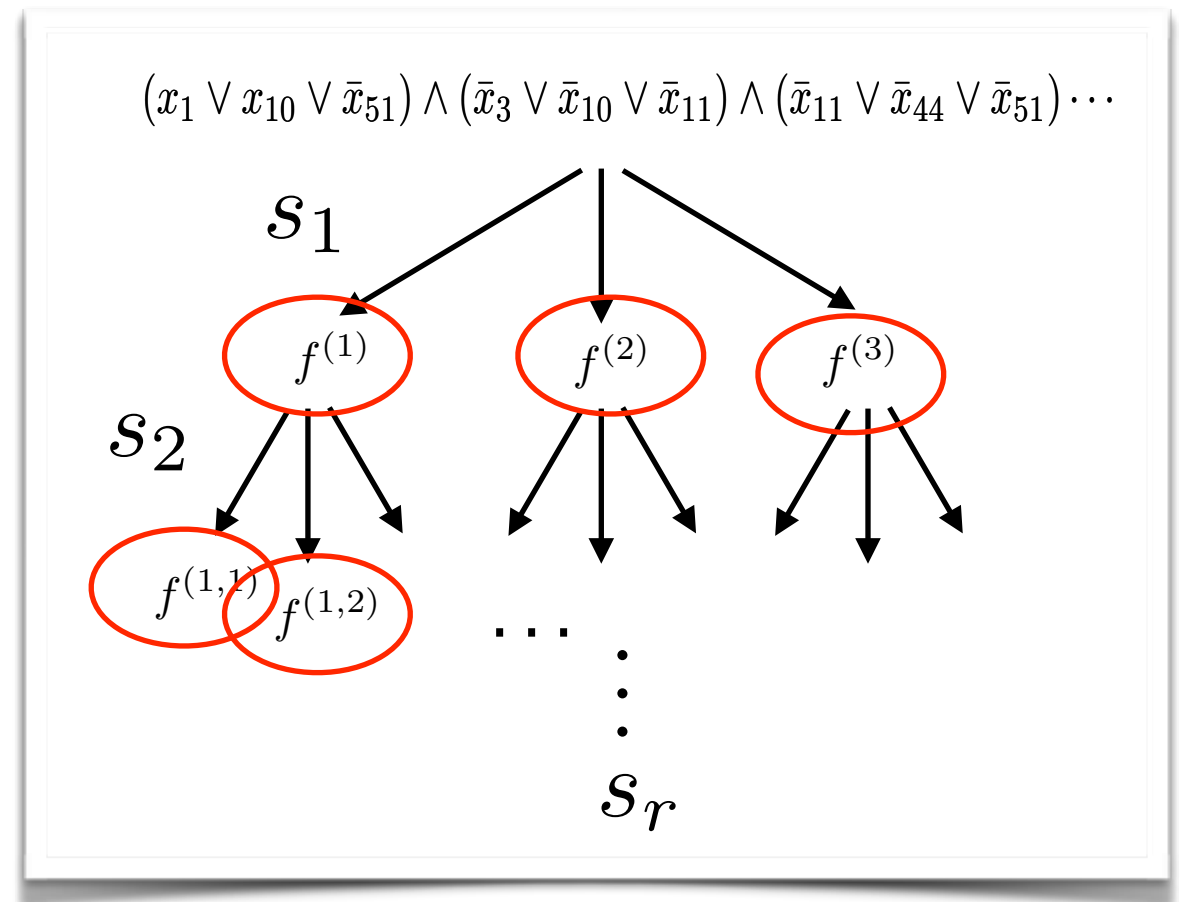
SAT solving a-la Schöning...

- 1) derandomized Schöning...
- 2) ...reduces to PromiseBallSAT...
- 3) ...which recurses itself on **smaller instance**...



SAT solving a-la Schöning...

- 1) derandomized Schöning(n)...
- 2) ...reduces to PromiseBallSAT(r)...
- 3) ...which recurses itself on **smaller r** ...



the “*hybrid approach*” for PromiseBallSAT:

- 1) find a quantum implementation (QPBS) which is fast, and uses few qubits (ideally r)
- 2) Run recursive algorithm, call QPBS once **r is small enough**

**How fast the end result is depends on
how big a r we can handle given QC of size m**

Critical: #needed qubits must not depend on initial size

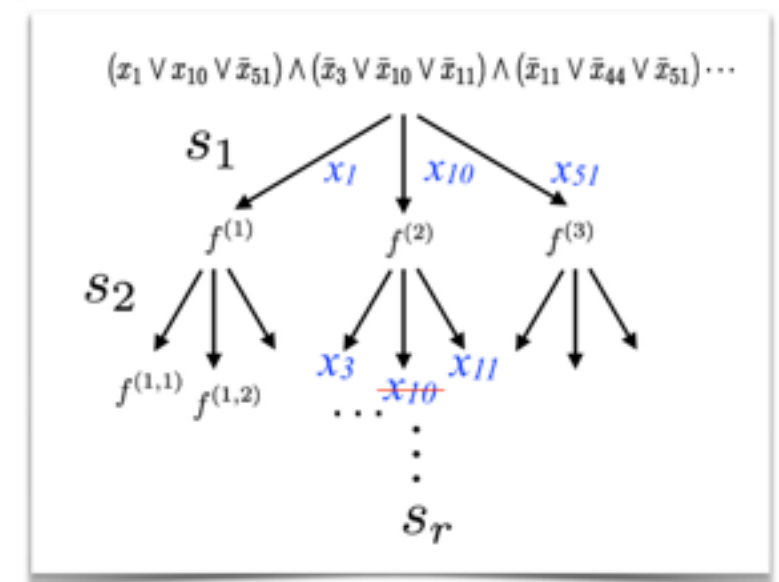
$$\text{PromiseBallSat}(\vec{x}, r) \rightarrow \text{PromiseBallSat}_{\vec{x}}(r)$$

Key observation: only carry **r trits**. **Could be independent from n.**

Only need to keep track of which bits to flip.

Only need 3 ancillas to check each clause sequentially

$$|s_1, \dots, s_r\rangle |0\rangle |0\rangle \xrightarrow{\text{QBALL}_{12}} \underbrace{|s_1, \dots, s_r\rangle}_{\text{QBALL}_1} \underbrace{|V\rangle |F(\mathbf{x}_V)\rangle}_{\text{QBALL}_2}$$



SAT solving a-la Schönning...

- 1) derandomized Schönning...
- 2) ...reduces to PromiseBallSAT...
- 3) ...which recurses itself on smaller instance...
- 4) ...call size almost independent from n ...

Is it n-independent *enough*?

Main step of algorithm: **keeping track of flipped variables.**

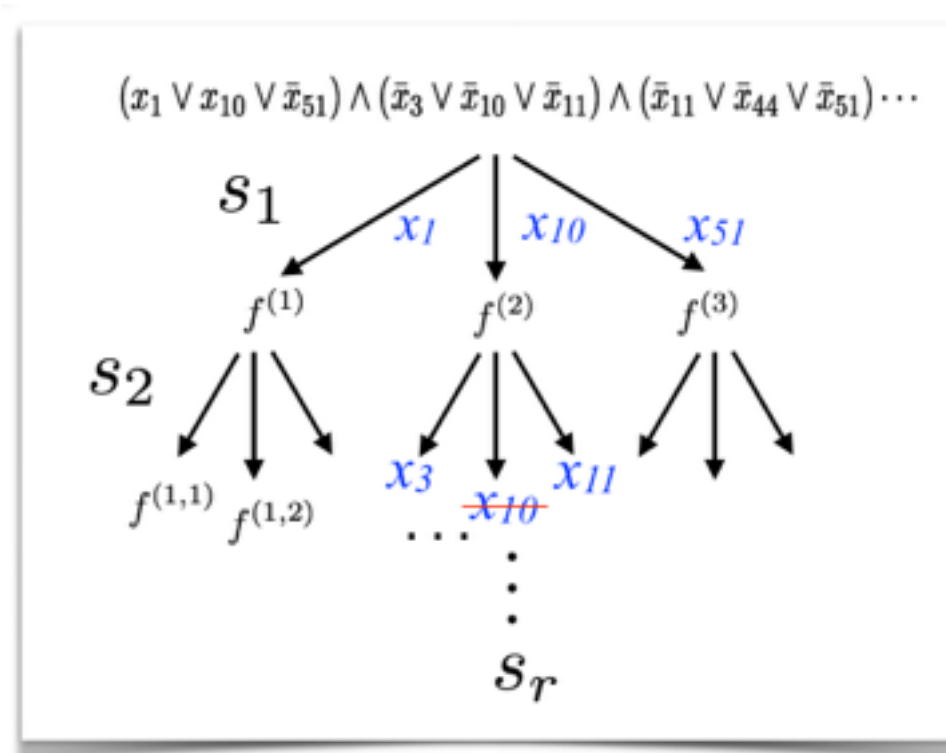
$$|s_1, \dots, s_r\rangle |V(k)\rangle \rightarrow |s_1, \dots, s_r\rangle |V(k+1)\rangle$$

$V(k+1) = V(k)$ appended with
 $(k+1)^{st}$ variable to be flipped

This is where the problem structure is exploited

Recall:

- when m is limited, how big “ r ” we can handle influences when quantum speed-ups kick in
- interesting cases when m/n is constant



Is it n-independent enough? **actually, non-triv...**

Main step of algorithm: keeping track of flipped variables.

$$|s_1, \dots, s_r\rangle |V(k)\rangle \rightarrow |s_1, \dots, s_r\rangle |V(k+1)\rangle$$

$V(k+1) = V(k)$ appended with
 $(k+1)^{st}$ variable to be flipped

What is V? Ordered list, then $O(r \log(n))$

Problem! Effective r we can handle
decays with $\log(n)$, **when m/n is constant !**

Is it n-independent enough? **actually, non-triv...**

Main step of algorithm: keeping track of flipped variables.

$$|s_1, \dots, s_r\rangle |V(k)\rangle \rightarrow |s_1, \dots, s_r\rangle |V(k+1)\rangle$$

$V(k+1) = V(k)$ appended with
 $(k+1)^{st}$ variable to be flipped

What is V? Ordered list, then $O(r \log(n))$

Problem! Effective r we can handle
decays with $\log(n)$, **when m/n is constant !**

If it is a **set**, need $O(r \log(n/r))$

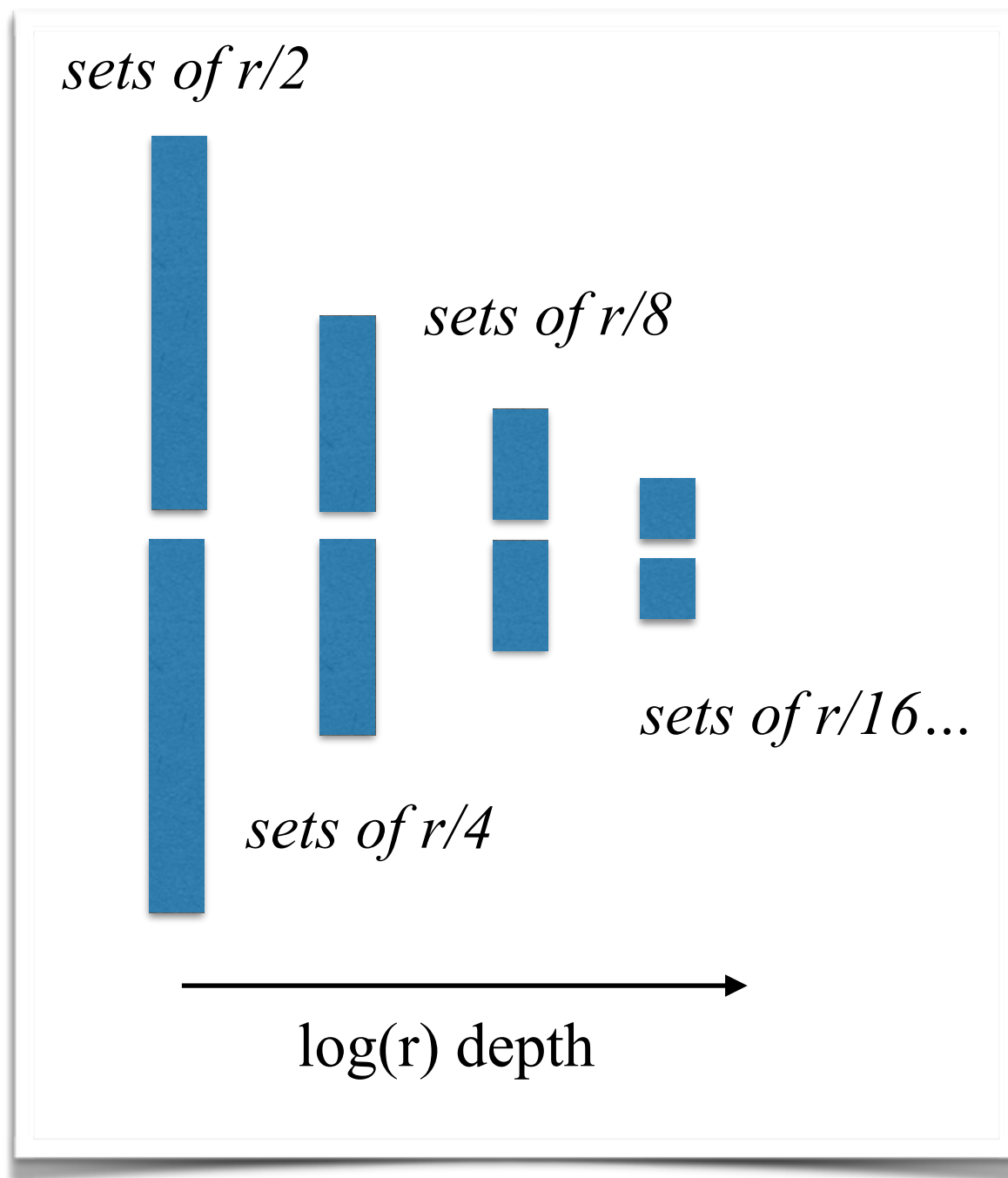
Now, this is an **n-independent fraction!**

Problem! Main step is no longer reversible!

Direct algorithmic deletion?

deletion recurses on r : $\exp(r)$ cost, no go

Solution: special memory structure and algorithmic deletion



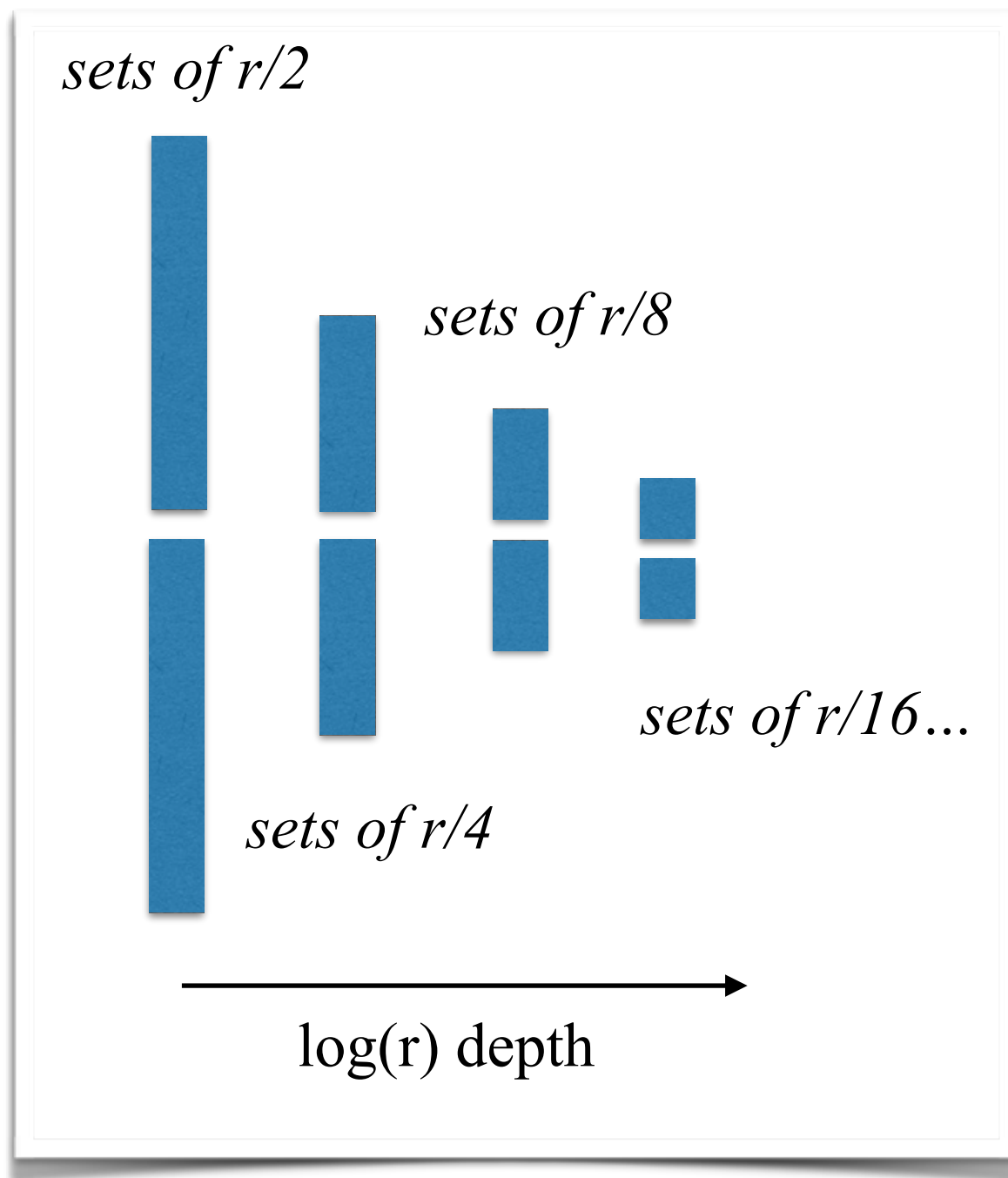
Fill k -th level:

1. Fill two $k-1$ levels
2. Join and copy to k^{th} level
3. Delete two $k-1$ levels

Recursion of depth $\log(r)$,
so in $2^{O(\log(n))} \in poly(n)$

Time AND memory efficient!

Solution: special memory structure and algorithmic deletion



Fill k -th level:

1. Fill two $k-1$ levels
2. Join and copy to k^{th} level
3. Delete two $k-1$ levels

Means: given QC of size m s.t. $m/n = \text{const.}$
we can quantum-solve $\text{PromiseBall}(r)$
where r/n is const.
Leads to true speedups.

Complete algorithm: combine fastest de-randomized Schöning, which speeds-up PromiseBall.

Total complexity:

$$O^* \left(2^{(\gamma + \varepsilon - f(m/n))n} \right)$$
$$f(x) \in \Theta(x / \log(1/x))$$

ε - can be made arbitrarily small
polynomial speedup!

Final statement: quantum enhancement for de-randomized Schöning's algorithm of Moser & Scheder
improving for any constant ratio m/n

Hard problems use structure *less...* and this *may be an advantage for near term devices*
Combined with an “*AI resilient to noise*”-type evidence
this provides further potential AI — QIP conspiracies.

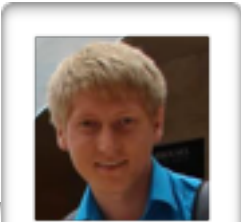
Acknowledgements:



theoretical
physics



Briegel



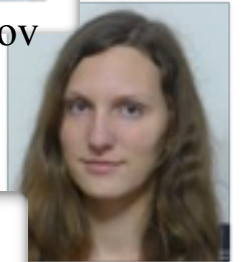
Melnikov



Wölk



Makmal



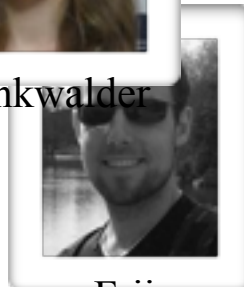
Trenkwalder



Poulsen Nautrup



Orsucci



Friis



Taylor



Liu



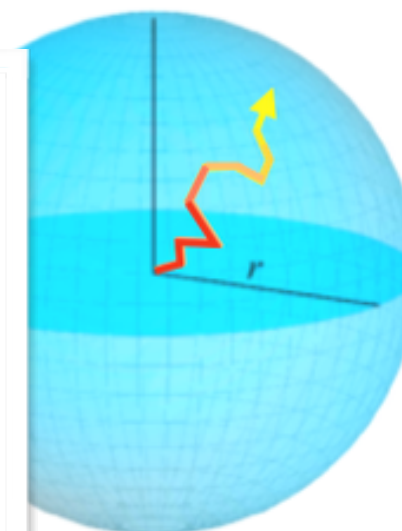
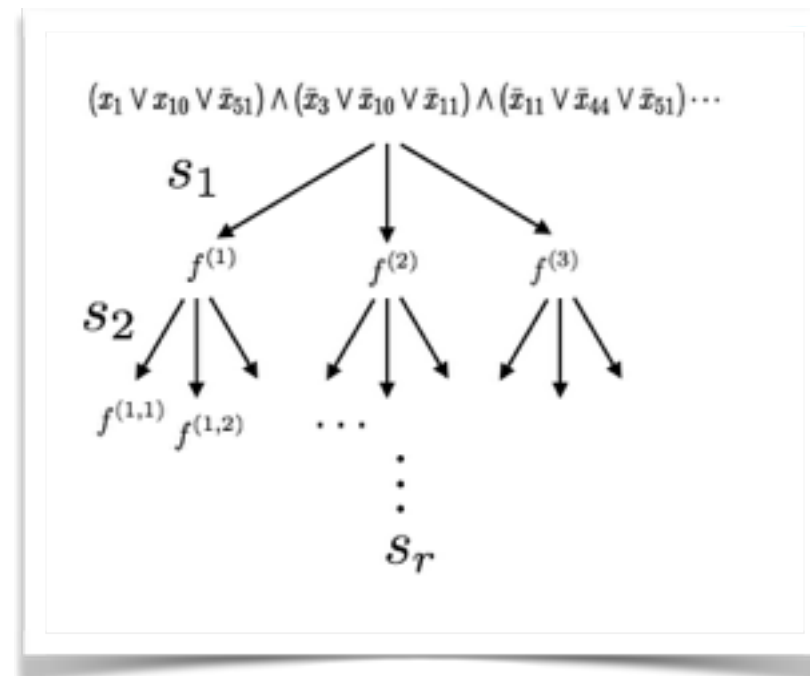
Wu



Cirac



Ge



Thank you

