Artificial Intelligence 174 (2010) 1508–1539

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Measuring universal intelligence: Towards an anytime intelligence test

José Hernández-Orallo [a,*], David L. Dowe [b]

[a] Departament de Sistemes Informàtics i Computació, Universitat Politècnica de València, Camí de Vera s/n, E-46022, València, Spain
[b] Computer Science & Software Engineering, Clayton School of I.T., Monash University, Clayton, Victoria, 3800, Australia

ARTICLE INFO

Article history:
Received 16 December 2009
Received in revised form 24 September 2010
Accepted 24 September 2010
Available online 29 September 2010

Keywords:
Measurement of intelligence
Artificial intelligence
Psychometrics
Algorithmic information theory
Kolmogorov complexity
Algorithmic probability
Turing test
Universal intelligence
Computerized adaptive testing
Compression
Inductive inference
Prediction
Minimum Message Length (MML)
Reinforcement learning

ABSTRACT

In this paper, we develop the idea of a universal anytime intelligence test. The meaning of the terms "universal" and "anytime" is manifold here: the test should be able to measure the intelligence of any biological or artificial system that exists at this time or in the future. It should also be able to evaluate both inept and brilliant systems (any intelligence level) as well as very slow to very fast systems (any time scale). Also, the test may be interrupted at any time, producing an approximation to the intelligence score, in such a way that the more time is left for the test, the better the assessment will be. In order to do this, our test proposal is based on previous works on the measurement of machine intelligence based on Kolmogorov complexity and universal distributions, which were developed in the late 1990s (C-tests and compression-enhanced Turing tests). It is also based on the more recent idea of measuring intelligence through dynamic/interactive tests held against a universal distribution of environments. We discuss some of these tests and highlight their limitations since we want to construct a test that is both general and practical. Consequently, we introduce many new ideas that develop early "compression tests" and the more recent definition of "universal intelligence" in order to design new "universal intelligence tests", where a feasible implementation has been a design requirement. One of these tests is the "anytime intelligence test", which adapts to the examinee's level of intelligence in order to obtain an intelligence score within a limited time.

1. Introduction

Artificial intelligence has arguably given preference to the goal of developing intelligent systems over the goal of evaluating them. The evaluation of artificial intelligence has almost always been addressed in the context of what artificial intelligence should be, or even in the context of whether it was possible. In fact, we are uncertain of how much the field of artificial intelligence has progressed in precise measurable terms since its inception. However, the eventual achievement of true and general artificial intelligence does not entail, per se, that intelligence can be evaluated. Nor does a *definition* of intelligence entail that intelligence can be *evaluated* with it. From an engineering point of view, the most desirable sequence of events would be to have a definition, then a feasible measurement procedure, and from these, to build intelligent systems that could be ultimately certified and evaluated with the measurement procedure.

The evaluation and measurement of intelligence is related to the notion of *test*, which is the daily basis of psychometrics; however, it is also the approach taken by the so-called Turing test [1], its variants, and extensions. In artificial intelligence, the major concern has been more about when and how our artefacts will be able to pass any of these tests, and not as

* Corresponding author.
E-mail addresses: jorallo@dsic.upv.es (J. Hernández-Orallo), david.dowe@infotech.monash.edu.au (D.L. Dowe).

0004-3702/$ – see front matter
doi:10.1016/j.artint.2010.09.006

much about how well our artificial intelligence systems will score in them. In the best case, evaluation is performed as a competition among several contenders, but even the results of two consecutive contest editions are not comparable.

From a scientific point of view, we could instead think about the desiderata for an intelligence measurement procedure, and then, with an engineer's perspective, see whether these desiderata are physically attainable. In other words, a scientifically ambitious view of a measurement of intelligence should have the following properties:

- It must be "universal" in the sense that it cannot be designed to only measure or favour some particular kind of intelligent system. It must allow us to measure any kind of intelligent system (biological or computational).
- It must be derived from well-founded computational principles with precise formalisations of the concepts involved in the measurement such as environment, action, goal, score, time, etc.
- It should be meaningful in the sense that what is being measured accounts for the most general notion of intelligence.
- It must be able to evaluate and score any present intelligent system or any system that may be built in the future, thereby allowing the comparison between different generations of intelligent systems up to and beyond human intelligence.
- The measurement should handle any level of intelligence and any time scale of the system. It must be able to evaluate inept and brilliant systems (any intelligence level) as well as very slow to very fast systems (any time scale).
- The quality (precision) of the assessment should be tuneable and should mostly depend on the time that is provided for the measurement. The evaluation can be interrupted at any time, producing an approximation to the intelligence score. The longer the evaluation time, the better the assessment (anytime test).

It is, of course, questionable whether a measurement that follows the above requirements is even possible since no measurement or test of intelligence presented to date fulfils all of these requirements. In fact, most of them do not take any of them into account.

This paper presents the first general and feasible test of intelligence, which can be valid for both artificial and biological systems, of any intelligence degree and of any speed. The test is not anthropomorphic; it is gradual, anytime, and is exclusively based on computational notions, such as Kolmogorov complexity [2]. It is also meaningful since it averages the capability of succeeding in different environments.

In order to do this, we build upon previous works on measuring or defining intelligence [3–9] based on the notions of inductive inference, prediction, compression, and randomness, all of which are properly formalised and understood in the context of Algorithmic Information Theory, Kolmogorov complexity, Occam's razor or the Minimum Message Length (MML) principle [2,10–12]. Our starting point is the same as in [9]; we set up the evaluation in a framework where an agent interacts with a set of environments. This is a very simple and natural idea since intelligence can be seen as good performance in a variety of environments. This also looks appealing, that is, if we do not care about the number of environments and the amount of time that is required for each of them. However, if we want the evaluation to be reliable with a small sample of environments and a small number of interactions in each of them, then we must confront several problems. We need to establish some conditions over the environments. The environments must be discriminative; they must react immediately; and they must be balanced (in terms of the expected score) in order to ensure that their aggregation is appropriate. We need to define a proper and computable measure of complexity for environments, and we need to design a way to derive a sample of these environments that adjusts to the subject's level of intelligence.

In what follows, we analyse the limitations of the definition of "universal intelligence" by Legg and Hutter [9]. We also introduce many new ideas that develop previous "compression tests" [3–5,13], "comprehension tests" [7], and the definition of Universal Intelligence [9] into a new "anytime intelligence test", with its actual implementation in mind. One of the key concepts that is worked out in this paper is the relation between time and intelligence and how to incorporate time into the measurement. As a result, this paper includes four operative intelligence test definitions: one that does not include time and is not anytime (but solves many other previous issues); one that does include time but is not anytime; one that does not include time and is anytime; and one that includes time and is anytime. A summary can be found in Table 4 at the end of this paper.

At the theoretical level, intelligence is defined as an average of performance in many environments, where complex environments are more relevant than in Legg and Hutter's definition. An adaptive test is not only a good thing in practice to get a reliable score in less time, but it is also a way out of the dilemma of choosing between simple and complex environments and the time required in each of them. At a practical level, we show some examples of what these tests should look like in practice. The tests are designed to evaluate the abilities of purported intelligent agents (artificial, human, non-human animal or extraterrestrial) in a feasible way.

Although the general idea is to measure general intelligence (which is accomplished if unbiased universal machines are used to generate the set of environments), the tests we introduce here can be used to evaluate more specialised capabilities by appropriately choosing a class of environments. For instance, if environments were chosen so that actions only affect rewards but not observations (passive environments), the tests would be able to evaluate sequence prediction capability (ignoring planning capabilities). Similarly, by using different classes of environments, performance on several tasks (classification, mazes, board games, etc.) in many (if not all) the areas in artificial intelligence could be efficiently evaluated. In Section 6, we present five examples with their corresponding environment classes to evaluate the following: very simple inference capabilities; performance in environments generated by finite state machines; performance in playgrounds where

some other agents can co-exist in the environment (typical in mazes and other kinds of games); performance in a more general class with a universal generation of spaces, agents and behaviours; and performance in a class of game environments based on the AAAI game competition and the game description language used therein.

The rest of the paper is organised as follows. In Section 2, we present a brief overview of previous approaches for the measurement of intelligence in the areas of psychometrics, comparative cognition, and artificial intelligence. We also include some general notions and notation about agents and environments that will be used throughout the rest of the paper. Section 3 starts with a short introduction to the basic concepts of Algorithmic Information Theory, Kolmogorov complexity, and related areas in order to pave the way for the explanation of the closest precedents of this work, i.e., intelligence definitions or measurements based on Kolmogorov complexity and related notions. We also re-visit Legg and Hutter's definition. Section 4 discusses several problems found in this definition, leading to the proposal of solutions for the selection of environments, the use of practical interactions, and the aggregation of the measurement into a single score. Section 5 analyses the relation between time and intelligence, how this affects rewards, and it introduces the anytime algorithm (considering time or ignoring it). Section 6 presents several examples of some interesting environment classes, from simple and specialised cases to more general scenarios. Section 7 discusses the main features of the new tests and comments on their implementation and applicability. Finally, Section 8 closes the paper with the implications of this work and directions for future work.

2. Background

In this section we summarise the many perspectives of intelligence measurement, including human, animal, and machine intelligence. Although these come from very different disciplines, we recognise some ingredients that are common to all of them, such as agents, environments, and rewards. As a result, we introduce some notation and concepts around these issues using the terminology and uses in artificial intelligence (especially from reinforcement learning). We also re-visit some characterisations that have previously been given about the way tasks, environments, and agents should be.

2.1. Intelligence measurement from different perspectives

Psychometric tests have a long history [14]; they are effective, easy to administer, fast, and quite stable when used on the same (human) individual over time. In fact, they have provided one of the best practical definitions of intelligence: "intelligence is what is measured by intelligence tests". However, psychometric tests are anthropomorphic; they cannot evaluate the intelligence of systems other than Homo sapiens. They are also static and are based on a time limit. New approaches in psychometrics such as Item Response Theory (IRT) allows for item selection based on their cognitive demand features, providing results for understanding what is being measured and adapting the test to the level of the individual being examined. Items generated from cognitive theory and analysed from IRT are a promising tool, but these models have not been fully implemented in mainstream testing [15].

These and other efforts have attempted to establish "a priori" what an intelligence test should be (e.g. [16]) and then find adaptations for different kinds of subjects. However, in general, we need different versions of the psychometric tests to evaluate children at different ages and to evaluate adults with varying pathologies since the psychometric tests for sane adult Homo sapiens rely on certain skills and knowledge.

The same thing occurs for other animals. Comparative psychologists and other scientists in the area of comparative cognition usually devise specific tests for different species. An example of these specialised tests for children and apes can be found here [17]. It has also been shown that psychometric tests do not work for machines at the current stage of progress in artificial intelligence [13] since they can be cheated upon by relatively simple and specialised computer programs. However, the main drawback of psychometric tests for evaluating subjects other than humans is that there is no mathematical definition behind them.

The first machine intelligence tests were first proposed by Alan M. Turing [1], who developed the imitation game (commonly known as Turing test) [18]. In this test, a system is considered intelligent if it is able to imitate a human (i.e., to be indistinguishable from a human) during a period of time and subject to a (tele-text) dialogue with one or more judges. Although it is still broadly accepted as a reference to eventually check whether artificial intelligence will reach the intelligence of humans, it has long generated debates. Of course, many variants and alternatives have been suggested [18]. The Turing test and related ideas present several problems as a machine intelligence test: the Turing test is anthropomorphic (it measures humanity, not intelligence); it is not gradual (it does not give a score); it is not practical (it is increasingly easy to cheat and requires a long time to get reliable assessments); and it requires a human judge.

A recent and singular approximation to a machine intelligence test is what is called a CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) [19,20]. CAPTCHAs are any kind of simple question that can be easily answered by a human but not by current artificial intelligence technology. Typical CAPTCHAs are character recognition problems where letters appear distorted. These distortions make it difficult for machines (bots) to recognise the letters. The immediate objective of a CAPTCHA is to tell humans and machines apart. The ultimate goal is to prevent bots and other kind of machine agents or programs from being able to create accounts, post comments or other tasks that only humans should do. The problem with CAPTCHAs is that they are becoming more and more difficult for humans, since bots are being specialised and improved to read them. Whenever a new CAPTCHA technique is developed, new bots appear that

have chances of getting through the test. This forces CAPTCHA developers to change them again, and so on and so forth. Although CAPTCHAs work reasonably well today, in about 10 or 20 years, they will need to make things so difficult and general, that humans will require more time and several attempts in order to resolve them.

There are, of course, many other less well-known proposals for intelligence measurement, most of which are informal or merely philosophical, and none of which have been put into practice. From an engineering point of view, there have also been many proposals. We can most prominently highlight the series of workshops on Performance Metrics for Intelligent Systems (see, e.g., [21]) held since 2000, which typically address very limited or specialised scenarios, systems, or applications. And, finally, there has been a vague and a catch-all use of the term *Machine Intelligence Quotient* (MIQ). It has been used in different ways in [22–24], especially in the area of fuzzy systems, but without a precise definition. In any case, the very notion of MIQ is inappropriate because the *quotient*[1] in psychometrics is obtained from a population of individuals from a species at a given stage of its development (child, adult), which is possible for humans, but it makes no sense when there is no notion of *species* or homogeneous sample for artificially intelligent systems.

2.2. Environments, agents and scores

Although very different from each other, there are three dimensions where any approach for intelligence measurement has to make choices. In every intelligence assessment setting, there is (1) a subject or agent to be examined, (2) a set of problems, tasks, or environments, and (3) a protocol for the application of the measurement and the derivation of one or more performance scores. Depending on the assumptions or constraints placed on each of these issues, we have, as a result, a different assessment framework. Nevertheless, in the artificial intelligence literature, focus has been primarily given to (1) and (2), especially in the area of cognitive architectures where, implicitly or explicitly, a set of constraints or requirements is established for agents and environments.

For instance, Soar [25,26], one of the most successful and well-known cognitive architectures in artificial intelligence, implicitly assumes some features about agents and environments. Some of these assumptions become explicit requirements in [27], where the relation between intelligence requirements and cognitive architecture requirements are elucidated. More precisely, this work outlines eight characteristics of the environments, tasks, and agents that are deemed to be important for human-level intelligence, from which they derive twelve requirements for general cognitive architectures. Here we are not interested in the requirements for the architecture, i.e., for building intelligent agents. However, the characteristics for the environments, tasks, and agents are a good basis for the requirements on agents and environments that have appeared in the literature in the last fifty years. The characteristics can be summarised as follows: environments must be dynamic, reactive, and complex. They must contain diverse interacting objects, as well as some other agents that affect performance. Tasks must be complex, diverse, and novel. There must be regularities at multiple time scales. Agent perception may be limited, and the environment may be partially observable. Agents must be able to respond quickly to the dynamics of the environment, but agent computation resources must be considered limited. Finally, agent existence is long-term and continual, so it must balance immediate task or reward completion with more long-term goals. None of the measurement approaches mentioned in Section 2.1 follow these requirements.

The notion of agent is nowadays mainstream in artificial intelligence, and, apart from the previous (or other) requirements, it does not deserve further clarification. The distinction between goal, task, and environment is a more complex issue since it depends on the intention and will of the agent. In child psychometrics and animal comparative cognition, intelligence tests cannot assume that we are able to program certain goals or to explicitly explain a task to the examinee. Consequently, testing is generally performed using rewards, which is a conditioning approach to make subjects indirectly focus on a task. Interestingly, this is the same approach taken in reinforcement learning [28]. Although reinforcement learning is typically seen as a formalisation of this setting in the context of artificial intelligence and machine learning, it can also be viewed from a broader perspective since reinforcement learning is the study of how animals and artificial systems optimise their behaviour conditioned by rewards and punishments.

The most general picture of this setting is the interaction between an agent and an environment through actions, rewards, and observations. This is also similar to the setting typically used in control or system theory, which is outlined in Fig. 1.

Actions are limited by a finite set of symbols $\mathcal{A}$ (e.g. $\{left, right, up, down\}$); rewards are taken from any subset $\mathcal{R}$ of rational numbers between 0 and 1; and observations are also limited by a finite set $\mathcal{O}$ of possibilities (e.g., a grid of binary cells of $n \times m$ and the objects therein, 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 i (or, more loosely, state). Rewards and observations are the outputs of the environment. A pair $\langle r_i, o_i \rangle$ is also known as a perception. The order of events is always reward, observation and action.[2] Therefore, a sequence of events is a string such as $r_1 o_1 a_1 r_2 o_2 a_2 \cdots$. Both the agent and the environment are

[1] Originally, the quotient was a real quotient of a child's estimated mental age over their biological age. Nowadays, the 'Q' in IQ is a historical relic; the quotient is a normalised score according to the scores of a population, typically assuming a normal distribution with mean equal to 100. This normalisation is much harder to do with machines, as it is not clear which machines to average over. So, it is this latter *relative* interpretation of intelligence quotient and its application to machines which we criticise here.

[2] Typically, in artificial intelligence and in the cognitive sciences, the sequence of events is observation, action and reward. Legg and Hutter [9], however, use the sequence observation, reward, and action because the pair ⟨observation, reward⟩ is seen as a whole (the "perception"). The reason is that grouping environment outputs simplifies the formal definition of environment interaction as a sequence of perception–action pairs. In this paper, we use

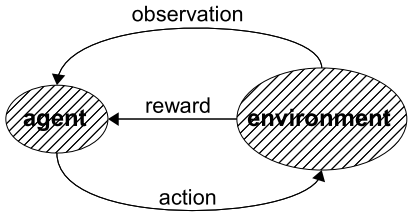

Fig. 1. Interaction between an agent and an environment [9].

defined as a probabilistic measure. For instance, given an agent, denoted by π, the term $\pi(a_k|r_1o_1a_1r_2o_2a_2\cdots r_ko_k)$ denotes the probability of agent π executing action a_k after the sequence of events $r_1o_1a_1r_2o_2a_2\cdots r_ko_k$. In a similar way, an environment μ is also a probabilistic measure that assigns probabilities to each possible pair of observation and reward. For instance, $\mu(r_ko_k|r_1o_1a_1r_2o_2a_2\cdots r_{k-1}o_{k-1}a_{k-1})$ denotes the probability in environment μ of outputting r_ko_k after the sequence of events $r_1o_1a_1r_2o_2a_2\cdots r_{k-1}o_{k-1}a_{k-1}$. Note that if for every cycle there is one action/perception with probability 1 (and 0 for the rest), we then have a deterministic agent/environment. If we combine the probability measures for agent and environment we have a probability measure for the interaction history or sequence. Interaction histories will be deterministic (respectively, computable) if both agent and environment are deterministic (respectively, computable). A sequence of actions can be denoted by $a_{1:n} = a_1a_2\cdots a_n$ where $a_i \in \mathcal{A}$. A sequence of actions determines (and hence can also denote) a *state* in a deterministic environment. We will denote by $a_i^{\mu,\pi}$, $r_i^{\mu,\pi}$ and $o_i^{\mu,\pi}$ the action, reward, and observation at interaction or cycle i for environment μ and agent π.

Example 1. Consider a test setting where a chimpanzee (the agent) can press one of three possible buttons ($\mathcal{A} = \{B_1, B_2, B_3\}$). Rewards are just the handing over (or not) of a banana ($\mathcal{R} = \{0, 1\}$) and the observation set is derived from three cells where a ball must be inside exactly one of them ($\mathcal{O} = \{C_1, C_2, C_3\}$). An example of a possible environment is defined by the following rules about observations and rewards. The observation o_k is randomly generated with a uniform distribution between the three possibilities in $\mathcal{O}$. The first reward is 1 (we start the game by giving a banana to the chimpanzee). The remaining rewards follow these properties:

$$\mu(r_k|r_1o_1a_1r_2o_2a_2\cdots r_{k-1}o_{k-1}a_{k-1})$$
$$= \begin{cases} 1, & \text{if } \begin{cases} \{(a_{k-1} = B_1 \text{ and } o_{k-1} = C_1) \text{ or } (a_{k-1} = B_2 \text{ and } o_{k-1} = C_2) \text{ or } (a_{k-1} = B_3 \text{ and } o_{k-1} = C_3)\} \\ \quad \text{and } (r_k = +1) \end{cases} \\ 1, & \text{if } \begin{cases} \neg\{(a_{k-1} = B_1 \text{ and } o_{k-1} = C_1) \text{ or } (a_{k-1} = B_2 \text{ and } o_{k-1} = C_2) \text{ or } (a_{k-1} = B_3 \text{ and } o_{k-1} = C_3)\} \\ \quad \text{and } (r_k = 0) \end{cases} \\ 0, & \text{otherwise} \end{cases}$$

According to this environment, a chimpanzee that always selects the button that corresponds to the cell where the ball is shown would score +1 reward (one banana) for each cycle. For example, if the environment shows C_2 and the chimpanzee presses B_2, then a banana is given.

Although this example is quite simple, the general setting shown in Fig. 1 using environments, agents, actions, observations, and rewards is powerful enough to represent the requirements stated in [27], as well as many intelligence measurement setting seen in Section 2.1 (except, perhaps, some types of psychometric tests, CAPTCHAs and the Turing test due to the absence of rewards).

In this setting, we must first think about a set of environments that are complex enough to meet the above requirements. In other words, we cannot assume any constraint on environments and observations if we want the setting to be general enough for an intelligence measurement. In reinforcement learning, in particular, many techniques assume that the environment is a Markov Decision Process (MDP). In other cases, however, it is assumed that environments are fully-observable, i.e., that there is a function between observations and states. We cannot assume this since many real-world problems are not fully-observable. This is especially the case in social contexts where other individuals may have different (and partial) views of the same situation.

In [29], a taxonomy of environments is developed that distinguishes among many kinds of environments. For instance, passive environments are those for which agents' actions can only affect rewards but not observations. A special subcategory is sequence prediction as used in classical psychometric tests, and also the typical classification problems in

an intermediate approach between the classical sequence of events used in artificial intelligence and Legg and Hutter's. We still maintain the sequence perception–action, but we change the order of the perception pair from ⟨observation, reward⟩ to ⟨reward, observation⟩. With this mostly irrelevant change, we finally have the same sequence of events that is typical in artificial intelligence (although with an initial reward starting the series). In any case, the order chosen is mostly a matter of taste and does not significantly affect the results in the rest of this paper.

machine learning. Some other kinds of environments are nth-order Markov Decision Processes (MDPs), where the next observation can only depend on the n last observations and the n last actions. It can be shown that nth-order MDPs can be reduced to first-order MDPs (or simply MDPs). In this case, it is natural to talk about "states", like many board games and some mazes, since the next reward and observation only depend on the previous observation and action (there is no 'perceptual aliasing' problem if the state is fully observable). Ergodic MDPs are MDPs of a special kind, which are characterised by making any possible observation reachable (in one or more cycles) from any state. That means that at any state, agents' actions can recover from a bad previous decision. We think this is a strong limitation. As many others advocate (e.g. [27]), we need to be as general as possible in the class of environments that is considered.

This is one of the issues that recurrently appears throughout the rest of this paper. We will try to consider the most general concept (or class) of environment and the most general concept of agent in order to allow a universal test that will permit a variety of contexts and be applicable to any kind of agent. For the moment, we only assume that environments are infinite (i.e. no sequence of actions makes them stop) and that they are model-based (i.e., we have a description, program, or model behind them). We also assume that this model is computable.

Apart from the characteristics of environment and agent, when we address the issue of evaluating the agent, it is necessary to determine whether any constraint on the reward distribution is needed, and, most especially, how rewards are aggregated. In reinforcement learning, several aggregation or payoff functions have been defined. For instance, the most common way of evaluating the performance of an agent π in an environment μ is to calculate the expected value of the sum of all the rewards, i.e.:

Definition 1 *(Expected cumulative reward).*

$$V_{\mu}^{\pi} := E\left(\sum_{i=1}^{\infty} r_i^{\mu,\pi}\right)$$

This is not the only option, since we have to determine whether more relevance is given to the immediate rewards or to long-term rewards by using different types of weighting or discounting, in order to make up for greedier or more explorative policies. This is related to the (expected) life of the agent (the number of interactions allowed) and also whether there is a bound on the rewards [30].

3. Defining and measuring intelligence using algorithmic information theory

In this section, we present a short introduction to the area of Algorithmic Information Theory and the notions of Kolmogorov complexity, universal distributions, Levin's Kt complexity, and its relation to the notions of difficulty, compression, randomness, the Minimum Message Length (MML) principle, prediction, and inductive inference. Then, we will survey the approaches that have appeared using these formal notions in order to give mathematical definitions of intelligence or to develop intelligence tests from them, starting from the compression-enhanced Turing tests, the C-test, and Legg and Hutter's definition of Universal Intelligence.

3.1. Kolmogorov complexity, universal distributions and inductive inference

Algorithmic Information Theory is a field in computer science that properly relates the notions of computation and information. The key idea is the notion of the Kolmogorov complexity of an object, which is defined as the length of the shortest program p that outputs a given string x over a machine U. Formally,

Definition 2 *(Kolmogorov complexity).*

$$K_U(x) := \min_{p \text{ such that } U(p)=x} l(p)$$

where $l(p)$ denotes the length in bits of p and $U(p)$ denotes the result of executing p on U.

For instance, if $x = 1010101010101010$ and U is the programming language Lisp, then $K_{Lisp}(x)$ is the length in bits of the shortest program in Lisp that outputs the string x. The relevance of the choice of U depends mostly on the size of x. Since any universal[3] machine can emulate another, it holds that for every two machines U and V, there is a constant $c(U, V)$, which only depends on U and V and does not depend on x, such that for all x, $|K_U(x) - K_V(x)| \leqslant c(U, V)$. The value of $c(U, V)$ is relatively small for sufficiently long x.

From Definition 2, we can define the universal probability for machine U as follows:

[3] A universal machine is any computer that is able to calculate any computable function, also known as Turing-complete. Turing machines, lambda calculus (or λ-calculus), re-writing systems and most programming languages are theoretical examples of universal machines. Any real computer is also a (memory-bounded) universal machine.

Definition 3 *(Universal distribution).* Given a prefix-free machine[4] U, the universal probability of string x is defined as:

$$p_U(x) := 2^{-K_U(x)}$$

which gives higher probability to objects whose shortest description is small and gives lower probability to objects whose shortest description is large. When U is universal, this distribution is similar (up to a constant difference) to the universal distribution for any other different universal machine, since one can emulate the other (given a translation program of finite length independent of the target string, x). Considering programs as hypotheses in the hypothesis language defined by the machine, this paves the way for the mathematical theory of inductive inference and prediction. This theory was developed by Solomonoff [31], formalising Occam's razor in a proper way for prediction, by stating that the prediction maximising the universal probability will eventually discover any regularity in the data. This is related to the notion of Minimum Message Length for inductive inference [10–12,32] and is also related to the notion of data compression.

The notions of *prediction* and *induction* are closely related, but not identical, since a prediction can be obtained by a (e.g. Bayesian) combination of *several* plausible models, while induction usually aims at discovering the most plausible model and it usually entails some explanation of the observations.[5] However, these notions are frequently used as synonyms. In fact, Solomonoff's seminal paper [31] refers to the "theory of inductive inference" while, under our interpretation (and also by some of his subsequent publications, see e.g. [33]), it would refer to a "theory of prediction" (or, perhaps, the inductive inference of a probability distribution). Moreover, there are also important differences between one-part compression and two-part compression (MML induction). In the former, the model does not distinguish between pattern and exceptions while the latter explicitly separates regularities (main pattern) from exceptions. See [11, Section 8], [12, Section 10.1] and [32] (part of Section 0.3.1 referring to Solomonoff) for more details on this.

One of the main problems of Algorithmic Information Theory is that Kolmogorov complexity is uncomputable. One popular solution to the problem of computability of K for finite strings is to use a time-bounded or weighted version of Kolmogorov complexity (and, hence, the universal distribution which is derived from it). One popular choice[6] is Levin's Kt complexity [2,36]:

Definition 4 *(Levin's Kt complexity).*

$$Kt_U(x) := \min_{p \text{ such that } U(p)=x} \left\{ l(p) + \log time(U, p, x) \right\}$$

where $l(p)$ denotes the length in bits of p, $U(p)$ denotes the result of executing p on U, and $time(U, p, x)$ denotes the time[7] that U takes executing p to produce x.

Finally, despite the uncomputability of K and the computational complexity of its approximations, there have been some efforts to use Algorithmic Information Theory to devise optimal search or learning strategies. Levin (or universal) search [36] is an iterative search algorithm for solving inversion problems based on Kt, which has inspired other general agent policies such as Hutter's AIXI, an agent that is able to adapt optimally[8] in all environments where any other general purpose agent can be optimal [37], for which there is a working approximation [38,39].

3.2. Developing mathematical definitions and measures of intelligence

Following ideas from A.M. Turing, R.J. Solomonoff, E.M. Gold, C.S. Wallace, M. Blum, G. Chaitin and others, between 1997 and 1998 some works on enhancing or substituting the Turing Test by inductive inference tests were developed, using Solomonoff prediction theory [31] and related notions, such as the Minimum Message Length (MML) principle. On the one hand, Dowe and Hajek [3–5] suggested the introduction of inductive inference problems in a somehow *induction-enhanced* or *compression-enhanced* Turing test [1] (they arguably called it non-behavioural) in order to, among other things, completely

[4] For a convenient definition of the universal probability, we need the requirement of U being a prefix-free machine (see, e.g. [2] for details). Note also that even for prefix-free machines there are infinitely many other inputs to U that will output x, so $p_U(x)$ is a strict lower bound on the probability that U will output x (given a random input).

[5] For example, in physics, we have several competing theories (Ptolemy, Newton, Einstein special relativity, relativistic quantum mechanics, Einstein general relativity, etc.). For the purposes of prediction, we can do a weighted average over these. But such a mixture is not a model or explanation of the laws of nature — rather, it is only pragmatically suited for prediction. See [12, Chapter 10].

[6] There are other options, as [34] also suggests, such as fixing a bound on time (but this would require setting an arbitrary constant), logical depth [2], the speed prior [35] or through the use of redundant Turing machines [32, Section 0.2.7].

[7] Here *time* does not refer to physical time but to computational time, i.e., computation steps taken by machine U. This is important, since the complexity of an object cannot depend on the speed of the machine where it is run.

[8] Optimality has to be understood in an asymptotic way. First, because AIXI is uncomputable (although resource-bounded variants have been introduced and shown to be optimal in terms of time and space costs). Second, because it is based on *a* universal probability over *a* machine, and this choice determines a constant term which may very important for small environments.

$k = 9$: a, d, g, j, ... Answer : m
$k = 12$: a, a, z, c, y, e, x, ... Answer : g
$k = 14$: c, a, b, d, b, c, c, e, c, d, ... Answer : d

Fig. 2. Examples of series of Kt complexity 9, 12, and 14 used in the C-test [7].

Table 1
Intelligence tests in passive and active environments (from [9]).

	Universal agent	Universal test
Passive environment	Solomonoff induction	C-test
Active environment	AIXI	Universal intelligence

dismiss Searle's Chinese room [40] objection, and also because an inductive inference ability is a necessary (though possibly "not sufficient") requirement for intelligence.

Quite simultaneously and similarly, and also independently, in [6,7], intelligence was defined as the ability to comprehend, giving a formal definition of the notion of comprehension as the identification of a 'predominant' pattern from a given evidence, derived from Solomonoff prediction theory concepts, Kolmogorov complexity and Levin's Kt. The notion of comprehension was formalised by using the notion of "projectible" pattern, a pattern that has no exceptions (no noise), so being able to explain *every* symbol in the given sequence (and not only most of it).

From these definitions, the basic idea was to construct a test as a set of series whose shortest pattern had no alternative projectible patterns of similar complexity. That means that the "explanation" of the series had to be much more plausible than other plausible hypotheses. The main objective was to reduce the subjectivity of the test — first, because we need to choose one reference universal machine from an infinite set of possibilities; secondly, because, even choosing one reference machine, two very different patterns could be consistent with the evidence and if both have similar complexities, their probabilities would be close, and choosing between them would make the series solution quite uncertain. With the constraints posed on patterns and series, both problems were not completely solved but minimised.

The definition was given as the result of a *test*, called C-test [6], formed by computationally-obtained series of increasing complexity. The sequences were formatted and presented in a quite similar way to psychometric tests (see Fig. 2) and, as a result, the test was administered to humans, showing a high correlation with the results of a classical psychometric (IQ) test on the same individuals. Nonetheless, the main goal was that the test could eventually be administered to other kinds of intelligent beings and systems. This was planned to be done, but the work from [13] showed that machine learning programs could be specialised in such a way that they could score reasonably well on some of the typical IQ tests. This unexpected result confirmed that C-tests had important limitations and could not be considered universal, i.e., embracing the whole notion of intelligence, but perhaps only a part of it.

Since then, other intelligence definitions or tests using ideas from algorithmic information theory or compression theory have also been proposed (e.g. [41], whose Large Text Compression Benchmark eventually converged with Jim Bowery's C-Prize into the Hutter's Prize[9]). The main controversy has been around the assumption that intelligence is all about induction, prediction, and (one-part or two-part) compression. The case against this is that some people who are commonly considered intelligent only excel on some specific cognitive abilities. On the contrary, the rationale in favour is that in order to perform induction, memory is also required (to store, match, and retrieve observations and models), as well as deduction (to check internal and external model consistency with observations and other models) and some other more basic cognitive abilities. Consequently, many argue that if inductive inference is not the only factor for intelligence, at least it brings the greatest accolades (see, e.g. [13, Section 5.2] or [42, Section 7.3] for a discussion on this).

Nonetheless, a compression or induction test was already considered limited even as far back as its inception. A factorisation (and, hence, extension) of these inductive inference tests was outlined in order to explore which other abilities could shape a complete (and, hence, sufficient) test [43]. Additionally, in order to apply the test for systems with low intelligence, (still) unable to understand natural language, the proposal for a dynamic/interactive extension of the C-test was expressed like this: "the presentation of the test must change slightly. The exercises should be given one by one and, after each guess, the subject must be given the correct answer (rewards and penalties could be used instead)" [7].

Recent works by Legg and Hutter (e.g. [8,9]) have followed the previous steps and, strongly influenced by Hutter's theory of AIXI optimal agents [44], have given a new definition of machine intelligence, dubbed "universal intelligence", also grounded in Kolmogorov complexity and Solomonoff's ("inductive inference" or) prediction theory. The key idea is that the intelligence of an agent is evaluated as some kind of sum (or weighted average) of performances in all the possible environments.

The comparison with the works from Hernández-Orallo (and also Dowe and Hajek's compression test) is summarised by Legg and Hutter as shown in Table 1 (reproduced from [9]). In fact, the definition based on the C-test can now be

[9] This prize is actually called the "Prize for Compressing Human Knowledge" (http://prize.hutter1.net/).

Table 2
Intelligence tests in passive and active environments (clarification).

	Universal agent	Universal definition	Universal test
Passive environment	Solomonoff prediction	Comprehension ability based on C-test [7], inductive ability	C-test [6], induction-enhanced Turing test [3]
Active environment	AIXI	Universal intelligence	?

considered a static precursor[10] of Legg and Hutter's work, where the environment outputs no rewards, and the agent is not allowed to make an action until several observations are seen (the inductive inference or prediction sequence). The point in favour of active environments (in contrast to passive environments) is that the former not only require inductive and predictive abilities to model the environment but also some planning abilities to effectively use this knowledge through actions. Additionally, perceptions, selective attention, and memory abilities must be fully developed. Not all this is needed to score well in a C-test, for instance.

In our opinion, one of the most relevant contributions in [9] is that their definition of *universal intelligence* allows one to formally evaluate the theoretical performance of some agents: a random agent, a specialised agent, etc., or a super-intelligent agent, such as AIXI [44], which is claimed to be the agent that, if ever constructed, would score the best in the *universal intelligence test*.

Legg and Hutter devote a special section in [9] to address typical criticisms that a formal definition of intelligence based on ideas borrowed from Solomonoff prediction might raise. We basically agree with Legg and Hutter's defence of these criticisms because we have faced similar ones in the past. This also means that we refer the reader to [3–5,7,9,13]. Thus, we will not devote any further space in this paper to typical issues on the whys and whens of machine intelligence measurement, its relation to the Turing test, and other tests proposed so far.

Taking Legg and Hutter's definition of Universal Intelligence as a basis, in the quest for a refinement and improvement of their work (as their work can be seen as an interactive variant of ours), we must first address some issues that, in our opinion, may require a clarification or a correction and, once they are clarified, we will concentrate on developing an anytime *universal intelligence test*.

First of all, Table 1 from [9] should, in our opinion, be understood as shown in Table 2. That means, as Legg and Hutter admit, that their definition is not a practical test and, for the moment, cannot be used to evaluate intelligent agents. On the contrary, other tests [3–5,7] have shown that evaluation is feasible (with some limitations, as shown in [13]). In any case, [9] is a good example of the fact that having a definition of intelligence does not directly provide us with a means of measuring intelligence (i.e., an intelligence test). Nevertheless, here we claim the reverse is true: having a general, well-grounded, and formal intelligence *test* provides us with a definition of intelligence.

3.3. Definition of Universal Intelligence

We have just stated above that having a definition of intelligence does not directly provide us with an intelligence test. In order to see why Legg and Hutter's definition of Universal Intelligence is not valid for testing (even with several modifications), we need to take a look at the definition. Their definition is based on the reasonable idea that intelligence is performance in a variety of environments. However, this "variety" of environments is addressed by Legg and Hutter in a straightforward way: choose all of them. Then, given an agent π, its universal intelligence Υ (using the expected cumulative reward from Definition 1) can be given by the following definition:

Definition 5 *(Universal Intelligence [9]).*

$$\Upsilon(\pi, U) := \sum_{\mu=i}^{\infty} p_U(\mu) \cdot V_\mu^\pi = \sum_{\mu=i}^{\infty} p_U(\mu) \cdot E\left(\sum_{i=1}^{\infty} r_i^{\mu,\pi} \right)$$

where μ is any environment coded on a universal machine U, with π being the agent to be evaluated. With $p_U(\mu)$, we assign a probability to each environment, although these probabilities will not add to 1.

Since we have infinitely many environments, we cannot assign a uniform distribution to the environments. The solution is to use the universal distribution over a given machine U, as seen in Section 3.1 (Definition 3), by properly adapting the

[10] In fact, in Legg's Ph.D. dissertation (Section 3.1.7) [29], it is shown that sequence prediction (which is used in Hernández-Orallo's and Dowe and Hajek's works) is a special case of what they call chronological environments (which is used in Universal Intelligence). One important detail is that in the C-test, a universal distribution based on Kt (Levin's resource-bounded version of Kolmogorov complexity) is used instead of (the non-computable) K (original version of Kolmogorov complexity).

coding of environments as strings in U, and assuming that the class of environments is recursively enumerable (see p. 63 of [29] for details).

And, finally, since we have a sum of infinitely many environments, Legg and Hutter discuss several ideas to avoid the accumulated rewards V_μ^π being infinite. In their discussion, the recurrent issue of environments where agents can be greedy or more long-term far-sighted appears [30] (an issue which is also persistent in the area of reinforcement learning). But, in the end, they come up with a single constraint that they finally impose on every environment:

Definition 6 *(Reward-bounded environment).* An environment μ is reward bounded iff for all π:

$$V_\mu^\pi \leqslant 1$$

Example 1 in Section 2.2 does not conform to this limitation because a 1 reward can be obtained several times. However, Example 2 (below) is reward bounded:

Example 2. Consider a test setting where a robot (the agent) can press one of three possible buttons ($\mathcal{A} = \{B_1, B_2, B_3\}$), rewards are just a variable score ($\mathcal{R} = [0 \ldots 1]$) and the observation set derives from three cells where a ball must be inside one of them ($\mathcal{O} = \{C_1, C_2, C_3\}$). An example of a possible environment is:

$$\mu(r_k|r_1 o_1 a_1 r_2 o_2 a_2 \cdots r_{k-1} o_{k-1} a_{k-1}) = \begin{cases} 1, & \text{if } \begin{cases} \{(a_{k-1} = B_1 \text{ and } o_{k-1} = C_1) \text{ or } (a_{k-1} = B_2 \text{ and } o_{k-1} = C_2) \text{ or } (a_{k-1} = B_3 \text{ and } o_{k-1} = C_3)\} \\ \quad \text{and } (r_k = 1/2^{k-1}) \end{cases} \\ 1, & \text{if } \begin{cases} \neg\{(a_{k-1} = B_1 \text{ and } o_{k-1} = C_1) \text{ or } (a_{k-1} = B_2 \text{ and } o_{k-1} = C_2) \text{ or } (a_{k-1} = B_3 \text{ and } o_{k-1} = C_3)\} \\ \quad \text{and } (r_k = 0) \end{cases} \\ 0, & \text{otherwise} \end{cases}$$

The observation o_k in both cases is randomly generated with a uniform distribution between the three possibilities in $\mathcal{O}$. The first reward (r_1) is 0 (we start the game giving nothing to the agent). The robot has the behaviour of always pressing button B_1, i.e., $\pi(B_1|X) = 1$ for all sequences X. Consequently, the performance of the robot in this environment is:

$$V_\mu^\pi = E\left(\sum_{i=1}^{\infty} r_i^{\mu,\pi}\right) = r_1 + E\left(\sum_{i=2}^{\infty} r_i^{\mu,\pi}\right) = 0 + \frac{1}{3}\sum_{k=2}^{\infty}\frac{1}{2^{k-1}} = \frac{1}{3}\sum_{k=1}^{\infty}\frac{1}{2^k} = \frac{1}{3}$$

So, in this environment, rewards get smaller as k gets larger. This means that most of the overall reward depends on the first actions.

The numerical value given for each reward r_i is used to compute the overall expected reward V_μ^π, but physical rewards can be a function of each reward r_i, in order to keep the attention of the agent. For instance, for an ape, a banana can be given whenever $r_i > 0$, independently of the magnitude of r_i.

4. Addressing the problems of the definition of Universal Intelligence

Definition 5, although very simple, captures one of the broadest definitions of intelligence: "the ability to adapt to a wide range of environments". However, there are three obvious problems in this definition regarding making it practical. First, we have two infinite sums in the definition: one is the sum over all environments, and the second is the sum over all possible actions (agent's life in each environment is infinite). And, finally, K is not computable. Additionally, we also have the dependence on the reference machine U. This dependence takes place even though we consider an infinite number of environments. The universal distribution for a machine U could give the higher probabilities $(0.5, 0.25, \ldots)$ to quite different environments than those given by another machine V.

Despite all these problems, it seems that just making a random finite sample on environments, limiting the number of interactions or cycles of the agent with respect to the environment and using some computable variant of K, is sufficient to make it a practical test. However, on the one hand, this is not so easy, and, on the other hand, the definition has many other problems (some related and others unrelated).

4.1. On the difficulty of environments

The first issue concerns how to sample environments. Just using the universal distribution for this will mean that very simple environments will be output again and again. Note that an environment μ with $K(\mu) = 1$ will appear half of the time. Of course, repeated environments must be ruled out, but a sample would almost become an enumeration from low to high K. This will still omit or underweight very complex environments because their probability is so low. As an example, the (approximately) 16 environments of complexity that are less or equal to 4 would have about 95% of the weight and a

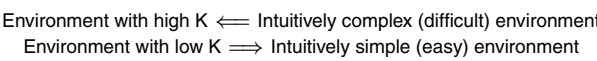

Fig. 3. Relation between K and intuitive complexity.

ridiculous 5% for the (overwhelming) rest, where more interesting things might happen. Furthermore, measuring rewards on very small environments will get very unstable results and be very dependent on the reference machine. And even ignoring this, it is not clear that an agent that solves all the problems of complexity that are lower than 20 bits and none of those that are more than 20 bits is more intelligent than another agent who does reasonably well on every environment. As an example, a genius who is not very adept at simple things would score badly.[11,12]

Before going on, we need to clarify the notions of simple/easy and complex/difficult that are used here. For instance, just choosing an environment with high K does not ensure that the environment is indeed complex.

Example 3. Consider, for instance, that a relatively simple environment μ_1 with high K has a behaviour with no pattern until cycle $i = 1000$ and then it repeats the behaviour from then on indefinitely. K will be high because 1000 cycles must be coded, but the pattern is relatively simple (a repetition) if the agent has a big memory and interacts for thousands of cycles.

Now consider a second environment μ_2 with high K that goes as follows: for any action, output the same observation o and reward r, except when the interaction i is a power of 2. In this case (and only in this case), the observation and reward depend on a quite complex formula over the previous actions. It is easy to see that in a much higher number of cycles the behaviour of the environment is simple while only in a few cycles the environment will require a complex agent's behaviour.

Finally, consider, e.g., an environment μ_3 with only two possible actions such that every sequence of actions leads to very simple "subenvironments" (subtrees in the decision tree), except one specific sequence of actions $a_1, a_2, \ldots, a_n$ which leads to a complex subenvironment. Logically, the complexity of this environment is high, but only 1 of 2^n action *combinations* will make the complex subenvironment accessible and visible for the agent. Consequently, with a probability of $(2^n - 1)/2^n$, the agent will see this environment μ_3 as very simple.

The issues raised with the first two environments μ_1 and μ_2 can also take place with strings, but the third environment μ_3 shows that the notion of easy or difficult is not the same for strings as for environments. In the case of an environment, an agent will only explore a (generally) small part of it. In general, the relation between *intuitive* complexity and K is one of the recurrent issues in Kolmogorov complexity, and this has motivated the development of variants (such as logical depth, see [2] for details).

As Fig. 3 illustrates, the relation is unidirectional; given a low K, we can affirm that the environment will look simple. On the other hand, given an intuitively complex environment, K must be necessarily high.

Given the relation shown in Fig. 3, only among environments with high K will we find complex environments, and, among the latter, not all of them will be difficult. From the agent's perspective, however, this is more extreme, since many environments with high K will contain difficult patterns that will never be accessed by the agent's interactions. As a result, the environment will be *probabilistically* simple. This will be crucial later on, as we can consider most environments with high K as a kind of aggregate of the environments of lower K. And this also means that environments are typically seen by agents as being much simpler than they really are since many of the patterns that these environments include can be (probabilistically) inaccessible. Therefore, only very simple patterns will be shown. Thus, giving most of the probability to environments with low K means that most of the intelligence measure will come from patterns that are extremely simple.

4.2. Selecting discriminative environments

Furthermore, many environments (either simple or complex) will be completely useless for evaluating intelligence, e.g., environments that stop interacting, environments with constant rewards, or environments that are very similar to other previously used environments, etc. Including some, or most, of them in the sample of environments is a waste of testing resources; if we are able to make a more accurate sample, we will be able to make a more efficient test procedure. The question here is to determine a non-arbitrary criterion to exclude some environments. For instance, Legg and Hutter's definition forces environments to interact infinitely, and since the description must be finite, there must be a pattern. This pattern can eventually be learned (or not) by the examinee. However, this obviously includes environments such as "always output the same observation and reward". In fact, they are not only possible but highly probable on many reference machines.

[11] Hibbard [45] proposes only including environments of complexity greater than a positive integer L. Besides, this reduces the dependency on the reference machine. However, how to choose an appropriate value for L is not clear. Furthermore, the exclusion of simple environments is a problem for evaluating subjects with low intelligence levels. Also, large environments usually require more time for an agent to interact inside in order to get a reliable assessment.

[12] Wallace [12] (Sections 2.14 and 2.15) gives a way of modifying the weights from $w_j = 2^{-j}$ so that they decay very slowly but still add to 1. Another (very related) possibility is to modify the original distribution very slightly to a distribution that is uniform over all low-complexity environments up to some threshold and then has very slowly decreasing weights.

Another pathological case is an environment that "outputs observations and rewards at random".[13] However, this has a high complexity if we assume deterministic environments. In both cases, the behaviour of any agent on these environments would almost be the same. In other words, they do not have *discriminative power*. Therefore, these environments would be useless for discriminating between agents.[14]

In an interactive environment, a clear requirement for an environment to be discriminative is that what the agent does must have consequences on rewards. Without any restriction, many (most) simple environments would be completely insensitive to agents' actions. As mentioned above, in [29], a taxonomy of environments is developed, and the concept of ergodic MDPs is presented. Ergodic MDPs are characterised by making any possible observation reachable (in one or more cycles) from any state. That means that at any state, agents' actions can recover from a bad previous decision. However, this taxonomy and the class of ergodic MDPs is not used to refine the definition given in [9]. In any case, we think that ergodic MDPs are quite a restriction, since many real environments do not give us a "second chance". If "second chances" are always available, agents' behaviours tend to be greedier and less reflective. Furthermore, it seems easier to learn and succeed in this class of environments than in a general class.

Instead, we will restrict environments to be sensitive to agents' actions. That means that a wrong action (e.g., going through a wrong door) might lead the agent to part of the environment from which it can never return (non-ergodic), but at least the actions taken by the agent can modify the rewards in that subenvironment. More precisely, *we want an agent to be able to influence rewards at any point in any subenvironment*. This does not imply ergodicity but reward sensitivity at any moment. That means that we cannot reach a point from which rewards are given independently of what we do (a dead-end). This can be formalised as follows:

Definition 7 *(Reward-sensitive environment).* Given a deterministic[15] environment μ, we say it is n-actions reward-sensitive if, for every sequence of actions $a_1a_2\cdots a_k$ of length k, there exists a positive integer $m \leqslant n$ such that there are two sequences of actions $b_1b_2\cdots b_m$ and $c_1c_2\cdots c_m$ such that the sum of rewards that are obtained by the sequence of actions $a_1a_2\cdots a_kb_1b_2\cdots b_m$ is different[16] to the sum of rewards of the sequence $a_1a_2\cdots a_kc_1c_2\cdots c_m$.

Note that Definition 7 does not mean that *any* action has an impact on rewards (immediately or subsequently), but that *at any point/time* there are always at least two different sequences of actions that can lead the agent to get different accumulated rewards for n interactions. That means that these environments can have an agent stuck for a time (in a "hole") if the good actions are not taken, but there is a way to get out of there or at least to find different rewards inside the hole. In other words, they do not have heaven/hell points nor do they have a passive "observer" behaviour, so at any point the agent can strive to increase its rewards (or to keep them from decreasing).

According to the above definitions, many table games we know are not reward-sensitive environments. For instance, there are English draughts (checkers) positions where unavoidably any move leads you to a different way of losing/winning (assuming certain ability on the opponent's side or using the perfect solution given by [46]). But also note that it is not very difficult to modify the scoring to make it fully reward-sensitive by assigning points to the score that depend on the moves and position before losing (e.g., losing in 45 moves is better than losing in 25 moves).

The restriction of environment classes given by Definition 7 is simply a practical thing. We do not want to use environments or subenvironments to evaluate agents when anything the agent can do is useless for changing the reward. There are many other options to restrict environments in order to be more discriminative, but probably with a loss of generality. For instance, environments with high *diversity* would be desirable, meaning that taking different actions would usually lead to quite different subenvironments (situations); however, this is extremely difficult (if not impossible) to formalise.

There is an interesting relation of reward-sensitive environments and the kind of rewards that are allowed by Definition 3.3 in Section 6 (reward-bounded environment). In [9], a long discussion about how to distribute rewards is included and only one restriction is ultimately set: the *total reward* must be lower than 1 (Definition 3.3). The reason is mainly that, otherwise, all agents could accumulate an infinite value in the limit, and we could have the same score for all. Note that with the original definition, an environment that gives reward 1 after the first action and then always gives 0 complies with the reward-bounded restriction, but it is quite useless. The reward sensitive condition makes this impossible since, at any point in the environment, part of the total reward must remain in order to be shared among all the following subenvironments. This does not exclude, in principle, an environment that gives 0 rewards for thousands of initial actions and then starts giving rewards after this "dead" period. That is the reason why we will typically specify a small n in the definition of

[13] In practice, this should be pseudo-random, which implies that there is a pattern, like the program for a pseudo-random number generator or a simple program that outputs the digits of the number π.

[14] The C-test [6] avoided some of the previous problems through the use of the notion of *projectibility* of sequences and the use of patterns of several complexities. Literally adapting this idea to environments is possible, but it would require some highly elaborate concepts, such as a similarity between environments, which would be difficult to implement in practice.

[15] The restriction to deterministic environments is made because, otherwise, the definition should be defined in terms of the expectation of the sum of rewards. This is perfectly possible, like choosing between a fair dice and a loaded dice produces different expected results. For the sake of simplicity, we have assumed deterministic environments.

[16] We could set up several degrees of sensitivity by establishing a minimum threshold on this difference. For instance, $\max(R) - \min(R)$, i.e., the difference between the maximum possible reward and the minimum possible reward, could be a good choice (if $n \gg 1$).

n-actions reward-sensitive environment, which implies that there must be some reward 'spent' after each n or more actions. An extreme case is when $n = 1$, where some amount must appear in at least one of the rewards of the possible actions at any point. Example 2 seen above is a case of this.

4.3. On practical interactions

We are now closer to being able to construct a finite sample of 'proper' environments. However, in the original definition, we still have an infinite interaction with the environment, and we also have that K is not computable.

Limiting the number of interactions is easy to do. We just need to set a limit of interactions n_i for each environment by simply modifying the definition of expected reward[17]:

Definition 8 *(Cumulative reward on a finite number of interactions).*

$$V_\mu^\pi(n_i) := \sum_{k=1}^{n_i} r_k^{\mu,\pi}$$

It would be adequate to make the environments n_r-actions reward-sensitive, with $n_r \leqslant n_i$ so that actions have an effect on rewards in the limited period of interaction with the environment. A good choice is to make $n_r = n_i$.

With this restriction to n_i interactions, administering the test seems to be a finite task, but it is not. Apart from the problem that K is not computable, it has an additional very important problem: very inefficient environments are possible since the time that elapses between two observations can be too large. This means that, even with a short number of environments and interactions, some tests can be extremely long to take.

There are some options to computably approximate K, with Levin's Kt complexity being one popular choice. Although Kt is a computable approximation to K, this does not mean that it is an efficient approximation. Thus, in some cases, we would need to check a huge number of operations for a very small program. Nonetheless, the main problem with this measure is that the number of environment interactions is infinite. Consequently, there is no bound on the time required to *execute* the environment. A possibility here is to consider the average time (per bit or, in the case of environments, for each interaction). However, averaging the time taken on an infinite string (or environment interaction) can be done in many different ways. Besides, convergence and computability are not always ensured, and very variable time-slots can appear for the interactions. A simpler (and more practical) approach is to consider the maximum time for each output. First, for a universal Turing machine U and an environment program p in that machine, we define $\Delta ctime(U, p, a_{1:i})$ as the time required to print the pair $\langle r_{i+1}, o_{i+1}\rangle$ after the sequence of actions $a_{1:i}$, i.e. the (*cycle*) *response* time after the sequence of actions $a_{1:i}$. From here, we can set the upper bound to the maximum computational time that the environment can take to output the reward and the observation after the agent's action.[18]

Definition 9 *(Kt complexity weighting interaction steps).*

$$Kt_U^{\max}(\mu, n) := \min_{p \text{ such that } U(p)=\mu} \left\{ l(p) + \log\Big(\max_{a_{1:i}, i \leqslant n} \big(\Delta ctime(U, p, a_{1:i})\big)\Big)\right\}$$

which means that we sum the length of the coding of the environment (the length of the program in U) plus the logarithm of the maximum response time of that environment coding with the machine U. The complexity measure is bounded by a maximum number of interactions or cycles n, making its definition computable.[19] Note that this upper bound can be used in the implementation of environments, especially for making their generation computable. This limit n is not only necessary for computability; it is also practical in some other cases where computability is not a problem but there is not a maximum. For instance, consider an environment whose ith output depends on the calculation of whether number i is prime or not.[20] In this case, the maximum $\Delta ctime$ would not be bounded and, hence, $Kt_U^{\max}$ of this sequence would be infinite. Therefore, the environment would be ruled out if we do not set a limit n.

The complexity function given by Definition 9 ensures that the response time at any interaction with an environment is bounded, but we still preserve Occam's razor in the derived probability. With this new distribution, we refine the definition for universal intelligence as follows:

[17] Apart from having a finite number of interactions, we also assume the environment to be deterministic, so this definition turns out to be a *calculated* reward.

[18] This definition is clearly inspired by both Levin's Kt and the speed prior [35].

[19] Another option is to only consider environments whose maximum response time can be computed (i.e., proven) (and possibly include the time for proving its response time in the complexity measure). This would make parameter n unnecessary for $Kt^{\max}$.

[20] We now know it is a polynomial problem (it can be solved in polynomial time with respect to the number whose primality we want to check) [47].

Definition 10 *(Universal intelligence (finite set of reward-sensitive environments, finite number of interactions, $Kt^{\max}$ complexity)).*

$$\Upsilon^{ii}(\pi, U, m, n_i) := \frac{1}{m} \sum_{\mu \in S} V_\mu^\pi(n_i)$$

where S is a finite subset of m environments being n_i-actions reward sensitive extracted with $p_U^t(\mu) := 2^{-Kt_U^{\max}(\mu, n_i)}$.

Definition 10 now makes explicit that the reference machine, the number of environments, and the number of interactions in each are parameters of the definition. While U is a theoretical necessary choice, both m and n_i are practical requirements to make the test finite. The good thing is that if both m and n_i get higher, their relevance is smaller, and it is also true that the choice of U is also less important.

Given a finite value for m and n_i, we can evaluate several subjects and compare their scores. The subject with highest score will be considered the most intelligent, so a test defined from Definition 10 is, in principle, useful for comparing subjects. However, since rewards are positive, the value of Υ^{ii} that is returned is somehow meaningless. This is because as n_i gets higher the value also gets higher, since rewards are always positive and we just accumulate scores. This is now clear because we have removed all the other sources of infinity. The attempt by [9] to bound rewards was in the right direction, but if we infinitely accumulate them for larger n_i, we will get higher and higher values (up to the established bound). Note that this does not happen for increasing values of m, since m is in the denominator. Therefore it seems that just putting n_i in the denominator would solve the problem. However, it is precisely the choice of the accumulated reward being between 0 and 1 which would yield:

$$\forall \mu, \pi: \quad \lim_{n_i \to \infty} \frac{V_\mu^\pi(n_i)}{n_i} = \lim_{n_i \to \infty} \frac{E(\sum_{k=1}^{n_i} r_k^{\mu,\pi})}{n_i} \leqslant \lim_{n_i \to \infty} \frac{1}{n_i} = 0$$

The use of reward-bounded environments creates many questions about the distribution of the rewards since most of the bounded total amount will be typically given during the first interactions. Therefore, agents will be very hasty trying to get positive rewards before they are exhausted. Generally, postponing an important quantity of the total amount to late cycles requires higher description sizes than do environments that return most of it during the first interactions. Bounding the rewards may solve some issues, but it also creates others, and it seems that having rewards throughout along the life of the agent (as is customary in reinforcement learning) seems a much more natural option to keep the attention of the examinee.

As we will see in Section 4.5, we do not bound rewards. Instead, we distribute them by maintaining the constraint of being reward-sensitive on a finite n. However, the relation between expected intelligence score and the number of interactions is still quite unclear. This relation is discussed below.

4.4. On aggregating an absolute intelligence score

The problem of getting an "intelligence score" from Definition 10 is much more cumbersome than it might seem at first glance. The relation between the expected score and the values of m and n_i is intricate. Let us first analyse the influence of m, which determines the number of environments and, indirectly, the $Kt^{\max}$ values of the sample.

The figurative plot shown in Fig. 4 shows an example of the expected score, i.e., assuming an infinite number of interactions, of four agents depending on the complexity $Kt^{\max}$ of the environment. As the figure shows, the curve for each agent has a lot of information that is difficult to summarise in a single number.

Three of the four agents behave better for low-complexity environments; even though their curves are not monotonic, the four of them are able to get some score for more complex environments. This is also a general result since rewards are positive and, even though rewards can be very sparse in large environments, this will typically be greater than 0 in the limit. If we compute the sum of the expected rewards per agent for all the environments, it is clear why we usually get an infinite value. In fact, we get an infinite value for the four since the integral of each and all of the four curves between 1 and ∞ is ∞. If, instead of that, we perform an average (dividing by m) as in Definition 10 and we let m become larger, it boils down to measuring the value when $Kt^{\max} \to \infty$, ignoring the start of the curve. In this case, agents 1, 3 and 4 would have the same intelligence (0.2) while agent 2 would have 0.15. If we use a probability that overweights simple environments (as Legg and Hutter do), agent 4 would dominate over agent 2, agent 2 over agent 1 and (mostly) agent 1 over agent 3. It is, in fact, almost like only considering the leftmost part of the curve.

Again, one dilemma we face here is whether performing in simple environments is as important as performing in complex environments, as we discussed at the beginning of this section. In other words, is the beginning of the curve important? This dilemma vanishes if we realise that complex environments also include simple patterns and these will remain much more frequent than the complex ones. In the case of environments, it is much easier to see why simple patterns are frequent in complex environments, as we discussed in Section 4.1. Consider, for instance, an environment μ defined as follows: if the first action is a_1, then behave as environment μ_1; otherwise, behave as environment μ_2. If μ_1 and μ_2 are independent, the overall complexity would be approximately the sum of both; it is clear that μ has two simpler patterns, each with

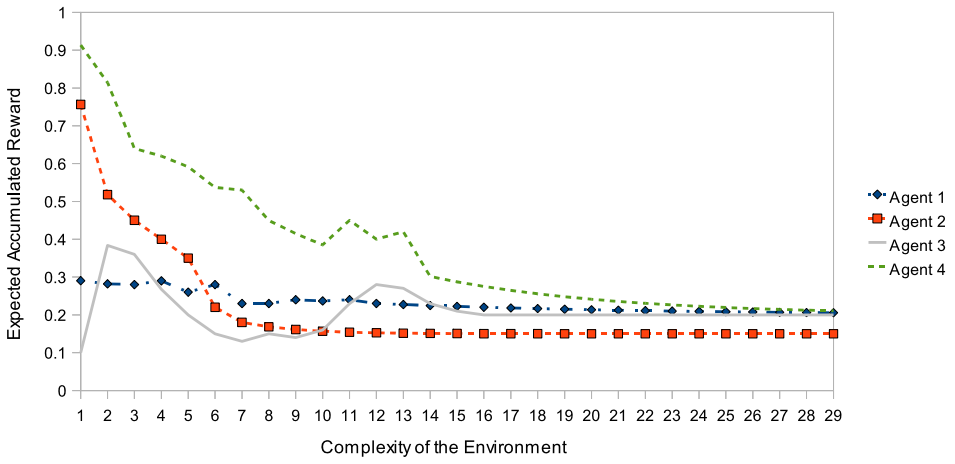

Fig. 4. Figurative evolution of expected accumulated rewards vs. complexity of environment.

complexity equal to the complexities of μ_1 and μ_2 alone.[21] Furthermore, in large environments, it is more likely to find subenvironments where a policy ensures very good rewards (as it is also more likely to find subenvironments where a policy ensures very bad rewards). This is basically because there can be more diversity. In very simple environments, however, series of very good rewards might simply not exist. And if these simple patterns exist in complex environments, intelligent agents will be able to take advantage of them (note that the number of interactions in Fig. 4 is assumed to be infinite), so rewards will be slightly better than those obtained by a random agent with infinite interactions. Consequently, the end of the curve is some kind of aggregate of all the previous complexities.

As a result, as suggested by [45], should we go directly to environments of very high complexity as a means of getting an approximate value of intelligence? This is completely against Occam's razor. One way out of this dilemma is to think that results for low-complexity environments are less reliable for the overall score than results for high-complexity environments. However, the problem of using high-complexity environments is that they also require longer interactions to get good reliability (Fig. 4 shows increasing stability on the right because we are assuming an infinite number of interactions). Consequently, going to very complex environments would require too much time.

And this is where n_i also has to be considered. If we give few interactions to a very complex environment we get poor reliability. In the end if the number of interactions increases with a fixed given complexity, then an intelligent agent has more data (more observations) to learn from, and the results are expected to be better (than those obtained by a random agent). This is seen in Fig. 5. Agent 1 and agent 2 start behaving as a random agent, but as the history of observations grows through an increasing number of interactions, they are able to learn and behave better. As a figurative example, agent 2 is shown to be more eager than agent 1 and starts scoring better initially, while agent 1 perhaps takes some more risks initially to learn the environment better and then scores better in the limit. Note also that since we are plotting accumulated reward (and rewards are always positive), it also grows for a random agent (independently of whether or not a discounting factor is used). This is counterintuitive, since the expected value for a random agent should be the same, independently of the number of interactions. One of the most distinctive signs of an agent being intelligent is that its expected reward increases (in our context, higher than random) as long as more interactions are given. Consequently, if random agents do not score 0 on average (or another constant) for every environment, we need to calibrate results for each environment in order to compare agent performance in each of them and also in order to compare and aggregate results from different environments.

However, the key point is in the relation between m (the number of environments, which determines the average complexity) and n_i (the number of interactions which has to be considered in each environment). For instance, a value of $n_i = 1000$ for an environment of $Kt_U^{\max} = 5$ will usually be enough to reach the stable part of the curves in Fig. 5. However, the same value for a very complex environment will typically be insufficient. A very intelligent agent might score as a random agent simply because it has not been given enough observations to capture the underlying models or once it has captured it, most of the reward has been exhausted since we are considering reward-bounded environments. Therefore, the result will not be much different from that of a random agent. This also suggests that for a finite number of interactions, the appearance of Fig. 4 might be different. The expected reward for very complex environments could be the same for a relatively intelligent agent as for a random agent since the former would not have enough interaction steps to learn from the environment.

[21] This is related (but different) to the fact that any incompressible object (e.g., a sequence) must have compressible parts (subsequences) (see, e.g. [2]). Note also that the existence of simple patterns in large environments is not obtained by the existence of "dead code" since this is impossible because we are using $K_U(\mu)$ (or $Kt_U^{\max}(\mu)$) instead of $l(\mu)$.

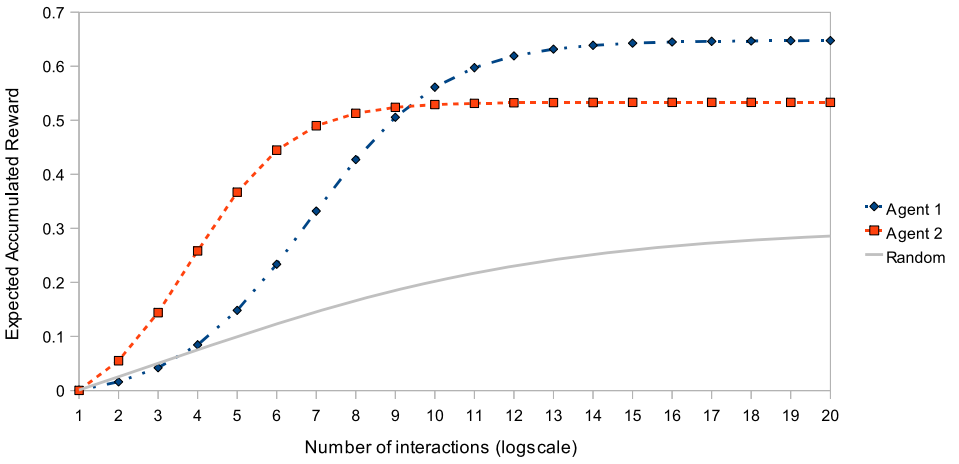

Fig. 5. Figurative evolution of expected accumulated rewards vs. number of interactions.

From this discussion, we see that increasing values for m may entail a worse expected result (since more complex environments are allowed) and increasing values for n_i certainly entail better results (a larger number of interactions is allowed to learn the pattern). This suggests that we could apply a correction in the score. The following definition revisits the previous definitions of accumulated reward and formalises this idea:

Definition 11 *(Adjusted reward).* Given an environment μ of complexity $\xi_\mu = Kt^{\max}(\mu, n_i)$, with n_i being the number of completed interactions, then the adjusted cumulative reward for agent π is defined as follows:

$$W_\mu^\pi(n_i) = V_\mu^\pi(n_i) \cdot \Phi(n_i, \xi_\mu)$$

such that $\Phi(x, y)$ is a function that is decreasing in x and increasing in y.

The function $\Phi(x, y)$ should be determined theoretically or experimentally (or a combination of both). However, this does not seem easy, especially because this relation depends on each environment. This would possibly include some bias and would be a source of permanent discussion. In what follows, and especially in Section 5 with the notion of an adaptive test, we will try to get rid of this adjustment.

4.5. Symmetric rewards and balanced environments

Note that the influence of n_i in Υ (Definition 10) is twofold. On the one hand, we have the effect of having more observations to let an intelligent agent learn and get better rewards. On the other hand, we have the effect of using cumulative rewards as a payoff function that affects any agent, including a random agent. Thus, an increase in the score may originate from a really good behaviour on the environment or just because more rewards are accumulated since they are always positive. As a result, an average reward seems a better payoff function. However, an average is not possible with Legg and Hutter's reward-bounded constraint (since this would converge to 0). Consequently, we are going to revisit the constraints on rewards. Our proposal is to use symmetric rewards, which can range between -1 and 1:

Definition 12 *(Symmetric rewards).* We say an environment has symmetric rewards when:

$$\forall i: \quad -1 \leqslant r_i \leqslant 1$$

Note that this does not preclude the accumulated reward at a certain point from being greater than 1 or lower than -1. Thus, if we do many good actions in a row, we can have an accumulated reward that is greater than 1. With regard to physical implementation, negative rewards do not have to be associated with punishment, which is considered unethical for biological individuals. For instance, if we are evaluating an ape, rewards from -1 to $-1/3$ might imply nothing, from $-1/3$ to $1/3$ a piece of fruit, and from $1/3$ to 1 two pieces. Or a negative reward may imply removing a previously awarded fruit.

If we set symmetric rewards, we also expect environments to be symmetric, or more precisely, to be balanced on how they give rewards. This can be seen in the following way. In a reliable test, we would like that many (if not all) environments give an expected 0 reward to random agents. The following definition formalises this.

Definition 13 *(Balanced environment).* An environment μ is balanced iff

1. It has symmetric rewards, i.e.: $\forall i\colon\ -1 \leqslant r_i \leqslant 1$.
2. Given a random agent π_r, the following equality holds[22]:

$$V_\mu^{\pi_r} = E\left(\sum_{i=1}^{\infty} r_i^{\mu,\pi}\right) = 0$$

This excludes both hostile and benevolent environments, i.e., environments where doing randomly will get more negative (respectively positive) rewards than positive (respectively negative) rewards. In many cases it is not difficult to prove that a particular environment is balanced. For complex environments, the previous constraint could be tested experimentally. Another approach is to set a reference machine that only generates balanced environments (as shown in some examples in Section 6).

Note that the previous modifications on rewards now allow us to use an average rather than an accumulated reward, namely:

Definition 14 *(Average reward).* Given an environment μ, with n_i being the number of completed interactions, then the average reward for agent π is defined as follows:

$$v_\mu^\pi(n_i) = \frac{V_\mu^\pi(n_i)}{n_i}$$

And we can calculate the expected value (although the limit may not exist) of the previous average, denoted by $E(v_\mu^\pi)$, for an arbitrarily large value of n_i. Let us see this with an example:

Example 4. Consider a modification of the test setting that we saw in Example 2. A robot (the agent) can press one of three possible buttons ($\mathcal{A} = \{B_1, B_2, B_3\}$), rewards are just the handing over of no bananas, a banana or two bananas ($\mathcal{R} = \{-1, 0, 1\}$), and the observation set derives from three cells where a white and black ball must each be inside one (but a different) cell, namely ($\mathcal{O} = \{0WB, 0BW, W0B, BOW, WB0, 0BW\}$), where W denotes the cell has a white ball, B denotes the cell has a black ball, and 0 denotes that it is empty. With x, we denote a variable symbol denoting any possible symbol in $\{0, W, B\}$ where two separate occurrences of x do not have to be equal. An example of a possible environment is:

$$\mu(r_k|r_1 o_1 a_1 r_2 o_2 a_2 \cdots r_{k-1} o_{k-1} a_{k-1}) = \begin{cases} 1, & \text{if } \begin{cases} \{(a_{k-1} = B_1 \text{ and } o_{k-1} = Wxx) \text{ or } (a_{k-1} = B_2 \text{ and } o_{k-1} = xWx) \text{ or } (a_{k-1} = B_3 \text{ and } o_{k-1} = xxW)\} \\ \text{and } (r_k = +1) \end{cases} \\ 1, & \text{if } \begin{cases} \{(a_{k-1} = B_1 \text{ and } o_{k-1} = Bxx) \text{ or } (a_{k-1} = B_2 \text{ and } o_{k-1} = xBx) \text{ or } (a_{k-1} = B_3 \text{ and } o_{k-1} = xxB)\} \\ \text{and } (r_k = -1) \end{cases} \\ 1, & \text{if } \begin{cases} \neg\{(a_{k-1} = B_1 \text{ and } o_{k-1} = Wxx) \text{ or } (a_{k-1} = B_2 \text{ and } o_{k-1} = xWx) \text{ or } (a_{k-1} = B_3 \text{ and } o_{k-1} = xxW)\} \\ \text{and} \\ \neg\{(a_{k-1} = B_1 \text{ and } o_{k-1} = Bxx) \text{ or } (a_{k-1} = B_2 \text{ and } o_{k-1} = xBx) \text{ or } (a_{k-1} = B_3 \text{ and } o_{k-1} = xxB)\} \\ \text{and } (r_k = 0) \end{cases} \\ 0, & \text{otherwise} \end{cases}$$

The observation o_k is generated as a uniformly random choice between these four observations $\{0WB, 0BW, W0B, WB0\}$. The first reward r_1 is 0.

A first robot (π_1) has the behaviour of always pressing button B_1, i.e., $\pi_1(B_1|X)$ for all sequences X. Consequently, the performance of π_1 in this environment is:

$$E\left(v_\mu^{\pi_1}\right) = E_{n_i\to\infty}\left(\frac{\sum_{k=1}^{n_i} r_k^{\mu,\pi_1}}{n_i}\right) = \frac{2}{4}\lim_{n_i\to\infty}\frac{n_i}{n_i} + \frac{2}{4}\lim_{n_i\to\infty}\frac{0}{n_i} = \frac{1}{2}$$

A second robot (π_2) has the behaviour of always pressing button B_2, i.e. $\pi_1(B_2|X)$ for all sequences X. Consequently, the performance of π_2 in this environment is:

$$E\left(v_\mu^{\pi_2}\right) = E_{n_i\to\infty}\left(\frac{\sum_{k=1}^{n_i} r_k^{\mu,\pi_2}}{n_i}\right) = \frac{1}{4}\lim_{n_i\to\infty}\frac{n_i}{n_i} + \frac{2}{4}\lim_{n_i\to\infty}\frac{-n_i}{n_i} + \lim_{n_i\to\infty}\frac{0}{n_i} = -\frac{1}{4}$$

[22] Note that the expected value must be 0 in the limit. For infinitely many n_i, the expected value can be different from 0. Even for deterministic environments, the series of accumulated rewards might be divergent, i.e., the value $\lim_{n_i\to\infty} V_\mu^{\pi_r}(n_i)$ might not exist and we assume a Cesàro limit ($\lim_{n_i\to\infty}\frac{1}{n_i}\sum_{j=1}^{n_i} V_\mu^{\pi_r}(n_i)$).

A third robot (π_3) has the behaviour of always pressing button B_3, i.e. $\pi_3(B_3|X)$ for all sequences X. Consequently, the performance of π_3 in this environment is:

$$E\left(v_\mu^{\pi_3}\right) = E_{n_i\to\infty}\left(\frac{\sum_{k=1}^{n_i} r_k^{\mu,\pi_3}}{n_i}\right) = \frac{1}{4}\lim_{n_i\to\infty}\frac{n_i}{n_i} + \frac{2}{4}\lim_{n_i\to\infty}\frac{-n_i}{n_i} + \lim_{n_i\to\infty}\frac{0}{n_i} = -\frac{1}{4}$$

A fourth robot (π_4) has a random behaviour. Then the performance of π_4 is:

$$E\left(v_\mu^{\pi_4}\right) = E_{n_i\to\infty}\left(\frac{\sum_{k=1}^{n_i} r_k^{\mu,\pi_4}}{n_i}\right) = \frac{3}{3}\left(\frac{2}{4}\lim_{n_i\to\infty}\frac{n_i}{n_i} + \frac{1}{4}\lim_{n_i\to\infty}\frac{-n_i}{n_i} + \frac{1}{4}\lim_{n_i\to\infty}\frac{-n_i}{n_i}\right) = 0$$

Equivalently, we can get this by averaging π_1, π_2 and π_3.

Consequently, agent π_1 is better than random (π_4) in this environment, and π_2 and π_3 are worse. And, finally, since the expected overall reward of a random agent is 0, this environment is balanced.

Let us now give a more refined definition of universal intelligence using the new symmetric rewards, the balanced environments and the average rewards:

Definition 15 *(Universal intelligence (finite set of reward-sensitive and balanced environments, finite number of interactions, $Kt^{\max}$ complexity) with adjusted score).*

$$\Upsilon^{iii}(\pi, U, m, n_i) := \frac{1}{m\cdot n_i}\sum_{\mu\in S} W_\mu^\pi(n_i)$$

where S is a finite subset of m balanced environments being n_i-actions reward sensitive extracted with $p_U^t(\mu) := 2^{-Kt_U^{\max}(\mu,n_i)}$.

We still maintain the weighted reward (from Definition 11), which includes the function Φ.

5. Time and intelligence

Definition 15 given above is now feasible and stable with respect to varying m and n_i. But the main criticism is that, provided that "universal intelligence could be viewed as generalising the C-test from passive to active environments" [9], there is no reference to (physical) *time*. How can an interactive extension disregard time? How can a very slow agent be considered as equally intelligent as a very quick agent with the same result? Also, how can we administer a test disregarding physical time? We need to evaluate intelligence in a finite period of time. And the use of physical time may refer *either* to the environment *or* to the agent since both interact and both of them can be either fast or slow.

If we consider how physical time may affect an environment, i.e., the *environment's speed*, it is unacceptable to have an interactive test where the agent has to wait several hours after each action in order to see the reward and the observation. We expect the environment to react *immediately*. With the use of $Kt^{\max}$, and the complexity levels we expect to reach, it will almost always be the case that the environment's reactions will be perceived as immediate. Under these circumstances, there need not be a great concern about the environment's speed in our test setting.

On the other hand, when we generally refer to time when measuring intelligence, especially in non-interactive tests, it is assumed that we are talking about the *agent's speed*. This is much more controversial in general (and also in our setting). A quick reference is made on p. 426 in [9]: "while we do not consider efficiency to be a part of the definition of intelligence, this is not to say that considering the efficiency of agents is unimportant". The reference to time has also been made in [5,7] and [32, footnote 202].

This issue is also recurrent in psychometrics, with a strong debate about the correlation of "inspection time" and intelligence. Nonetheless, time is generally introduced in typical IQ tests in an indirect way since the examinees have a fixed time to do the test. Conversely, one can think that speed and intelligence are different things. We could measure intelligence and the speed of answers/reactions separately. In fact, in psychometrics, tests are usually categorised into either speed tests or potential tests, with intelligence tests being inside the latter category. The rationale behind this is that many difficult problems seem unsolvable by dull agents even if they are given infinite time. Even though this is probably true, considering time and intelligence to be independent is quite an assumption. Some systems are able to produce a reasonable action in a short period of time, and they can improve the action if they are given more time. Intuitively, these systems seem more adaptable than others that do not come up with any good action until a long period of time has passed.

Also, there are other conceptual problems that are associated with ignoring time in an intelligence definition: constructing intelligent systems would seem less difficult than it actually is since disregarding computational complexity would be an option. For instance, an inefficient and exhaustive search method (see, e.g. [32, footnote 199]) might be appropriate for intelligence.

Consequently, there is a need to consider time, either as a limit to get agents' actions or as a component of the final score. The inclusion of time in the test can be done in such a way that its influence can be adjusted, from cases where we want to test good and quick responses to other cases where time is not so relevant. Another option is to incorporate time in a discrete way (as we will see in some definitions in the following sections).

One possibility of incorporating time without modifying the original Legg and Hutter formulation would be to de-couple observations-rewards from actions in the way that we can get several rewards and observations in a run without any action. This is in fact implicitly contemplated in the original definition since a special action known as "no action" can be considered. However, the time scale between observations and actions (imagine an observation each nanosecond, or an observation every thousand years) seems to be a parameter that looks quite anthropomorphic if we set it to some seconds. Thus, we need to explore some other options.

5.1. Time and rewards

Apparently, there are many options for incorporating time. Considering that we have an overall time τ for an environment, one option is to set a time-out τ_o for each action (with $\tau_o \ll \tau$) such that if the agent does not select an action within that time, reward 0 is given (or a random action is performed). The shorter the time-out is, the more difficult the test is. However, apart from the problem of setting this time-out in an appropriate way for different kinds of individuals, consider a very fast agent who does slightly better than random. If we evaluate this agent during a time τ over an environment, its strategy would be to perform as many actions as possible in order to accumulate maximum reward, since the more interactions, the higher the accumulated score. Apart from the solution of averaging the rewards and using balanced environments, an alternative possible solution would be to set a fixed time, a time-slot τ_s (instead of a time-out) for each interaction (with $\tau_s \ll \tau$). But, again, given an overall time τ, how many equal-length time-slots should we generate? Considering (randomly chosen) different-length time-slots for several interactions, a quick agent would be able to perform appropriate actions for more interactions than a slow agent with the same potential intelligence. However, it is not easy to tune these time-slots independently from the agent and, in any case, it is not very sensible to make the agent wait for some observations and rewards if we want to make a practical and efficient test.

As a result, if we do not assign time-slots, necessarily the rewards obtained in an environment during an overall time τ must be averaged, otherwise very fast but dull (slightly better than random) agents would perform well. The natural idea is to average by the number of interactions that the agent finally performs in time τ as seen in Definition 14. However, a *shrewd* policy here would be to act as a fast random agent until the average reward becomes larger than a threshold (this can happen with greater or lower probability depending on the threshold) and then stop acting. For instance, consider an agent that performs one action randomly. If the reward is positive, then stop (no other action is performed). If the reward is negative, then act fast and randomly until the average reward is positive and then stop. Note that this strategy ensures a positive reward in balanced environments. Consequently, an agent could get a very good result by very fast (and possibly lucky) first interactions and then rest on its laurels, because the average so far was good.

One way to minimise this problem is to use the time left from the last action until time τ as a discount over the recent history. Namely,

Definition 16 *(Average reward with diminishing history).*

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

where $\breve{v}_\mu^\pi \| \tau$ means the average reward until time τ (possibly externally interrupted), n_τ is the number of completed interactions made by π in μ in time τ, and t_{n_τ} denotes the total time elapsed until the last action a_i was made.

This definition scales the number of evaluated cycles proportionally to the time from the beginning until the last completed action τ. That means that if the most recent actions have been good and we delay future actions or let time pass, the measure will soon ignore the recent (good) rewards. If we stop, in the limit, the measure reaches 0, so it also avoids stopping policies (for further justifications on these choices, see [48]).

It is clear that as we leave more physical time, generally, an agent can get more observations (very quickly, almost randomly at first) and then try to use this knowledge in some other actions. Thus, we again have that results are expected to improve when τ grows, while they are expected to worsen when the complexity of environment ξ_μ grows (if τ remains constant). Consequently, if we were able to relate τ and ξ_μ, such that complex environments were allotted more time (as we will do in Section 5.2), we could get rid of Φ. For the moment, we need the adjusted version of the aggregated reward:

$$\breve{w}_\mu^\pi \| \tau := \Phi'(\tau, \xi_\mu) \cdot \breve{v}_\mu^\pi \| \tau$$

The function Φ' is similar to Φ, but it considers physical time and not the number of interactions as the first parameter. And now, with an average reward and physical time, the goal of Φ' becomes different. If we want to give more or less weight to the speed of the agent in the measure of intelligence, then Φ' should be different from the case in which we just

want to correct the influence of time. Of course, we can also tune Φ' to give more or less weight for simple or complex environments.

Finally, if interactions are limited by physical time, the definition is as follows:

Definition 17 *(Universal intelligence considering time (finite set of reward-sensitive and balanced environments, finite number of interactions, $Kt^{\max}$ complexity) with adjusted score and using physical time to limit interactions).*

$$\Upsilon^{iv}(\pi, U, m, n_i, \tau) := \frac{1}{m} \sum_{\mu \in S} \breve{w}_\mu^\pi \| \tau$$

where S is a finite subset of m balanced environments that are also n_i-actions reward-sensitive. S is extracted with $p_U^t(\mu) := 2^{-Kt_U^{\max}(\mu, n_i)}$.

Note that we use the same value n to bound the limit of consecutive actions that have to be reward-sensitive and also to make Kt computable. We should consider here a value of n that is large enough and still makes the computation of Kt feasible. If the number of interactions used to calculate Kt is eventually surpassed, we could just limit the response time for subsequent interactions to become the calculated maximum response time until interaction n.

5.2. Anytime evaluation with time

Using two different parameters for the number of environments (m) and the overall time left (τ) for each environment is an important bias (apart from the effect they have on the measurement) even if minimised by function Φ'. If we allow many environments with a small number of interactions in each, many agents will be unable to find patterns and take advantage of them. Leaving many interactions to let agents find the patterns would allow us to explore only a few environments. In other words, there are also practical reasons to find a trade-off between the parameters m and n_i in Definition 15, and the relation between m and τ in Definition 17. If we want to test a system with high intelligence, these values should be chosen accordingly: apparently we should use few but more complex environments with more time interacting with them. On the other hand, if we want to test a less competent system, we should use many simple environments with possibly less time interacting with them. This suggests that in order to take advantage of a limited available time, we should carefully choose some parameters before applying the test according to our expectations about the examinee. Although this is typical in psychometrics, and we have different kinds of tests for different levels of intelligence (non-human animals, infant or adult humans, etc.), it is not acceptable for a general test. The key point here is that tests cannot be independently configured before administration, since the complexity and time of the exercises would be set beforehand depending on the subject, and the results would not be comparable. Instead, we need tests to be adaptive to the examinee's level of intelligence. And this adaptation must be done automatically and equally for all (quite differently to a Turing test, for instance, where there is an adaptation, but it is quite subjectively conducted by a human).

The idea of adapting the test to the examinee is known in psychometrics as Adaptive Testing and, since the tool which is typically used to adapt the questions/items to the examinee is a computer, the area is known as Computerized Adaptive Testing (C.A.T., see, e.g. [49]). C.A.T. is based on a calibration of items in order to know their difficulty. This must be done by taking information from previous test administrations, so necessarily the difficulty of an item is *subjective* to the population that has been used as examinees in the past. Following this idea, a more sophisticated approach that is generally used in C.A.T. is known as Item Response Theory (IRT) [50,51], where not only a difficulty score is assigned to each item but a richer Item Response Function or Curve is assigned. These functions allow for an optimised item selection based on the examinee's cognitive demand features, providing results that help understand what is being measured and adapting the test to the level of the individual being examined. An extra sophistication is when the adaptation to the examinee is not purely statistical but tries to learn from the examinee using cognitive theory or even artificial intelligence. Items generated from cognitive theory and analysed with IRT are a promising tool, but these models are not yet mainstream in testing [15,16,52].

Not only do we want a test to be adaptive, but we want to have a test that, given a small slot of time, we could have a rough approximation of the intelligence of the agent. On the other hand, if given more time, we could have a better approximation. It would also be interesting to be able to stop the evaluation at any moment and get an approximation. In other words, we would like to have an *anytime* test (an anytime algorithm is an algorithm that can be stopped at any time, giving a reasonable answer for the time given). Again, some approaches from C.A.T. fulfil (or can be easily adapted to) this requirement.

All of this research in C.A.T. can be useful to a greater or lesser extent for constructing a universal and adaptive test. In our case, however, the difficulty of each item is determined theoretically and not experimentally. We propose the idea of adapting both the complexity of the environments and the testing time that we assign to each of them. One possibility is to define two counters: one for the ($Kt^{\max}$) complexity (denoted by ξ) of the environment and another for (physical) time (denoted by τ), the time that we leave to play with one environment before switching to a different one. Therefore, since we do not know the intelligence or the speed of the agent, we need to start with very simple problems and very small time slots to play with each environment. We will produce progressively more complex problems and we will leave more time for each environment. But since many individuals will only be able to learn and interact reasonably at some small ranges of

complexity if given a limited time, there must be a mechanism to also degrade the complexity if the agent does not score well on the environments[23] (this is the meaning of adaptive, here).

Therefore, to design the adaptation policy of the modified test, the first question is to determine the complexity level and the speed scale at which we should start. In C.A.T., a typical choice is to start at an *intermediate* level of ability/difficulty because we have some expectations on the individual's ability or, at least, we have a wide range from which we can establish a middle point. However, when evaluating an unknown agent (which might be a very intelligent human or an incredibly dull software agent), we cannot assume any base intelligence. Similarly, we cannot assume any speed scale (humans have typical response times in the order of seconds, but a software agent might have response times which might be much more variable, from microseconds to hours of computation). Consequently, the proposal is to start with the smallest possible value for complexity ($\xi = 1$, or the lowest value such that its ($Kt^{\max}$) complexity gives a valid environment) and the smallest possible value for time ($\tau =$ Planck time, $\approx 10^{-43}$ seconds, or, more practically, a value around current agent technology, e.g. 1 microsecond). It is easy to modify these start values if we have some information about the capabilities of the examinee.

The second question is the pace and directions at which ξ and τ have to be modified. If we think about complexity, the idea is to apply a *servo-control* adaptation to increase ξ when the agent is able to deal with the environment's complexity and the time scale and to decrease ξ otherwise. This kind of regulation is also typical in C.A.T. Thus, ξ will be incremented when the agent succeeds in an environment, and ξ will be decremented when the agent fails in an environment. A detail we have to consider is that we will not allow an agent to play with the same environment twice (which would allow rote memory learning with an already seen environment). Since there is a limited number of environments with a complexity lower than a given ξ, if a very bad agent exhausts all the environments with complexity lower than a given ξ, we will increase ξ in order to avoid environment repetitions.

With regard to time, we need to increase it until it matches the agent's time resolution/scale. The difference here is that if an agent reacts with an action in approximately a second, initially it will not have time to perform any action if we start with microseconds. In a few iterations, we will reach $\tau = 1$ second and we will start seeing how the agent performs a few actions. Finally, with some more iterations, when this value is incremented further (e.g. $\tau = 2$ minutes), we will allow more actions in the same environment and the agent will work within its time scale.

The final question is therefore the precise pace of increasing/decreasing complexity and increasing time. To allow quick adaptation, an exponential growth for time could be used, and complexity variations could depend on the reward value. Definition 18 below specifies this.

The definition has four parameters. Apart from the agent, we have a universal machine (or an environment class), a complexity function (a function that returns a positive real number with the complexity of each environment in the universal machine or in the environment class), and a time limit (which can be given in advance or used to stop the algorithm anytime). The complexity function can be precalculated for a sufficiently large random sample or pool of environments. As discussed above, our proposal for H is the function[24] $Kt^{\max}$; however, any other computable complexity function could be used instead, especially for restricted environment classes (such as those that will be seen in the examples in Section 6).

The selection of environments is made in line 11 of Definition 18. Note that this function will typically depend on precalculations or prefilters. For instance, it is preferable to use a universal environment class that only includes balanced reward-sensitive environments, such as some of those shown in the examples in Section 6 in this paper (or the one introduced in [53]). Also, the measurement is more efficient and robust if we leave each environment running against a random agent for a while before using the environment. This eliminates any start-up garbage (see, e.g. [53] for a full discussion on this).

Note that complexity and time have similar updating policies, even though complexity depends on rewards. Note that complexity is not necessarily a positive integer number, but a positive real number, so fractions in ξ are perfectly valid. If rewards are positive, complexity increases. If rewards are negative, complexity decreases. This dependence on rewards is important in order to avoid cheating. For example, an agent could cheat by scoring well (near 1) for very simple environments and the complexity would be upgraded. Afterwards, the agent could just score sufficiently bad (near -0.01) in order to be downgraded. This would allow the agent to try easy environments again and to score very well (near 1). This could be iterated indefinitely. On average, the scores would be around 0.5, even though the agent has never left the very simple levels. Although this behaviour is quite pathological, with the formula in line 20, the increase/decrease of complexity is proportional to the rewards (like a servomechanism), so going back to easy complexities implies important negative rewards, which are finally counted into the measure Υ.

[23] Again, using lower values for ξ ensures simple environments, but higher values for ξ do not guarantee difficult environments (nor are they much more likely, but just possible). Moreover, in complex environments, we may find subenvironments where there is a policy that ensures very good rewards. On the contrary, for simple environments, even though we assume reward-sensitivity, we will seldom find a sequence of actions that ensure good indefinite rewards. In other words, large environments have more diversity and offer more opportunities.

[24] A sufficiently large value for the parameter n can be used if the environment class does not ensure a provable bounded execution time per interaction.

Line 18 is the part of the algorithm where the agent interacts with the environment and where the algorithm waits for the agent's action. All of the time that is spent by the algorithm must be included in the environment's response time, and should be much shorter than the agent's time scale. Thus, the agent would perceive that the environment reacts immediately.

Definition 18 *(Anytime universal intelligence test taking time into account).* We define $\Upsilon^{v}(\pi, U, H, \Theta)$ as the result of the following algorithm, which can be stopped anytime:

```
1.  ALGORITHM: Anytime Universal Intelligence Test
2.  INPUTS: π (an agent), U (a universal machine), H (a complexity function),
            Θ (test time, not as a parameter if the test is stopped anytime)
3.  OUTPUTS: a real number (approximation of the agent's intelligence)
4.  BEGIN
5.    Υ ← 0                                  (initial intelligence)
6.    τ ← 1 microsecond                      (or any other small time value)
7.    ξ ← 1                                  (initial complexity)
8.    S_used ← ∅                             (set of used environments, initially empty)
9.    WHILE (TotalElapsedTime < Θ) DO
10.     REPEAT
11.       μ ← Choose(U, ξ, H, S_used)        (get a balanced, reward-sensitive environment with ξ − 1 ⩽ H ⩽ ξ not already in S_used)
12.     IF (NOT FOUND) THEN                  (all of them have been used already)
13.       ξ ← ξ + 1                          (we increment complexity artificially)
14.     ELSE
15.       BREAK REPEAT                       (we can exit the loop and go on)
16.     END IF
17.   END REPEAT
18.   Reward ← V_μ^π ‖τ                      (average reward until time-out τ stops)
19.   Υ ← Υ + Reward                         (adds the reward)
20.   ξ ← ξ + ξ · Reward/2                   (updates the level according to reward)
21.   τ ← τ + τ/2                            (increases time)
22.   S_used ← S_used ∪ {μ}                  (updates set of used environments)
23.  END WHILE
24.  Υ ← Υ/|S_used|                          (averages accumulated rewards)
25.  RETURN Υ
26. END ALGORITHM
```

A figurative trace of the algorithm is as follows. Initially, with a very small τ, the agent has no time to perform an action (or it has to do something very quickly without time for thinking, such as a random choice). Therefore, the agent has an expectation of 0 reward. This means that ξ would not be increased (since the increase depends on the reward), and the agent would go for another environment with $\xi = 1$ (if it exists, otherwise $\xi = 2$, and so on). In practice, this means that ξ will hover around small numbers for a while (since repeated environments are not allowed), but not too long, since τ grows exponentially. When time gets closer to the agent's time scale, the agent can perform better actions, and, if rewards are positive, increase ξ. The larger Θ is, the higher ξ can be, and more time is left in each of the environments. This gives more reliability to the aggregated reward and less dependence on the reference machine. Logically, if very little time is allowed, speed is crucial. Consequently, the relevance of speed in the measurement is determined by the time available to do the test.

Whenever the test is stopped, we have an approximation of the agent's intelligence by averaging the rewards obtained in all the environments. We can also have a look at the evolution (the curve) of results depending on ξ. This information should not be seen as an alternative measurement of intelligence, but just as some extra information that can help understand how the test worked out.

The expected results for different kinds of agents are difficult (if not impossible) to estimate precisely because they depend on the environment class used and the reference machine. The only easy case is a random agent, whose expectation is 0.

Nonetheless, we can make an estimation of the expected intelligence results of agents with some learning abilities, or to show the maximum value that can be obtained.[25] Table 3 shows how the test should work for some specific types of agents.

[25] Note that as $\Theta \to \infty$ and $\xi \to \infty$, we have very large environments which contain patterns that can be exploited by the agent. In fact, the probability of an environment μ appearing (dominated by $2^{-K(\mu)}$) is similar to the probability of *subenvironment* μ being reached by a random interaction inside a randomly chosen μ' (according to the same probability distribution but with $K(\mu) \ll K(\mu')$). Whether this holds for $Kt_U^{\max}$ is to be confirmed in order to ensure non-zero convergence for $\Theta \to \infty$.

Table 3
Figurative expected results on the anytime test for several prototypical agents for Υ (from Definition 18) and the expected complexity level which may be attained.

	Expected Υ and attained level if test stopped very early (e.g. microseconds)	Expected Υ and attained level if test stopped after several minutes	Expected Υ and attained level if test could go on indefinitely (actual value)
Random very fast agent	$\Upsilon = 0, \xi > 0$	$\Upsilon = 0, \xi \gg 0$	$\Upsilon = 0, \xi = \infty$
Relatively slow intelligent agent (e.g. a human)	$\Upsilon = 0, \xi = 0$	$\Upsilon > 0, \xi > 0$	$\Upsilon > 0, \xi = \infty$
Very fast super-intelligent agent (or an oracle)	$\Upsilon > 0, \xi \gg 0$	$\Upsilon > 0, \xi \gg 0$	$\Upsilon > 0, \xi = \infty$

Although Table 3 shows figurative results for a human when the test is administered for several minutes, it is still an open question to know how much time is required to have a good assessment of intelligence of, let us say, a human or an ape, and whether this required time would make the tests practical. It is different for machines since we can invest more time for intelligence assessment (since the test is completely automated and it does not require human intervention). The reason for showing the complexity level (and not only the aggregated value of intelligence) in Table 3 is to help understand how the test works. It is not our intention to use ξ as another cognitive factor or to define intelligence as a tuple of values that represent different factors.

The effect on the time required for the test is an issue since, given a fixed time Θ, the reliability of the estimation of intelligence of slow agents will be poorer than the estimation of intelligence of fast agents.[26] Speed is also considered in Definition 18 in the score since a slow agent will receive many 0 results until the values for τ reach the agent's time scale. According to Definition 17, only when the agent can see enough observations will it be able to make good actions and improve its average result. Finally, there is a relation between the time available for the test and the relevance of speed: the higher Θ is, the lower the influence of speed in the measurement is.

5.3. Anytime evaluation disregarding time

Although the algorithm in Definition 18 has been designed to consider physical time, it can be easily converted into a test that ignores the agent's speed by replacing physical time by interaction steps, as follows:

Definition 19 *(Anytime universal intelligence test disregarding time).* We define $\Upsilon^{vi}(\pi, U, H, \Theta)$ as the result of a modified version of the algorithm in Definition 18, which can be stopped anytime. In Definition 18, we replace physical time by interaction steps (line 6 changes into "$n \leftarrow 1$ (step)"), and these are updated by the formula $n \leftarrow \lceil n + n/2 \rceil$ (line 21). And $V_\mu^\pi \| \tau$ is replaced by $V_\mu^\pi(n)$ (line 18).

Note that in this case, testing a very slow agent would take much more time than testing a very fast agent.

6. Examples

In order to illustrate some of the features of the test specified in Definition 18, we present some examples of environment classes. We show simple examples on restricted classes first, and then we show some richer environment classes considering universal (Turing-complete) machines.

6.1. A very simple environment class

Consider the same test setting as in Example 2 with some modifications. In this setting, the agent can press one of n (with $n > 2$) possible buttons ($\mathcal{A} = \{B_1, B_2, \ldots, B_n\}$), raw rewards are just the removal of one banana, no banana, or the handing over of one banana ($\mathcal{R} = \{-1, 0, 1\}$), and the observation set is specified by n cups where some (or all) of them are black and the rest are white. Initially, the colours of the cups are set randomly. Each button (action) switches the colour of the corresponding cup. The environment also changes the colour of one cup in each action i, with the cup being i mod n. Initially, the colours of the cups are set randomly. Raw rewards are -1 if all the cups are black, $+1$ if all the cups are white, and 0 otherwise (not all cups are of the same colour).

It is easy to see that this class of environments (let us call it U_s) is reward-sensitive, and it is also balanced. An agent π_w that always presses the button that corresponds to a black cup (unless no black cup is left, where any other button is pressed randomly) makes (using Definition 18):

$$\lim_{\Theta \to \infty} \Upsilon^v(\pi_w, U_s, H, \Theta) > 0$$

[26] For ultra-intelligent systems, a small value of Θ could also give an underestimated score.

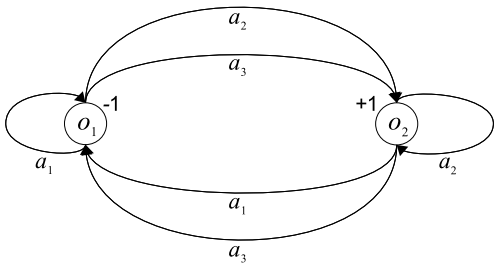

Fig. 6. A simple automaton.

where the complexity function is $H(\mu) = n$, with n being the number of cups. For low values of Θ and a slow agent, the result may typically be equal to 0, such as a random agent, since the agent would not have time to react in time.

It is clear that this environment class is very restrictive and uses a very simplistic complexity function H, but it can be used to understand the adaptation process of Definition 18.

6.2. Environments as finite state machines

Consider a reference machine U_1 which codes regular automata (finite state machines), such that each state has a fixed raw reward in the range between -1 and 1, transitions can only be made through three different actions $\mathcal{A} = \{a_1, a_2, a_3\}$, and observations are always a subset of the set of possible states $\mathcal{O} = \{o_1, o_2, o_3, \ldots\}$, where o_1 is always the initial state.

Additionally, we have some particular constraints:

- All the states must be accessible. That is, the resulting graph must be fully connected.
- From each state it must be possible to access any other state with a different reward in one or more steps. This implies that environments are reward-sensitive.
- Any infinite random walk through the automaton has an expected reward of 0. This implies that environments are balanced.

Given this setting, we can enumerate all the automata and compute their $Kt^{\max}$ using a language or grammar to represent this class of automata, and use this measure of complexity for H. One of the simplest possible automata is shown in Fig. 6. Note that the optimal behaviour for that environment is: a_2*. Of course, in general, we would need an agent that is able to learn regular patterns to behave well in other environments in this class.

This second example is much more interesting and richer than the first one. Finite State Machines are an important class of machines in computer science (and also in linguistics). This means that even with the set of constraints we have added, many environments and problems that can be modelled with Finite State Machines and regular languages can be evaluated with the anytime test. The novelty here would be that we start the test with simple automata, such as the one shown in Fig. 6, which would be used for a few interactions. Then we would move to more complex automata with more interactions. This allows for a more effective measuring of any agent with a short time limit. Regular automata is an important class, but it is not universal (type 3 in Chomsky's hierarchy). Also it does not contain other agents and objects, as discussed in Section 2 (a requirement advocated by many, such as [27]). Nonetheless, it is still an interesting environment class to assess learners, before moving to type 2, type 1, and ultimately type 0.

6.3. A very simple spatial environment

Consider a reference machine U_2 which codes a playing space or grid of 9×9 cells. We denote cells by (X, Y), where X is the horizontal co-ordinate and Y is the vertical co-ordinate. Our agent can move towards four possible directions or stay in the same cell, with actions $\mathcal{A} = \{L, R, U, D, S\}$, which represent *left*, *right*, *up*, *down*, and *stay*, respectively. The limits of the grid can be surpassed (so the grid movement is toroidal), appearing at the corresponding cell at the other side of the grid (e.g. if placed at cell $(1, 5)$, we move Left (L), then we go to $(9, 5)$).

Two objects, called *Good* and *Evil*, move around the space. The evaluated agent can be at the same cell as either *Good* or *Evil*, but *Good* and *Evil* cannot share the same cell (except for the initial state). The sequence of movements of *Good* and *Evil* are given by a non-empty finite sequence of actions (a path) which are repeated when exhausted. For instance, if the path for *Good*, which is denoted by $path_{Good}$, is $UULRDD$, it means that it will move according to the pattern $UULRDDUULRDD\ldots$ forever. If the movements of *Good* and *Evil* make them go to the same cell, this is avoided by randomly letting one of them move and keeping the other at its previous cell.

Observations are 4×3 matrices where the i-index indicates the content of the four adjacent cells in this order (*Left*, *Right*, *Up*, and *Down*) and the j-index indicates the presence or absence of *Good*, *Evil*, and *Limit*. *Limit* indicates that we are at some of the limits of the grid (which, as we have stated, can be surpassed).

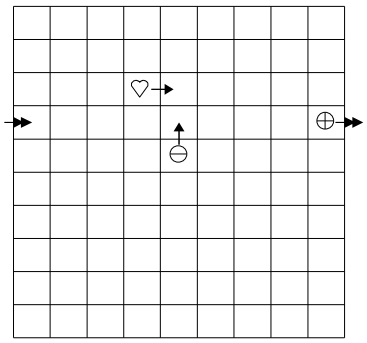

Fig. 7. A representation of this class of environments. ⊕ denotes Good, ⊖ denotes Evil and ♡ is the agent.

Raw rewards for an agent π are defined as the inverse of the Euclidean distance plus 1 to *Good* minus the inverse of the Euclidean distance plus 1 to *Evil*, i.e.:

$$r = \frac{1}{d(\pi, Good) + 1} - \frac{1}{d(\pi, Evil) + 1}$$

and using the surpassing rule to compute distances (i.e., toroidal distance). Thus, the distance between cell $(1, 5)$ and $(9, 5)$ is 1.

It is easy to see that these environments are reward-sensitive (note that both *Good* and *Evil* cannot be in the same cell), and they are also balanced (one *Good* and one *Evil* with a symmetric reward function).

Given this setting, we can enumerate all the possible behaviours for *Good* and *Evil* and compute their $Kt^{\max}$. One of the simplest possible environments μ is the following one:

$$path_{Good} = L$$
$$path_{Evil} = R$$

In this environment μ, a random agent would score 0 (as it could not be otherwise since all the environments in this class are balanced). A slightly more intelligent agent could move randomly until it gets a reward that is greater than 0.5. For instance, an example that this is possible is when *Good* is at $(4, 3)$, *Evil* at $(6, 7)$, and the agent is at $(4, 3)$ where the reward is $(1 - 1/(5 + 1)) = (1 - 0.17) = 0.83$. Note that a reward that is greater than 0.5 always implies that the agent is at the same position as *Good*. When this is the case, the agent will move left all the time with rewards close to 1. If a collision between *Good* and *Evil* takes place, in half of the cases, it will make a reward of less than 0.5. In this case, the agent will start moving randomly again until a reward greater than 0.5 is found. With this policy, it is clear that the average reward will be close to 1 in the limit.

This example is a good class of environments to show that the agent can interact, in some way, with other simulated agents (very naïve in this case).

6.4. A general spatial environment class

The above example can be easily extended to include objects. The grid can be converted into any graph with a different (and variable) topology and actions (e.g. using walls, as mazes), and many more objects and agents can be introduced using Turing-complete languages to generate their movements. The more we are able to generalise the better. This is what we have developed in [53], a hopefully unbiased environment class with spaces and agents with universal descriptional power, which can be summarised as follows:

- Space (cells and actions): The space is defined as a directed labelled graph of nodes (or vertices), where each node represents a cell and arcs represent actions. The topology of the space can be quite varied. Since it is generated by a randomly-generated set of rules (using a universal distribution), it can include any complex topology.
- Objects/agents: Cells can contain objects (agents). Objects can have any behaviour as long as they follow the space topology. Agents can be reactive to other agents and can be defined to act deterministically (or not) with different actions according to their observations. Objects perform one and only one action at each interaction of the environment. *Good* and *Evil* are special agents that must have the same behaviour.
- Observations and actions: Actions allow the evaluated agent to move in the space. Observations show the (adjacent) cell contents.
- Rewards: We use the notion of reward trace (left by *Good* and *Evil*) and the notion of "cell reward". An intuitive way of seeing this is that *Good* and *Evil* leave a positive and a negative trace, respectively.

For the space (the graph), and the behaviour of all the agents, we use a Turing-complete language based on rewriting rules (Markov algorithms). Of course, this makes it difficult to compute $Kt^{\max}$ (especially for agents because, unlike the space, we need to check that the computation ends for each interaction). This environment class is shown in [53] to be balanced and reward-sensitive, and an interface has been defined so that it can be used to test biological subjects and machines. For more details we refer the reader to [53].

6.5. A game environment class

In this last example of environment class, we are going to focus on games, which is a restricted (but still important) set of tasks in artificial intelligence. There have been several game languages that have been used to describe games in a general way. For instance, Metagame [54,55] is a generalisation of symmetric chess-like games, which uses generative grammars to create many new games within a family or class of games. A more elaborate approach is the Game Description Language (GDL), which is used for the AAAI General Game Playing Competition [56]. The first problem of using GDL in our tests is that the computation of $Kt^{\max}$ in this language would be very difficult. For example, tic-tac-toe (noughts and crosses) is represented in about 40 rules. It can surely be expressed with fewer rules, but calculating this is a challenging problem with current computing power even for this simple game. Also, it would be very difficult to define the notion of balanced environments for these games. We should change scores in order to make a random agent score 0, which is not a typical result for random agents in games. Furthermore, many of them are not reward-sensitive. Nonetheless, we could use the number of rules (or characters) as an approximation of $Kt^{\max}$ for each of the 90 games that have already been coded. We could then calibrate the games by executing them on random agents repeatedly (and change the scores accordingly), and we could also check whether there is a dead-end in some of the games in order to ensure sensitivity. With this, we could construct a pool of games to execute the anytime test.

The advantages of this arrangement (over the selection and contest rules of the AAAI General Game Playing Competition) are that the test would be anytime, and the complexity of games would be adjusted automatically by using a (rough) approximation to $Kt^{\max}$. An interesting observation here is that the notion of difficulty of a game is generally considered unrelated to the length of the shortest description for the game (its Kolmogorov complexity, see e.g. [4, Section 6]). For instance, the game *Go* is a very simple game in terms of its rules, but it gets more elaborate as more matches are played. In other words, it is one thing to solve a game (i.e., to play optimally, as was recently done in [46] with checkers), and it is a different thing to simply learn the rules of play. It is much easier to learn the rules of *Go* than to learn the rules of Chess, and this is clearly related to the size of the description of the game. In fact, in [38,39], an agent learns to play tic-tac-toe and a partially observable pac-man from scratch (including the rules of the game) by only using rewards. This *conditioning* setting is enough for the agent to learn what the legal moves are, and what the goals of the game are. As a result, trying to adapt the anytime testing philosophy and concepts to this environment class would be a great source of results and insights into the area of (artificial intelligence in) game playing.

7. Discussion

There is a common thesis in all intelligence tests (which is relatively well known in psychometrics and comparative cognition) that states that *the time which is required to evaluate the intelligence of a subject depends* (1) *on the knowledge or features which are known about the subject* (*same species, same culture, same language, etc.*) *and* (2) *on the adaptability of the examiner*. The former (1) is taken to the extreme in human classical psychometrics (except for C.A.T.), where tests are just a form with questions explained in natural language. The latter (2) is taken to the extreme with the Turing test or any kind of interview evaluation, where the examiners are humans who dynamically adapt their questions according to the subject's answers. In this work, since we do not want to assume anything about the subject and we want to evaluate any intelligence range, then we are forced to devise an adaptive test. Adaptivity has many advantages, even in the case of evaluating one single species or a segment from it as C.A.T. and Item Response Theory practitioners advocate. However, adaptivity becomes necessary when we want to get the best assessment of any kind of agent (human, non-human animal or machine) in a limited period of time. A different aspect is how to specifically design this adaptivity. We have made some choices, but there may obviously be other possibilities.

An interesting discussion is that the adaptivity we consider here is a non-intelligent adaptability. An example of intelligent adaptability of the test is precisely the case where the examiner is an intelligent system. The prototypical case is when the examiner is a human, such as in a Turing test or in any classical informal psychological test. We do not discard a possible evaluation by an artificial intelligence examiner, provided it is clear how the examiner operates in order to ensure no bias for any of the examinees. For the time being, simpler notions of adaptive examiners such as those included in this paper suggest a huge amount of possibilities, discussion, and material for further investigation.

7.1. Main new features

Let us sum up the changes we have made to the original definition of Universal Intelligence. Legg and Hutter's definition had three main formal limitations: the use of uncomputable K; the use of all the environments; and the use of all the interactions. Their definition also had other trickier problems (time was ignored, the possible use of very slow environments)

Table 4
Relation among intelligent agents, intelligence definitions, and tests.

Envinonment	Time	Universal agent	Universal definition	Universal tests
Passive	No	Solomonoff prediction [31]	Comprehension ability based on C-test [7]	C-test [6], induction-enhanced Turing test [3,4]
Active	No	AIXI [44]	Universal intelligence (infinite number of environments and interactions, simple environments having more weight) [9]	Impossible (as originally defined)
Active	No	Brute-force or computationally intractable algorithms (such as AIXI) [44]	Universal intelligence (finite number of environments and interactions, use of $Kt^{\max}$, average rewards, complex environments aggregate simpler environments)	Definition 15 (parametric, Υ^{iii}), Definition 19 (anytime, Υ^{vi})
Active	Yes	Levin search agent [36], Monte Carlo AIXI [38,39], AIMML	Universal intelligence (finite number of environments and interactions, use of $Kt^{\max}$, average rewards, complex environments aggregate simpler environments, considering time)	Definition 17 (parametric, Υ^{iv}), Definition 18 (anytime, Υ^{v})

and other general issues that affect any measurement based on several exercises (how to weight them, especially if there are infinitely many).

The following items summarise the main features of the various new intelligence tests we have introduced:

- The distribution of environments is based on $Kt^{\max}$ (a bounded and computable version of K). There are many reasons for this: we cannot wait indefinitely for the environment; it is also computable and allows us to make the sample.
- The definition now includes a sample of environments, instead of all environments. The most important constraint to make this sample more discriminative is that the environment must be reward-sensitive.
- In the anytime versions of the test, the complexity of the environments is also progressively adjusted in order to make the test more effective and less dependent on the chosen distribution and preference over simple or complex environments.
- Interactions are not infinite. Rewards are averaged by the number of actions instead of accumulated. This makes the score expectation less dependent on the available test time.
- Time is included. The agent can only play with a single environment for a fixed time. This time limit progressively grows to make the test anytime.
- Rewards and penalties are both included (rewards can range from -1 to 1). Environments are required to be balanced, meaning that a random agent would score 0 in the limit in these environments. Otherwise, a very inept but proactive/quick agent would obtain good results.

Some other less relevant features have also been introduced to make the measurement consistent and feasible, especially in the anytime version of the test.

7.2. Applicability and implementation

The above modifications shape several new intelligence tests and hence new intelligence definitions. Some tests include time and others do not. In the same way, some tests are anytime and some others are not. In order to clarify this, the relation among agents,[27] definitions and tests is summarised in Table 4.

Although the recommended test is the anytime test that considers time (Definition 18), Table 4 suggests that several kinds of tests can coexist and the most suitable one can be chosen depending on the application, the subjects, or the environment class. In fact, the different choices of either ignoring time or taking time into account can be useful in refining or getting a better understanding of the relation between speed and intelligence for a specific individual. For instance, Definitions 15 and 19 ignore time, so they should give the same score to two agents π_1 and π_2, where π_2 behaves exactly the same as π_1 but π_2 is slower. On the other hand, consider an agent π'_1 that improves with experience (i.e., as long as it gets more observations, its reward expectancy increases) and another agent π'_2 that behaves exactly the same as π'_1 but π'_2 is n times slower (with $n > 1$), meaning that it takes n times longer to perform the same action in the test than π'_1 does. For both Definitions 17 and 18, we could expect that the intelligence score for π'_1 should be greater than for π'_2.

[27] AIMML in Table 4 refers to a modification of AIXI that would consider the best action following MML induction (a single theory) instead of a Bayesian posterior aggregation of theories as AIXI does.

According to this rationale, a good practice to evaluate an unknown agent would be to first apply the time-sensitive anytime test (Definition 18) and then use the time-insensitive anytime test (Definition 19) to see whether the bad or good results can be attributed to a bad or good intelligence level or because the agent is too fast or two slow. For instance, if we get very bad results for a test with Definition 18 and then very good results with a test with Definition 19, we can conclude that we have a very slow, but intelligent, agent. However, for a slow agent, Definition 19 would require a lot of time. Another option is to use the history of rewards and times used in the anytime version of the test to calculate a pair of scores (potential intelligence, speed) instead of a single aggregated value.

With regard to the implementation of the tests, many issues appear, especially if we want to find a single environment class that can be used to evaluate, for instance, adult humans, children, robots, software agents, chimpanzees, and dolphins. With the examples in Section 6 we have got an idea of what a real test on universal environments could look like. The question of feasibility is obviously a crucial one, since in any case some approximations should be used. It is clear that the generation of a pool of environments will require more effort, time, and computational power than the C-test required (which also was based on a Turing-complete machine, an accumulator machine, and the use of a computable, but intractable, complexity function, such as Kt). The main reason is that now we deal with environments, and not with sequences, making things harder. But again, the startpoint is a computable definition, and the part that is computationally the hardest only needs to be completed offline.

In any case, we do not expect to generate environments such as "checkers" randomly using any reasonable (and not-biased) environment class, as we do not find these problems in IQ tests. What we require is to generate a set of environments of different complexity that can be used in a test, in such a way that the intelligence of the system is inferred from the test, without measuring the success in real scenarios. This is exactly the approach that was taken in IQ tests for almost a century in order to evaluate human intelligence. Instead of arbitrarily constructing and selecting the exercises from previous tests to existing intelligent machines (which we may or may not have), we propose constructing the exercises from a theoretical base and objectively determining their a priori complexity.

It is also useful to see some related concepts that can also help to implement the tests. Since psychometric tests are quite similar to the C-tests, or an imitation contest is similar to a Turing test, a question that arises here is: Is there something we know that would be comparable to these "anytime intelligence tests"? The answer is affirmative: games, children's games, where children start with easy ones and soon move to more sophisticated ones. To find an even closer parallelism, the anytime intelligence test is quite similar to a random collection of videogames where players play for a short time with the easy ones first and then they are given more time to play with other more difficult ones if they succeed on the easiest ones. As usual, easy levels can be passed quickly and difficult levels require more time.

The difference here is that the environments and interactions are much more carefully chosen and controlled, and the main goal is not to choose the environments that best entertain the agent, but those that are more discriminative. In fact, many experiments in animal cognition have taken place with virtual environments, which very much resemble computer games (the evolution from 2D to 3D perception on the environments has also taken place in this area, see, e.g. [57]).

The selection of tasks and environments in psychometrics for human and non-human animals, as well as in artificial intelligence, is typically performed in such a way that only a specific cognitive ability (or frequently, just a task ability) is evaluated, such as memory, planning, pattern recognition, chess, etc. In our case, the use of a sample of environments that is universal helps the test to measure the ability to perform well in a variety of environments. Consequently, the main criticism of the C-test and compression-based tests, i.e., *that only induction/compression ability was measured*, is no longer valid in our setting. Of course, a biased sample of environments can favour some abilities over others, but the inaccuracy and bias can only come by the choice of the universal machine taken as a reference and by time constraints that preclude us from making thorough tests with a sufficiently large sample of environments.

Finally, a recurrent issue is whether a random selection of environments matches the environments where real intelligent species are successful. This is related to the notion of "social intelligence". In psychometrics and comparative cognition and, more recently, in artificial intelligence, the role of "social intelligence" has been more and more vindicated, versus purely instrumental tests. Some studies support the cultural intelligence hypothesis that states that the quality step in human intelligence is social intelligence, which is present to a much lower extent in apes and other animals. For instance, [17] developed several specialised tests for children and apes (chimpanzees and orangutans) that "support the cultural intelligence hypothesis and contradict the hypothesis that humans simply have more general intelligence". In particular, they "found that the children and chimpanzees had very similar cognitive skills for dealing with the physical world, but that the children had more sophisticated cognitive skills than either of the ape species for dealing with the social world".

How can we create environments so that they have intelligent agents *inside*? It is enlightening (but perhaps of little practical use) to think that some extremely complex infinite environments we consider as possible in the test could contain "life". In some of them, we could even find "intelligent beings". And, in some of them, these "intelligent beings" would be around from the very start of the interaction with the environment. The only assumption for this is to consider intelligence to be a merely computable thing. When we say that it is perhaps of little practical use, it is because the complexity of these environments is extremely high and the probability of one of them appearing by chance is almost zero. Therefore, we cannot bind the evaluation of social intelligence to this remote chance.

However, this a priori remote probability is in fact a much higher a posteriori probability if we think in terms of evolution. The notion of life, as we know it, implies several individuals and several species competing in the same environment. It is then natural to expect that any intelligent biological being has come through millions of years of evolution interacting

with other individuals, competing with other species, and possibly collaborating (and developing languages) with individuals from its own or other species. This means that a social environment should not only be probable but even necessary, and that a great proportion of the world's complexity surrounding a natural individual is given by other *agents* that we usually refer to as animals and plants. Consequently, we require inserting these other agents into the environments (see [32, Section 0.2.7] for a similar discussion).

One option would be to first evaluate many agents alone (including the agent that we want to be evaluated). Then we could *insert* some agents of similar intelligence into some environments (many possibilities exist here) and then evaluate the behaviour of the agent in these enriched environments where agents have to compete and/or collaborate, using, e.g., the anytime test framework.

8. Conclusions

This paper represents a very important challenge which might have strong and direct implications in many fields (e.g., artificial intelligence, psychometrics, comparative cognition, and philosophy). We have developed a set of tests and, especially, an anytime intelligence test that can be applied to any kind of intelligent system (biological or artificial, fast or slow).

The name *anytime* comes from the idea that we can obtain a rough approximation of the intelligence of an agent in a small amount of time and much better approximations in more time. The term also originates from the fact that we introduce time in the definition of intelligence and we also adapt the time scale to the agent's in order to be able to evaluate very slow and very fast intelligent agents, by also incorporating these times into the measurement.

It is fair to recognise that the first anytime intelligence test in artificial intelligence is the Turing test. The Turing test requires a human for its implementation, it is anthropomorphic, and it tests humanity rather than intelligence; however, it is anytime at least in the sense that the more time the examiner is allowed to interact with the examinee, the higher the accuracy of the result. Any test based on an interview (by a human) is usually anytime, since it is generally adaptive.

One of the key claims and hypotheses that is considered here is that intelligence is defined as an average that converges in the limit, i.e., for infinitely large environments. Many very complex environments have simple patterns and, consequently, the performance of an agent that is slightly better than random should be slightly greater than 0. Note that for this to be true, it is important to realise that no "dead code" appears in the environments; therefore, using a $Kt^{\max}$ (or K) to evaluate the complexity of the environments is a *conditio sine qua non* (using the length of the description of the environment or some other notions of complexity not based on algorithmic information theory would not work).

Some other less determinant choices are our selection of environments. We have restricted environments to be reward-sensitive and to be balanced, which changes rewards from a range between 0 and 1 into a range between -1 and 1, which must be centred for random agents. We think that in general we can modify any environment to be balanced, while preserving its essence. In fact, as we show in some examples in Section 6, we present environment classes with universal behaviours that comply with all these properties.

Despite its scientific interest from a theoretical point of view, we expect that, in the near future, practical applications and a plethora of test instances and variants may arise for multi-agent systems, games, collaborative platforms, social networks, psychometrics, animal comparative cognition, etc. In the medium term, the results of this research will be of utmost relevance to grade, classify, and certify a surfeit of intelligent agents and bots, according to their universal intelligence.

More precisely, the acceptance and use of these tests could allow new research breakthroughs to take place:

- Progress in artificial intelligence could be boosted because systems would be evaluated. Contests and competitions would foster and provide enormous feedback information to improve intelligent systems. This would not only be possible on artificial general intelligence with universal reference machines, but we could also evaluate restricted artificial problems (mazes, sequence predictions, classification problems, games, etc.) using restricted versions of the reference machines (classes of environments), like those shown in some examples here, and a proper choice of the samples and assessment of the complexities.
- New generations of CAPTCHAs [19,20] that take into account some of the ideas of these tests could be developed. For instance, CAPTCHAs are usually non-interactive, and they typically have one single question. In the medium term, more sophisticated CAPTCHAs will be needed since it is becoming easier and easier to crack them by bots. In the long term, very fast adaptations of the anytime tests could be used instead.
- Certification tests would be devised in order to automatically decide whether an unknown agent can be accepted in a service or a collaborative project. In other words, we would be able to establish cognitive requirements in order to admit an agent in a project, service, or application. If it passes, then we can make the agent/assistant learn its tasks through the use of rewards.
- In the long term, these tests will be necessary to determine when we reach the "Technological Singularity" (the point in evolution where a species is able to build a system as intelligent as itself[28]). This point is placed by some (optimistic)

[28] The term 'singularity' can be traced back to a conversation between Von Neumann and Ulam which is referred by Ulam [58] in 1958. A more explicit link to artificial intelligence is made through the term "intelligence explosion", introduced by Good [59] in 1965. The magnitude of the "future shock" that we can expect from our AI expanded scientific community and on social effects was first analysed by Solomonoff in 1985 as the "infinity point" [60]. Finally, Vinge [61] popularised the term "Technological Singularity" in the 1990's as we know it today.

researchers about twenty years from now. This means that in approximately that time, a battery of tests for different kinds of factors and environment classes will be required, since intelligent systems will converge sooner on some intelligent factors than others. And once the technological singularity is surpassed, we will require a test to measure the evolution of intelligence beyond human intelligence (and it is clear that the Turing test or related contests will not be useful for that). Additionally, a test like the one presented here (and the theory behind it) will help to focus the ethical debate that will be generated as the moment of the singularity nears.

Making an analogy between machine intelligence tests and psychometrics, we would expect at least the same applications that psychometrics have today in education, recruitment, and therapy in the world of artificial intelligence, where the areas would be knowledge acquisition, agent cognitive certification, and intelligent system (re)design.

As a consequence, future work around many different lines is possible. Implementation and experimentation using the tests are at the top of the list. Experimentation on any kind of subjects (humans, non-human animals, artificial intelligent systems[29]) would bring valuable information from which we could learn lessons and improve the tests. The battery of experiments should be enlarged with the use of different classes of environments. The implementation using universal machines and its administration to real subjects (e.g., humans, as we did with the C-test in [6,7]) is certainly a challenge because of the computational resources needed, but it is one of the combinations where we can obtain the most useful information. Nevertheless, implementations on restricted environment classes are also of utmost relevance in artificial intelligence. We now have a general theory of how to evaluate agent performance for specific fields such as optimisation and control problems, machine learning tasks, games, etc. Many specific areas in artificial intelligence have different notions of complexity and different "standards" to evaluate the performance of their systems in their tasks. For instance, "maze learning" is clearly a problem (like any other problem) that can be interpreted as a restricted environment class. In fact, a maze is not very different from the environment that we used in the example of Section 6.3. Zatuchna and Bagnall [62], for instance, analyse mazes used in research in the last two decades, develop a specific measure of "complexity", and try to determine which kind of learning agents behave best depending on the complexity, by also using a specific measure of performance (based on correctness, convergence, and memory). Extensive works, such as Zatuchna and Bagnall's paper, are not frequent (because they require an enormous amount of work), but they are crucial for the real evaluation of progress in artificial intelligence. In our opinion, these evaluation works would be much more productive if they could be homologated under a grounded and common measurement of performance. The evaluation made in Zatuchna and Bagnall's paper (and many others, such as [63]) can be done as an instance of the anytime intelligence test.

Much needs to be done on the reliability and optimality of the test. Constructs from Computerized Adaptive Testing and Item Response Theory (IRT) can be adapted here. An interesting open problem is whether it is possible to determine a theoretical item response function given an environment. This would allow a direct adaptation of IRT here. The relation between speed and intelligence is also an area where further research is needed. We think it is possible to develop tests that are able to measure intelligence and speed at the same time, without a batch combination of tests as we suggested in Section 7.

There is also much theoretical work ahead. Some of the decisions that we made in some of the definitions could be presumably refined or improved. Some theoretical results could be obtained for some of the tests (convergence, optimality, etc.), as well as some expected scores proven for different kinds of agents and classes of environments (as Legg and Hutter do for the AIXI agent and as we have (trivially) done here for random agents). In this regard, the completion of the taxonomy of environments, as it appears in [29], is one of the most appealing things to do first, including the new environment definitions that we have introduced here. A formalisation of the notion of social environment, its parametrisation, and its inclusion in the taxonomy would also be necessary.

Overall, the practical experimentation and the theoretical exploration of the notion of anytime universal intelligence test will help understand the implications of some of the choices introduced in this paper.

Acknowledgements

This work has benefited from discussions and suggestions made from many people from different areas. We thank Shane Legg and Marcus Hutter for some early discussions in 2005 about the relation between their work and ours. These discussions and their work on universal intelligence finally provided us with enough motivation (and also some more scientific basis and constructs) to resume our earlier work and embark on this work. The primitive idea of an anytime intelligence test and its applicability to areas out of artificial intelligence matured from insightful comments from María Victoria Hernández-Lloreda and Sergio España. Finally, this paper has been greatly improved with the comments and suggestions from the reviewers. The authors are also grateful for a grant from the Spanish *Ministerio de Educación y Ciencia* (MEC) and funding from Monash University during 2004 for a three-month research stay of one of the authors to collaborate with the other, as well as the funding for the MEC projects EXPLORA-INGENIO TIN 2009-06078-E, CONSOLIDER-INGENIO 26706 and TIN 2007-68093-C02, and GVA project PROMETEO/2008/051.

[29] Intelligent systems, either biological or artificial, can be typically evaluated as individuals, but they can also be evaluated as a group, society, colony or swarm. This is especially the case when the individuals are very simple entities that collaborate to have higher 'emergent' intelligence. See [32, Section 0.2.7, p. 545, Col. 1] and [42, Section 7.3] for further discussion.

References

[1] A.M. Turing, Computing machinery and intelligence, Mind 59 (1950) 433–460.
[2] M. Li, P. Vitányi, An Introduction to Kolmogorov Complexity and Its Applications, 3rd ed., Springer-Verlag, New York, 2008.
[3] D.L. Dowe, A.R. Hajek, A computational extension to the Turing test, in: Proceedings of the 4th Conference of the Australasian Cognitive Science Society, University of Newcastle, NSW, Australia, 1997.
[4] D.L. Dowe, A.R. Hajek, A computational extension to the Turing test, Technical report #97/322, Dept. Computer Science, Monash University, Melbourne, Australia, 9 pp., http://www.csse.monash.edu.au/publications/1997/tr-cs97-322-abs.html.
[5] D.L. Dowe, A.R. Hajek, A non-behavioural, computational extension to the Turing test, in: International Conference on Computational Intelligence & Multimedia Applications (ICCIMA'98), Gippsland, Australia, 1998, pp. 101–106.
[6] J. Hernández-Orallo, 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), ICSC Press, 1998, pp. 146–163.
[7] J. Hernández-Orallo, Beyond the Turing test, Journal of Logic, Language and Information 9 (4) (2000) 447–466.
[8] S. Legg, M. Hutter, A universal measure of intelligence for artificial agents, in: International Joint Conference on Artificial Intelligence, vol. 19, 2005, p. 1509.
[9] S. Legg, M. Hutter, Universal intelligence: A definition of machine intelligence, Minds and Machines 17 (4) (2007) 391–444, http://www.vetta.org/documents/UniversalIntelligence.pdf.
[10] C.S. Wallace, D.M. Boulton, An information measure for classification, Computer Journal 11 (2) (1968) 185–194.
[11] C.S. Wallace, D.L. Dowe, Minimum message length and Kolmogorov complexity, Computer Journal 42 (4) (1999) 270–283, Special issue on Kolmogorov complexity.
[12] C.S. Wallace, Statistical and Inductive Inference by Minimum Message Length, Springer-Verlag, 2005.
[13] P. Sanghi, D.L. Dowe, A computer program capable of passing IQ tests, in: Proceedings of the 4th ICCS International Conference on Cognitive Science (ICCS'03), Sydney, Australia, July 2003, pp. 570–575.
[14] C. Spearman, General intelligence, objectively determined and measured, American Journal of Psychology 15 (2) (1904) 201–292.
[15] S.E. Embretson, K.M.S. McCollam, Psychometric approaches to understanding and measuring intelligence, in: R. Sternberg (Ed.), Handbook of Intelligence, Cambridge University Press, 2000, pp. 423–444.
[16] S.E. Embretson, A cognitive design system approach to generating valid tests: Application to abstract reasoning, Psychological Methods 3 (3) (1998) 380–396.
[17] E. Herrmann, J. Call, M.V. Hernández-Lloreda, B. Hare, M. Tomasell, Humans have evolved specialized skills of social cognition: The cultural intelligence hypothesis, Science 317 (5843) (2007) 1360–1366.
[18] G. Oppy, D.L. Dowe, The Turing test, in: E.N. Zalta (Ed.), Stanford Encyclopedia of Philosophy, Stanford University, 2008, http://plato.stanford.edu/entries/turing-test/.
[19] L. Von Ahn, M. Blum, J. Langford, Telling humans and computers apart automatically, Communications of the ACM 47 (2) (2004) 56–60.
[20] L. Von Ahn, B. Maurer, C. McMillen, D. Abraham, M. Blum, RECAPTCHA: Human-based character recognition via web security measures, Science 321 (5895) (2008) 1465.
[21] R. Madhavan, E. Tunstel, E. Messina, Performance Evaluation and Benchmarking of Intelligent Systems, Springer, September 2009.
[22] L.A. Zadeh, Fuzzy logic, neural networks, and soft computing, Communications of the ACM 37 (3) (1994) 84.
[23] L.A. Zadeh, Toward human level machine intelligence — Is it achievable? The need for a paradigm shift, IEEE Computational Intelligence Magazine 3 (3) (2008) 11–22.
[24] V.C.I. Ulinwa, Machine Intelligence Quotient, VDM Verlag, Saarbrücken, 2008.
[25] J. Laird, A. Newell, P. Rosenbloom, Soar: An architecture for general intelligence, Artificial Intelligence (33) (1987) 1–64.
[26] J.E. Laird, Extending the Soar cognitive architecture, in: P. Wang, S. Franklin (Eds.), Artificial General Intelligence 2008: Proceedings of the First AGI Conference, IOS Press Inc., 2008, pp. 224–235.
[27] J.E. Laird, R.E. Wray III, Cognitive architecture requirements for achieving AGI, in: M. Hutter, E. Baum, E. Kitzelmann (Eds.), Proceedings of the 3rd International Conference on Artificial General Intelligence AGI, in: Advances in Intelligent Systems Research, Atlantis Press, 2010, pp. 79–84.
[28] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, 1998.
[29] S. Legg, Machine Super Intelligence, Department of Informatics, University of Lugano, June 2008.
[30] M. Hutter, General discounting versus average reward, in: J.L. Balcazar, P.M. Long, F. Stephan (Eds.), ALT, in: Lecture Notes in Computer Science, vol. 4264, Springer, 2006, pp. 244–258.
[31] R.J. Solomonoff, A formal theory of inductive inference. Part I, Information and Control 7 (1) (1964) 1–22.
[32] D.L. Dowe, Foreword re C.S. Wallace, The Computer Journal 51 (5) (2008) 523–560, Christopher Stewart Wallace (1933–2004) memorial special issue.
[33] R.J. Solomonoff, Does algorithmic probability solve the problem of induction?, in: D.L. Dowe, K.B. Korb, J.J. Oliver (Eds.), Proceedings of the Conference on Information, Statistics and Induction in Science (ISIS), World Scientific, Melbourne, Australia, ISBN 981-02-2824-4, 1996, pp. 7–8.
[34] M. Hutter, Open problems in universal induction and intelligence, Algorithms 2 (3) (2009) 879–906.
[35] J. Schmidhuber, The speed prior: A new simplicity measure yielding near-optimal computable predictions, in: Computational Learning Theory, Springer, 2002, pp. 123–127.
[36] L.A. Levin, Universal sequential search problems, Problems of Information Transmission 9 (3) (1973) 265–266.
[37] M. Hutter, Universal algorithmic intelligence: A mathematical top → down approach, in: B. Goertzel, C. Pennachin (Eds.), Artificial General Intelligence, Cognitive Technologies, Springer, Berlin, 2007, pp. 227–290.
[38] J. Veness, K.S. Ng, M. Hutter, D. Silver, A Monte Carlo AIXI approximation, in: CoRR, preprint arXiv:0909.0801, 2009.
[39] J. Veness, K.S. Ng, M. Hutter, D. Silver, Reinforcement learning via AIXI approximation, in: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI'10), 2010, pp. 605–611.
[40] J. Searle, Minds, brains, and programs, Behavioral and Brain Sciences 3 (3) (1980) 417–457.
[41] M. Mahoney, Text compression as a test for artificial intelligence, in: Proceedings of the National Conference on Artificial Intelligence, AAAI/John Wiley & Sons Ltd., 1999, p. 970.
[42] D.L. Dowe, MML, hybrid Bayesian network graphical models, statistical consistency, invariance and uniqueness, in: J. Wood, M.R. Forster, P. Bandyopadhyay (Eds.), Handbook of the Philosophy of Science — Philosophy of Statistics, Elsevier, 2010, pp. 861–942.
[43] J. Hernández-Orallo, On the computational measurement of intelligence factors, in: A. Meystel (Ed.), Performance Metrics for Intelligent Systems Workshop, National Institute of Standards and Technology, Gaithersburg, MD, USA, 2000, pp. 1–8.
[44] M. Hutter, Universal Artificial Intelligence: Sequential Decisions Based on Algorithmic Probability, Springer, 2005.
[45] B. Hibbard, Bias and no free lunch in formal measures of intelligence, Journal of Artificial General Intelligence 1 (1) (2009) 54–61.
[46] J. Schaeffer, N. Burch, Y. Bjornsson, A. Kishimoto, M. Muller, R. Lake, P. Lu, S. Sutphen, Checkers is solved, Science 317 (5844) (2007) 1518.
[47] M. Agrawal, N. Kayal, N. Saxena, PRIMES is in P, Annals of Mathematics 160 (2) (2004) 781–793.
[48] J. Hernández-Orallo, On evaluating agent performance in a fixed period of time, in: M. Hutter, E. Baum, E. Kitzelmann (Eds.), Proceedings of the 3rd International Conference on Artificial General Intelligence AGI, in: Advances in Intelligent Systems Research, Atlantis Press, 2010, pp. 25–30.

[49] H.E. Wainer, Computerized Adaptive Testing: A Primer, 2nd ed., Lawrence Erlbaum Associates, Mahwah, NJ, 2000.
[50] F.M. Lord, Applications of Item Response Theory to Practical Testing Problems, Erlbaum, Mahwah, NJ, 1980.
[51] S.E. Embretson, S.P. Reise, Item Response Theory for Psychologists, Lawrence Erlbaum, 2000.
[52] S.E. Embretson, Measuring human intelligence with artificial intelligence, in: R.J. Sternberg, J. Pretz (Eds.), Cognition and Intelligence: Identifying the Mechanisms of the Mind, Cambridge University Press, 2005, pp. 251–267.
[53] J. Hernández-Orallo, A (hopefully) non-biased universal environment class for measuring intelligence of biological and artificial systems, in: M. Hutter, E. Baum, E. Kitzelmann (Eds.), Proceedings of the 3rd International Conference on Artificial General Intelligence, in: Advances in Intelligent Systems Research, Atlantis Press, 2010, pp. 182–183.
[54] B. Pell, Strategy generation and evaluation for metagame playing, PhD thesis, University of Cambridge, 1993.
[55] B. Pell, A strategic metagame player for general chesslike games, in: Proceedings of the 12th National Conference on Artificial Intelligence, Association for the Advancement of Artificial Intelligence (AAAI), 1994, pp. 1378–1385.
[56] M. Genesereth, N. Love, B. Pell, General game playing: Overview of the AAAI competition, AI Magazine 26 (2) (2005) 62.
[57] D.A. Washburn, R.S. Astur, Exploration of virtual mazes by Rhesus monkeys (Macaca mulatta), Animal Cognition 6 (3) (2003) 161–168.
[58] S. Ulam, Tribute to John von Neumann, Bulletin of the American Mathematical Society 64 (3) (1958) 1–49.
[59] I.J. Good, Speculations concerning the first ultraintelligent machine, Advances in Computers 6 (1965) 31–88.
[60] R.J. Solomonoff, The time scale of artificial intelligence: Reflections on social effect, Human Systems Management 5 (1985) 149–153.
[61] V. Vinge, Technological singularity, in: VISION-21 Symposium Sponsored by NASA Lewis Research Center and the Ohio Aerospace Institute, vol. 30, March 1993, p. 31.
[62] Z. Zatuchna, A. Bagnall, Learning mazes with aliasing states: An LCS algorithm with associative perception, Adaptive Behavior 17 (1) (2009) 28–57.
[63] D. Weyns, H. Parunak, F. Michel, T. Holvoet, J. Ferber, Environments for multi-agent systems, state-of-the-art and research challenges, in: Environments for Multi-Agent Systems, in: Lecture Notes in Computer Science, vol. 3374, Springer-Verlag, 2005, pp. 1–48, Held with the 3rd Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS.