Private memory confers no advantage

Samuel Allen Alexander[1] [0000−0002−7930−110X]

[1]The U.S. Securities and Exchange Commission, samuelallenalexander@gmail.com

Abstract. Mathematicians and software developers use the word "function" very differently, and yet, sometimes, things that are in practice implemented using the software developer's "function", are mathematically formalized using the mathematician's "function". This mismatch can lead to inaccurate formalisms. We consider a special case of this meta-problem. Various kinds of agents might, in actual practice, make use of private memory, reading and writing to a memory-bank invisible to the ambient environment. In some sense, we humans do this when we silently subvocalize thoughts about the actions we are taking (at least when the environment we're in is too primitive to probe the contents of our brains). Mathematical function formalizations of agents often ignore this ability. We show that in a general agent-environment framework (of which reinforcement learning is a special case), in a technical sense, such private memories do not enable qualitatively different agent behavior.

1 Introduction

There are many different ways to formalize the interaction between an agent and an environment. At a high level, the idea is simple: agent and environment take turns. On the agent's turn, the agent takes an action. On the environment's turn, the environment generates a percept for the agent to see. But precisely how to formalize this depends on various questions about things like what the agent and environment can remember, whether or not agent and/or environment can have an element of randomness, and so on. For example, if we are only interested in memoryless agents who see nothing but the most recent environmental percept, then we can formalize an agent as a function which takes a percept as input and outputs an action (or an action probability distribution). Examples of such agents include the so-called *policies* in the Stable Baselines3 reinforcement learning package [7]. On the other hand, if we are interested in agents who remember the entire interaction so far, then we should formalize an agent as a function which takes a *history* as input, not just a single percept. Examples of such agents include the agents in the Legg-Hutter formalization of reinforcement learning [4, 6].

And when formalizing agents in the latter way, there are still questions about what needs to be included in the *history* which is input into the agent-function. For example, if the agent's actions are completely deterministic, then a history need only include the sequence of percepts seen so far. From these, the agent's actions can be inferred, provided the agent is deterministic. It is superfluous

(at least ignoring computational efficiency concerns) for a deterministic agent to remember its own past actions, provided that it remembers the percepts that prompted those actions: the actions themselves, being deterministic, are determined by the percepts. But, if the agent is non-deterministic (i.e., if agents are functions that output action probability distributions rather than just actions), then the history which we input into the agent should include both the past environmental percepts and also the agent's own past actions. Past actions can not, generally, be inferred from past environmental percepts, if said actions had an element of randomness to them.

In addition to memories of past percepts and past actions, we can also imagine agents who have additional *private memory*. For example, we might imagine that the agent has access to a piece of scratch paper stored outside of the environment itself (and thus invisible to the environment) on which the agent can write notes to its own future self. The human hippocampus is not, in reality, an example of such a private memory bank, because, in reality, the human hippocampus is part of the environment (and thus vulnerable to being observed or modified by the outside world). But if, as simplifying assumption, the human agent is modelled as having his brain perfectly isolated, then the human hippocampus does become a place for such private memories to be stored. In such an idealized model, the human agent can silently think to herself things like: "I'm going to see what happens when I push this button. If it hurts me, then I won't push any more buttons like it in the future." And then, later on, the human agent has memory of such internal chatter that she made in the past.

In this paper, we consider agents who have memory of past environmental percepts and of their own past actions. We are concerned with the question: does private memory (in addition to memory of past actions and past environmental percepts) confer any advantage? We will show that the answer is "No," in the following sense. We will show that if π is any agent with private memory in addition to memory of the past agent-environment interaction, then there is an agent π' with only memory of past agent-environment interaction (and no private memory), such that for every finite initial history h of percepts and actions, the probably of h occurring when π interacts with a given environment is exactly equal to the probably of h occurring when π' interacts with that environment. This is trivial if private memories are written deterministically (π' can then reconstruct π's private memories by simulating π on the past percepts and actions which π' remembers). But if private memories have an element of randomness to them, then this result is much less obvious.

Although private memory confers no advantage in the above formal sense, there are other senses in which is might be considered to confer advantage. First of all, the π' described above is not guaranteed to have as good of a runtime complexity as the original π, and indeed, the π' which we will construct in our proof will generally be much more expensive than π. Second, the ability to use private memory might simplify the task of actually implementing a given agent. And thirdly, our proof only applies to traditional environments, not to environments capable of simulating the agent itself, as in [1] or [3].

2 Formal definitions

Throughout the paper, we fix a nonempty set $\mathcal{O}$ of *percepts*, a nonempty set $\mathcal{A}$ of *actions*, and a nonempty set $\mathcal{M}$ of *private memories*. We assume $\mathcal{O}$, $\mathcal{A}$, and $\mathcal{M}$ are pairwise disjoint.

Definition 1. *If X is any set, by a* probability distribution on X *we mean a function $f : X \to [0,1]$ such that $\sum_{x \in X} f(x) = 1$. For each $x \in X$, we say that f assigns probability $f(x)$ to x. We write $\Delta(X)$ for the set of all probability distributions on X.*

2.1 Agents without private memory

Definition 2. *(Histories)*

- *By an* environment-seen history*, we mean a sequence $\langle o_1, a_1, \ldots, o_n, a_n \rangle$, where each $o_i \in \mathcal{O}$ and each $a_i \in \mathcal{A}$. We also consider the empty sequence $\langle\rangle$ to be an environment-seen history.*
- *By an* agent-seen history*, we mean a sequence $\langle o_1, a_1, \ldots, o_{n-1}, a_{n-1}, o_n \rangle$, where each $o_i \in \mathcal{O}$ and each $a_i \in \mathcal{A}$. Note that for each percept $o \in \mathcal{O}$, the length-1 sequence $\langle o \rangle$ is an agent-seen history.*
- *By a* history *we mean an environment-seen history or an agent-seen history.*

Definition 3. *(Environments) By an* environment *we mean a function μ which takes as input an environment-seen history h, and outputs a probability distribution $\mu(h) \in \Delta(\mathcal{O})$. If h is an environment-seen history and $o \in \mathcal{O}$, we write $\mu(o|h)$ for $(\mu(h))(o)$ (the probability of o according to $\mu(h)$).*

Definition 4. *(Deterministic agents without private memory) By a* deterministic agent without private memory *we mean a function π which takes as input an agent-seen history h, and outputs an action $\pi(h) \in \mathcal{A}$.*

Definition 5. *(Non-deterministic agents without private memory) By a* non-deterministic agent without private memory *we mean a function π which takes as input an agent-seen history h, and outputs a probability distribution $\pi(h) \in \Delta(\mathcal{A})$. If h is an agent-seen history and $a \in \mathcal{A}$, we write $\pi(a|h)$ for $(\pi(h))(a)$ (the probability of a according to $\pi(h)$).*

When a deterministic agent π without private memory interacts with an environment μ, we imagine a sequence $o_1, a_1, o_2, a_2, \ldots$ chosen randomly as follows:

- o_1 is chosen randomly using the probability distribution $\mu(\langle\rangle)$.
- $a_1 = \pi(\langle o_1 \rangle)$.
- o_2 is chosen randomly using the probability distribution $\mu(\langle o_1, a_1 \rangle)$.
- $a_2 = \pi(\langle o_1, a_1, o_2 \rangle)$.
- And so on.

When a non-deterministic agent π without private memory interacts with an environment μ, we imagine a sequence $o_1, a_1, o_2, a_2, \ldots$ chosen randomly as follows:

- o_1 is chosen randomly using the probability distribution $\mu(\langle\rangle)$.
- a_1 is chosen randomly using the probability distribution $\pi(\langle o_1\rangle)$.
- o_2 is chosen randomly using the probability distribution $\mu(\langle o_1, a_1\rangle)$.
- a_2 is chosen randomly using the probability distribution $\mu(\langle o_1, a_1, o_2\rangle)$.
- And so on.

But instead of reasoning about these randomly-generated sequences, it is easier to reason about the probability of individual finite histories. In the following definition and throughout the rest of the paper, $\frown$ denotes concatenation.

Definition 6. *(The probability of a history) Suppose μ is an environment and π is a (deterministic or non-deterministic) agent without private memory. For every history h, let $P^\pi_\mu(h)$ be the probability that h would be an initial sequence of the sequence $o_1, a_1, \ldots$ randomly generated from letting π interact with μ as described above.*

Some authors, such as [5], would write $P(h)$ or a variation thereof for $P^\pi_\mu(h)$, if π and μ are clear from context.

2.2 Agents with private memory

Definition 7. *By an* agent-seen private-memory-augmented history *we mean a sequence $\langle o_1, a_1, m_1, \ldots, o_{n-1}, a_{n-1}, m_{n-1}, o_n\rangle$ where each $o_i \in \mathcal{O}$, each $a_i \in \mathcal{A}$, and each $m_i \in \mathcal{M}$. Note that for each percept $o \in \mathcal{O}$, the length-1 sequence $\langle o\rangle$ is an agent-seen private-memory-augmented history.*

Definition 8. *(Deterministic agents with deterministic private memory) By a* deterministic agent with deterministic private memory *we mean a function π which takes as input a private-memory-augmented agent-seen history h, and outputs an action-memory pair $\pi(h) \in \mathcal{A} \times \mathcal{M}$.*

Definition 9. *(Deterministic agents with non-deterministic private memory) By a* deterministic agent with non-deterministic private memory *we mean a function π which takes as input a private-memory-augmented agent-seen history h, and outputs $\pi(h) \in \mathcal{A} \times \Delta(\mathcal{M})$.*

Definition 10. *(Non-deterministic agents with deterministic private memory) By a* non-deterministic agent with deterministic private memory *we mean a function π which takes as input an agent-seen history h, and outputs $\pi(h) \in \Delta(\mathcal{A}) \times \mathcal{M}$.*

Definition 11. *(Non-deterministic agents with non-deterministic private memory) By a* non-deterministic agent with non-deterministic private memory *we mean a function π which takes as input an agent-seen history h, and outputs $\pi(h) \in \Delta(\mathcal{A} \times \mathcal{M})$.*

When a deterministic agent π with deterministic private memory interacts with an environment μ, we imagine a sequence $o_1, a_1, m_1, o_2, a_2, m_2, \ldots$ chosen randomly as follows:

- o_1 is chosen randomly using the probability distribution $\mu(\langle\rangle)$.
- $(a_1, m_1) = \pi(\langle o_1 \rangle)$.
- o_2 is chosen randomly using the probability distribution $\mu(\langle o_1, a_1 \rangle)$ (note the absense of m_1).
- $(a_2, m_2) = \pi(\langle o_1, a_1, m_1, o_2 \rangle)$.
- o_3 is chosen randomly using the probability distribution $\mu(\langle o_1, a_1, o_2, a_2 \rangle)$ (note the absense of m_1, m_2).
- $(a_3, m_3) = \pi(\langle o_1, a_1, m_1, o_2, a_2, m_2, o_3 \rangle)$.
- And so on.

When a deterministic agent π with non-deterministic private memory interacts with an environment μ, we imagine a sequence $o_1, a_1, m_1, o_2, a_2, m_2, \ldots$ chosen randomly as follows:

- o_1 is chosen randomly using the probability distribution $\mu(\langle\rangle)$.
- $a_1 = a$ and m_1 is chosen randomly using the probability distribution f, where $(a, f) = \pi(\langle o_1 \rangle)$.
- o_2 is chosen randomly using the probability distribution $\mu(\langle o_1, a_1 \rangle)$ (note the absense of m_1).
- $a_2 = a$ and m_2 is chosen randomly using the probability distribution f, where $(a, f) = \pi(\langle o_1, a_1, m_1, o_2 \rangle)$.
- o_3 is chosen randomly using the probability distribution $\mu(\langle o_1, a_1, o_2, a_2 \rangle)$ (note the absense of m_1, m_2).
- $a_3 = a$ and m_3 is chosen randomly using the probability distribution f, where $(a, f) = \pi(\langle o_1, a_1, m_1, o_2, a_2, m_2, o_3 \rangle)$.
- And so on.

When a non-deterministic agent π with deterministic private memory interacts with an environment μ, we imagine a sequence $o_1, a_1, m_1, o_2, a_2, m_2, \ldots$ chosen randomly as follows:

- o_1 is chosen randomly using the probability distribution $\mu(\langle\rangle)$.
- $m_1 = m$ and a_1 is chosen randomly using the probability distribution f, where $(f, m) = \pi(\langle o_1 \rangle)$.
- o_2 is chosen randomly using the probability distribution $\mu(\langle o_1, a_1 \rangle)$ (note the absense of m_1).
- $m_2 = m$ and a_2 is chosen randomly using the probability distribution f, where $(f, m) = \pi(\langle o_1, a_1, m_1, o_2 \rangle)$.
- o_3 is chosen randomly using the probability distribution $\mu(\langle o_1, a_1, o_2, a_2 \rangle)$ (note the absense of m_1, m_2).
- $m_3 = m$ and a_3 is chosen randomly using the probability distribution f, where $(f, m) = \pi(\langle o_1, a_1, m_1, o_2, a_2, m_2, o_3 \rangle)$.
- And so on.

When a non-deterministic agent π with non-deterministic private memory interacts with an environment μ, we imagine a sequence $o_1, a_1, m_1, o_2, a_2, m_2, \ldots$ chosen randomly as follows:

- o_1 is chosen randomly using the probability distribution $\mu(\langle\rangle)$.
- (a_1, m_1) is chosen randomly using the probability distribution $\pi(\langle o_1\rangle)$.
- o_2 is chosen randomly using the probability distribution $\mu(\langle o_1, a_1\rangle)$ (note the absense of m_1).
- (a_2, m_2) is chosen randomly using the probability distribution

$$\pi(\langle o_1, a_1, m_1, o_2\rangle).$$

- o_3 is chosen randomly using the probability distribution $\mu(\langle o_1, a_1, o_2, a_2\rangle)$ (note the absense of m_1, m_2).
- (a_3, m_3) is chosen randomly using the probability distribution

$$\pi(\langle o_1, a_1, m_1, o_2, a_2, m_2, o_3\rangle).$$

- And so on.

But instead of reasoning about these randomly-generated sequences, it is easier to reason about the probability of individual finite histories.

2.3 Conditional probabilities

Definition 12. *(Conditional probabilities of histories) Suppose μ is an environment, π is an agent (deterministic or non-deterministic, with or without deterministic or non-deterministic private memory), and h is a history.*

- *Let $P^\pi_\mu(h)$ be the probability that when π interacts with μ, h is an initial segment of the percept-action sequence randomly generated using π and μ as described above.*
- *Let $P^\pi(h)$ be the probability that h would be an initial segment of the percept-action sequence randomly generated as described above if π were to interact with some environment, on the condition that, in said random generation process, the percepts in h are always selected.*
- *Let $P_\mu(h)$ be the probability that h would be an initial segment of the percept-action sequence randomly generated as described above if some agent were to interact with μ, on the condition that, in said random generation process, the actions in h are always selected.*

Lemma 1. *For any μ, π, h as in Definition 12, $P^\pi_\mu(h) = P^\pi(h)P_\mu(h)$.*

Proof. By the multiplication rule of probability.

3 Private memory confers no advantage

Definition 13. *(Agent equivalence) Let π be an agent (deterministic or non-deterministic, with or without deterministic or non-deterministic private memory). Let ρ be an agent (deterministic or non-deterministic, with or without deterministic or non-deterministic private memory). We say $\pi \sim \rho$ if the following requirement holds: for every environment μ, for every history h, $P_\mu^\pi(h) = P_\mu^\rho(h)$.*

Theorem 1. *(Private memory confers no advantage)*

1. *If π is a deterministic agent with deterministic private memory, there exists a deterministic agent π' without private memory, such that $\pi' \sim \pi$.*
2. *If π is a deterministic agent with non-deterministic private memory, there exists a non-deterministic agent π' without private memory, such that $\pi' \sim \pi$.*
3. *If π is a non-deterministic agent with (deterministic or non-deterministic) private memory, there exists a non-deterministic agent π' without private memory, such that $\pi' \sim \pi$.*

Proof. Fix some $a_0 \in \mathcal{A}$.

First, we will prove (2) and (3) together. Then, we will prove (1).

(2) and (3). Define π' so that for each agent-seen history h and each $a \in \mathcal{A}$,

$$\pi'(a|h) = \begin{cases} \frac{P^\pi(h \frown a)}{P^\pi(h)} & \text{if } P^\pi(h) \neq 0; \\ 1 & \text{if } P^\pi(h) = 0 \text{ and } a = a_0; \\ 0 & \text{if } P^\pi(h) = 0 \text{ and } a \neq a_0. \end{cases}$$

Claim: π' is a non-deterministic agent without private memory. The only thing required to show is that π' outputs probability distributions. Thus, we must show that for any agent-seen history h, $\sum_{a \in \mathcal{A}} \pi'(a|h) = 1$, and for all $a \in \mathcal{A}$, $0 \leq \pi'(a|h) \leq 1$. The latter inequalities are easy to show by induction. To show $\sum_{a \in \mathcal{A}} \pi'(a|h) = 1$, it suffices to consider the case $P^\pi(h) \neq 0$ (the other case is trivial). Suppose a percept-action sequence is randomly generated by letting π interact with some environment, and that the percepts and actions so generated initially match h. The next action must be *some* action in $\mathcal{A}$. Thus $\sum_{a \in \mathcal{A}} P^\pi(h \frown a) = P^\pi(h)$. Therefore

$$\sum_{a \in \mathcal{A}} \pi'(a|h) = \sum_{a \in \mathcal{A}} \frac{P^\pi(h \frown a)}{P^\pi(h)} = \frac{1}{P^\pi(h)} P^\pi(h) = 1,$$

as desired, proving the claim.

It remains to show $\pi' \sim \pi$. Let μ be any environment. We will show by induction on h that for every history h, $P_\mu^{\pi'}(h) = P_\mu^\pi(h)$.

Case 1: $h = \langle\rangle$. Then $P_\mu^{\pi'}(h) = P_\mu^\pi(h) = 1$.

Case 2: $h = h_0 \frown o$ for some $o \in \mathcal{O}$. Then

$$\begin{aligned} P_\mu^{\pi'}(h) &= P_\mu^{\pi'}(h_0 \frown o) \\ &= P_\mu^{\pi'}(h_0)\mu(o|h_0) && \text{(Multiplicative rule)} \\ &= P_\mu^{\pi}(h_0)\mu(o|h_0) && \text{(Induction)} \\ &= P_\mu^{\pi}(h_0 \frown o) = P_\mu^{\pi}(h). && \text{(Multiplicative rule)} \end{aligned}$$

Case 3: $h = h_0 \frown a$ for some $a \in \mathcal{A}$.

Subcase 3.1: $P^\pi(h_0) = 0$. Then also $P^\pi(h_0 \frown a) = 0$, since one cannot randomly generate initial percepts and actions $h_0 \frown a$ without generating initial percepts and actions h_0 first. Thus $P_\mu^\pi(h_0 \frown a) = 0$ by Lemma 1. Thus

$$\begin{aligned} P_\mu^{\pi'}(h) &= P_\mu^{\pi'}(h_0 \frown a) \\ &= P_\mu^{\pi'}(h_0)\pi'(a|h_0) && \text{(Multiplicative rule)} \\ &= P_\mu^{\pi}(h_0)\pi'(a|h_0) && \text{(Induction)} \\ &= 0\pi'(a|h_0) = 0 = P_\mu^\pi(h_0 \frown a) = P_\mu^\pi(h). \end{aligned}$$

Subcase 3.2: $P^\pi(h_0) \neq 0$. Then

$$\begin{aligned} P_\mu^{\pi'}(h) &= P_\mu^{\pi'}(h_0 \frown a) \\ &= P_\mu^{\pi'}(h_0)\pi'(a|h_0) && \text{(Multiplicative rule)} \\ &= P_\mu^{\pi}(h_0)\pi'(a|h_0) && \text{(Induction)} \\ &= P^\pi(h_0)P_\mu(h_0)\pi'(a|h_0) && \text{(Lemma 1)} \\ &= P^\pi(h_0)P_\mu(h_0)P^\pi(h_0 \frown a)/P^\pi(h_0) && \text{(Definition of } \pi') \\ &= P_\mu(h_0)P^\pi(h_0 \frown a) && \text{(Algebra)} \\ &= P_\mu(h_0 \frown a)P^\pi(h_0 \frown a) && \text{(Clearly } P_\mu(h_0 \frown a) = P_\mu(h_0)) \\ &= P_\mu^\pi(h_0 \frown a) = P_\mu^\pi(h). && \text{(Lemma 1)} \end{aligned}$$

(1) Let π' be the deterministic agent without private memory, defined so that for every agent-seen history h:

- If $P^\pi(h) \neq 0$ then $\pi'(h) = a$ where $(a, m) = \pi(h) \in \mathcal{A} \times \mathcal{M}$.
- If $P^\pi(h) = 0$ then $\pi'(h) = a_0$.

Adopt the following convention: for every action a and agent-seen history h, write $\pi(a|h)$ for 1 if $\pi(h) = a$ or 0 if $\pi(h) \neq a$. Since π is deterministic, clearly if $P^\pi(h) \neq 0$ then $P^\pi(h) = 1$. In that case, if $(a, m) = \pi(h)$, then clearly $P^\pi(h \frown a) = 1$, and by definition $\pi'(h) = a$, so $\pi'(a|h) = 1 = P^\pi(h \frown a)/P^\pi(h)$. On the other hand, if $P^\pi(h) = 0$, then by definition $\pi'(h) = a_0$, so that $\pi'(a|h) = 1$ if $a = a_0$ and $\pi'(a|h) = 0$ if $a \neq a_0$. So altogether,

$$\pi'(a|h) = \begin{cases} \frac{P^\pi(h \frown a)}{P^\pi(h)} & \text{if } P^\pi(h) \neq 0; \\ 1 & \text{if } P^\pi(h) = 0 \text{ and } a = a_0; \\ 0 & \text{if } P^\pi(h) = 0 \text{ and } a \neq a_0, \end{cases}$$

exactly as in the proof of (2) and (3) above. Thus, an identical argument as above shows that $\pi' \sim \pi$.

Note that the π' constructed above is much more computationally expensive than π. If $\mathcal{M}$ is finite, then computing $P^\pi(h)$ (in general) "merely" requires computing π on a number of private-memory-augmented histories which is exponential in the length of h. If $\mathcal{M}$ is infinite, then computing $P^\pi(h)$ (in general) requires computing π infinitely many times and summing infinite series[1].

4 The reinforcement learning special case and generalization of prior work

One special case of agents and environments is *reinforcement learning* (or RL), in which the percepts generated from an environment include numerical rewards. In RL, agents are considered to perform better or worse depending on their ability or inability to maximize average rewards across the whole space of environments (or some suitable subspace thereof).

At the beginning of Section 2 we assumed a nonempty set $\mathcal{O}$ of percepts. All the results of the paper continue to hold if additional structure is imposed on those percepts, e.g., if we further require that each percept include a numerical reward. Thus, this paper's results automatically apply to RL. Let us impose exactly that requirement for the remainder of this section. For each $o \in \mathcal{O}$, let $r(o)$ denote the numerical reward included in o.

The following notation is standard in RL.

Definition 14. *If μ is an environment and π is an agent (deterministic or non-deterministic), with or without (deterministic or non-deterministic) private memory, let V_μ^π denote the expected total reward π would obtain from μ (i.e., the expected value of the sum $r(o_1) + r(o_2) + \cdots$ if we generate actions $a_1, a_2, \ldots$, percepts $o_1, o_2, \ldots$, and possibly private memories $m_1, m_2, \ldots$, as in Section 2), assuming this expected value exists. If not, V_μ^π is undefined.*

For an example in which V_μ^π might be undefined, take μ to be an environment which, every other turn, assigns probability 100% to a percept with reward 1, and every other turn, assigns probability 100% to a percept with reward -1. Any agent interacting with μ would obtain expected total reward $1 - 1 + 1 - 1 + \cdots$, a divergent series.

Proposition 1. *Suppose π, ρ are as in Definition 13. If $\pi \sim \rho$ then for every environment μ, $V_\mu^\pi = V_\mu^\rho$ (and the left-hand side is defined iff the right-hand side is defined).*

[1] And if π is sophisticated enough that the machinery on which π is run somehow experiences consciousness when π is computed on various inputs, then there could potentially even be ethical implications about plugging so many inputs into π in order to compute π'. This was perhapse foreshadowed by Nietzsche's doctrine of the eternal return.

Proof. For each agent σ (of whatever kind), let $V^{\sigma}_{\mu,n}$ denote the expected total reward that σ would obtain after interacting with μ until n percepts are generated. This can be computed by considering all possible histories terminating in an nth percept: for each such history h, multiply the probability $P^{\sigma}_{\mu}(h)$ of that history by its total reward $r(h)$ (defined as the sum of $r(o)$ where o ranges over the percepts in h), to obtain the expected total reward contributed by h to $V^{\sigma}_{\mu,h}$. Thus,

$$V^{\pi}_{\mu,n} = \sum_{h} P^{\pi}_{\mu}(h) r(h)$$

where h varies over the set of histories terminating in an nth percept. By identical reasoning, $V^{\rho}_{\mu,h} = \sum_h P^{\rho}_{\mu}(h) r(h)$. But $\pi \sim \rho$, so each $P^{\pi}_{\mu}(h) = P^{\rho}_{\mu}(h)$. Thus each $V^{\pi}_{\mu,n} = V^{\rho}_{\mu,n}$. Taking the limit as $n \to \infty$ proves the proposition.

Proposition 1 and Theorem 1 together justify this paper's title: private memory confers no advantage. Not only do equivalent agents have the same probability of resulting in any given history, but if different percepts come with different numerical rewards, then Proposition 1 shows that equivalent agents obtain the same expected total reward from each environment. Thus, as numerically measured by RL rewards, equivalent agents have the same exact numerical performance. Theorem 1 shows that for any agent which makes use of private memories, there is an equivalent agent which does not make use of private memories. Thus, private memories do not enable out-performance of agents without private memories.

This work generalizes the main result of [2]. The authors of that paper considered sequences $\vec{\pi} = (\pi_1, \ldots, \pi_n)$ of non-deterministic RL agents (without private memory), along with weights $\vec{w} = (w_1, \ldots, w_n)$ (each $w_i > 0$, with sum $w_1 + \cdots + w_n = 1$). The authors showed that for every such weighted distribution, there is a so-called mixture agent (a non-deterministic agent without private memory) $\vec{w} \cdot \vec{\pi}$ with the property that for every environment μ, $V^{\vec{w}\cdot\vec{\pi}}_{\mu} = \vec{w} \cdot (V^{\pi_1}_{\mu}, \ldots, V^{\pi_n}_{\mu})$. Expressed aphoristically: "the performance of the weighted mixture is the weighted mixture of the performances". But it is trivial to construct such an agent if the agent is allowed to have non-deterministic private memories. Assume $|\mathcal{M}| = n$, say $\mathcal{M} = \{m_1, \ldots, m_n\}$. Let π be the non-deterministic agent, with non-deterministic private memory, defined as follows (where h is a private-memory-augmented history).

- If $h = \langle o \rangle$ then let $\pi(h)$ assign to each pair $(a, m_i) \in \mathcal{A} \times \mathcal{M}$ the probability $\pi(a, m_i | h) = w_i \pi_i(a|h)$. In plain English: with probability w_i, act as π_i and remember (using private memory m_i) having done so.
- Otherwise, h is of the form $(o, a, m_i) \frown h_0$ for some $o \in \mathcal{O}$, $a \in \mathcal{A}$, $m_i \in \mathcal{M}$. Let h^- be the history obtained by deleting all private memories from h. For each pair $(a, m_j) \in \mathcal{A} \times \mathcal{M}$, let $\pi(h)$ assign the probability $\pi(a, m_j | h) = \pi_i(a|h^-)$ if $j = i$, or probability $\pi(a, m_j|h) = 0$ otherwise. In plain English: act as π_i, where i is the agent which you previously remembered (and repeat said memory).

Informally, for its initial turn, π randomly chooses one of the agents $\pi_1, \ldots, \pi_n$ (using the given weights), plays as that agent on that turn, and commits (using private memory) to play as that agent thereafter. Clearly for every environment μ, $V_\mu^\pi = \vec{w} \cdot (V_\mu^{\pi_1}, \ldots, V_\mu^{\pi_n})$. Proposition 1 and Theorem 1 together show there is a non-deterministic agent π' *without private memory* with the same performance. Thus, the main result from [2] is a special case of this paper.

5 Further generalization

The results of this paper could be further generalized to agents who, in response to input h:

- (Pre-rationalizing) First generate a memory m (depending on h), and then generate an action which may depend on both h and m.
- (Post-rationalizing) First generate an action a (depending on h), and then generate a memory which may depend on both h and a.
- (Pre-and-post-rationalizing) First generate a memory m_{pre} (depending on h), then an action a (depending on h and m_{pre}), and finally a memory m_{post} (depending on h, m_{pre}, and a).

The detailed formalization, however, becomes very verbose, so we leave it, and the corresponding version of Theorem 1, to the reader.

6 Conclusion

Mathematicians and software developers understand the word "function" in different ways. The software developer's "function" can, in the process of computing an output, do various things which the mathematician's function cannot do, for example, reading from or writing to a persistent memory-bank. We typically use the mathematician's "function" when we formalize certain things, because the mathematician's function is easier to reason about and prove things about. But the mismatch could, a priori, lead to inaccurate formalisms. In this paper, we considered agents, with or without such "private memory", in a very general agent-environment framework, and showed that in a technical sense, every agent with such private memory is equivalent to one without. Thus, in some sense, the software developer's agent's ability to use private memory confers no advantage. This partially justifies formalizing said agents as *mathematical* functions. Of course, software developers' agents can also do various other things that a mathematician's agent cannot do (such as query the system clock, access the computer's camera, or even surf the internet). At least, they can in principle—they might not do so very often in practice. Perhaps it is an advantage of the mathematical formalism that, by formalizing agents as *mathematical* functions, we automatically rule out things like agents who check the system clock or surf the internet in order to compute their outputs. We also rule out private memory, but as this paper shows, that's not a big loss.

Acknowledgements

We gratefully acknowledge Arthur Paul Pedersen, Len Du, Oscar Martinez, and the reviewers for feedback and discussion.

References

1. Samuel Allen Alexander, Michael Castaneda, Kevin Compher, and Oscar Martinez. Extending environments to measure self-reflection in reinforcement learning. *Journal of Artificial General Intelligence*, 13(1):1–24, 2022.
2. Samuel Allen Alexander, David Quarel, Len Du, and Marcus Hutter. Universal agent mixtures and the geometry of intelligence. In *AISTATS*. PMLR, 2023.
3. James Henry Bell, Linda Linsefors, Caspar Oesterheld, and Joar Skalse. Reinforcement learning in Newcomblike environments. In *NeurIPS*, 2021.
4. Marcus Hutter. *Universal artificial intelligence: Sequential decisions based on algorithmic probability*. Springer, 2004.
5. Marcus Hutter. Discrete MDL predicts in total variation. *Advances in Neural Information Processing Systems*, 22, 2009.
6. Shane Legg and Marcus Hutter. Universal intelligence: A definition of machine intelligence. *Minds and machines*, 17(4):391–444, 2007.
7. Antonin Raffin, Ashley Hill, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, and Noah Dormann. Stable Baselines3. https://github.com/DLR-RM/stable-baselines3, 2019.