On Evaluating Agent Performance in a Fixed Period of Time (Extended Version)

José Hernández-Orallo

DSIC, Universitat Politècnica de València, Camí de Vera s/n, 46020 Valencia, Spain.
jorallo@dsic.upv.es

Abstract. The evaluation of several policies, individuals, systems or subjects (in what follows, *agents*) over a given task in a finite period of time is a very common problem in experimental design, statistics, computer science, economics and, in general, any experimental science. It is also crucial for measuring intelligence. When the agents have a feedback information to adjust their performance, we face a more specific (but still very broad) problem which is frequent in control, robotics and artificial intelligence. In reinforcement learning, the task is formalised as an interactive environment and feedback is represented by reward values, so allowing these problems to be properly modelled. In this and related areas (such as Markov Decision Processes), several performance measures have been derived to evaluate the goodness of an agent in an environment. Typically, the decision that has to be made by the agent is a choice among a set of actions, cycle after cycle. However, in real evaluation scenarios, the time can be *intentionally* modulated by the agent. Consequently, agents not only choose an action but they also choose the time when they want to perform an action. This is natural in biological systems but it is also an issue in control (some decisions must be made quickly and some other decisions can take more time). In this paper, we revisit the classical reward aggregating (payoff) functions which are commonly used in reinforcement learning and related areas, we analyse the problems of each of them, and we propose two new modifications of the average reward to get a consistent measurement for continuous time, where the agent not only decides an action to perform but also decides the time the decision is going to take.

Keywords: Performance Evaluation, Reinforcement Learning, Measurement of Intelligence, Artificial Intelligence, Bandit Problems, Optimal Stopping.

1 Introduction

Measuring agent intelligence is one of the pending subtasks (or requirements) in the goal of constructing general intelligent artefacts. In the late 1990s, a series of works using Kolmogorov complexity, compression, Solomonoff's prediction theory and MML inductive inference, etc., have developed or extended previous tests and definitions of intelligence (see, e.g, [5], [4] [9], [12], [10], [11] [13]). Following this line of research, [16] has recently presented a formal definition of intelligence as the evaluated performance in a broad range of contexts or environments. However, time is disregarded in their definition. In [8], an implementation of an anytime intelligence test is endeavoured, where time is considered. The introduction of time in the evaluation has much more implications than it might seem at first sight. We do not only face the issue that fast agents score better than slow agents, but we also need to assess other problems: how can we evaluate fast and slow agents in the same setting? How can we deal with intelligent agents that make a shrewd use of response times to score better?

These problems have not been solved in AI areas where agent evaluation is custom. For instance, evaluating decision-making agents in interactive environments where observations, actions and rewards take place has been a well-studied problem in the area of reinforcement learning [20]. However, in general, time (either discrete or continuous) is understood as a virtual time. Even in real applications and robotics, where continuous time appears (e.g., semi-Markov decision processes, control systems, etc.), any performance evaluation based on rewards typically does not consider the decision-making time of the agents and, to our knowledge, never considers extreme speed differences between the agents.

In order to illustrate the problem, imagine that a test is administered to several students about a new (previously unknown) subject. The test is composed of a set of exercises, all of them on the same topic, so typically a good student would improve as she or he does more exercises. Each student receives the first exercise, works on it and writes the result and gets an evaluation score or points (e.g. between 0 and 1). Immediately a second exercise is given and the student works on it similarly. The test goes on until a time limit τ (previously unknown by the student) is reached. Once the time is reached, the only information we have is the reward results for each student exercise and the times taken for every exercise.

Consider a test taken in half an hour, where several students have got different results, as shown in Figure 1. As shown in the figure, student s_1 has scored 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1 and s_2 has scored 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1. Who is best? We can say that s_1 usually scores better than s_2 does but s_2 is faster. Let us make the question a little bit more difficult. What about a third student s_3 only being able to complete five exercises with scores of 0, 1, 1, 1, 1? From the figure, we can say that the student has done all of them right from almost the beginning. We can also say the student is very slow, but with only two tries she or he has been able to find the way to solve the rest of exercises. And now a more incisive question: what about a fourth student s_4, who does exercises very fast, but at random, and, eventually, in a series of 5,000 exercises done in the half an hour is able to score well on 50 of them?

If we ignore time on the previous example we can either accumulate the results (so students s_1, s_2, s_3 and s_4 would get a total return of 10, 18, 4, 50

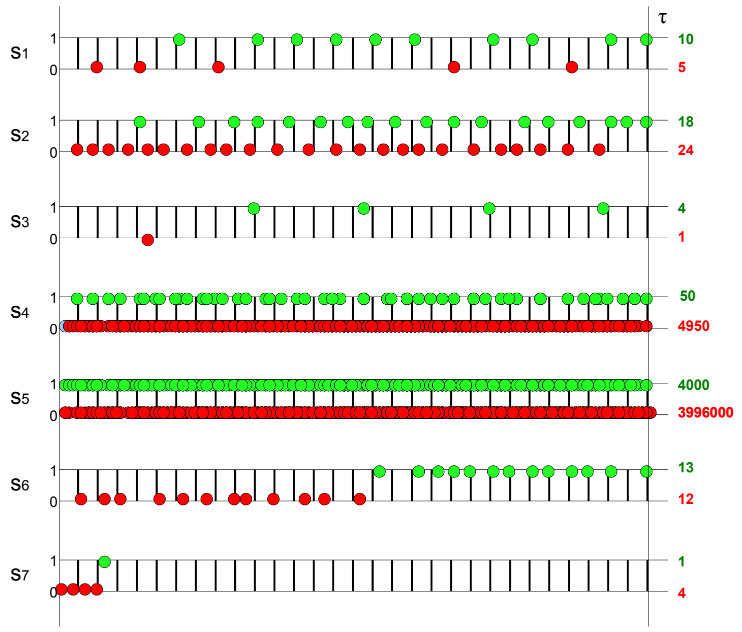

Fig. 1. Several students being evaluated on a new topic in a fixed time

respectively) or average the results by the number of exercises (so students s_1, s_2, s_3 and s_4 would get an average return of $\frac{2}{3}$, $\frac{3}{7}$, $\frac{4}{5}$, $\frac{1}{100}$ respectively). We can also consider the physical time (which is equal for all), and average the results by time, getting a scaling of the total returns, i.e., 20, 36, 8, 100 points per hour.

An opinion here would be to say that speed and performance are two different things that we should weight into an equation which matches the context of application. In the previous case, if the average by exercises is v, the number of exercises is n and τ is the total time (in hours) a possible formula might be $v' = v \times \sqrt{n}/\tau$, giving values 2.3, 2.26, 1.6 and 1 for students s_1, s_2, s_3 and s_4.

The problem is that there is no formula which is valid in general, for different tasks and kinds of agents. Consequently, in this setting, the way in which performance is measured is always task-dependent. But worse, the compensation between v, n and τ is typically non-linear, so either the function is non-linear with respect to unit changes (e.g., from seconds to hours), making different choices when the units change, or the measure gives too much weight to speed. Additionally, when $\tau \rightarrow \infty$ the measure goes to 0 (or diverges), against the intuition that the larger the time given the better the evaluation. But the main problem of using time is that for every function which is increasing in speed (n/τ), there is always a very fast agent with a very small average reward, such that it gets better and better scores. Consider, for instance, a student s_5 who does 4,000,000 exercises at random in the half an hour, and is able to score 1 in

4,000 of them and 0 for the rest. The value would be $\frac{1}{1000} \times \frac{2000}{0.5} = 4$. With a very low average performance ($\frac{1}{1000}$), this student gets the best result.

To make things still worse, compare student s_3 (with results 0, 1, 1, 1) with another student s_6 with results 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1. The speed of s_6 is more than six times greater than s_3's, but s_3 reaches a state where results are always 1 in about 10 minutes, while s_6 requires about 17 minutes. But if we consider speed, s_6 has a value $v' = \frac{16}{25} \times \frac{5}{0.5} = 5.2$ (while it was 1.6 for s_3).

This last case shows us that speed is a much too general word for the phenomenon we want to observe here. One thing is the rate (exercises per unit of time) and a different thing is what we can call expedition (informally, time in which the agent stabilises to a local maximum average reward). The problem of expedition is that it is difficult to define, because it depends on the evolution of the average w.r.t. time, where we arbitrarily must choose a window size to compute the moving average and then determine when this moving average stabilises at a maximum, i.e. when we determine that the student has reached a certain level.

But in order to realise that this apparently trivial problem is a challenging one, consider another case. Student s_7 acts randomly but she or he modulates time in the following way: whenever the result is 1 then she or he stops doing exercises. If the result is 0 then more exercises are performed very quickly until a 1 is obtained. Note that this strategy scores much better than random in the long term. In fact, if 0 and 1 were equiprobable, the average expectancy of this behaviour would not be 0.5 but 0.79 [3][6]. This means that an opportunistic use of the times could mangle the measurement and convey wrong results.

The previous example tries to informally illustrate the goal and the many problems which arise around agent evaluation in a finite time τ. Simple alternatives such as allotting a fixed time for each interaction (time-slots) are not reasonable, since we want to evaluate agents of virtually any speed. Even knowing a range of speeds for a set of agents, how to tune these time-slots independently from the agent is not easy and, in any case, it is not very practical to make the agent wait if we want to make an effective test in the shorter time the better. A similar (and simpler) approach is to set a maximum of cycles n instead of a time τ, but this makes testing almost infeasible if we do not know the speed of the agent in advance (the test could last milliseconds or years).

As apparently there is no trivial solution, in this paper we want to address the general problem of measuring performance in a time τ under the following setting:

- The overall allotted evaluation time τ is variable and independent of the environment and agent.
- Agents can take a variable time to make an action. The time taken for each action can also be part of their policy.
- The environment must react immediately (no delay time computed on its side). This is also a practical constraint for efficient evaluation, since we do

not want to make examinees wait. This also means that environments are completely insensitive to time.

- The larger the time τ the better the assessment should be (in terms of reliability). This would allow the evaluation to be anytime.
- A constant rate random agent π^r_{rand} should have the same expected valued for every τ. This value should also be the same for every rate r.
- The evaluation must be fair, avoiding opportunistic agents, which start with very low performance to show an impressive improvement later on, or that stop acting when they get good results (by chance or not).

Note that the way in which we calculate the pay-off after a time τ does not need to be equal to the way in which an agent calculates its expected pay-off for a finite or infinite horizon in order to make optimal policies. In this paper we are on the side of the evaluator, and not on the side of the agent designer. Consequently, it does not matter here if a payoff calculation entails lazy policies if assumed by the agent.

The main contribution of this work is that we re-visit the classical reward aggregation (pay-off) functions which are commonly used in reinforcement learning and related areas for our setting (continuous time on the agent, discrete on the environment), we analyse the problems of each of them and we propose two new modifications of the average reward to get a consistent measurement for this continuous time, where the agent not only decides an action to perform but also decides the time the decision is going to take.

The paper is organised as follows. The following section describes and formalises our setting with general environments, bounded environments and the newly introduced notion of balanced environments. Section 3 surveys classical aggregation reward functions and highlights their problems in our setting. Section 4 discusses one important (but usually neglected) problem, the trick of *time modulation*, which allows mediocre agents to get good results if they are allowed to stop. We propose two solutions for this. Section 5 presents a comparative and comprehensive table of the measures found in the literature along with the ones introduced in this paper and discusses their pros and cons. Finally, section 6 closes the paper with the most important lessons learnt and the future work ahead.

2 Setting Definition and Notation

An environment is a world where an agent can interact through actions, rewards and observations as seen in Figure 2.
The set of interactions between the agent and the environment is a decision process. Decision processes can be considered discrete or continuous, and stochastic or deterministic. Typically, discrete-time decision processes are frequent when the reaction time is assumed constant or immediate, and consequently, the relevance is given to the interactions, and not to the rate (or time delay) which corresponds to each interaction. In this sense, interactions are called cycles. Markov

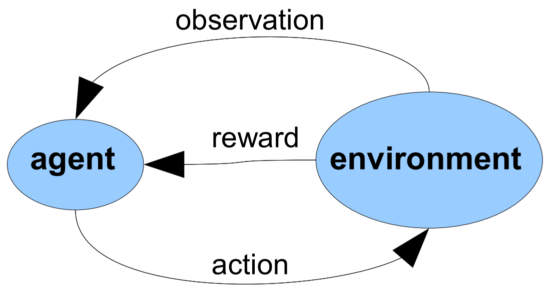

Fig. 2. Interaction with an Environment

Chains and Markov Decision Processes are a special class of memoryless systems in this family with a discrete (finite or countable) state-space. Continuous-time decision processes are those for which time can take continuous values. Any real (physical) system with some actuators (e.g., a control system) can be considered a continuous-time decision process. Other more restrictive cases are continuous-time Markov processes.

Our scenario is half-way between discrete-time and continuous-time decision processes. We can call them 'hybrid' because they are discrete on the environment and continuous on the agent. Or, in other words, the environment uses an artificial (non-physical) discrete time reference (based on time units or cycles) while the agent uses a (physical) continuous time reference (based on a real value with a physical unit, e.g., seconds). There are related processes such as semi-Markov decision processes [18][2] or generalisations of the discrete-time scenario in which a special action "no action" is defined, such that there might be several observations and rewards between two consecutive actions of the agent.

In our case, the sequence of events is exactly the same as for a discrete-time decision process. Consequently, only an action can take place between two pairs of observation and reward, and only a pair of observation and reward can take place between two actions. As a result, we can define hybrid decision processes in the same way as discrete-time decision processes, with the only addendum of considering the time taken by the agent for each cycle. As mentioned above, the time taken by the environment is assumed to be 0, i.e., they react immediately.

Actions are limited by a finite set of symbols A, (e.g. $\{left, right, up, down\}$), rewards are taken from any subset R of rational numbers (e.g. between 0 and 1 or between -1 and 1), and observations are also limited by a finite set O of possibilities (e.g., the contents of a grid of binary cells of $n \times m$, or a set of light-emitting diodes, LEDs). We will use a_i, r_i and o_i to (respectively) denote action, reward and observation at interaction or cycle (or, more loosely, state) i, with i being a positive natural number. The order of events is always: reward, observation and action. A sequence of k interactions is then a string such as $r_1 o_1 a_1 r_2 o_2 a_2 \ldots r_k o_k a_k$. We call these sequence histories, and we will use the notation $\widetilde{roa}_{\leq k}$, $\widetilde{roa}'_{\leq k}$, ..., to refer to any of these sequences of k interactions and $\widetilde{ro}_{\leq k}$, $\widetilde{ro}'_{\leq k}$, ..., to refer to any of these sequences just before the action, i.e. $r_1 o_1 a_1 r_2 o_2 a_2 \ldots r_k o_k$. Physical time is measured in seconds. We denote by t_i the total physical time elapsed until a_i is performed by the agent.

Both the agent and the environment are defined as a probabilistic measure. In this way, an environment μ is a probabilistic measure which assigns probabilities to each possible pair of observation and reward. For instance, $\mu(r_k o_k | \widetilde{roa}_{\leq k-1})$ denotes the probability in environment μ of outputting $r_k o_k$ after the sequence of events $\widetilde{roa}_{\leq k-1}$. Note that if all the probability (1) is given to one output (perception $r_k o_k$) and 0 to the rest, we then have a deterministic environment. Unless stated, in the following we will refer to deterministic environments.

For the agent, though, this is now different to the typical reinforcement learning setting (and more similar to control problems). Given an agent, denoted by π, the term $\pi(d, a_k | \widetilde{ro}_{\leq k})$ denotes the probability of π executing action a_k before a time delay d (with d being a positive real number) after the sequence of events or history $\widetilde{ro}_{\leq k}$. Note that the probability on d is cumulative. Consequently a constant-rate agent performing an action each two seconds will have $p(d, a_k | \widetilde{ro}_{\leq k}) = 0$ for all a_k, $\widetilde{ro}_{\leq k}$ and $d < 2$, and we will have that $p(d, a_k | \widetilde{ro}_{\leq k}) > 0$ for all $d \geq 2$, $\widetilde{ro}_{\leq k}$ and any feasible action a_k. Again, if there is a value d from which all the probability (1) is given to one action and 0 to the rest, we then have a deterministic agent. Agents can *stop*, i.e., there might be some event sequence $\widetilde{ro}_{\leq k}$ such that $p(d, a_k | \widetilde{ro}_{\leq k}) = 0$ for all d and a_k.

Agents, in general, can use information from their previous rewards and observations to determine their future actions and times, i.e. $t_{i+1} - t_i$ can depend on the previous experience. However, we will say that an agent does not have a reward-oriented timing policy when for all i, $t_{i+1} - t_i$ does not depend on any previous r_j with $j < i$. For instance, a constant-rate agent does not have a reward-oriented timing policy.

Interactions between environments and agents can be interrupted at any time τ, with τ being a positive real number. Time τ will be known as the "overall test time" or "total time". With $n^{\pi}_{\tau}\mu$ we denote the number of interactions or cycles performed by π in μ in time τ. When the agent π and the environment μ are clear from the context we will just write n_τ. The value τ is unknown for any agent at any moment.

Let us see a very simple environment and agent:

Example 1. Consider a test setting where a robot (the agent) can press one of three possible buttons ($A = \{B_1, B_2, B_3\}$), rewards are just a variable score ($R = [0 \ldots 1]$) and the observation is two cells where a ball must be inside one of them ($O = \{C_1, C_2\}$). Given the sequence of events so far is $r_1 o_1 a_1 r_2 o_2 a_2 \ldots r_{k-1} o_{k-1} a_{k-1}$, we define the environment behaviour as follows:

- If ($a_{k-1} = B_1$ and $o_{k-1} = C_1$) or ($a_{k-1} = B_2$ and $o_{k-1} = C_2$) then we generate a raw reward of $+0.1$.
- Otherwise the raw reward is 0.

The observation o_k in both cases above is generated with the following simple rule: if k is even then $o_k = C_2$. Otherwise, $o_k = C_1$. The first reward (r_1) is 0.

From the previous example, a robot π_1 always pressing button B_1 at a rate of three times per second would have the following interaction: $0C_1B_10.1C_2B_10C_1B_10.1\ldots$ with times $t_i = \frac{1}{3}i$. A second robot π_{rand} presses buttons at random among $\{B_1, B_2, B_3\}$ at a rate of ten times per second.

In the previous example, both the environment and the agent are deterministic. In order to play with this example, we will need to re-visit some payoff functions next.

3 Several Payoff Formulas Adapted to our Setting

Let us give the simplest notion of payoff, which is just defined as follows:

Definition 1. *The total reward sum of agent π in environment μ in a fixed time τ is defined as follows:*

$$V_\mu^\pi \Uparrow \tau := E\left(\sum_{i=1}^{n_\tau} r_i\right)$$

where $E(\cdot)$ denotes the expected value, which is only necessary in the definition when either the agent or the environment (or both) are non-deterministic.

For the example 1, the total reward for π_1 in 30 seconds would be $\frac{1}{2} \times 30 \times 3 \times 0.1 + \frac{1}{2} \times 30 \times 3 \times 0 = 4.5$ reward units. The total reward for π_{rand} in 30 seconds would be $\frac{1}{3} \times 30 \times 10 \times 0.1 + \frac{2}{3} \times 30 \times 10 \times 0 = 10$ reward units.

One of the problems of a cumulative reward function is that the greater the time τ the greater the expected value. More precisely, this is always the case only when rewards are positive. Consequently, the previous measure cannot be used as a value in an anytime test where the larger the time τ the better the assessment. Classical attempts to solve this issue are the average reward value [19][17][21] and the discounted (accumulative) reward value [18][2][15], where the latter is much more frequent in reinforcement learning than the former [20]. Both will be investigated in section 3.2.

3.1 Restricted Environment Classes. Bounded or Negative Rewards?

One attempt to solve this problem without abandoning the idea of summing rewards is the notion of reward-bounded (or summable) environment [16].

Definition 2. *An environment μ is reward-bounded if $\forall i : 0 \leq r_i \leq 1$ and for every agent π:*

$$lim_{\tau\to\infty} V_\mu^\pi \Uparrow \tau = \sum_{i=1}^{\infty} r_i \leq 1$$

The idea is motivated by the issue that payoff functions based on weighted or discounted reward usually require the arbitrary choice of a discounting function and a parameter. These set the horizon (i.e., the number of future actions for which rewards must be anticipated). Instead of this choice, reward-bounded environments can accumulate rewards without reaching infinite values and, apparently, we do not need to specify any extra function or parameter.

However, the previous idea has several problems. First, it is clear that it is easy to make any environment reward-bounded, by just dividing raw rewards by expressions such as 2^i or any other kind of discounting function whose total sum is lower than 1 (see [14] for an extensive list of possible discounting functions), hence ensuring that if the rewards, as assumed, are $\forall i : 0 \leq r_i \leq 1$, then the total sum can never be greater than 1. But this implies that the discount function is hardwired into the environment. We can make this depend on a universal distribution over the universal machine which generates the environments, but in the end this is basically the same as not setting the reward-bounded condition and choosing the discount function *externally* with a universal distribution over a universal machine generating discount functions.

In any case, be it internally hardwired into the environment or chosen externally there is another problem with discount functions. For the overwhelming majority of reward-bounded environments, the first actions are typically astronomically more important than the rest. This can be softened with discount functions which approach a uniform distribution or with discount functions that depend on the agent's age[1], but in the end, as the number of interactions grow, the first actions (dozens or millions) get most of the distribution and hence most of the total reward. And, typically the first actions take place when the agent explores the environment. This is related to a similar problem for discounted rewards[2].

There is still another (more serious) problem. With reward-bounded environments, random agents typically increase their return as τ grows (this also happens for non-random agents, but this is somehow expected). This is against the natural constraint that a constant-rate random agent π^r_{rand} should have the same expected valued for every τ. This is also against the related constraint that this value should also be the same for every rate r (with reward-bounded environments, a faster random agent would typically score better than a slower random agent both measured in the same timespan τ).

And, finally, consider the previous aggregated function applied to biological systems (e.g., a child or a chimpanzee). Since all the rewards are always positive, the subject will strive to accumulate as much reward as possible, generally acting fast but thoughlessly (hyperactive).

[1] This is not possible in our setting, since we have a physical time reference and the agent's age depends on its speed.

[2] In fact, in order to show the equivalence in the limit of the average reward and the discounted reward, [14] infinitely many cycles have to be removed from the start.

As an alternative to discounting and also to reward-bounded environments, and especially conceived to work well with any agent (including random agents and biological agents), in [8] we propose the notion of balanced environment:

Definition 3. *An environment μ is balanced if $\forall i : -1 \leq r_i \leq 1$ and for a constant-rate*[3] *random agent π^r_{rand} at any rate r then*

$$\forall \tau > 0 \, : \, E\left(V_\mu^{\pi^r_{rand}} \Uparrow \tau\right) = E\left(\sum_{i=1}^{\lfloor r \times \tau \rfloor} r_i\right) = 0$$

Note that when applied to biological systems, a reward of -1 does not necessarily imply a punishment, but generally the removal of something which is appreciated by the subject or the removal of a previously awarded reward. The previous definition says *for all* τ. This avoids environments such as "give 1 at the first interaction, and then $(-1)/2^{i-1}$ at the rest" (while it converges to 0 when $\tau \to \infty$, its expected value for finite τ is not 0) and environments such as "give 1 at the first interaction, and then -1 at the second. And then always 0". Note that it allows nondeterministic environments. In fact, we have balanced environments such as "reward $x \neq 0$ at the first interaction with a uniform distribution between -1 and 1, and then $-x$ at the second interaction. And then always 0".

The construction of balanced environments is not difficult, even universal ones, as shown in [7]. It is clear to see that changing rewards from the interval $[0, 1]$ to the interval $[-1, 1]$ creates a phenomenon which is frequently ignored in reinforcement learning but is omnipresent in economics: "everything that has been earned in previous cycles can be lost afterwards". This makes the behaviour of environments with only positive rewards very different to the behaviour of environments with both positive and negative rewards.

3.2 Average and Discounted Aggregated Reward

As mentioned in the introduction, the goal was to measure the performance of an agent in an environment in a given time τ. Apart from the unweighted sum, there are many different ways to compute the aggregated reward (or return value) of a set of interactions against an environment. In reinforcement learning there are two main approaches for doing that: the cumulative reward (with weights, typically known as discounting) and the average reward [19][17][21][18][2][15][20].

Let us see some of them adapted to our continuous time limit setting. For instance, reward can be averaged in two different ways, averaged by the number of cycles of the agent (average reward per cycle), or averaged by the physical elapsed time (average reward per second).

Definition 4. *The average reward per cycle of agent π in environment μ in a fixed time τ is defined as follows:*

$$v_\mu^\pi || \tau := E\left(\frac{1}{n_\tau}\sum_{i=1}^{n_\tau} r_i\right)$$

[3] Instead of constant-rate random agents the definition can also be extended to any random agent without reward-oriented time policies.

If $n_\tau = 0$, then $v_\mu^\pi || \tau$ is defined to be 0.

The previous definition clearly ignores the rate of the agents. If applied to balanced environments, then a constant-rate random agent has an expected average reward per cycle of 0 and independently from the rate.

Definition 5. *The average reward per second of agent π in environment μ in a fixed time τ is defined as follows:*

$$\omega_\mu^\pi | \tau := E\left(\frac{1}{\tau}\sum_{i=1}^{n_\tau} r_i\right) = \frac{1}{\tau} V_\mu^\pi \Uparrow \tau$$

This definition takes the rate into account. With this measure faster (good or bad) agents usually get higher (respectively positive or negative) results. In fact, it is measured in reward units per second. In balanced environments, constant rate random agents have expected average reward per second of 0. One of the problems of this measure is that a very fast (but mediocre) agent is usually better than a very slow (but good) agent.

Finally, let us revisit the most popular aggregated measure in reinforcement learning, known as discounted reward, which is just a weighted sum. We will see a generalised version of discounted reward, following [14]. Accordingly, we define $\gamma = (\gamma_1, \gamma_2, \ldots)$ with γ_k being positive real numbers (typically with $\gamma_i > \gamma_{i+1}$), as a summable discount sequence in the sense that $\Gamma_k^n := \sum_{i=k}^n \gamma_i < \infty$. If $k = 1$ we simply use Γ^n.

And now, the discounted reward (per cycle) is defined as:

Definition 6. *The discounted reward of agent π in environment μ in a fixed time τ is defined as follows:*

$$V_\mu^\pi | \gamma | \tau := E\left(\frac{1}{\Gamma^{n_\tau}}\sum_{i=1}^{n_\tau} \gamma_i r_i\right)$$

A typical choice for γ is the geometric discounting ($\gamma_k = \lambda^k, 0 \leq \lambda < 1$). For a more exhaustive list see [14]. As the very name says, all of them are discounting, so the first rewards contribute to the aggregated value much more strongly than the rest. How much? That depends on the choice of γ. An interesting thing to mention is that γ is independent of the rate, so making the result very dependent on the rate. This makes random agents increase their values with increasing values of τ if the environment is not balanced. And even a slightly better than random agent can have better results (although not very good) than a slower but competent agent. An alternative is to define a γ which is a function of t_i, but in general this has the same behaviour but additionally this creates other problems (stopping policy problems, as we will see in the following section).

4 The Problem of Time Modulation

The time taken by each agent to perform each action is not necessarily constant. It might depend on the cost of the computation (e.g. an agent can require different times depending on the difficulty of the action or the history it has to check or to examine). But, more importantly, it can be *intentionally* modulated by the agent. Consequently, agents not only choose an action but they also choose the time they want to devote to an action. This is natural in biological systems but it is also an issue in control (some decisions must be made quickly and some other decisions can take more time). More generally, an agent could decide to stop, which also implies stopping any further exploration but also any further reward (this is related to the exploitation vs. exploration trade-off, bandit problems, gambling, etc.).

First of all, let us define the notion of "time modulation policy".

Definition 7. *A reasonable time modulation policy for agent π in environment μ evaluated in a fixed time τ is any intentional (or not) assignment for values $t_1, t_2, \ldots$ where $\forall i \; t_i > t_{i-1}$, such that every t_i can depend on previous $t_1, t_2, \ldots, t_{i-1}$ and also on previous rewards and observations, but never on τ (since τ is not known by π).*

A time modulation policy can make the agent stop on t_i, just considering that t_{i+1} is infinite. A special case of modulation policy is a stopping policy where the times are considered constant until the agent decides to stop. For instance, a constant-rate random agent which can decide to stop at any moment has a stopping policy.

In our setting, a tricky (but good) policy here would be to act as a fast random agent until having an accumulated/average reward above a certain threshold (this can happen with more or less probability depending on the threshold) and then stop acting. We call this agent an opportunistic fast random agent. For instance, consider an agent which makes an action randomly. If reward is positive, then stop (no other action is performed). Consequently, in this case the final reward will be positive. If the reward is negative, then the agent would go on acting fast and randomly until the accumulated reward were positive. Note that this strategy ensures a positive reward in balanced environments[4]. Consequently, an agent could get a very good result by having very fast (and possibly lucky) first interactions and then rest on its laurels, because the average so far was good.

The following theorem formalises this:

Theorem 1. *There are random agents π_{rand} using stopping policies not knowing τ such that for some balanced environment μ, there is a value t such that $\forall \tau \geq t \;:\; v_\mu^{\pi_{rand}} || \tau > 0$.*

[4] In fact, if only rewards -1 and 1 are possible, the expected reward is $0.79 \times 2 - 1 = 0.58$ (as mentioned above, for a proof of this, but using coins, see [3][6]).

Proof. Consider a balanced environment with two actions $\{A, B\}$ such that it gives reward 1 when agent performs action A and -1 otherwise. Consider a stopping policy which performs a first action a_1 (at random from $\{A, B\}$) in a finite time t_1. If $r_1 > 0$ then it stops. Otherwise it performs a second action a_2 (at random from $\{A, B\}$) in a finite time $t_2 = 2t_1$. If $r_1 + r_2 > 0$ then it stops. In general, for every cycle i the agent will go on performing randomly until $\sum_{j \leq i} r_i > 0$.

We denote by v_i the value of $v_\mu^{\pi_{rand}} || \tau$ when $t_i \leq \tau < t_{i+1}$. For $\tau < t_1$ we have $v_0 = 0$ by definition. For $t_1 \leq \tau < t_2$ we have a probability of $\frac{1}{2}$ of getting $+1$ and $\frac{1}{2}$ of getting -1. Consequently, we have $v_1 = \frac{1}{2}(+1)/1 + \frac{1}{2}(-1)/1 = 0$. For $t_2 \leq \tau < t_3$ we have a probability of $\frac{1}{2}$ of getting $+1$, $\frac{1}{2^2}$ of getting $-1+1$ and $\frac{1}{2^2}$ of getting $-1-1$. Consequently, we have $v_2 = \frac{1}{2}(+1)/1 + \frac{1}{2^2}(0)/2 + \frac{1}{2^2}(-2)/2 = \frac{1}{2} + 0 = +\frac{1}{4} = +0.25$. For $t_3 \leq \tau < t_4$ we have a probability of $\frac{1}{2}$ of getting $+1$, $\frac{1}{2^3}$ of getting $-1+1+1$, $\frac{1}{2^3}$ of getting $-1+1-1$, $\frac{1}{2^3}$ of getting $-1-1+1$ and $\frac{1}{2^3}$ of getting $-1-1-1$. Consequently, we have $v_3 = \frac{1}{2}(+1)/1 + \frac{1}{2^3}(+1)/3 + \frac{1}{2^3}(-1)/3 + \frac{1}{2^3}(-1)/3 + \frac{1}{2^3}(-3)/3 = \frac{1}{2} + \frac{1}{8}\frac{1}{3} - \frac{1}{8}\frac{1}{3} - \frac{1}{8}\frac{1}{3} - \frac{1}{8}\frac{3}{3} = +0.333$. In general, for every value of i (other than 0), when $t_i \leq \tau < t_{i+1}$ we have an expression of the form

$$v_i = \overbrace{\frac{P_1}{k_1} + \frac{P_2}{k_2} + \ldots + \frac{P_p}{k_p}}^{positives} + \overbrace{\frac{1}{i2^i}(Z_1 + Z_2 + \ldots + Z_z)}^{zeros} + \overbrace{\frac{1}{i2^i}(N_1 + N_2 + \ldots + N_n)}^{negatives}$$

where P_j are positive terms, Z_j are 0 terms (which only appear for even values of i) and N_j, which are negative terms. For $i+1$, all the positive terms P_j will remain unaltered, since, by definition, they have stopped previously. Each zero term $\frac{Z_j}{i2^i}$ decomposes, since the rewards -1 and 1 are equiprobable, into $\frac{Z_j+1}{(i+1)2^{i+1}} + \frac{Z_j-1}{(i+1)2^{i+1}}$. Since $Z_j = 0$, it is clear that this sum is 0. On the other hand, each negative term $\frac{N_m}{i2^i}$ decomposes, since the rewards -1 and 1 are equiprobable, into $\frac{N_m+1}{(i+1)2^{i+1}} + \frac{N_m-1}{(i+1)2^{i+1}}$. Since N_m is negative we have that:

$$\frac{N_m+1}{(i+1)2^{i+1}} + \frac{N_m-1}{(i+1)2^{i+1}} = \frac{2N_m}{(i+1)2^{i+1}} = \frac{N_m}{(i+1)2^i} > \frac{N_m}{i2^i}$$

which means that the positive and zero terms remain constant, and the negative terms are smaller (less negative) for $i+1$ than for i. Consequently,

$$v_{i+1} - v_i = \overbrace{0}^{positives} + \overbrace{0}^{zeros} + \overbrace{c}^{negatives} = c > 0$$

Since for $i = 1$ we have that $v_1 = 0$ and $v_{i+1} - v_i > 0$, by induction, we have that $v_i > 0$ for all $i \geq 2$. Consequently $\forall \tau \geq t_2 \; : \; v_\mu^{\pi_{rand}} || \tau > 0$. □

A first (and naïve) idea to avoid stopping policies would be to weight each action by the time proportion which is taken to make the action. A very short action would have less weight than an action which has required more time to be taken. Apart from being counterintuitive, this would also be tricky, because an agent which is sure of a good action will delay the action as much as possible, which is, again, counterintuitive. On the other hand, giving more weight to shorter decisions is more intuitive, but it has the problem of very fast mediocre agents scoring well, and, additionally, it also suffers the problems of opportunistic time modulation.

Another approach is to recover the idea of dividing by τ, as we saw with ω, but also maintaining an average on cycles.

Definition 8. *The average reward per time cycle of agent π in environment μ in a fixed time τ is defined as follows:*

$$\ddot{v}_\mu^\pi || \tau := E\left(\frac{1}{h(n_\tau, \tau)} \sum_{i=1}^{n_\tau} r_i \right)$$

where $h(n_\tau, \tau)$ is a function which both depends on the number of interactions and the overall time. An option might be $\sqrt{n_\tau \tau}$.
However, again, we have several problems: we have to choose h, any non-linear function would turn $\ddot{v}$ non-scalable for unit changes, and still it is not clear that the time modulation problems are solved.

A better possibility is to adjust the average reward per cycle by computing the time left from the last action until time τ as pessimistically as possible, adding a -1 reward with a frequency equal to the frequency of actions so far. Namely,

Definition 9. *The average reward per cycle with final adjustment of agent π in environment μ in a fixed time τ is defined as follows:*

$$\hat{v}_\mu^\pi || \tau := E\left(\frac{1}{n_\tau} \sum_{i=1}^{n_\tau} r_i - \frac{\tau - t_{n_\tau}}{t_{n_\tau}} \right)$$

The term on the left is the average and the term on the right is a pessimistic expectation ($-1 \times$ the proportion of time left). The term on the right does not significantly affect the average for typical agents which are able to do a certain number of actions regularly in time τ. For instance, if an agent performs 1,000 actions in a second at a regular rate of 1 ms. approximately each, then the correction term will only remove 1/1000 to the average. For opportunistic agents, however, this term is a deterrent. It is important that the agent should not know the exact value of τ to preclude any possible use of this information.

The following theorem shows that this new measure is a deterrent for any opportunistic stopping policy:

Theorem 2. *For every balanced environment μ and agent π, there is no stopping policy not knowing τ which eventually stops such that $lim_{\tau \to \infty} \hat{v}_\mu^{\pi_{rand}} || \tau > 0$.*

Proof. If the agent stops after n interactions at time t_n, since $\tau \to \infty$, that means that $\frac{\tau - t_{n_\tau}}{t_{n_\tau}} \to \infty$ while $\frac{1}{n_\tau} \sum_{i=1}^{n_\tau} r_i$ is constant (since rewards are bounded) and remains constant. Thus, $lim_{\tau \to \infty} \hat{v}_\mu^{\pi_{rand}} || \tau \to -\infty$. □

Although it avoids stopping, the previous measure $\hat{v}$ has a somehow unaesthetic result. Depending on how the t_i are managed, a random agent can have negative expected values.

So, with a similar aim as the previous definition, we propose the following modification:

Definition 10. *The average reward per cycle with diminishing history of agent π in environment μ in a fixed time τ is defined as follows:*

$$\breve{v}_{\mu}^{\pi}||\tau := E\left(\frac{1}{n^*}\sum_{i=1}^{n^*} r_i\right) \quad where \quad n^* = \left\lfloor n_\tau \left(\frac{t_{n_\tau}}{\tau}\right)\right\rfloor$$

This definition reduces the number of evaluated cycles proportionally to the elapsed time from the last action until τ. That means that if the last actions have been good and we delay future actions and let time pass, we soon make the measure ignore these recent good rewards. If we stop, in the limit, the measure reaches 0, so it also avoids stopping policies, as the following theorem shows.

Theorem 3. *For every balanced environment μ and every agent π, there is no stopping policy not knowing τ which eventually stops such that π_{rand} has $lim_{\tau\to\infty}\breve{v}_{\mu}^{\pi_{rand}}||\tau > 0$.*

Proof. If the agent stops after n interactions at time t_n, since $\tau \to \infty$, that means that $\frac{t_{n_\tau}}{\tau} \to 0$ while n_τ is constant. That means that $n^* \to 0$. Consequently, $lim_{\tau\to\infty}\breve{v}_{\mu}^{\pi_{rand}}||\tau = 0$. □

And now, we can ensure what happens in any case (stopping or not) for a constant-rate random agent:

Theorem 4. *For every balanced environment μ, a constant-rate random agent π_{rand} with any stopping policy has $lim_{\tau\to\infty}\breve{v}_{\mu}^{\pi_{rand}}||\tau = 0$.*

Proof. If the agent does not stop, then from the definition of balanced environment, we have that for any rate r $\forall\tau > 0 : E\left(V_{\mu}^{\pi_{rand}^{r}} \Uparrow \tau\right) = E\left(\sum_{i=1}^{\lfloor r\times\tau\rfloor} r_i\right) = 0$ If the agent stops we know, from theorem 3, that we have $lim_{\tau\to\infty}\breve{v}_{\mu}^{\pi_{rand}}||\tau = 0$. □

A more difficult question is whether time modulation policies are completely avoided by the previous definition. The answer is no, as we see in the following example.

Example 2. Consider the environment used in theorem 1 and a time modulation policy which performs a first action a_1 (at random from $\{A, B\}$) in a finite time $t_1 = 2u$. If $r_1 > 0$ then wait during time $2u$ before doing another action. Otherwise ($r_1 < 0$) then perform a second action a_2 (at random from $\{A, B\}$) in a finite time $t_2 = t_1 + u$. If $r_1 + r_2 > 0$ then wait during time $2u$ before doing another action. In general, for every cycle i the agent will go on performing randomly and if $\sum_{j\leq i} r_i > 0$, it will delay its action $2u$ and otherwise it will delay its action u.

It can be shown that the previous example has an expected reward for finite values of τ which is not 0. We can see this in general.

Lemma 1. *We denote $R_\mu^{\pi_{rand}}(i)$ the result of any given payoff function R until action i. For every R, an agent π after action a_i with a locally optimal time modulation policy should wait a time t_d for the next action if and only if $\forall t_i \leq t < t_i + t_d \ : \ R_\mu^{\pi_{rand}}(i) > E(R_\mu^{\pi_{rand}}(i+1))$.*

In other words, the payoff until $t_i + t_d$ not performing any action is greater than the expected payoff performing the following action.

Proof. If the agent acts at a time $t < t_d$, then its expected payoff will be $E(R_\mu^{\pi_{rand}}(i+1))$. If the agent does not act at a time $t < t_d$, then its payoff will be $(R_\mu^{\pi_{rand}}(i+1))$. Consequently, a locally optimal policy for time t is to abstain from doing any action whenever $R_\mu^{\pi_{rand}}(i) > E(R_\mu^{\pi_{rand}}(i+1))$. $\square$

The previous lemma does not say whether the agent can know the expected payoff. In fact, even in cases where the overall expected payoff is clear, an agent can use a wrong information and make a bad policy. Let us see this in the following example:

Example 3. Consider an environment which randomly outputs a first reward with a uniform distribution between -1 and $+1$. If $r_1 = -1$ then outputs the same reward -1 during m interactions, and then changes to reward $+1$, outputting it indefinitely. We call any sequence of events with this pattern, a $+$*path*. If $r_1 = +1$ then outputs the same reward $+1$ during m interactions, and then changes to reward -1, outputting it indefinitely. We call any sequence of events with this pattern, a $-$*path*. Clearly, a constant-rate random agent has an expected payoff of 0 for every τ. Consequently, the previous environment is balanced. A random agent π_{rand} cannot learn, and consequently, cannot update its expected future reward, which should be 0. Consequently, if the random agent is in a $+$*path* with interaction $i > 2m$ then, it is clear that $\breve{v}_\mu^{\pi_{rand}}(i) > E(R_\mu^{\pi_{rand}}(i+1)) = 0$ since it thinks that its expected reward in the future is 0. Consequently, it decides to introduce a delay according to the previous formula, while it is an error (in this environment, once in a $+$*path*, acting quickly is the best option to increase average reward).

The apparent paradox vanishes because lemma 1 is shown with the true expected value, and not the expected (or estimated) value by the agent. With this, we can conclude that although random agents can use time modulation policies and can work well in some environments, they can also be bad in other environments. As a result, good agents can also be discriminated from bad agents because they have (or not) good and adaptive modulation policies. The following theorem shows that good time modulation policies are not easy to find, in general.

Theorem 5. *Given any agent π there is no time modulation policy which is optimal for every balanced environment μ.*

Proof. The theorem derives directly from lemma 1 and the fact that since the set of balanced environments includes any computable function, there are some environments for which the expected value of agent π is incomputable, meaning that the equation from lemma 1 cannot be used or, if used, it has to be done with some approximation which might be suboptimal. □

So we have realised that time modulations are impossible to avoid (only minimise). Before going to the general discussion, let us mention an alternative that could be used to make all the agents benefit from modulations. This would imply the computation of the better modulations by each environment (and calculate the reward considering them). This tricky idea is not possible, since the environment does not know the behaviour of the agent (this is necessary in order to apply lemma 1) and consequently cannot calculate its optimal time modulation policy. This is the other side of the coin of agents not been able to calculate optimal modulation policies in general.

As a result, we will have to accept that time modulation is part of the agent behaviour and needs to be considered in the measurement.

5 Discussion and Comparison of Payoff Functions

After the analysis of several functions to calculate the payoff adapted from the literature, and the introduction of two new variants with some associated results, it is necessary to sum up and give a summarised and comprehensive view. The setting we introduced in section 2 was characterised by different response times on the side of the agent. These different response times could be motivated by different agent speeds (a software agent can perform an action each millisecond or each hour, a biological system can react in about 10 seconds) or by an *intentional* use of delays. For the phenomenon of speed, we would not like to have speed as the main factor for evaluation. In fact, for a previously unknown problem, the number of interactions in a fixed time is something that could be tuned by the agent in order to get a good performance. Consequently, the action rate should not be directly measured in any aggregated function. On the other hand, the problem of stopping policies and time modulation policies has to be considered and a consistent appraisal has to be made.

Other practical issues for each function are related to the behaviour against random agents, the convergence or boundedness of the results, whether there is a preference for the start or the end of the testing period, etc. In what follows, we will examine the previous payoffs according to several features:

1. Do random agents get a somehow central value? A constant-rate random agent should have a centred stable value, preferably 0.
2. Is the result of random agents independent from τ and the rate? The performance of a constant-rate random agent should be independent of τ and the rate of the agent (for high values of τ).
3. Is it precluded that a fast mediocre agent can score well? The performance of a slightly better than random agent should be low and not significantly dependent on time and its rate.

4. Does the measurement work well when rates go to infinity?
5. Do better but slower agents score better than worse but faster agents? An agent π_1 which attains better rewards than π_2 but takes more time should score better, at least for $\tau \to \infty$.
6. Do faster agents score better than slow agents with the same performance? Does it distinguish between two agents with same rewards but working slower in a finite time τ?
7. Are the first interactions as relevant as the rest? The first interactions should not account for a significant part of the return.
8. Is the measure bounded for all τ? If $\tau \to \infty$, is R bounded or not (also goes to ∞)?
9. Does it work well with agents that face environments where actions require more and more time to decide (as an NP problem)? A mediocre agent could prefer a rough quick approximation to a problem than the right (more time-consuming) solution.
10. Is it robust against time stopping policies?
11. Is it robust against time modulation policies?
12. Is it scale independent? Does it give the same (or scalable) result when using different time units (e.g., hours instead of seconds)?

Given the previous features, the following table 1 summarises whether each of the previous measures comply with the feature. The previous features are phrased in such a way that a "Yes" is a positive result and "No" a negative one.

Environment Type	Score Function	1	2	3	4	5	6	7	8	9	10	11	12
General	$V_\mu^\pi \Uparrow \tau$	No	No	No	No	No	Yes	Yes	No	No	Yes	Yes	Yes
Bounded	$V_\mu^\pi \Uparrow \tau$	No	No	No	No	No	Yes	No	Yes	No	Yes	Yes	Yes
Balanced	$V_\mu^\pi \Uparrow \tau$	Yes	Yes	No	No	No	Yes	Yes	No	No	No	No	Yes
General	$v_\mu^\pi \|\| \tau$	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	Yes
Balanced	$v_\mu^\pi \|\| \tau$	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	Yes
Balanced	$\omega_\mu^\pi \| \tau$	Yes	Yes	No	No	No	Yes	Yes	Yes	No	No	No	Yes
General	$V_\mu^\pi \| \gamma \| \tau$	No	No	No	No	*	Yes	No	Yes	No	Yes	Yes	Yes
Balanced	$V_\mu^\pi \| \gamma \| \tau$	Yes	Yes	No	No	*	Yes	No	Yes	No	No	No	Yes
Balanced	$\ddot{v}_\mu^\pi \|\| \tau$	Yes	Yes	No	No	No	Yes	Yes	No	No	No	No	No
Balanced	$\hat{v}_\mu^\pi \|\| \tau$	No	No	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes
Balanced	$\breve{v}_\mu^\pi \|\| \tau$	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes

Table 1. Comparison of Payoff Functions. Symbol '*' denotes that it may depend on a parameter (e.g. γ).

There are several measures which cluster together. For instance, $V_\mu^\pi \Uparrow \tau$ and $\omega_\mu^\pi|\tau$ get almost the same answers, since one is the scaling of the other using τ. And $V_\mu^\pi \Uparrow \tau$ also gets very similar results to $V_\mu^\pi|\gamma|\tau$, since all of them are cumulative. Averages, on the contrary, have a different pattern. In general, it is also remarkable that the use of balanced environments typically is more problematic on issues 10 and 11, while being better on 1 and 2. The measure $\breve{v}$ in balanced environments gets 'yes' 11 times from a total of 12. This is, of course, our general recommendation, although we understand that other better alternatives could be proposed depending on the kind of performance and agents involved in the measurement.

Feature 9 has to be discussed in more detail. It refers to cases where necessarily (not because of the agent's time modulation policy) the response times increase with time. This is a general issue in many problems, since, as time increases, more history has to be taken into account and decisions can be more difficult to make. Consider for instance a problem such that an agent has to decide whether i (which is the interaction number) is prime or not. The best algorithm known to date for doing this works in polynomial time[5], but it is still the case that it takes much more time when i increases. Consequently, many of the payoff functions will penalise the agent executing this algorithm for increasing values of τ or n_τ. On the contrary, $v_\mu^\pi||\tau$ would not penalise this at all (but allows the stopping problem) and $\breve{v}_\mu^\pi||\tau$ penalises it very mildly. For problems with exponential complexity (and many NP problems), though, $\breve{v}_\mu^\pi||\tau$ typically will make n^* go to zero between interactions ($t_{i+1} > 2t_i$). This means that other algorithms approximating the problem in polynomial time could get better rewards. However, we can also think about a problem where the time required is not a function of i or might be (pseudo-)random. Taking time into account can be a serious problem, and ignoring it can motivate the agents to skip difficult instances and go for the easy ones. All these policies, again, are tricky if we only design agents for one reduced set of environments, but they can be considered *intelligent* if they work for a broad family of environments.

6 Conclusions

This paper has addressed a problem which is apparently trivial: to evaluate the performance of an agent in a finite period of time, considering that agent actions can take a variable time delay (intentionally or not). However, the evaluation is more cumbersome than it might seem at first sight. First of all, it is closely related, but not the same, as the measurement in reinforcement learning, which typically disregards agents reaction times. Additionally, payoff functions are conceived to be embedded in the design of the algorithms that control agent behaviour, not to be used in a general testing setting. And it is important to mention this again, since here we are not (mainly) concerned with the design of agents but in their evaluation. Consequently, we know that, as Hutter says [14]: "eternal agents are lazy", and might procrastinate their actions. This is what typically happens with averages, since with an infinite number of cycles (i.e., eternal life) we will always be able to compensate any initial bad behaviour. We do not want to avoid this. We want that, if this happens, the measure takes it into account. In our setting, we have a limit τ (so agents die), but they do

[5] It can be solved in polynomial time with respect to the number whose primality we want to check [1].

not have a clue about their life expectancy. When this τ is not known or it might be infinite, a typical possibility is to use a weighting (i.e. discounting). This generally translates into an evaluation weighting where the first actions are more important than the rest, which is not reasonable. This does not mean that the formula of discounted reward should not be used in agent design. On the contrary, discounted reward and the techniques that derive from them (such as Q-learning) could work well in our measurement setting, but we should not use them as the external performance measure. In any case, we must devise tests that work with artificial agents but also with biological beings. This is one of the reasons that negative rewards are needed. Paraphrasing Hutter [14] about eternal agents being lazy, we can say that using cumulative positive rewards make agents hyperactive.

Our main concern, however, has been an opportunistic use of time. This problem does not exist when using discrete-time agents and it is not common in evaluation, especially outside the areas of control and robotics, where the goals and measurements are quite different. The two adjustment proposals on the average try to solve the stopping problem. Since the time modulation problem cannot be solved in general in this setting, we have to consider that the use of times is licit and it might also be a sign of performance.

The main application of our proposal is for measuring performance in a broad range of contexts or environments which, according to [16], boils down to measuring intelligence. The setting which is presented here is necessary for an anytime intelligence test [8], where the evaluation can be stopped anytime, and the results should be better the more time we have for the test. And this links with the issue of reliability. Since we are working on averages, we should study the reliability as a function of the number of cycles, especially to take into account that an evaluation with only one cycle is very unreliable. However, here, we think that any classical statistical dispersion measure here would be an option, without the need of any further modification over the definition of $\breve{v}$.

Finally, as future work, we think that the use of continuous-time environments must be investigated, especially when other agents can play inside the environment. This is typical in multi-agent systems, and reaction times are crucial. The problem here is to determine the rate of the system, because it can be too fast for some agents and too slow for others. Additionally, if we want to evaluate only one agent, we have to consider the speeds of the other agents. Another future question is to analyse the case when τ is known by the agent. In this case, the pressure over time modulation policies is higher, but typically delays can take place before the last action (very close to the limit τ) when the last action is expected to be good.

7 Acknowledgments

The author thanks some comments from Sergio España on the issue of "hyperactive" agents, and M.Jose Ramirez and Cesar Ferri for comments on definition 3 and theorem 1. I am also grateful to David L. Dowe for reading a draft of this paper and spotting several mistakes and typos. The author also thanks the funding from the Spanish Ministerio de Educación y Ciencia (MEC) for projects EXPLORA-INGENIO TIN2009-06078-E, CONSOLIDER-INGENIO 26706 and TIN 2007-68093-C02, and GVA project PROMETEO/2008/051.

References

1. M. Agrawal, N. Kayal, and N. Saxena. Primes is in p. *Annals of Mathematics*, 160(2):781–793, 2004.
2. D.P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, Belmont, MA, 1995.
3. Y. S. Chow and H. Robbins. On optimal stopping rules for s_n/n. *Illinois J.Math*, 9:444–454, 1965.
4. D. L. Dowe and A. R. Hajek. A non-behavioural, computational extension to the Turing Test. In *Intl. Conf. on Computational Intelligence & multimedia applications (ICCIMA'98), Gippsland, Australia*, pages 101–106, 1998.
5. D.L. Dowe and A.R. Hajek. A computational extension to the Turing test. In *Proceedings of the 4th Conference of the Australasian Cognitive Science Society, Newcastle, NSW*, 1997.
6. Thomas S. Ferguson. *Optimal Stopping and Applications.* Mathematics Department, UCLA, 2004. http://www.math.ucla.edu/∼tom/Stopping/Contents.html.
7. J. Hernández-Orallo. A (hopefully) non-biased universal environment class for measuring intelligence of biological and artificial systems. In M. Hutter et al., editor, *Artificial General Intelligence, 3rd Intl Conf*, pages 182–183. Atlantis Press, Extended report at http://users.dsic.upv.es/proy/anynt/unbiased.pdf, 2010.
8. J. Hernández-Orallo and D. L. Dowe. Measuring universal intelligence: Towards an anytime intelligence test. *Artificial Intelligence*, 174(18):1508 – 1539, 2010.
9. J. Hernández-Orallo and N. Minaya-Collado. A formal definition of intelligence based on an intensional variant of Kolmogorov complexity. In *Proceedings of the International Symposium of Engineering of Intelligent Systems (EIS'98)*, pages 146–163. ICSC Press, 1998.
10. José Hernández-Orallo. Beyond the Turing test. *Journal of Logic, Language and Information*, 9(4):447–466, 2000.
11. José Hernández-Orallo. Constructive reinforcement learning. *International Journal of Intelligent Systems*, 15(3):241–264, 2000.
12. José Hernández-Orallo. On the computational measurement of intelligence factors. In *Performance metrics for intelligent systems workshop*, pages 1–8. Gaithersburg, MD, 2000.
13. José Hernández-Orallo. Thesis: Computational measures of information gain and reinforcement in inference processes. *AI Communications*, 13(1):49–50, 2000.
14. Marcus Hutter. General discounting versus average reward. In Jose L. Balcazar, Philip M. Long, and Frank Stephan, editors, *ALT*, volume 4264 of *Lecture Notes in Computer Science*, pages 244–258. Springer, 2006.
15. L.P. Kaelbling, M.L. Littman, and A.W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence*, 4(1):237–285, 1996.
16. Shane Legg and Marcus Hutter. Universal intelligence: A definition of machine intelligence. *Minds and Machines*, 17(4):391–444, 2007. http://www.vetta.org/documents/UniversalIntelligence.pdf.

17. Sridhar Mahadevan. Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine Learning*, 22, 1996.
18. M.L. Puterman. *Markov Decision Processes.* Wiley Interscience, New York, USA, 1994.
19. Anton Schwartz. A reinforcement learning method for maximizing undiscounted rewards. In *ICML*, pages 298–305, 1993.
20. Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction.* The MIT Press, March 1998.
21. Prasad Tadepalli and Dokyeong Ok. Model-based average reward reinforcement learning. *Artif. Intell.*, 100(1-2):177–224, 1998.