On Evaluating Agent Performance in a Fixed Period of Time

José Hernández-Orallo
DSIC, Univ. Politècnica de València,
Camí de Vera s/n, 46020 Valencia, Spain. jorallo@dsic.upv.es

Abstract

The evaluation of several 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 intelligence evaluation. In reinforcement learning, the task is formalised as an interactive environment with observations, actions and rewards. 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. In this paper we revisit the classical reward aggregating functions which are commonly used in reinforcement learning and related areas, we analyse their problems, and we propose a modification of the average reward to get a consistent measurement for continuous time.

Introduction

Measuring agent intelligence is one of the pending subtasks (or requirements) in the goal of constructing general intelligent artefacts. (LH07) presents 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 (HOD09), 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 glance. 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 (SB98). But, in general, time (either discrete or continuous) is understood as a virtual time. Even in real applications, where continuous time appears, 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 (composed of several exercises is passed to several students. All exercises deal about the same (previously unknown) subject, so typically a good student would improve as s/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 (previously unknown) time limit τ is reached.

Consider a test taken in half an hour, where several students have got different results, as shown in Figure 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? From the figure, we can say that it has done all of them right from almost the beginning. We can also say it is very slow, but with only two attempts 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?

In 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 respectively) or average the results by the number of exercises (so we 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 a possible formula might be $v' = v \times \sqrt{n}/\tau$, giving values

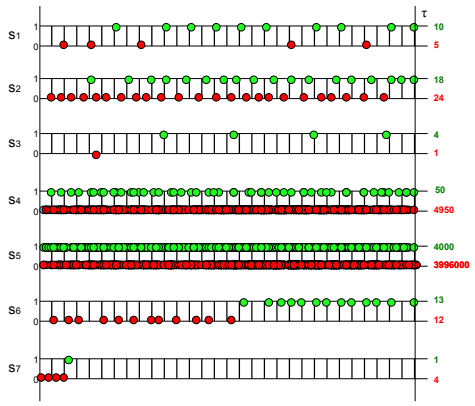

Figure 1: Several students evaluated in a fixed time.

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, making different choices when units change, or the measure gives too much weight to speed. Additionally, when $\tau \to \infty$ 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 on 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 s_3 with s_6 as shown in Figure 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).

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. 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 using fixed time slots are not reasonable, since we want to evaluate agents of virtually any speed, without making them wait. A similar (and simpler) approach is to set a maximum of cycles n instead of a time τ, but this makes testing almost unfeasible if we do not know the speed of the agent in advance (the test could last miliseconds 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, which can also be part of their policy.
- The environment must react immediately (no delay time computed on its side).
- 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 and rate r.
- The evaluation must be fair, avoiding opportunistic agents, which start with low performance to show an impressive improvement later on, or that stop acting when they get good results (by chance or not).

The main contribution of this work is that we revisit the classical reward aggregation (payoff) 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 a new modification of the average reward to get a consistent measurement for this case, where the agent not only decides an action to perform but also decides the time the decision is going to take.

Setting Definition and Notation

An environment is a world where an agent can interact through actions, rewards and observations. 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.

In our case, the sequence of events is exactly the same as discrete-time decision process. 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, and observations are also limited by a finite set O of possibilities. 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 $\widetilde{roa}_{\leq k-1}$. 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 to execute action a_k before a time delay d after the sequence of events or history $\widetilde{ro}_{\leq k}$. Note that the probability on d is cumulative. 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 its previous rewards and observations to determine its future actions and times, i.e. $t_{i+1} - t_i$ can depend on the previous experience.

Interactions between environments and agents can be interrupted at any time τ, known as the "overall or total test time". The value τ is unknown for any agent at any moment. With $n_\tau^\pi \mu$ (or just n_τ) we denote the number of interactions or cycles performed by π in μ in time τ. Let us see a very simple environment and agent:

Example 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 ($R = [0 \dots 1]$) 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 $r_1 o_1 a_1 r_2 o_2 a_2 \dots r_{k-1} o_{k-1} a_{k-1}$, 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: $0 C_1 B_1 0.1 C_2 B_1 0 C_1 B_1 0.1 \dots$ 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 $t_i = \frac{1}{10} i$.

Payoff and Environment Classes

Let us give the simplest notion of payoff:

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

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

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

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.

One attempt to solve this problem without abandoning the idea of summing rewards is the notion of reward-bounded (or summable) environment (LH07).

Definition 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 rewards usually require the arbitrary choice of a discounting function and a 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 (Hut06) for an extensive list of possible discounting functions). But this implies that the discount function is hardwired in 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 choose the discount function *externally* with a universal distribution over a universal machine generating discount functions.

In any case, be it internally hardwired in 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 that approach a uniform distribution or that depend on the agent's age, 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_{rand}^r should have the same expected

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

[2] In fact, in order to show the equivalence in the limit of the average reward and the discounted reward, (Hut06) infinite many cycles have to be removed from the start.

valued for every τ and this value should also be the same for every rate r.

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 rashly (hyperactive).

As an alternative to discounting and also to reward-bounded environments, and especially conceived to work well with any agent, in (HOD09) we propose this:

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

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

The construction of balanced environments is not difficult, even universal ones, as shown in (HO09a). 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".

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 payoff (or 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 (Mah96)(Ber95)(KLM96)(SB98).

Let us see some of them adapted to our continuous time limit setting. For instance, reward can be averaged in two different ways, by the number of cycles of the agent (average reward per cycle), or by the physical elapsed time (average reward per second). Since the second boils down to $\frac{1}{\tau} V_\mu^\pi \Uparrow \tau$ (so inheriting most of the problems of $V_\mu^\pi \Uparrow \tau$), we will just analyse the first.

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

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

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

Let us also 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 (Hut06). 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 $\Gamma^n_k := \sum_{i=k}^n \gamma_i < \infty$. If $k = 1$ we simply use Γ^n.

Hence, the discounted reward (per cycle) is:

Definition 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 (Hut06). As the very name says, all of them are discounting, so the first rewards contribute to the aggregated value much stronger than the rest. How much? That depends on the choice of γ. In any case, the result very dependent on the rate. For instance, 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 γ as 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).

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. But, more importantly, it can be *intentionally* modulated by the agent. Thus, 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. More generally, an agent could decide to stop, which implies stopping any further exploration but also any further reward.

First, we see the notion of "time modulation policy":

Definition 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).

A time modulation policy can make the agent stop on t_i (and, hence t_{i+1} is infinite). In our setting, a tricky (but good) policy here would be to act as a fast random agent until having an average reward over a threshold and then stop acting. We call this agent an opportunistic fast random agent. If the threshold is 0 this strategy ensures a positive reward in balanced environments[3]. 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 [4] *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 $\forall \tau \geq t \ : \ v_\mu^{\pi_{rand}} || \tau > 0$.*

[3] In fact, if only rewards -1 and 1 are possible, the expected reward is $0.79 \times 2 - 1 = 0.58$ (see (Fer04)).

[4] Due to space limitations, proofs are found in (HO09b).

A first (and naïve) idea to avoid stopping policies would be to give less weight to quick actions and more weight to slow actions. 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 quick decisions is more intuitive, but very fast mediocre agents can score well, and, additionally, it also suffers the problems of opportunistic time modulation. A better possibility is shown next:

Definition 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. 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 2 *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 $lim_{\tau\to\infty}\breve{v}_\mu^{\pi_{rand}}||\tau > 0$.*

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

Theorem 3 *For every balanced environment μ, a constant-rate random agent π_{rand} with any stopping policy has $lim_{\tau\to\infty}\breve{v}_\mu^{\pi_{rand}}||\tau = 0$.*

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

Lemma 4 *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 $\forall t_i \le t < t_i + t_d \;:\; R_\mu^{\pi_{rand}}(i) > E(R_\mu^{\pi_{rand}}(i+1))$.*

In other words, the payoff until $t_i + t_d$ not performing any action is greater than the expected payoff performing the following action. 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. Note that lemma 4 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 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.*

So we have realised that time modulations are impossible to avoid (only minimise). As a result, we will have to accept time modulation as part of the agent behaviour and needs to be considered in the measurement.

Comparison of Payoff Functions

After the analysis of several payoff functions adapted from the literature, and the introduction of a new variant with some associated results, it is necessary to recapitulate and give a comprehensive view. The setting we introduce in this paper is characterised by different response times on the side of the agent. These different response times could be motivated by different agent speeds or by an intentional use of delays.

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 of the testing period, etc. In what follows, we will examine the previous payoff functions according to several features, as shown in table 1.

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 11 'yes' from a total of 12.

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 choosing the right decision at interaction i has an increasing polynomial time complexity. 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 other 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.

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

Environment Type	General	Bounded	Balanced	General	Balanced	General	Balanced	Balanced
Score Function	$V_\mu^\pi \Uparrow \tau$	$V_\mu^\pi \Uparrow \tau$	$V_\mu^\pi \Uparrow \tau$	$v_\mu^\pi \|\|\tau$	$v_\mu^\pi \|\|\tau$	$V_\mu^\pi \|\gamma\|\tau$	$V_\mu^\pi \|\gamma\|\tau$	$\tilde{v}_\mu^\pi \|\|\tau$
1. Do random agents get a somehow central value (preferrably 0)?	No	No	Yes	No	Yes	No	Yes	Yes
2. Is the result of random agents independent from τ and the rate?	No	No	Yes	No	Yes	No	Yes	Yes
3. Is it avoided that a fast mediocre agent can score well?	No	No	No	Yes	Yes	No	No	Yes
4. Does the measurement work well when rates $\to \infty$?	No	No	No	Yes	Yes	No	No	Yes
5. Do better but slower agents score better than worse but faster agents?	No	No	No	Yes	Yes	*	*	Yes
6. Do faster agents score better than slow ones with equal performance?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
7. Are the first interactions as relevant as the rest?	Yes	No	Yes	Yes	Yes	No	No	Yes
8. Is the measure bounded for all τ?	No	Yes	No	Yes	Yes	Yes	Yes	Yes
9 .Does it work well when actions require more and more time to decide?	No	No	No	Yes	Yes	No	No	Yes
10.Is it robust against time stopping policies?	Yes	Yes	No	No	No	Yes	No	Yes
11.Is it robust against time modulation policies?	Yes	Yes	No	No	No	Yes	No	No
12.Is it scale independent (different time units)?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

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

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 agent 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 test 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 (Hut06): "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. When the lifetime τ is not known or is 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 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, 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 uncommon in evaluation, especially outside control and robotics, where the goals and measurements are different. The adjustment proposal on the average tries to solve the stopping problem.

The main application of our proposal is for measuring performance in a broad range of environments which, according to (LH07), boils down to measuring intelligence. The setting which is presented here is necessary for an anytime intelligence test (HOD09), where the evaluation can be stopped anytime, and the results should be better the more time we have for the test.

Finally, as future work, 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. 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.

Acknowledgments

The author thanks the funding from the Spanish Ministerio de Educación y Ciencia (MEC) for projects Explora-Ingenio TIN2009-06078-E, Consolider 26706 and TIN 2007-68093-C02, and Prometeo/2008/051 from GVA.

References

D.P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, Belmont, MA, 1995.

T. S. Ferguson. *Optimal Stopping and Applications*. Maths. Department, UCLA, 2004. http://www.math.ucla.edu/∼tom/Stopping/Contents.html.

J. Hernández-Orallo. A (hopefully) unbiased universal environment class for measuring intelligence of biological and artificial systems. Extended Version. *available at http://users.dsic.upv.es/proy/anynt/*, 2009.

J. Hernández-Orallo. On evaluating agent performance in a fixed period of time. Extd. Version. *available at http://users.dsic.upv.es/proy/anynt/*, 2009.

J. Hernández-Orallo and D. L. Dowe. Measuring universal intelligence: Towards an anytime intelligence test. *Under Review, available at http://users.dsic.upv.es/proy/anynt/*, 2009.

M. Hutter. General discounting versus average reward. In *ALT*, volume 4264 of *LNCS*, pages 244–258. Springer, 2006.

L.P. Kaelbling, M.L. Littman, and A.W. Moore. Reinforcement learning: A survey. *J. Artif. Intel.*, 4(1):237–285, 1996.

S. Legg and M. Hutter. Universal intelligence: A definition of machine intelligence. *Minds & Mach.*, 17(4):391–444, 2007.

S. Mahadevan. Average reward reinforcement learning: Foundations, algorithms, empirical results. *Mach. Learn.*, 22, 1996.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, March 1998.