Reinforcement learning's connection to quantum mechanics via the Schrödinger equation

甄景贤 (King-Yin Yan)

General.Intelligence@Gmail.com

Abstract. This paper contains enough details for implementation, and a prototype system is currently under development. We adopt an abstract style of exposition so that the reader can understand there is a large number of variations possible under this architecture.

Keywords: cognitive architecture, reinforcement learning, deep learning, logic-based artificial intelligence

0 Summary

We propose an AGI architecture:

1. With **reinforcement learning** (RL) as top-level framework
   - The external environment is turned "inward"
   - State space = mental space
2. **Logic** structure is imposed on the **knowledge representation** (KR)
   - State transitions are given by logic rules
   - Actions in RL = right-hand side of logic rules
3. The set of logic rules is approximated by a deep-learning neural network (**deep NN**)
   - Logic conjunctions are **commutative**, so the NN should be made **symmetric** using an algebraic trick (§**??**)
   - **Policy-gradient** methods (and variants) may be employed to speed up learning
   - Logic propositions are embedded in "continuous" space, so we have **continuous actions** in RL. The probability distribution over actions can be modeled by **Gaussian kernels** (radial basis functions).

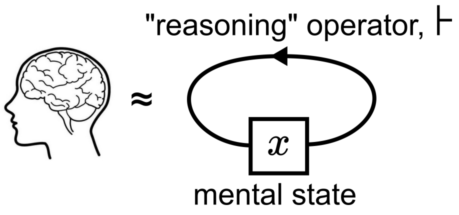
The rest of this paper will explain these design features in detail.

1 Reinforcement-learning architecture

The **metaphor** in the title of this paper is that of RL controlling an autonomous agent to navigate the maze of "thoughts space", seeking the optimal path:

(1)

The main idea is to regard "thinking" as a **dynamical system** operating on **mental states**:

 (2)

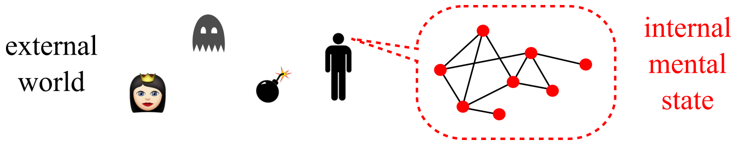
A mental state is a **set of propositions**, for example:

- I am in my room, writing a paper for AGI-2019.
- I am in the midst of writing the sentence, "I am in my room, ..."
- I am about to write a gerund phrase "writing a paper..."

Thinking is the process of **transitioning** from one mental state to another. As I am writing now, I use my mental states to keep track of where I am at within the sentence's syntax, so that I can construct my sentence grammatically.

1.1 "Introspective" view of reinforcement learning

Traditionally, RL deals with acting in an *external* environment; value / utility is assigned to *external* states. In this view, the *internal* mental state of the agent may change without any noticeable change externally:

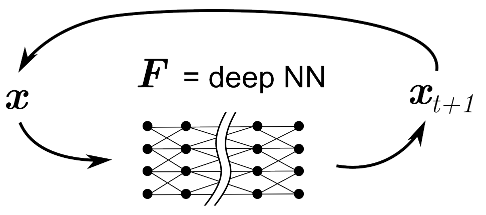
 (3)

1.2 Actions = cognitive state-transitions = "thinking"

Our system consists of two main algorithms:

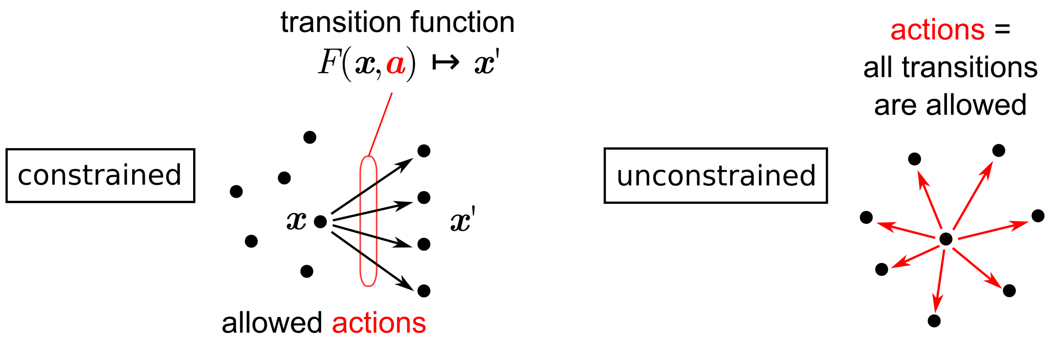
1. Learning the transition function $\vdash$ or $\boldsymbol{F} : \boldsymbol{x} \mapsto \boldsymbol{x}'$. $\boldsymbol{F}$ represents the **knowledge** that constrains thinking. In other words, the learning of $\boldsymbol{F}$ is the learning of "static" knowledge.
2. Finding the optimal trajectory of the state $\boldsymbol{x}$. This corresponds to optimal "thinking" under the constraints of static knowledge.

In our architecture, $\boldsymbol{F}$ can implemented as a simple feed-forward neural network (where "deep" simply means "many layers"):



(4)

In traditional reinforcement learning (left view), the system chooses an action $\boldsymbol{a}$, and the transition function $\boldsymbol{F}$ gives the probability of reaching each state $\boldsymbol{x}$ given action $\boldsymbol{a}$. In our model (right view), all possible cognitive states are potentially **reachable** from any other state, and therefore the action $\boldsymbol{a}$ coincides with the next state $\boldsymbol{x}'$.



(5)

1.3 Comparison with AIXI

AIXI's environmental setting is the same as ours, but its agent's internal model is a universal Turing machine, and the optimal action is chosen by maximizing potential rewards over all programs of the UTM. In our (minimal) model, the UTM is restricted to a neural network, where the NN's **state** is analogous to the UTM's **tape**, and the optimal weights (program) are found via Bellman optimality.

1.4 Infinite-dimensional control

The cognitive state is a vector $\boldsymbol{x} \in \mathbb{X}$ where $\mathbb{X}$ is the space of all possible cognitive states, the reasoning operator $\vdash$ or $\boldsymbol{F}$ is an **endomorphism** (an **iterative map**) $\mathbb{X} \to \mathbb{X}$.

Mathematically this is a **dynamical system** that can be defined by:

$$\boxed{\text{discrete time}} \qquad \boldsymbol{x}_{t+1} = \boldsymbol{F}(\boldsymbol{x}_t) \tag{6}$$

$$\text{or } \boxed{\text{continuous time}} \qquad \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) \tag{7}$$

where $\boldsymbol{f}$ and $\boldsymbol{F}$ are different but related [1]. For ease of discussion, sometimes I mix discrete-time and continuous-time notations.

A **control system** is a dynamical system added with the control vector $\boldsymbol{u}(t)$:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \tag{8}$$

The goal of control theory is to find the optimal $\boldsymbol{u}^*(t)$ function, such that the system moves from the initial state $\boldsymbol{x}_0$ to the terminal state $\boldsymbol{x}_\perp$.

A typical control-theory problem is described by:

$$\boxed{\text{state equation}} \qquad \dot{\boldsymbol{x}}(t) = \boldsymbol{f}[\boldsymbol{x}(t), \boldsymbol{u}(t), t] \tag{9}$$

$$\boxed{\text{boundary condition}} \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0\,,\ \boldsymbol{x}(t_\perp) = \boldsymbol{x}_\perp \tag{10}$$

$$\boxed{\text{objective function}} \qquad J = \int_{t_0}^{t_\perp} L[\boldsymbol{x}(t), \boldsymbol{u}(t), t] dt \tag{11}$$

and we seek the optimal control $\boldsymbol{u}^*(t)$.

According to control theory, the condition for **optimal path** is given by the Hamilton-Jacobi-Bellman equation:

$$\boxed{\text{Hamilton-Jacobi-Bellman}} \qquad 0 = \frac{\partial J^*}{\partial t} + \min_u H \tag{12}$$

$$\frac{d}{dt} V(x,t) = \min_u \{ C(x,u) + \langle \nabla V(x,t), f(x,u) \rangle \} \tag{13}$$

1.5 Reinforcement learning / dynamic programming

Reinforcement learning is a branch of machine learning that is particularly suitable for controlling an **autonomous agent** who interacts with an **environment**. It uses **sensory perception** and **rewards** to continually modify its **behavior**. The exemplary image you should invoke in mind is that of a small insect that navigates a maze looking for food and avoiding predators:

A reinforcement learning system consists of a 4-tuple:

$$\boxed{\text{reinforcement learning system}} = (\boldsymbol{x} \in \text{States}, \boldsymbol{u} \in \text{Actions}, R = \text{Rewards}, \pi = \text{Policy}) \tag{14}$$

For details readers may see my *Reinforcement learning tutorial* [?].

U is the total rewards of a sequence of actions:

$$\overset{\text{total value of state 0}}{U(\boldsymbol{x}_0)} \quad = \quad \sum_t \overset{\text{reward at time } t}{R(\boldsymbol{x}_t, \boldsymbol{u}_t)} \tag{15}$$

[1] They are related by: $\boldsymbol{x}(t+1) = \boldsymbol{F}(\boldsymbol{x}(t))$, $\boldsymbol{x}^{-1}(\boldsymbol{x}(t)) = t = \int_{\boldsymbol{x}_0}^{\boldsymbol{x}_t} \frac{d\boldsymbol{x}}{\boldsymbol{f}(\boldsymbol{x}(t))}$, and $f(\boldsymbol{x}) = \frac{1}{(\boldsymbol{x}^{-1})'(\boldsymbol{x}(t))}$. So we can just solve the functional equation $\boldsymbol{x}^{-1}(\boldsymbol{F}(\boldsymbol{x})) - \boldsymbol{x}^{-1}(\boldsymbol{x}) = 1$. See [?] §8.2.3.

For example, the value of playing a chess move is not just the immediate reward of that move, but includes the consequences of playing that move (eg, greedily taking a pawn now may lead to checkmate 10 moves later). Or, faced with delicious food, some people may choose not to eat, for fear of getting fat.

The goal of **reinforcement learning** is to learn the **policy function**:

$$\text{policy}: \quad \text{state} \overset{\text{action}}{\longmapsto} \text{state'} \tag{16}$$

when we are given the **state space**, **action space**, and **reward function**:

$$\text{reward}: \boxed{\text{state}} \times \boxed{\text{action}} \to \mathbb{R} \tag{17}$$

The action a is the same notion as the control variable u in control theory.

The central idea of **Dynamic programming** is the **Bellman optimality condition**, which says: "if we cut off a tiny bit from the endpoint of the optimal path, the remaining path is still an optimal path between the new endpoints."

$$\boxed{\text{Bellman equation}} \quad \overset{\text{value of entire path}}{U^*(\boldsymbol{x})} = \max_{\boldsymbol{u}}\{\ \overset{\text{reward of choosing } \boldsymbol{u} \text{ at current state}}{R(\boldsymbol{u})} + \overset{\text{value of rest of path}}{U^*(\boldsymbol{x}_{t+1})}\ \} \tag{18}$$

This seemingly simple formula is the entire content of dynamic programming; What it means is that: When seeking the path with the best value, we cut off a bit from the path, thus reducing the problem to a smaller problem; In other words, it is a **recursive relation** over time.

In AI reinforcement learning there is an oft-employed trick known as Q-learning. Q value is just a variation of U value; there is a U value for each state, and Q is the **decomposition** of U by all the actions available to that state. In other words, Q is the utility of doing action $\boldsymbol{u}$ in state $\boldsymbol{x}$. The relation between Q and U is:

$$U(\boldsymbol{x}) = \max_{\boldsymbol{u}} Q(\boldsymbol{x}, \boldsymbol{u}) \tag{19}$$

The advantage of Q is the ease of learning. We just need to learn the value of actions under each state. This is so-called "**model free learning**".

The **Bellman equation** governs reinforcement learning just as in control theory:

$$\begin{aligned} \boxed{\text{optimal path}} = &\ \text{choose max reward on current path segment} \\ &+ \boxed{\text{the rest of optimal path}} \end{aligned} \tag{20}$$

In math notation:

$$U_t^* = \max_u\{\ \boxed{\text{reward(u, t)}} + U_{t-1}^*\ \} \tag{21}$$

where U is the "long-term value" or **utility** of a path.

1.6 Connections with Hamiltonian and quantum mechanics

This section is optional.

In **reinforcement learning**, we are concerned with two quantities:

- $R(\boldsymbol{x}, \boldsymbol{u})$ = **reward** of doing action $\boldsymbol{u}$ in state $\boldsymbol{x}$
- $U(\boldsymbol{x})$ = **utility** or **value** of state $\boldsymbol{x}$

Simply put, **utility** is the integral of instantaneous **rewards** over time:

$$\boxed{\text{utility } U} = \int \boxed{\text{reward } R} \, dt \tag{22}$$

In **control-theoretic** parlance, it is usually defined the **cost functional**:

$$\boxed{\text{cost } \ J} = \int L dt + \Phi(\boldsymbol{x}_\perp) \tag{23}$$

where L is the **running cost**, ie, the cost of making each step; Φ is the **terminal cost**, ie, the value when the terminal state $\boldsymbol{x}_\perp$ is reached.

In **analytical mechanics** L is known as the **Lagrangian**, and the time-integral of L is called the **action**:

$$\boxed{\text{action } \ S} = \int L dt \tag{24}$$

Hamilton's **principle of least action** says that S always takes the **stationary value**, ie, the S value is extremal compared with neighboring trajectories.

The **Hamiltonian** is defined as $H = L + \frac{\partial J^*}{\partial \boldsymbol{x}} \boldsymbol{f}$, which arises from the method of **Lagrange multipliers**.

All these refer to essentially the same thing, so we have the following correspondence:

Reinforcement learning	**Control theory**	**Analytical mechanics**
utility or value U	cost J	action S
instantaneous reward R	running cost	Lagrangian L
action a	control u	(external force?)

(25)

Interestingly, the reward R corresponds to the **Lagrangian** in physics, whose unit is "energy"; In other words, "desires" or "happiness" appear to be measured by units of "energy", this coincides with the idea of "positive energy" in pop psychology. Whereas, long-term value is measured in units of [energy $\times$ time].

This correspondence between these 3 theories is explained in detail in Daniel Liberzon's book [**?**]. The traditional AI system is discrete-time; converting it to continuous-time seems to increase the computational burden. The recent advent of **symplectic integrators** [?] are known to produce better numerical solutions that retain qualitative features of the exact solution, eg. quasi-periodicity.

An interesting insight from control theory is that our system is a Hamiltonian dynamical system in a broad sense.

Hamilton's **principle of least action** says that the trajectories of dynamical systems occuring in nature always choose to have their action S taking **stationary values** when compared to neighboring paths. The action is the time integral of the Lagrangian L:

$$\boxed{\text{Action S}} = \int \boxed{\text{Lagrangian L}}\ dt \tag{26}$$

From this we see that the Lagrangian corresponds to the instantaneous "rewards" of our system. It is perhaps not a coincidence that the Lagrangian has units of **energy**, in accordance with the folk psychology notion of "positive energy" when we talk about desirable things.

The **Hamiltonian** H arises when we consider a typical control theory problem; The system is defined via:

$$\begin{aligned} \text{state equation:} &\quad \dot{\boldsymbol{x}}(t) = \boldsymbol{f}[\boldsymbol{x}(t), \boldsymbol{u}(t), t] &(27)\\ \text{boundary condition:} &\quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0\,,\ \boldsymbol{x}(t_\perp) = \boldsymbol{x}_\perp &(28)\\ \text{objective function:} &\quad J = \textstyle\int_{t_0}^{t_\perp} L[\boldsymbol{x}(t), \boldsymbol{u}(t), t] dt &(29)\end{aligned}$$

The goal is to find the optimal control $\boldsymbol{u}^*(t)$.

Now apply the technique of **Lagrange multipliers** for finding the maximum of a function, this leads to the new objective function:

$$U = \int_{t_0}^{t_\perp} \{L + \boldsymbol{\lambda}^T(t)\,[f(\boldsymbol{x}, \boldsymbol{u}, t) - \dot{\boldsymbol{x}}]\} dt \tag{30}$$

So we can introduce a new scalar function H, ie the Hamiltonian:

$$H(\boldsymbol{x}, \boldsymbol{u}, t) = L(\boldsymbol{x}, \boldsymbol{u}, t) + \boldsymbol{\lambda}^T(t) f(\boldsymbol{x}, \boldsymbol{u}, t) \tag{31}$$

Physically, the unit of $\boldsymbol{f}$ is velocity, while the unit of L is energy, therefore $\boldsymbol{\lambda}$ should have the unit of **momentum**. This is the reason why the phase space is made up of the diad of (position, momentum).

According to control theory, the **optimal path** is given by the Hamilton-Jacobi-Bellman equation:

$$\boxed{\text{Hamilton-Jacobi-Bellman}} \qquad 0 = \frac{\partial S^*}{\partial t} + \min_u H. \tag{32}$$

With the substitution $\Psi = e^{iS/\hbar}$ into the Hamilton-Jacobi equation, one can obtain the **Schrödinger equation** in quantum mechanics:

$$\boxed{\text{Hamilton-Jacobi}} \quad \frac{\partial S}{\partial t} = -H \quad \xRightarrow{\Psi = \exp\{iS/\hbar\}} \quad i\hbar\frac{\partial \Psi}{\partial t} = H\Psi \quad \boxed{\text{Schrödinger}} \tag{33}$$

which suggests that techiques in quantum mechanics can be applied to solve our AGI problem.

1.7 Diffusion equation

Recently, I accidentally discovered [?] [2] a precise transition from the classical H-J equation to the **Schrödinger equation** in quantum mechanics, via a simple substitution $\Psi = e^{iS/\hbar}$,

$$\boxed{\text{Hamilton-Jacobi}} \quad \frac{\partial S}{\partial t} = -H \quad \xRightarrow{\Psi=\exp\{iS/\hbar\}} \quad i\hbar\frac{\partial \Psi}{\partial t} = H\Psi \quad \boxed{\text{Schrödinger}}. \tag{34}$$

This implies that the Schrödinger equation is an alternative way of expressing the optimality condition for RL! It is also known that the Schrödinger equation in *imaginary time* becomes the **diffusion equation** and is related to stochastic processes (*cf* [?] Ch.6); This may lead to new algorithms.

We can also express J using the quantum-mechanical notation:

$$J = \langle \Psi | i\hbar \log \Psi | \Psi \rangle \tag{35}$$

All these "physical" ideas flow automatically from our definition of **rewards**, without the need to introduce them artificially. But these ideas seem not immediately useful to our project, unless we are to explore **continuous-time** models.

2 Weyl correspondence

Weyl proposed the following very general quantization rule: to a classical observable $a(x, p)$ depending on the position an momentum coordinates, one should associate an operator defined by

$$a_{\text{Weyl}}(\hat{x}, \hat{p}) = \frac{1}{2\pi\hbar}\int e^{\frac{i}{\hbar}(x\hat{x}+p\hat{p})} F a(x,p)dpdx \tag{36}$$

where F is the Fourier transform; this operator is formally obtained by replacing the variables x and p in the Fourier inversion formula by the non-commuting variables x and p used by Born, Jordan, and Heisenberg. Some easy algebra shows that if we quantize a classical Hamiltonian function

$$H = \frac{p^2}{2m} + V(x) \tag{37}$$

using Weyl' s rule one obtains the operator

$$H(\hat{x}, \hat{p}) = -\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + V(x) \tag{38}$$

which is the same as the one appearing in Schrödinger' s equation. So far, so good. The rub comes from the following observation: if we apply Weyl' s rule to the monomials $x^r p^s$ considered by Born, Jordan, and Heisenberg, we get the correspondence

$$x^r p^s \longrightarrow \frac{1}{2^s}\sum_{k=0}^{s}\binom{s}{k}\hat{p}^{s-k}\hat{x}^r\hat{p}^k \tag{39}$$

[2] The relation $S = i\hbar \log \Psi$ appeared in one of Schrödinger's 1926 papers, but is dismissed by him as "incomprehensible". This formula seems to be overlooked by physicists since that time, possibly including Feynman. I have yet to discuss / verify this with physicists.

where the $\binom{s}{k}$'s are the binomial coefficients; as is immediately seen by simple inspection, this rule is fundamentally different from Born and Jordan' s quantization rule, as soon as $r, s \geq 2$. Thus, if one wants to extend the Schrödinger picture to observables which are arbitrary functions of the variables x and p one obtains two different results depending on which quantization rule one uses. This fact has the following unwanted consequence: if one uses Weyl quantization, the Born-Jordan and Schrödinger pictures are no longer equivalent, and we thus have two different quantum mechanics.

2.1 Prior art: other cognitive architectures

The minimalist architecture based on reinforcement learning has been proposed by Itimar Ariel from Israel, in 2012 [**?**], and I also independently proposed in 2016 (precursor of this paper). The prestigious researcher of signal processing, Simon Haykin, recently also used the "RL + memory" design, cf. his 2012 book *Cognitive dynamic systems* [**?**]. Vladimir Anashin in the 1990's also proposed this kind of cognitive architecture [**?**]. There may exist more precedents, eg: [**?**].