Measuring Universal Intelligence: Towards an Anytime Intelligence Test

Jose Hernandez-Orallo[a,1], David L. Dowe[b]

[a] *Universitat Politècnica de València, Departament de Sistemes Informàtics i Computació, Camí de Vera 14, E-46022, València, Spain*
[b] *Computer Science & Software Engineering, Clayton School of I.T., Monash University, Clayton, Victoria, 3800, Australia*

ARTICLE INFO

Article history:
Submitted: December 2009

Keywords:
Measurement of Intelligence,
Artificial Intelligence,
Psychometrics,
Kolmogorov complexity,
Algorithmic Information Theory,
Turing Test,
Universal Intelligence,
Computerized Adaptive Testing,
Compression,
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 which exists at this time or in the future, and 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). Additionally, the test may be interrupted at any time, producing an approximation to the intelligence score, in such a way that the more time we leave 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 nineties (C-tests and compression-enhanced Turing tests) and 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, especially if we want to construct a both general and practical test. Consequently, we introduce many new ideas which 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 designing requirement. One of these tests is the "anytime intelligence test", which adapts to the examinee's level of intelligence to get an intelligence score within a limited time. We illustrate the way the test can look like in practice with some examples of environment classes. Finally, we explore some issues about the notion of emergent intelligence using the tests and the evaluation of social intelligence.

1. Introduction

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 [Solomonoff 1964] and related notions, such as the Minimum Message Length (MML) principle [Wallace and Boulton 1968] [Wallace and Dowe 1999a] [Wallace 2005], all of them based upon the modern mathematical theory of induction[2].

On one hand, Dowe and Hajek [Dowe and Hajek 1997a, 1997b, 1998] suggested the introduction of inductive inference problems in a somehow *induction-enhanced* or *compression-enhanced* Turing Test [Turing 1950] (they arguably called it non-behavioural), in order to, among other things, completely dismiss Searle's Chinese room [Searle 1980] objection, and also because an inductive inference ability is a necessary (though possibly "not sufficient") requirement for intelligence.

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

[2] 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. However, these notions are frequently used as synonyms. In fact, Solomonoff's seminal paper [Solomonoff 1964] refers to the "theory of inductive inference" while it actually refers to a "theory of prediction". Additionally, 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 [Wallace 2005] (sec. 10.1) and [Dowe 2008a] (part of sec. 0.3.1 referring to Solomonoff) for more details on this.

Quite simultaneously and similarly, and also independently, in [Hernandez-Orallo and Minaya-Collado 1998] [Hernandez-Orallo 2000a], 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[3], derived from Solomonoff prediction theory concepts, Kolmogorov complexity (also known as Algorithmic Information Theory) and Levin's variants and optimal search. The definition was given as the result of a *test*, called C-test [Hernandez-Orallo and Minaya-Collado 1998], formed by computationally-obtained series of increasing complexity. The sequences were formatted and presented in a quite similar way to psychometric tests, and as a result, the test was administered to humans. Nonetheless, the main goal was that the test could eventually be administered to other kinds of intelligent beings and systems.

Since then, the main controversy has been around the assumption that intelligence is all about induction, prediction and compression. The case against this is that some people who are commonly considered intelligent, do only excel on some specific cognitive abilities (e.g. people with a great memory who win TV quiz shows, mathematicians who excel on deductive abilities, psychopathic killers who plan and calculate all their movements to mislead the police, etc.). 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., section 5.2 in [Sanghi and Dowe 2003]).

Nonetheless, a compression test was already considered limited since its inception. A factorisation (and hence extension) of these inductive inference tests was sketched in order to explore which other abilities could shape a complete (and hence sufficient) test [Hernandez-Orallo 2000b]. 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)" [Hernandez-Orallo 2000a].

Recent works by Legg and Hutter (e.g. [Legg and Hutter 2005],[Legg and Hutter 2007]) have followed the previous steps and, strongly influenced by Hutter's theory of AIXI optimal agents [Hutter 2005], 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 many different environments. An environment is understood as it is usually done in artificial intelligence, comparative cognition or psychology, i.e., an interactive real or artificial environment where an agent receives observations, can make actions and can get some rewards depending on its behaviour.

The comparison with the works from Hernandez-Orallo (and also Dowe and Hajek's compression test) is summarised in [Legg and Hutter 2007] with the following table:

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

Table 1. Intelligence tests in passive and active environments (from [Legg and Hutter 2007]).

In fact, the definition based on the C-test can be now considered a static precursor of Legg and Hutter's work, where the environment outputs no rewards, and the agent is not allowed to make an action until a number of observations are seen (the inductive inference or prediction sequence). The only important detail would be that in the C-test, the 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). In fact, in Legg's Ph.D. dissertation (section 3.1.7) [Legg 2008], it is shown that sequence prediction (which is used in Hernandez-Orallo and Dowe and Hajek's works) is a special case of what they call chronological environments (which is used in Universal Intelligence). The point in favour for active environments (in contrast to passive environments) is that the former not only require inductive abilities to model the environment but also some planning abilities to effectively use this knowledge through actions.

[3] The basic idea was to construct series whose shortest pattern has no alternative "projectible" patterns of similar complexity. That means that the "explanation" of the series has 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.

[Legg and Hutter 2007] is an excellent piece of work in terms of coverage of related work (especially in psychometrics and artificial intelligence), scope, and also because it is sufficiently broad and formal at the same time without getting into difficult technicalities. In fact, it can be understood by scientists from different fields. In our opinion, one of the most relevant contributions in their work is that their definition of Universal Intelligence allows one to formally evaluate the theoretical performance of some agents: a random agent, a specialised agent, ..., or a super-intelligent agent, such as AIXI [Hutter 2005], which is claimed to be the agent that, if ever constructed, would score the best in the Universal Intelligent test. In a nutshell, Legg and Hutter's work is another step that helps to shape the things that should be addressed in the near future in order to eventually reach a theory of machine intelligence evaluation.

Their work has a special section which addresses 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 [Legg and Hutter 2007], [Dowe and Hajek 1997a, 1997b, 1998] and [Hernandez-Orallo 2000a] and we will not devote any space in this paper to typical issues on the whys and whens of machine intelligence measurement, its relation to Turing Test and other tests proposed so far.

So, taking Legg and Hutter's definition of Universal Intelligence as a basis, in the quest of 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 should, in our opinion, be understood as follows:

	Universal agent	Universal definition	Universal tests
Passive environment	Solomonoff prediction	Comprehension ability based on C-test / Inductive ability	C-test / Induction-Enhanced Turing Test (Dowe and Hajek)
Active environment	AIXI	Universal intelligence	?

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

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 [Dowe and Hajek 1997a, 1997b, 1998] [Hernandez-Orallo 2000a] have shown that evaluation is feasible (with some limitations [see, for instance, Sanghi and Dowe 2003]). In any case, [Legg and Hutter 2007] 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 contrary is true: having a general, well-grounded and formal intelligence test provides us with a definition of intelligence.

In what follows, we will analyse the limitations of the definition of universal intelligence and we will introduce many new ideas which develop previous "compression tests" and the definition of "universal intelligence" into a new "anytime intelligence test", having its actual implementation in mind. One of the key concepts which is worked out in this paper is the relation between time and intelligence and how to incorporate time in the measurement. As a result, this paper includes four operative intelligence test definitions, a first which does not include time and is not anytime (but solves many other previous issues), a second which does include time but is not anytime, a third which does not include time and is anytime and, finally, a fourth test which 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 now re-defined as an average of performance in many environments, where now complex environments are more relevant than in Legg and Hutter's definition. At a practical level, we present several feasible tests; we show some examples and figurations on how these tests should look like in practice. The tests are designed to evaluate, in a practical way, the abilities of purported intelligent agents (artificial, human, non-human animal or extraterrestrial). Although the general idea is to measure general intelligence (and this is accomplished if we use universal machines to generate the set of environments), we can also use the test we introduce here to evaluate more restricted capabilities, by appropriately choosing a class of environments. For instance, if environments were chosen such that actions only affect rewards but not observations (passive environments), we would have that the tests would be able to evaluate sequence prediction capability (ignoring planning capabilities). Similarly, using different classes of environments we would be able to efficiently evaluate performance on several tasks (classification, mazes, board games, etc.), in many (if not all) the areas in artificial intelligence. Some examples are shown at the end of this paper, where we give three examples with corresponding environment classes to evaluate very simple short-memory and inference capabilities (for apes, very simple agents or any other kind of subject), to evaluate performance in environments generated by finite state machines, and

finally to evaluate some orientation abilities where some other agents can co-exist in the environment (typical in mazes and other kinds of games).

The rest of the paper is organised as follows. In Section 2 we re-visit Legg and Hutter's definition and we introduce some notation. Section 3 discusses several problems found in this definition and we propose solutions for the selection of environments, the use of practical interactions and the aggregation of the measurement into a single score. Section 4 analyses the relation between time and intelligence, how this affects rewards, and introduces the anytime algorithm (variously both considering time and ignoring it). It also includes the main theoretical result of the paper, which supports the choice of averaging environments. Section 5 presents several examples of some interesting environment classes and some figurative evaluations. Section 6 discusses the main features of the new tests, the evaluation of social intelligence and the use of intelligence tests to boost artificial intelligence progress, related to the notion of emergent intelligence. Finally, Section 7 closes the paper with the implications of this work and the future work.

2. Definition of Universal Intelligence

We have said 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 practical for testing, we need to take a look at the definition. Their definition is based on the notion of environment, which is a common concept in artificial intelligence, psychology and biology. Following Legg and Hutter [Legg and Hutter 2007], an environment is a world where an agent can interact through actions, rewards and observations, as shown in Figure 1:

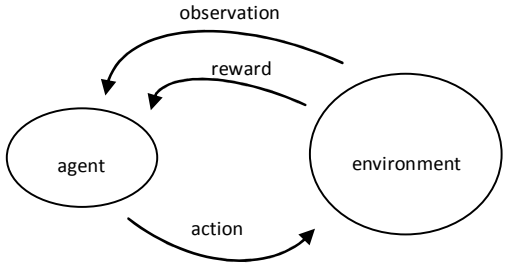

Figure 1. Interaction between an agent and an environment.

Actions are limited by a finite set of symbols A, (e.g. {*left*, *right*, *up*, *down*}), rewards are taken from any subset R of rational numbers between 0 and 1, and observations are also limited by a finite set O of possibilities (e.g., a grid of binary cells of $n \times m$, or a set of light-emitting diodes, LEDs). We will use a_i, r_i and o_i to (respectively) denote action, reward and observation at interaction or cycle i (or, more loosely, state). The order of events is always: reward, observation and action[4]. A sequence of events is then a string such as $r_1 o_1 a_1 r_2 o_2 a_2$. Both the agent and the environment are defined as a probabilistic measure. For instance, given an agent, denoted by π, the term $\pi(a_k \mid r_1 o_1 a_1 r_2 o_2 a_2 \ldots r_k o_k)$ denotes the probability of π to execute action a_k after the sequence of events $r_1 o_1 a_1 r_2 o_2 a_2 \ldots r_k o_k$. In a similar way, an environment μ is also a probabilistic measure which assigns probabilities to each possible pair of observation and reward. For instance, $\mu(r_k o_k \mid r_1 o_1 a_1 r_2 o_2 a_2 \ldots r_{k-1} o_{k-1} a_{k-1})$ denotes the probability in environment μ of outputting $r_k o_k$ after the sequence of events $r_1 o_1 a_1 r_2 o_2 a_2 \ldots . r_{k-1} o_{k-1} a_{k-1}$. Note that if all the probability (1) is given to one action/output and 0 to the rest, we then have a deterministic agent/environment. The symbol * will be used to denote any finite (possibly empty) sequence.

EXAMPLE 1.

Consider a test setting where a chimpanzee (the agent) can press one of three possible buttons ($A = \{B_1, B_2, B_3\}$), rewards are just the handing over (or not) of a banana ($R = \{0, 1\}$) and the observation is three cells where a ball must be inside exactly one of them ($O = \{C_1, C_2, C_3\}$). An example of a possible environment is:

[4] Typically, in artificial intelligence and in cognitive sciences, the sequence of events is observation, action and reward. Legg and Hutter, 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 simplify the formal definition of environment interaction as a sequence of perception-action pairs. In this paper, we use 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 which 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.

$\mu(1^* \mid r_1 o_1 a_1 r_2 o_2 a_2 \ldots r_{k-1} o_{k-1} a_{k-1})$ if ($a_{k-1} = B_1$ and $o_{k-1} = C_1$) or ($a_{k-1} = B_2$ and $o_{k-1} = C_2$) or ($a_{k-1} = B_3$ and $o_{k-1} = C_3$)

$\mu(0^* \mid r_1 o_1 a_1 r_2 o_2 a_2 \ldots r_{k-1} o_{k-1} a_{k-1})$ otherwise

The observation o_k in both cases is randomly generated with a uniform distribution between the three possibilities in O. The first reward is 1 (we start the game giving a banana to the chimpanzee).

According to the previous environment definition, a chimpanzee always selecting the button which 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.

Let us give some more notation and definitions. Given a deterministic environment μ, we respectively denote by $o_i^{\mu,\pi}$, $r_i^{\mu,\pi}$, $a_i^{\mu,\pi}$, the observation, reward and action i in environment μ when interacting with π. We denote the sequence of observations and rewards after several actions as:

$$O_\mu(a_1 a_2 \ldots a_k) := o_1 o_2 \ldots o_{k+1} \text{ after executing } a_1 a_2 \ldots a_k \text{ on } \mu.$$

$$R_\mu(a_1 a_2 \ldots a_k) := r_1 r_2 \ldots r_{k+1} \text{ after executing } a_1 a_2 \ldots a_k \text{ on } \mu.$$

We can compute the performance of the agent in the environment as the expected value of the sum of all the rewards for the agent π in the environment μ, i.e.:

DEFINITION 1. EXPECTED CUMULATIVE REWARD

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

Legg and Hutter discuss several ideas to avoid this value being infinite (in the previous example, a chimpanzee could get an infinite number bananas in the limit). In their discussion, the recurrent issue of environments where agents can be greedy or more long-term farsighted appears (an issue which is also persistent in the area of reinforcement learning). But, in the end, they come up with a single constraint they finally impose on every environment:

DEFINITION 2. REWARD-BOUNDED ENVIRONMENT

An environment μ is reward-bounded iff for all π:

$$V_\mu^\pi \leq 1$$

The previous example does not conform to this limitation because a 1 reward can be obtained several times. Let us see another example that does:

EXAMPLE 2.

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

$\mu((1/2^{k-1})^* \mid r_1 o_1 a_1 r_2 o_2 a_2 \ldots r_{k-1} o_{k-1} a_{k-1})$ if ($a_{k-1} = B_1$ and $o_{k-1} = C_1$) or ($a_{k-1} = B_2$ and $o_{k-1} = C_2$) or ($a_{k-1} = B_3$ and $o_{k-1} = C_3$)

$\mu(0^* \mid r_1 o_1 a_1 r_2 o_2 a_2 \ldots r_{k-1} o_{k-1} a_{k-1})$ otherwise

The observation o_k in both cases is randomly generated with a uniform distribution between the three possibilities in 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. $\pi(B_1 \mid X)$ 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 at 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.

All the previous definitions are more or less similar to any kind of interactive environment, which is typical in artificial intelligence or even in comparative cognition / psychology, where observations, actions and rewards are a custom.

Before evaluating performance on these environments, we have to face a crucial question. Which environments do we choose? The proposal from Legg and Hutter is straightforward: choose *all* of them. Then, given an agent, its universal intelligence Υ is given by the following definition:

DEFINITION 3. UNIVERSAL INTELLIGENCE [LEGG AND HUTTER 2007]

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

where μ is any environment coded on a universal machine[5] U, with π being the agent we want to be evaluated. With $p_U(\mu)$ we assign a probability to each environment.

Since we have infinitely many environments, we cannot assign a uniform distribution to the environments. The solution is "the" so-called universal distribution over the machine U (see e.g. [Wallace and Dowe 1999a][Wallace 2005][Li and Vitányi 2008]). This is defined over Kolmogorov Complexity, as the length of the shortest program which outputs a given string x over the machine U. Formally,

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

where $l(p)$ denotes the length in bits of p and $U(p)$ denotes the result of executing p on U. For instance, if U is the programming language Lisp and x = 1010101010101010, then $K_{\text{Lisp}}(x)$ 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 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, $|K_U(x) - K_V(x)| \leq c(U,V)$. The value of $c(U,V)$ is relatively small for sufficiently long x.

From the previous definition, we can now define the universal probability for machine U as follows[6]:

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

which gives higher probability to objects whose shortest description is small and lower probability to objects whose shortest description is large. In order to apply it to environments, we require them to be expressed as a string or sequence.

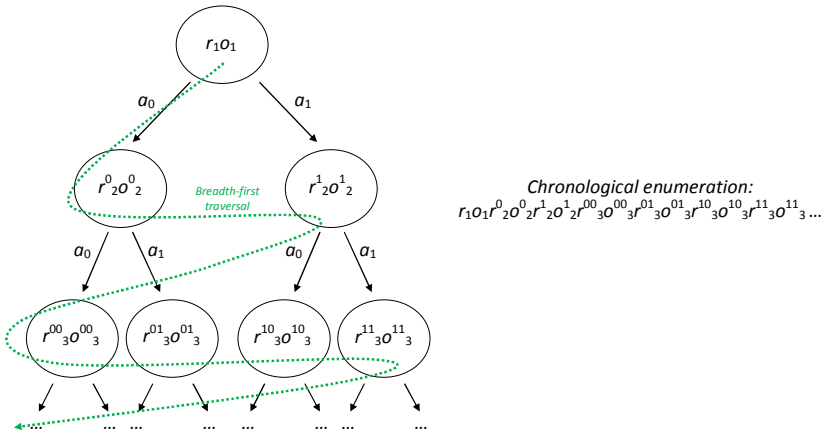

Figure 2. A tree representation of all the possible interactions in a deterministic environment (with only two actions, a_0 and a_1) and the corresponding chronological enumeration.

One way of doing this is to enumerate environments by any possible agent's action, which is known as "chronological environment enumeration", which means making a breadth-first traversal of the environment's interaction tree, as shown in Figure 2 (see [Legg 2008] for more details). Then, given an environment, the string x we use as the parameter for $K_U(x)$ will be its chronological enumeration.

Since we are using $K_U(x)$, it is important to mention that this (jointly with the enumeration) avoids any dead code in the environments. This is easy to see, since if any part of the environment description (program) is devoted to

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

[6] For a convenient definition of the universal probability, we have to add the requirement of U being a prefix-free machine (see e.g. [Li and Vitányi 2008] for details).

something which cannot be reached by any action, then the environment will necessarily have another shortest program without this part and its $K_U(x)$ must be necessarily lower.

Definition 3, although very simple, captures one of the broadest definitions of intelligence: "the ability to adapt to a wide range of environments".

3. Some Problems with the Definition of Universal Intelligence

There are three obvious problems in this definition to make it practical. First, we have two infinite sums in the definition: one is the sum over all the environments and the second is the sum over all the 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, since a machine U could give the higher probabilities (0.5, 0.25, …) to quite different environments than those given by another machine V.

Despite all the previous problems, at the end of Legg and Hutter's paper, 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. But, as we will see next, on the one hand, this is not so easy, and, on the other hand, the definition has many other (some related and some other unrelated) problems.

3.1. Selecting Discriminative Environments

The first issue is about 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 with $K()=1$ 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(), which will still omit or underweight very complex environments, because their probability is so low. As an example, the (approximately) 16 environments of complexity less or equal to 4 would have about 95% of the weight and a 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 very dependent to the reference machine. And even ignoring this, it is not clear that an agent solving all the problems of complexity lower than 20 bits and none of those of 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 on simple things would score badly.

[Hibbard 2009] proposes to only include 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. Additionally, large environments usually require more time for an agent interacting inside in order to get a reliable assessment.

Before going on we need to clarify the notions of simple/easy and complex/difficult used here. Just choosing an environment with high $K()$ does not ensure that the environment is indeed complex. Consider for instance an environment with high $K()$ which goes as follows: for any action, output the same observation o_i and reward r_i, except when the interaction i is a power of 2. In this case (and only in this case), the observation and reward must 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.

Another example of a relatively simple environment with high $K()$ is an environment which has a behaviour with no pattern until cycle i=1,000 and then it repeats the behaviour from then on indefinitely. $K()$ will be high because 1,000 cycles must be coded (and there is no pattern to be used for compression) but the pattern is relatively simple (a repetition) if the agent has a big memory and interacts for thousands of cycles to see the pattern.

The previous two examples can be applied to finite strings and environments. But environments are not just finite strings as the enumeration shown in Figure 2 suggests. Consider, e.g., an environment μ with only two possible actions such that every n-cycle sequence of actions $a_1,a_2, ..., a_n$, leads to very simple "subenvironments" (subtrees in the decision tree), except one specific sequence of actions which leads to a complex subenvironment. The complexity of this environment is high, but only 1 of the 2^n *beginnings* 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 as very simple. This example shows that the notion of easy or difficult is not the same for strings than for environments. In the case of an environment, an agent will only explore a (generally) small part of it.

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 [Li and Vitányi 2008] for details). As

Figure 3 illustrates, the relation is unidirectional; given a low $K()$ we can affirm that the environment will look simple. On the other side, given an intuitively complex environment, $K()$ must be necessarily high (if a simple pattern existed, $K()$ would be low).

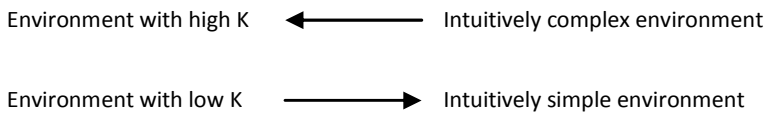

Figure 3. Relation between $K_U(\mu)$ and intuitive complexity.

Given the previous relation, only among environments with high $K()$ will we find complex environments, but not all of them will be difficult. From the agent's perspective, however, this is more extreme, since many environments with high $K()$ and with complex behaviours will be never accessed by the agent's interactions and will be *probabilistically* simple. This will be crucial later on, as we will consider environments with high $K()$ as a kind of aggregate of the environments of lower complexity. And this also means that environments with low $K()$ are typically seen by agents as simpler than they are, since many of the simple patterns that these environments include can be inaccessible, so only even simpler patterns will be shown. Consequently, giving most of the probability to environments with low $K()$ means that most of the intelligence measure will come from patterns which are dramatically simple.

After the previous discussion, we also have to realise that we are not (just) giving a formal definition of intelligence, but trying to construct an effective test. As a result, we have to clarify that one thing is the probability of an environment appearing and quite another thing is the weight that an environment should be assigned. So, a modification of the formula might be as follows.

DEFINITION 4. UNIVERSAL INTELLIGENCE (FINITE SET OF ENVIRONMENTS)

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

where S is a finite subset of m environments extracted with $p_U(\mu) = 2^{-K_U(\mu)}$

The probability $p_U(\mu)$ is, for the moment, the same as defined above. The sample S is understood as a set (i.e., a sample without repetitions), so the same environment cannot appear twice. Note that instead of averaging them through the probability, now we average them in a uniform way (just by summing all the environments in the sample S and dividing the result by its size). If S is infinite, then both definitions (Υ and Υ^{I}) are then different in the weighting of environments, so the change is more important than it seems at first sight. When we make a finite sample, since we avoid repetitions and we do not weight by the probability of computing the overall value, this means that if two environments with complexity 1 and 10 appear, both are weighted equally.

However, this modification does not solve some of the previous problems[7] but, additionally, many environments (either simple or complex) will be completely useless to evaluate intelligence, such as environments which stop interacting, or environments with constant rewards, or environments which are very similar to other previously used environments, etc. Including some, or most, of them in the sample is a waste of testing resources; if we are able to make a more accurate sample with environments we will be able to make a more efficient testing. 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 some pattern, which can be eventually learned (or not) by the examinee. But 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. Another pathological case is an environment which "outputs observations and rewards at random"[8], but 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*. So these environments would be useless in order to discriminate between agents.

The C-test [Hernandez-Orallo and Minaya-Collado 1998] 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[9] adapting

[7] For instance, it is not clear that the previous definition converges to a (finite) value greater than 0.

[8] 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 which outputs the digits of π.

[9] An approximation of this concept for environments has been studied in [Hernandez-Orallo 2009a]

this idea to environments is possible but it would require the definition of notions such as similarity between environments, which would be difficult to implement in practice.

So we will take another approach. In an interactive environment, a clear requirement for an environment to be discriminative is that what the agent does must have consequences on the observations and rewards. Without any restriction, many (most) simple environments would be completely insensitive to agent's actions. In [Legg 2008], a taxonomy of environments is developed, while distinguishing among many kinds of environments. For instance, passive environments are those for which agents' actions can only affect rewards but not observations. A special sub-category is sequence prediction as used in the C-test, the compression tests or the classical psychometric tests, and also the typical classification problems in machine learning. Some other kinds of environments are *n*th-order Markov Decision Processes (MDP), where the next observation can only depend on the *n* last observations and the *n* last actions. It can be shown that *n*th-order MDPs can be reduced to 1st-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 depends on the previous observation and action (no 'perceptual aliasing' problem). A special kind of MDPs are the ergodic MDPs, which are characterised by making any possible observation reachable (in one or more cycles) at any state. That means that at any state, agents' actions can recover from a bad previous decision. According to [Legg 2008], ergodic MDPs are one of the classes of environments that admit "self-optimising agents", the environments in which a universal agent will learn to behave optimally. However, they do not use this taxonomy and the class of ergodic MDPs to refine the definition given in [Legg and Hutter 2007].

In any case, we think that ergodic MPDs 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.

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 sub-environment. More precisely, *we want an agent to be able to affect on rewards at any point in any sub-environment*. This does not imply ergodicity but reward sensitivity at any moment. That means that we cannot reach a point from which rewards are fixed independently of what we do (a dead-end). This can be formalised as follows:

DEFINITION 5. REWARD-SENSITIVE ENVIRONMENT

Given a deterministic environment μ, we say it is reward-sensitive if for every cycle k, there exists a positive integer n such that there are two sequences of actions $b_1b_2...\ b_n$ and $c_1c_2...c_n$ such that $R_\mu(a_1a_2...a_kb_1b_2...\ b_n) \neq R_\mu(a_1a_2...a_kc_1c_2...c_n)$.

Note that the previous definition 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 rewards[10]. 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 have they a passive "observer" behaviour, so at any point the agent can strive to increase its rewards (or to avoid them decreasing).

Note that we have posed no constraints on observations, so these can be independent from the actions. In this sense, the following definition can be seen as a counterpart to the previous one.

DEFINITION 6. OBSERVATION-SENSITIVE ENVIRONMENT

Given a deterministic environment μ, we say it is observation-sensitive if for every cycle k, there exists a positive integer n such that there are two sequences of actions $b_1b_2...b_n$ and $c_1c_2...c_n$ such that $O_\mu(a_1a_2...a_kb_1b_2...b_n) \neq O_\mu(a_1a_2...a_kc_1c_2...c_n)$.

Although only Definition 5 would be strictly necessary, since rewards could be used as an "indirect" observation (as an agent without perception which can still feel when it is fed or not and could use some actuators), we will force our environments to be reward and observation sensitive. If we want to avoid big *holes* or *waiting periods*, we can restrict the two previous definitions to values of n which must be lower than a constant. In this way, we can talk about *n-actions observation/reward sensitive environments*.

According to the previous definitions, many table games we know are not reward sensitive environments (but they are observation sensitive environments). For instance, there are chess positions where unavoidably any move leads you to a different way of losing/winning. But also note that it is not very difficult to modify chess scoring to

[10] The previous definition can be made slightly stronger if we force the sum of rewards to be different (two different sequences of rewards can have equal sum).

make it fully reward sensitive, by assigning points to the score which depend on the moves before losing (e.g. losing in 45 moves is better than losing in 25 moves).

The restriction of environment class given by the previous two definitions is simply a practical thing. We do not want to use environments or sub-environments to evaluate agents when anything the agent can do is useless to change 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* at every *k* would be desirable, meaning that taking different actions would usually lead to quite different sub-environments (situations).

So, for the moment, we will just restrict environments to be observation and reward sensitive. Consequently, we will sample a subset of environments where we know that an action at every state may have consequences on present or future observations and rewards.

There is an interesting relation of reward-sensitive environments and the kind of rewards that have been defined in the previous section. In [Legg and Hutter 2007] a long discussion about how to distribute rewards is included and, finally, only one restriction is set, as we have seen before: the *total reward* must be lower than 1. 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[11]. Note that with the original definition, an environment which gives reward 1 after first action and then always 0 complies with the previous total reward restriction but it is quite useless. The reward sensitive condition makes this impossible as the following proposition shows:

PROPOSITION 1. A reward sensitive environment implies that at any point in the environment, part of the total reward must remain in order to be shared among all the following sub-environments.

PROOF. Trivial. Consider that the accumulated reward at point *k* is 1. Then, since rewards cannot be negative by definition, all the actions from this point on will have 0 reward (since 1 cannot be exceeded) and consequently the environment cannot be reward sensitive. Consequently, there must always be some remaining reward to share among all the descendants of an environment at any point, because otherwise it is impossible to have $R_\mu(a_1a_2...a_kb_1b_2...\ b_n) \neq R_\mu(a_1a_2...a_kc_1c_2...c_n)$. □

This does not exclude, in principle, an environment which gives 0 rewards for thousands of initial actions and then starts giving rewards after this "dead" period. But note that if instead of general reward sensitive environment we use the notion of *n-actions reward-sensitive environments* mentioned above, with a small *n*, that means there must be some reward spent at least each *n* actions. An extreme case is when *n*=1, where some amount must appear at least in one of the rewards of the possible actions at any point.

Considering then the class of environments which are both reward and observation sensitive, the definition of universal intelligence is modified as follows:

DEFINITION 7. UNIVERSAL INTELLIGENCE (FINITE SET OF REWARD AND OBSERVATION SENSITIVE ENVIRONMENTS)

$$\Upsilon^{II}(\pi, U, m) := \frac{1}{m}\sum_{\mu \in S} V_\mu^\pi \text{ where } S \text{ is a finite subset of } m \text{ reward and observation sensitive environments extracted with } p_U(\mu) = 2^{-K_U(\mu)}$$

Conceptually, all the previous changes and restrictions do not change the picture significantly from the original meaning of "universal intelligence", except the weighing (now complex environments have more weight, simply because of number, than simple environments). But, from a *practical* point of view, the re-definition can have enormous consequences on feasibly evaluating intelligence with a finite sample of environments.

3.2. On Practical Interactions

With the previous section we can now work with a sample of 'proper' environments, but in the original definition, we still have an infinite interaction with the environment and also we have that *K*() is not computable.

Limiting the number of interactions is easy to do. We need just to set a limit of interactions n_i for each environment, just modifying the definition of expected reward into a *calculated* reward:

[11] This restriction will be no longer necessary in the following sections, since we will not allow an infinite interaction with an environment, but a finite one.

DEFINITION 8. CUMULATIVE REWARD ON A FINITE NUMBER OF INTERACTIONS

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

But in this case it would be adequate to make the environments n_r-actions reward sensitive and n_o-actions observation sensitive, with $n_r \leq n_i$ and $n_o \leq n_i$, so actions have effect on rewards and observation at most in the limited period of interaction with the environment. An option is to make $n_r = n_o = n_i$.

So, with this restriction, administering the test seems to be a finite task. But it is not. The use of $K()$, apart from the problem that it is not computable, has an additional very important problem: very inefficient environments are possible, since the time which elapses between two observations can be too large. This can produce that some tests, even with a short number of environments and interactions, can be extremely long to take.

A solution to this problem and 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). Our first choice[12] is Levin's Kt complexity [Levin 1973] [Li and Vitányi 2008]:

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

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[13] that U takes executing p to produce x.

The problem of this measure is that environment enumerations are infinite, as the one shown in Figure 2. 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 is 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 we define:

$$\Delta time(U, p, i) := time(U, p, x, i) - time(U, p, x, i-1)$$

where $time(U, p, x, i)$ denotes the time taken to output string x until bit i, i.e., to output the first i bits of string x. Consequently, $\Delta time(U, p, i)$ denotes the time taken to output bit i of the string which is generated by p. Note that if i is greater than the length of the string (only for finite strings) we set $\Delta time = 0$. And now we define[14]:

$$Kt^{\max}{}_U(x) := \min_{p \text{ such that } U(p)=x} \left\{ l(p) + \log\left(\max_i (\Delta time(U, p, i)) \right) \right\}$$

which means that we sum the length of the program plus the logarithm of the maximum interaction time of that environment with the machine U.

It is easy to see that for finite strings, this measure is quite similar to the original Kt (although not identical, since time here will typically have a lower weight). Like Kt, $Kt^{\max}$ is computable for finite strings. The interesting difference comes up for infinite strings (as environments are infinite strings). For some strings, such as "1111....", it is easy to see that a program such as "always output 1" requires a constant computational time for each output. Consequently, $Kt^{\max}{}_U(1111...)$ will usually give a small finite number. However, for other strings, the previous definition gives an infinite value. For instance, the sequence "01101010001010001010..." which shows a 1 when bit i is prime and 0 otherwise, requires increasingly higher $\Delta time$ to tell whether number i is prime or not[15]. This means that the maximum $\Delta time$ is not bounded and hence $Kt^{\max}{}_U$ of this sequence is infinite. Hence, the environment would be ruled out[16].

This response limit is precisely what we require for an environment. The response time at any interaction with an environment must be bounded. Otherwise, we might face situations such that once an agent performs an action we might wait for several years to get a reward and an observation. This would be completely unacceptable for a test, as

[12] There are other options, as [Hutter 2009] also suggests, such as fixing a bound on time (but this would require to set an arbitrary constant), logical depth [Li and Vitányi 2008], the speed prior [Schmidhuber 2002] or through the use of redundant Turing machines [Dowe 2008a, sec. 0.2.7].

[13] 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 environment cannot depend on the speed of the machine where it is run.

[14] This definition is clearly inspired by both Levin's Kt and the Speed Prior [Schmidhuber 2002].

[15] Even though 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) [Agrawal et al. 2004].

[16] Of course, as said before, if we set a finite number of interactions for each environment, then we could compute Kt only up to this number. In this case, Kt would be finite unless halting problems occur.

it is unacceptable for an interactive application, a videogame or any other kind of interactive environment or virtual world.

Consequently, we will use the previous $Kt^{max}{}_U$ for environments. This can be easily translated from binary strings to environments by defining the $\Delta time$ as the time required to print the pair $<r_i,o_i>$ after action a_{i-1}, i.e. the *response* time. Consequently, the upper bound is set to the maximum computational time that the environment can take to output the reward and the observation after the agent's action. Note that this upper bound can be used in the implementation of environments, especially for making their generation computable.

And now, from this measure, we can also derive a probability and refine our definition for universal intelligence as follows:

DEFINITION 9. UNIVERSAL INTELLIGENCE (FINITE SET OF REWARD AND OBSERVATION-SENSITIVE ENVIRONMENTS, FINITE INTERACTIONS, KT^{MAX} COMPLEXITY)

$\Upsilon^{III}(\pi, U, m, n_i) := \frac{1}{m}\sum_{\mu \in S} V_\mu^\pi(n_i)$ where S is a finite subset of m environments being both n_r-*actions* reward and n_o-actions observation sensitive (with $n_r=n_o=n_i$) extracted with $p^t{}_U(\mu) = 2^{-Kt^{max}{}_U(\mu)}$

The previous definition 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 higher score is more intelligent, so the previous test is, in principle, useful for comparing subjects. However, since rewards are positive, the value of Υ^{IV} which is returned is somehow meaningless, as the following proposition shows:

PROPOSITION 2. For all π, U, m, n_i and n_i' if $n_i \geq n_i'$ then $\Upsilon^{III}(\pi, U, m, n_i) \geq \Upsilon^{III}(\pi, U, m, n_i')$.

PROOF. Trivial. Since rewards are always positive and we just accumulate scores, as the number of interactions increases, the accumulated score never decreases. □

And it is frequently the case that the relation is stricter, i.e. $\Upsilon^{III}(\pi, U, m, n_i) > \Upsilon^{III}(\pi, U, m, n_i')$ when $n_i > n_i'$. And this is now so obvious because we have removed all the other sources of infinity. The attempt by [Legg and Hutter 2007] to have rewards limited between 0 and 1 was in the right direction, but if we infinitely accumulate them for larger n_i, we will get higher and higher values. Note that this does not happen for increasing values of m, since m is in the denominator. So it seems that just adding n_i in the denominator solves the problem, but it is precisely the choice of the accumulated reward being between 0 and 1 which makes that:

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

The use of reward-bounded environments creates many questions about the distribution of the rewards, since most of the reward can be given during the first interactions (dozens or hundreds), since postponing an important amount of rewards to late cycles typically requires higher description sizes than environments which spend most of it during the first interactions. Even with the constraint of being reward-sensitive, we minimise this problem, but the relation between expected reward and number of interactions is quite unclear. This relation is discussed next.

3.3. On Aggregating an Absolute Value of Intelligence

The problem of getting a "value of intelligence" from the previous definition is much more cumbersome that it might seem at first sight. 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 Kt^{max} values of the sample.

The following figurative plot (Figure 4) shows an example of the expected score, i.e., assuming infinite interactions, of four agents depending on the complexity Kt^{max} of the environment. As we see, the curve for each agent has a lot of information which is difficult to summarise in a single number.

Three of the four agents behave better for low complexity environments but their curves are not monotonic (this is not generally the case in general, and agent 3 shows a less monotonic behaviour) but all of them are able to get some score for more complex environments (this is also a general result, since rewards are positive and despite they can be very sparse in large environments, this will typically be greater than 0 in the limit). If we compute the total of

the expected total reward per agent, we see why we get usually an infinite value. In fact, we get an infinite value for the four, since the integral of the four curves between 1 and ∞ is ∞. If, instead of that, we perform an average as we started to do since Definition 4, then, as m becomes larger, it boils down to measure 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 which overweights simple environments (as Legg and Hutter do), we would have that agent 4 dominates over agent 2, agent 2 over agent 1 and (mostly) agent 1 over agent 3. In fact, it is almost like only considering the leftmost part of the curve.

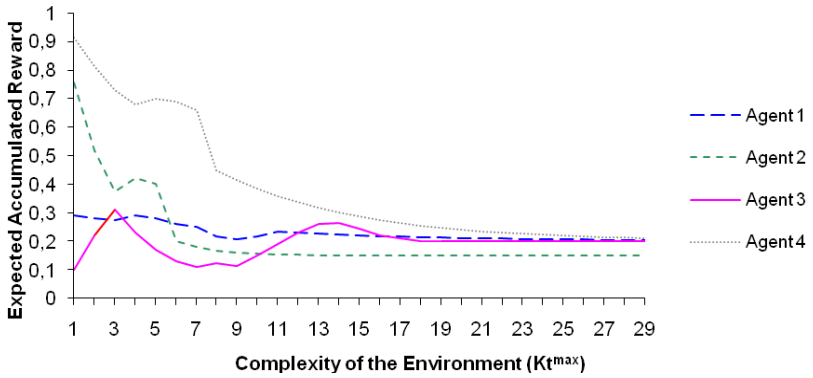

Figure 4. Figurative evolution of expected accumulated rewards vs. complexity of environment.

We can argue that behaving in simple environments is as important as behaving in complex environments and that the beginning of the curve is also important. But we have to realise that complex environments also include simple patterns and these will still be much more frequent than the complex ones[17]. And if these simple patterns exist in complex environments, agents with limited intelligence will be able to take advantage (note that the number of interactions 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 [Hibbard 2009], should we go directly to environments of very high complexity as a means of getting an approximate value of intelligence? This is completely the opposite of the probability we have used for the samples, making simple environments much more probable than complex environments. One way out from this dilemma is to think that results for low-complexity environments are less reliable for the overall score than results for high-complexity environments. But the problem of using high-complexity environments is that they also require longer interactions to get a good reliability (the previous figure shows increasing stability on the right because we are assuming an infinite number of interactions). So, 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 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 Figure 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 2 and starts scoring better initially, while agent 1 perhaps take some more risks initially to learn the environment better and then scoring better in the limit. Note also that since we are plotting accumulated reward, it also grows with a random agent[18] (this is one of the counterintuitive things of non-averaging the accumulated rewards). 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.

[17] This is related (but different) to the fact that any incompressible object (e.g., a sequence) must have compressible parts (subsequences) (see e.g. [Li and Vitányi 2008]). In the case of environments, it is much easier to see why simple patterns have to exist; consider, for instance, an environment 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, while it is clear that it has simpler patterns of complexity as μ_1 and μ_2 alone. Note also that the existence of simple patterns in large environments is not obtained by the existence of "dead code", since this is impossible since we are using $K_U(\mu)$ (or $Kt^{max}{}_U(\mu)$) instead of $l(\mu)$.

[18] This would even happen as well if we used a discount factor γ^t for future rewards as in reinforcement learning.

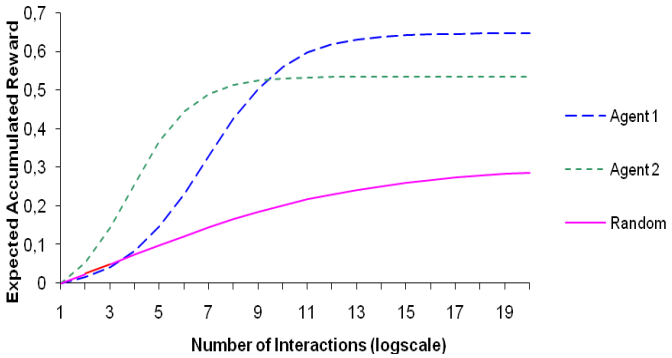

Figure 5. Figurative evolution of expected accumulated rewards vs. number of interactions.

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 =1,000 for an environment of $Kt^{max}{}_U$=5 will usually be enough to reach the stable part of the curves in Figure 5. However, the same value for a very complex environment will typically be insufficient, and a very intelligent agent might score as a random agent simply because it has not been given enough observations to capture the models underneath (or once it has captured it, most of the reward has been exhausted and randomly found, since we are considering reward-bounded environments, so the result will not be much different from a random agent). This also suggests that for a finite number of interactions, the appearance of Figure 4 might be different, and the expected reward for very complex environments would be the same for a relatively intelligent agent than for a random agent, because the former would not have enough interaction steps to learn from the environment. However, as environments with high $Kt^{max}{}_U$ also include simple subenvironments, it is not so clear that every agent would behave as a random agent for high values of m and low values of n_i.

In other words, increasing values for m entail a worse expected result and increasing values for n_i entail better results. This suggests that we need to apply a correction in the score. The following definition formalises this idea:

DEFINITION 10. ADJUSTED REWARD

Given an environment μ of complexity $\xi_\mu = Kt^{max}(\mu)$ and a number of interactions n_i, then the adjusted reward for agent π is defined as follows:

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

Such that $\Phi(x, y)$ is a function which is decreasing on x and increasing on y.

The function $\Phi(x, y)$ must be determined theoretically or experimentally (or a combination of both). But this does not seem easy. An example of such a function might be:

$$\Phi(n_i, c_\mu) := \frac{1 + 2^{-\left(\frac{n_i}{\xi_\mu}\right)}}{2}$$

Note that the maximum for Φ (which is 1) is obtained when $\xi_\mu \to \infty$ (or n_i=0) and the minimum (0.5) is obtained when $n_i \to \infty$ (or n_i=0). And this function makes that $\forall \mu, \pi, n_i : 0 \le W_\mu^\pi(n_i) \le 1$.

3.4. Symmetric Rewards and Balanced Environments

Note that the influence of n_i in Φ 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 an accumulative reward system which affects any agent, including a random agent, but an average is not possible with Legg and Hutter's constraint (the total accumulated reward cannot be greater than 1).

Consequently, we are going to revisit the constraints on rewards. Our proposal is to use symmetric rewards, which can range between −1 and 1, i.e.:

DEFINITION 11. SYMMETRIC REWARDS

$$\forall i : -1 \le r_i \le 1$$

Note that this does not preclude the accumulated reward at a certain point from being greater than 1 or lower than −1. So, if we do many good actions in a row, we can have an accumulated reward greater than 1. Regarding 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.

Note that this redefinition also implies that proposition 1 (which we saw in section 3, stating that there must always be some remaining reward) is no longer true since now we do not have a saturation point and, additionally, we can use two different negative values to make the environment reward sensitive.

If we set symmetric rewards, we also expect reference machines to be symmetric, or more precisely, to be balanced on how they give rewards. The following definition formalises this:

DEFINITION 12. WEAKLY BALANCED REFERENCE MACHINE

Given a reference machine U we say it is weakly balanced or unbiased for rewards if its program coding has no preference on positive rewards over negative rewards (and viceversa), i.e. for every program p there is one and only one symmetric program p^- with the same length such that the behaviour of the environments their generate (μ_p and μ_{p^-}) are the same but the rewards are symmetric (for every reward r in μ_p we have a reward $-r$ in μ_{p^-}).

And now, if the reference machine is weakly balanced, we can show the following:

PROPOSITION 3. Consider symmetric rewards and a reference machine U which is weakly balanced. Given a random agent π_r (an agent which randomly chooses among actions with a uniform distribution), then the expected value of $\Upsilon^{III}(\pi, U, m, n_i)$ for any value of m and n_i is 0.

Proof. Since the reference machine is not biased on positive or negative rewards, given an environment μ there is always a symmetric environment (denoted by μ^-) with equal coding size in terms of Kt^{max} where all the rewards are symmetric (any positive reward r becomes $-r$ and vice versa). A random agent with an expected accumulated reward ρ in environment μ for n_i interactions will necessarily have an expected accumulated reward $-\rho$ in environment μ^- for n_i interactions. Since the complexity is the same (their program lengths are the same), their probability is the same, and hence the expected total value for any sample (independently from its size m) is 0.□

However, although it is easy to find non-biased reference machines in these terms, when samples are small, there can be many environments where acting randomly can have a positive overall result (or a negative result), so typically a small sample of environments would give a quite variable (and hence unreliable) value of intelligence. In a reliable test, we would like that many (if not all) environments will give a 0 reward to random agents. The following definition formalises this.

DEFINITION 13. BALANCED ENVIRONMENT

An environment μ is balanced iff

1) $\forall i : -1 \leq r_i \leq 1$
2) Given a random agent π_r, the following equality holds[19]:

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

This excludes both hostile and benevolent environments, i.e. environments where doing randomly will get more negative (resp. positive) reward than positive (resp. negative). 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 which only generates balanced environments, such as the one proposed in [Hernandez-Orallo 2009b].

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

[19] 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.

DEFINITION 14. AVERAGE REWARD

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

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

EXAMPLE 3.

Consider a modification of the test setting we saw in EXAMPLE 2. A robot (the agent) can press one of three possible buttons ($A = \{B_1, B_2, B_3\}$), rewards are just the handing over of no bananas, a banana or two bananas ($R = \{-1, 0, 1\}$) and the observation is three cells where a white and black ball must be inside one (but different) cell, namely ($O = \{$0WB, 0BW, W0B, B0W, WB0, 0BW$\}$), where W denotes the cell has a white ball, B denotes the cell has a black ball and 0 denotes that it is empty. An example of a possible environment is:

$\mu((1)^*| r_1 o_1 a_1 \ldots r_{k-1} o_{k-1} a_{k-1})$ if ($a_{k-1} = B_1$ and $o_{k-1} =$ Wxx) or ($a_{k-1} = B_2$ and $o_{k-1} = x$Wx)
or ($a_{k-1} = B_3$ and $o_{k-1} = xx$W)

$\mu((-1)^*| r_1 o_1 a_1 \ldots r_{k-1} o_{k-1} a_{k-1})$ if ($a_{k-1} = B_1$ and $o_{k-1} =$ Bxx) or ($a_{k-1} = B_2$ and $o_{k-1} = x$Bx)
or ($a_{k-1} = B_3$ and $o_{k-1} = xx$B)

$\mu(0^*| r_1 o_1 a_1 \ldots r_{k-1} o_{k-1} a_{k-1})$ otherwise

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:

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

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:

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

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:

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

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

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

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 AND OBSERVATION SENSITIVE AND BALANCED ENVIRONMENTS, FINITE INTERACTIONS, Kt^{MAX} COMPLEXITY) WITH ADJUSTED SCORE.

$\Upsilon^{IV}(\pi, U, m, n_i) := \frac{1}{m \cdot n_i} \sum_{\mu \in S} W_\mu^\pi(n_i)$ where S is a finite subset of m balanced environments being both n_r-*actions* reward and n_o-actions observation-sensitive (with $n_r=n_o=n_i$) extracted with $p^t{}_U(\mu) = 2^{-Kt^{\max}{}_U(\mu)}$

We still maintain the weighted reward which includes the function Φ.

4. Time and Intelligence

The last definition given in the previous section is now feasible. But the main criticism is that, provided that "universal intelligence could be viewed as generalising the C-test from passive to active environments" [Legg and Hutter 2007], we do not understand why there is no reference to (physical[20]) *time* in their formula. How can an interactive extension disregard time? How can a very slow agent be considered equally intelligent as a very quick agent with the same result? And, additionally, 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*, just imagine an interactive test where the agent has to wait several hours after each action in order to see the reward and the observation. This would not only make the test impractical, but it would also affect the results (since the agent may get bored and lost, and even fall asleep or die, or, conversely, can take advantage of this extra time between states to meditate about the environment's behaviour). A quite similar issue appears in interactive games, such as chess. So, typically this problem has been ignored by assuming that environments used in *tests* react *immediately*. But this is a fallacy, since no machine reacts *immediately*. The use of $Kt^{\max}$ seems to minimise the problem, but it only refers to computational time (machine steps), which can take a lot of physical time on a slow machine. A more reasonable assumption is to think that the environment's speed is much higher than the agent's, so to the agent's scale, the environment's reactions seem immediate. In the case where the agent is a biological system and the environment is a computer, with the use of $Kt^{\max}$, it will almost always be the case that the environment's reactions will be "immediate". But when both the environment and the agent are machines, we will have to be more careful. Again, the use of $Kt^{\max}$ and assuming that both the environment and the agent are implemented in machines of comparable speed, the assumption will frequently hold. In any case, a small delay would be the same for all the examinees, so there must 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 at page 426 in [Legg and Hutter 2007]: "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 [Dowe and Hajek 1998] [Hernandez-Orallo 2000a] and [Dowe 2008a, footnote 202].

This issue is also recurrent in psychometrics, with a strong debate with the correlation of "inspection time" and intelligence. But, generally, time is 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 on one side and the speed of answers/reactions on the other side. In fact, in psychometrics, tests are usually categorised between speed tests and 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. Although this is 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.

Additionally, there are other conceptual problems associated with ignoring time in an intelligence definition: constructing intelligent systems would look less difficult than it is, since disregarding computational complexity would be an option. For instance, as we will see in the next section, a committee of agents might generally score

[20] In Legg's Ph.D. dissertation [Legg 2008] the most general environment type is called "chronological environment", but here, "time" is understood computationally (as machine states or steps) and not physically (we mean physical by the time used in Physics, where the standard unit is the second or, alternatively, the Planck time).

better than a single agent, if time is not taken into account. Or, even worse, an inefficient and exhaustive search method might be appropriate for intelligence.

So, 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 tuned, from cases where we want to test good and quick responses to other cases where time is not so relevant, or even in a discrete way (as we will see in some definitions in the following sections) where we will incorporate time or not.

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 which looks quite anthropomorphic if we set it to some seconds. So, we need to explore some other options.

4.1. 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 $\tau_o << \tau$) such that if the agent does not select an action inside 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 obvious solution of averaging the rewards (which will be investigated below), a possible solution would be to set a fixed time, a time-slot τ_s (instead of a time-out) for each interaction (with $\tau_s << \tau$). 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 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 effective 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. But a *good* policy here would be to act as a fast random agent until the average reward becomes larger than a threshold (this can happen with more or less probability depending on the threshold) and then stop acting. We call this agent, the "opportunistic fast random agent", denoted by π_o. The expected reward (in the limit) would be precisely the threshold times its probability. For instance, consider an agent which makes one action randomly. If 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 possibility is to use the time left from the last action until time τ and use it as a discount over the recent history.

DEFINITION 16. AVERAGE REWARD WITH DIMINISHING HISTORY

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

where n_τ is the number of interactions made by π in μ in time τ, and t_i denotes the total time elapsed until action a_i is made.

This definition reduces the number of evaluated cycles proportionally to the time left from the last action until τ. That means that if the last actions have been good and we delay future actions and let time pass, we can soon make the measure ignore the recent rewards. If we stop, in the limit, the measure reaches 0, so it also avoids stopping policies (for further justifications on these choices and a theorem showing this, see [Hernandez-Orallo 2009c]).

Finally, it is clear that as we leave more physical time, generally an agent can get more observations (very quickly, almost randomly first) and then try to use this knowledge in some other actions. This also means that a faster agent will generally score better. And the effect of the complexity of the environment in Φ has not been

affected by the recent changes. Consequently, we have again 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 the following subsection), we could get rid of Φ. For the moment, we need the adjusted version of the aggregated reward:

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

We must change Φ to consider physical time and not interactions as the first parameter. And now, with an average reward and physical time, the goal of Φ becomes different. If we want to consider the speed of the agent in the measure of intelligence, then Φ should treat τ differently from the case in which we 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, using interactions limited by physical time, the definition is as follows:

DEFINITION 17. UNIVERSAL INTELLIGENCE CONSIDERING TIME (FINITE SET OF REWARD AND OBSERVATION SENSITIVE BALANCED ENVIRONMENTS, FINITE INTERACTIONS, Kt^{MAX} COMPLEXITY) WITH ADJUSTED SCORE AND USING PHYSICAL TIME TO LIMIT INTERACTIONS.

$\Upsilon^V(\pi, U, m, \tau) := \frac{1}{m} \sum_{\mu \in S} \breve{w}_\mu^\pi \Big\| \tau$ where S is a finite subset of m balanced environments being both reward and observation sensitive extracted with $p^t{}_U(\mu) = 2^{-Kt^{\max}{}_U(\mu)}$

Note that we do not limit now the number of interactions that have to be reward and observation sensitive, since with physical time we may have a few or many interactions depending on the agent's speed.

4.2. Anytime Evaluation with Time

Using two different values for the number of environments and the overall time left for each environment is an important bias, apart from the effect they have on the measurement. If we allow many environments with a small number of interactions in each, many agents will be unable to find patterns and take advantage from them. On the other side, 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 9 and 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: 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, since 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 are set beforehand depending on the subject. 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).

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. [Wainer et al. 2000]). 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 which 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) [Lord 1980] [Embretson and Reise 2000], where not only a difficulty score is assigned to each item but a richer Item Response Function or Curve. 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 [Embretson 1998] [Embretson and Mc Collam 2000] [Embretson 2004].

However, not only do we want a test to be adaptive, but another important goal should be a test such that given a small slot of time for doing it, we could have an approximation of the intelligence of the agent, but if we were given

more time, a better approximation could be obtained. But not only this, it would 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 with the time given). Again, some approaches from C.A.T. fulfil (or can easily be adapted to) this requirement.

Clearly, all this research in C.A.T. is, to a greater or lesser extent, useful for constructing a universal and adaptive test. In our case, though, the difficulty of each item is determined theoretically and not experimentally and, additionally, we do not have a range of difficulties and time scales where we can assume, a priori, that an individual may be placed.

For that, we propose the idea of adapting both the complexity of the environments and the time we leave on each of them. One possibility is to define two counters: one for the Kt^{max} complexity ξ of the environment and another for time τ (*physical time*), the time that we leave to play with one environment before switching to a different one. So, since we do not know the intelligence and the speed of an agent, we need to start with very simple problems and very small time slots to play with each environment. Progressively, we will face 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, there must be a mechanism to also degrade the complexity if the agent does not score well on the environments (this is the meaning of adaptive, here).

So let us design the adaptation policy of the modified test. The first question is at which complexity level and at which speed scale 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 set 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 $Kt^{\max}$ gives a valid environment) and the smallest possible value for time (τ = Planck time, $\approx 10^{-43}$, or, more practically, a value around current agent technology, e.g. 1 microsecond). Obviously, it is easy to modify these start values in cases where we have some information about the capabilities of the examinee, in order to make the test more efficient.

And now the second question is the pace and directions in which ξ and τ have to be modified. If we think about complexity, the idea is to apply a *servo-control* adaptation to increase complexity when the agent is able to manage with the environment's complexity and the time scale, and decrease complexity otherwise. In fact this kind of regulation is also typical in C.A.T. So, complexity 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 (this would allow rote memory learning with an already seen environment). Since there are a limited number of environments with 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.

Regarding 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 approximately in a second, initially it will not have time to make any action if we start with microseconds. In a few iterations, we will reach τ=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. τ=10 minutes), we will allow more actions in the same environment and the agent will work within its time scale.

The final question is then the precise pace of increasing/decreasing complexity and increasing time. To allow quick adaptation, an exponential growth for time could be used, and complexity could depend on the reward value. Let us see all this in more detail in the following definition.

DEFINITION 18. ANYTIME UNIVERSAL INTELLIGENCE TEST HAVING TIME INTO ACCOUNT

We define $\Upsilon^{VI}(\pi, U, \Theta)$ 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)
3.  OUTPUTS: a real number (approximation of the agent's intelligence)
4.  BEGIN
5.  Y ← 0                          (initial intelligence)
6.  τ ← 1 microsecond              (or any other small time value)
7.  ξ ← 1                          (Kt^max initial complexity)
8.  S_used ← ∅                     (set of used environments, initially empty)
9.  WHILE total elapsed time < Θ DO
10.   REPEAT
11.     Compute S_ξ (U)            (the set of all the balanced, reward and observation
                                    sensitive environments with ξ − 1 ≤ Kt^max ≤ ξ)
12.     IF S_ξ (U) ⊂ S_used 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.   μ = Choose(S_ξ (U), S_used)  (uniformly random choice of one of the
                                    environments from S_ξ (U) which is not in S_used.)
19.    Reward ← V_μ^π ‖ τ          (average reward until time-out τ stops the interactions)
20.   Y ← Y + Reward               (adds the reward)
21.   ξ ← ξ + ξ · Reward / 2       (upgrades or degrades the level according to reward)
22.   τ ← τ + τ/2                  (increases time)
23. END WHILE
24. Y ← Y / | S_used |             (averages accumulated rewards)
25. RETURN Y
26. END ALGORITHM
```

Note that complexity and time have similar updating policies, although complexity depends on rewards (note that Kt^{max} complexity is not a positive integer number, but a positive real number, so fractions in ξ are perfectly valid and hence we need to get environments in the range $\xi - 1 \leq Kt^{max} \leq \xi$). If rewards go well, complexity increases quickly. If rewards are negative, complexity can be decreased. This dependence on rewards is important to avoid cheating. Imagine an agent that for very simple environments scores well (near 1), and after being upgraded, it could just score sufficiently bad (near –0.01) to be downgraded, in order to try easy environments again and to score very well (near 1). In average, the scores would be around 0.5, while the agent has never left the very simple levels. Although this behaviour is quite pathological, with this dependence, the increase/decrease of complexity is proportional to the rewards (as a servomechanism), so going back to easy complexities implies important negative rewards which finally will be counted into the measure Υ.

The generation and selection of environments is made in lines 11 and 18. Note that these two parts can be coordinated in order to produce a more efficient generation of environments. For instance, it is preferable to use a universal environment class which only includes balanced reward and observation-sensitive environments, such as the one introduced in [Hernández-Orallo 2009b]. Additionally, the measurement is more efficient and robust if we leave each environment running against a random agent for a while before using the environment, in order to eliminate any start-up garbage (see, e.g., [Hernandez-Orallo 2009a] for a full discussion on this).

The part of the algorithm in Definition 18 where the agent interacts is line 19 and where the algorithm waits for an agent's action. All the time spent by the algorithm for the rest of lines has to be counted on the environment's response time, and as we mentioned in the previous subsection, it should be much shorter than the agent's time scale, in order to give the appearance that the environment interacts almost immediately. That means that line 11 (which holds the most computationally expensive calculation) would require in practice some pre-calculations before starting the main loop.

A figurative trace of the algorithm is as follows. Initially, with τ so small, the agent has no time to make an action (or it has to do something very quickly without time for thinking, such as a random choice), so it has an expectation of 0 reward. This means that ξ would not be increased (since the increase depends on the reward) and we would go for another environment with ξ =1 (if it exists, otherwise ξ =2). In practice, this means that ξ will be

around small numbers for a while (since repeated environments are not allowed[21]), but not too much, since τ grows exponentially. When time starts to be closer to the agent's time scale, the agent can make better actions and if rewards are positive, increase ξ. As more time Θ is allowed, we get environments with higher ξ and more time is left in each of them, so giving more reliability to the aggregated reward and less dependence on the reference machine. Additionally, as more time is allowed, the speed of the agents is less relevant. 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 all the obtained rewards. Note that this average is performed in the same way we have discussed in the previous sections. Getting a single number (this average) is practical in some applications but in others we can have a look at the evolution (the curve) of results by increasing complexity. This information can complement the intelligence score.

The expected results for different kinds of agents is difficult (if not impossible) to estimate precisely, because it depends on the environment class used and the reference machine. The only easy case is a random agent, as we show below.

PROPOSITION 4. Consider a random agent π_r (the agent which randomly chooses among actions with a uniform distribution and a constant reaction time τ_r) to whom we administer the test from Definition 18 for a time Θ. Then, for every universal machine U, when $\Theta \to \infty$ then $\Upsilon^{VI}(\pi_r, U, \Theta) = 0$.

PROOF. Trivial from the definition of balanced environment. Since all the environments are balanced, the total expected reward in each environment tends to 0 since $V_\mu^{\pi_r} = 0$ in the limit and hence the accumulated Υ will be 0. □

A different issue is to tell the expected intelligence results of agents with some learning abilities, or to show the maximum value which can be obtained. Note that as $\Theta \to \infty$, $\xi \to \infty$, and very large environments (which are always balanced, reward-sensitive and observation-sensitive) will contain patterns that can be exploited by the agent[22].

It is interesting to compare the expected results with finite (and infinite) runs of the test. The following table shows how the test should work for some specific types of agents:

	Curve	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	Constant (always 0)	$\Upsilon_{mic} = 0$ $\xi > 0$	$\Upsilon_{nor} = 0$ $\xi >> 0$	$\Upsilon_{inf} = 0$ $\xi = \infty$
Relatively slow intelligent agent (e.g. a human)	Exponentionally decreasing with a base constant greater than 0 for ξ=∞	$\Upsilon_{mic} = 0$ $\xi = 0$	$\Upsilon_{nor} > 0$ $\xi > 0$	$\Upsilon_{inf} > 0$, $\Upsilon_{inf} \approx \Upsilon_{nor}$ $\xi = \infty$
Very fast super-intelligent agent	Exponentionally decreasing with a base constant greater than 0 for ξ=∞	$\Upsilon_{mic} > 0$ $\xi >> 0$	$\Upsilon_{nor} > 0$, $\Upsilon_{nor} \approx \Upsilon_{mic}$ $\xi >> 0$	$\Upsilon_{inf} > 0$, $\Upsilon_{inf} \approx \Upsilon_{nor}$ $\xi = \infty$
Oracle (the most intelligent and fastest agent maximising rewards at any environment)	Fairly constant (greater than 0 for ξ=∞)	$\Upsilon_{mic} > 0$ $\xi >> 0$	$\Upsilon_{nor} > 0$, $\Upsilon_{nor} \approx \Upsilon_{mic}$ $\xi >> 0$	$\Upsilon_{inf} > 0$, $\Upsilon_{inf} \approx \Upsilon_{nor}$ $\xi = \infty$

Table 3. Figurative expected results on the anytime test for several prototypical agents.

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 we will require 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. A different thing is for machines, since we can invest more time for intelligence assessment (since the test is completely automated and it does not require human intervention).

[21] Not allowing repetition logically allows an agent to increase ξ by exhausting all the explored environments of lower complexity. However, higher ξ does not necessarily entail lower scores, provided time is also increased. Note that considering repeated environments would allow agent with a very large memory to score well.

[22] In fact, the probability of an environment μ to appear ($2^{-K(\mu)}$) is similar to the probability of *subenvironment* μ to be reached by a random interaction inside a randomly chosen μ' (according to the same probability $2^{-K(\mu')}$ but with $K(\mu) << K(\mu')$). Whether this holds for $Kt^{max}{}_U$ is to be confirmed, to ensure non-zero convergence for $\Theta \to \infty$.

4.3. Anytime Evaluation Disregarding Time

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

DEFINITION 19. ANYTIME UNIVERSAL INTELLIGENCE TEST DISREGARDING TIME

We define $\Upsilon^{VII}(\pi, U, \Theta)$ 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 ← 1 (step)"), and these are updated by the formula $n \leftarrow \lceil n + n/2 \rceil$ (line 22). And $V_\mu^\pi \parallel \tau$ is replaced by $V_\mu^\pi(n)$ (line 19).

Clearly, Proposition 4 also holds for this variant of the test. Note that in this case, testing a very slow agent would take much more time than testing a very fast agent or, seen differently, given a fixed physical time the estimation of intelligence of slow agents will be poorer than the estimation of intelligence of fast agents.

5. Examples

In order to illustrate some of the features of the previous test, we are going to give some examples of environment classes. We will show simple examples, so instead of considering universal (Turing-complete) machines, we will choose some reference machines which are not universal. This choice also highlights the differences between examples.

5.1. A Simple (Non-Virtual) Environment Class

Consider the same test setting as in Example 2 with some modifications. In this setting, a chimpanzee can press one of n possible buttons ($A = \{B_1, B_2, B_3, ...\}$, *raw* rewards are just no bananas, one banana or two bananas ($R = \{-1, 0, 1\}$) and the observation is n cups where a black ball and a white ball must be inside one of the cups (both balls can be in the same cup). The value of n is part of the environment, so typically high and random values of n will imply higher complexities. In order to pose some difficulty, the observation is only visible to the chimpanzee for three seconds, then there is a dead period (or downtime) of another three seconds and then the chimpanzee can press any of the buttons.

Raw rewards are −1 if it presses the corresponding button where the black ball is, +1 if it presses the corresponding button where the white ball is, and 0 if either no ball is under the cup which corresponds to the button it has pressed or both balls are inside.

Initially, the white ball is placed at cup 1 and the black ball is placed at cup n. At the second iteration, the white ball is moved to cup n, while the black ball is not moved. At the third iteration, the white ball is moved to cup 1, while the black ball is not moved. Each subsequent interaction the white ball is placed exactly where the agent pressed two iterations ago (cup i if action i) and the black ball is always left where it was (always at cup n).

It is easy to see that the previous class of environments (let us call it U_s) is reward and observation sensitive, and it is also balanced.

An agent who always remembers where the white ball is and never presses the last button performs optimally. But we also know that an agent always pressing B_1 also behaves optimally in the limit.

Consequently, a chimpanzee who does not realise how observations go will only be able to rely on its memory, and its performance in this task (if we allow a sufficient period of time for complex environments with many buttons and cups to appear) will be:

$$\lim_{\Theta \to \infty} \Upsilon^{VI}(\pi, U_s, \Theta) = 0 \text{ (using Definition 18).}$$

Although in simple environments it can perform well, in more complex environments it cannot remember where it is and it behaves randomly.

But if the chimpanzee realises that always pressing button B_1 is a good policy, we would have that the chimpanzee would have

$$\lim_{\Theta \to \infty} \Upsilon^{VI}(\pi, U_s, \Theta) = 1 \text{ (using Definition 18).}$$

5.2. 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 $A=\{a_1, a_2, a_3\}$ and observations are always a subset of the set of possible states $O=\{o_1, o_2, o_{3,\ldots}\}$, where o_1 is always the initial state.

Additionally, we have some particular restrictions:

- 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 and observation 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}. One of the simplest possible automata is shown in Figure 6:

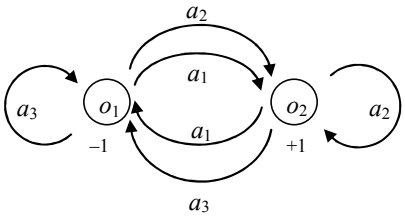

Figure 6. A simple automaton.

Note that the optimal behaviour for that environment is:

$$(a_1|a_2)a_2^*$$

And now let us consider a very complex automaton such that the following state exists:

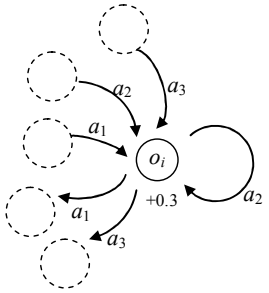

Figure 7. Part of a complex automaton showing a state which is much alike to the one in Figure 6.

Note that the existence of such a state is compatible with the restrictions of this family of environments. An obvious example can be seen in the automaton shown in Figure 6.

Eventually, if an agent reaches this state it can perform a_2^* and would score an average reward closer to 1 in a few actions.

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 restrictions we have added, many environments and problems that can be modelled with Finite State Machines and learning regular languages can be evaluated with the anytime test.

5.3. "Spatial" Environments. A very simple "social" environment class.

Consider now 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 staying in the same cell, $A=\{L,R,U,D,S\}$. 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 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 that either *Good* or *Evil*, but *Good* and *Evil* cannot share the same cell (except from 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*, denoted by $path_{Good}$ is UULRDD, it means that it will move according to this pattern UULRDDUULRDD… forever. If the movements of *Good* and *Evil* make them go to the same cell, this is avoided by randomly letting one move and keeping the other at its previous cell.

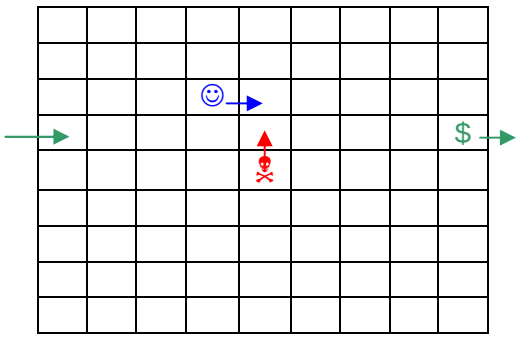

Figure 8. A representation of this class of environments. $ denotes Good, ☠ denotes Evil and ☺ is the agent.

Observations are 4×3 matrices where the *i*-index indicate the content of the four adjacent cells in this order (Left, Right, Up and Down) and the *j*-index indicates the presence or not of *Good*, *Evil* and *Limit*. *Limit* indicates that we are at some of the limits of the grid (which, as we have said, can be surpassed).

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 distance (i.e., toroidal distance). So, 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 on the same cell) and observation-sensitive (the agent can move and the limits can be perceived) and they are also balanced (one Good and one Evil with symmetric reward function). Note that for these conditions to hold the surpassing rule is very important (movements are toroidal when going out of the grid). Otherwise, more restrictions on movements should be imposed.

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:

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

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 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 greater than 0.5 always implies that we are at the same position as *Good*. When this is the case, the agent will move left all the time. Then, it is clear that the reward that comes from any future interaction cannot be lower than 0 (and in many cases it will be greater than 0), so it is clear that in a few interactions, the agent will reach accumulated reward 1 and will stay like that forever.

The previous example is a good class of environments where we see that the agent can interact, in some way, with other simulated agents (very naïve in this case). For a more complex and general (Turing-complete) environment following the properties of balancedness and reward and observation-sensitiveness, we refer the reader to [Hernandez-Orallo 2009b].

6. Discussion

There is a common thesis in all intelligence tests (relatively well-known in psychometrics, comparative cognition or artificial intelligence) which 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*. While (1) is taken to the extreme in human classical psychometrics (except from C.A.T.), where tests are just a form with questions explained in natural language, (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 have been forced to devise an adaptive test. Adaptivity has many advantages even in the case of evaluating one single species or a segment from it, such in psychometrics, as C.A.T. and Item Response Theory practitioners advocate (and it is getting greater

acceptance in psychometrics). But 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 thing 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 the Turing Test or many informal psychological tests. We do not discard a possible evaluation by an artificial intelligent 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 provide by themselves a huge amount of possibilities, discussion and material to further investigate.

6.1. Main New Features

Let us recapitulate the changes we have made to the original definition of Universal Intelligence. Legg and Hutter's definition had three main formal limitations (use of incomputable K, use of all the environments and use of all the interactions), some other trickier ones (time was ignored, possible use of very slow environments) and some other general issues which 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 several new intelligence tests we have introduced.

- The distribution of environments is based on Kt^{max} instead of K. There are many reasons for this change: we cannot wait indefinitely for the environment and, additionally, it is computable and allows us to make the sample.
- The definition now includes a sample of environments, instead of all them. The most important constraint to make this sample more discriminative is that the environment must be reward and observation sensitive.
- Additionally, the complexity of the environments is also progressively adjusted to match the intelligence of the agent, and their rewards weighted accordingly, in order to make the test more effective.
- Interactions are not infinite. Rewards are averaged by the number of actions instead of accumulated. This makes the score less dependent to 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 go 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 get 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.

6.2. Applicability and Implementation

The previous modifications shape *several* new intelligence tests and hence new intelligence definitions, because some tests include time and others not. In the same way, some tests are anytime and some others are not. In order to clarify this, the relation between definitions and tests is summarised in the following table:

Environment	Time	Universal agent	Universal definition	Universal tests
Passive	Ignored	Solomonoff prediction [Solomonoff 1964]	Comprehension ability based on C-test [Hernandez-Orallo 2000a]	C-test [Hernandez-Orallo 1998] / Induction-Enhanced Turing Test [Dowe and Hajek 1997]
Active	Ignored	AIXI [Hutter 2007]	Universal intelligence (simple environments having more weight) [Legg and Hutter 2007]	?[23]
Active	Ignored	?[24]	Universal intelligence (complex environments having more weight)	**Definition 15** (parametric) **Definition 19** (anytime)
Active	Considered	Levin search agent? [Levin 1973]	Universal intelligence (complex environments having more weight) considering time	**Definition 17** (parametric) **Definition 18** (anytime)

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

Although the recommended test is the anytime test considering time (Definition 18), the previous table suggests that several kinds of tests can coexist and the most suitable one can be chosen depending on the application. In fact, the

[23] As originally defined, this evaluation is not possible.

[24] Since time is not considered, brute-force or computationally intractable algorithms would be a possibility here, such as AIXI [Hutter 2005].

different choices of ignoring time or not can be useful to refine or get a better understanding on the relation between speed and intelligence for a specific individual. For instance, the following propositions give us some better understanding on the relation between Definition 15 and Definition 19 which ignore time, and Definition 17 and Definition 18 which consider time.

PROPOSITION 5. Consider two agents π_1 and π_2, where π_2 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. Using Definition 15, we have that for all U, m and n_i, we have that $\Upsilon^{IV}(\pi_1, U, m, n_i) = \Upsilon^{IV}(\pi_2, U, m, n_i)$ and using Definition 19, we have that for all U, $\Upsilon^{VII}(\pi_1, U) = \Upsilon^{VII}(\pi_2, U)$.

PROOF. Trivial from the definitions. Definition 15 and Definition 19 are both based on the reward function $V_\mu^\pi()$, which disregards time. □

PROPOSITION 6. Consider an agent π_1 which improve with experience (i.e. as long as they get more observations, their reward expectancy does not decrease). Consider as well that π_2 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. Using Definition 17 (and a function Φ which is non-decreasing on τ), we have that for all U, m and τ the intelligence $\Upsilon^{V}(\pi_1, U, m, \tau) \geq \Upsilon^{V}(\pi_2, U, m, \tau)$ and using Definition 18, we have that for all U and a sufficiently large Θ then, $\Upsilon^{VI}(\pi_1, U, \Theta) \geq \Upsilon^{VI}(\pi_2, U, \Theta)$.

PROOF. Definition 17 and Definition 18 are based on the reward function $V_\mu^\pi \| \tau$, which is limited by time τ. Since the time limit τ is the same, then π_1 will be able to run n times more actions than π_2. Since both π_1 but π_2 improve with experience, the expected reward for π_1 will be greater than for π_2. In the case of Definition 17 we require the assumption that function Φ which is non-decreasing on τ, and for Definition 18, since it has a adaptation which depends on the results, the samples and evolutions might be different, but as the test time Θ gets larger, the advantage of π_1 will prevail. □

According to the previous result, a good practice to evaluate an unknown agent would be to apply the time-sensitive anytime test (Definition 18) first and next 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.

Regarding implementation of the tests, many issues appear, especially if we want to find an environment class which can be used to evaluate, for instance, adult humans, children, robots, software agents, chimpanzees and dolphins. With the examples in the previous section (despite the fact they are on restricted classes of environments) we have got an idea of what a real test on universal environments could look like. The last example, the grid, can be easily extended to include walls, as mazes, the grid can be converted into another kind of graph with a different (and variable) topology, and many more objects and agents (such as Good and Evil) can be introduced, with universal machines used to generate their movements. The more we are able to generalise the better. This idea resembles Metagame [Pell 1993, 1994], a generalisation of symmetric chess-like games, which uses generative grammars to create many new games inside a family or class of games. However, this is a very restricted class of environments. An example of a universal environment class that could be used to implement the test is developed in [Hernandez-Orallo 2009b]

In general, other more familiar examples can be found, which 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 which arises here is: Is there something we know that would be comparable to these "anytime intelligence tests"? The answer is positive: games, child 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 we play a little time with the easy ones first and then we are given more time to play with some other more difficult ones if we 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 which better entertain the agent but those which are more discriminative. In fact, many experiments in animal cognition have taken placed with virtual environments, which much resemble computer games (and the evolution from 2D to 3D perception on the environments has also taken place in this area, see e.g. [Washburn and Astur 2003]).

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. The use of a sample of environments from a universal distribution ensures that the test measures 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 we can have a biased sample of environments favouring 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 which preclude us from making thorough tests with hundreds of environments.

Choosing environment classes and interfaces which are valid and unbiased for different kinds of agents (humans, non-human animals and machines), such as the one proposed in [Hernandez-Orallo 2009b], is of course a hypothesis we will have to check with further practical investigation on the implementation and administration of the tests.

6.3. Social Intelligence

The previous tests and definitions are aimed at evaluating the intelligence of a single agent. A set of agents working together (if they cooperate to do each action together) could also be measured (like an IQ test made by several people). The way in which the agents can collaborate is completely transparent to the test[25], which just evaluates behaviours.

A different issue is if want to evaluate that thing which is called "social intelligence". In psychometrics and comparative cognition and, more recently, in artificial intelligence, the role of "social intelligence" is more and more vindicated, in front of purely instrumental tests. Some studies support the cultural intelligence hypothesis which says that the quality step in human intelligence is social intelligence, which is present in a much reduced extent in apes and other animals. For instance, [Herrmann et al 2007] develop several specialised tests on children and apes (chimpanzees and orangutans) which "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".

So, we could have a similar controversy here. If this hypothesis is true, the universal tests would not be suitable to evaluate "social intelligence", if we understand "social intelligence" in our framework as the performance of an agent in environments where there are other intelligent agents *inside*, which is a quite specific and rare[26] class of environments.

How can we create environments such that they have intelligent agents *inside*? It is enlightening (but of little practical use) to think that some of the infinite complex 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 a mere computable thing. When we say that it is of little practical use is because the complexity of these environments is extremely high, and the probability of one of them appearing by chance is almost zero. So, we cannot bind the evaluation of social intelligence to this remote chance.

But 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 year of evolution interacting with other individuals, competing with other species and possibly collaborating 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* we usually refer to as animals and plants. Consequently, we require inserting these other agents in the environments.

If we do not want to insert humans and other anthropomorphic (or biological) elements into the environments, an idea is to insert artificial intelligent agents with a clear design, profile and capabilities. The first question about this is to determine the level of intelligence of the agents we want to insert. If we put very simple non-intelligent agents (such as those in the example of section 5.3), there is no possibility to make social learning and apply any social capability.

---

[25] Followers of Searle's Chinese room paradox would disagree here.

[26] So, if the hypothesis is true, the tests would still be valid as "universal" intelligence tests. The problem would be, though, that the environment class in which animals and humans have evolved and specialised their intelligence, (i.e., environments which are full of other animals and humans around, and survival depends mostly on their understanding), is a quite *specific* class of environments.

An option would be to first evaluate many agents alone (including the agent we want to be evaluated). Then we could "insert" some agents of similar intelligence into some environments (some 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. In the end, similar things, although much more arbitrary, are made in the area of multi-agent systems [Shoham and Leyton-Brown 2008] [Weyns et al 2005], and also the comparison with videogames is enlightening, since many videogames include some kind of enemies/friends/foods/platforms with simple or complex behaviours. Here, an option could be to categorise first each agent we incorporate into the environment and possibly classify the environment by the number of agents and their complexity or intelligence.

6.4. Emerging Intelligence

Legg and Hutter claim that their definition is a top-down approach to intelligence (see [Hutter 2005] [Legg and Hutter 2007]), especially because they define an optimal agent as a theoretical thing that we can use to develop less intelligent (but more feasible) approximations (as shown in [Veness et al 2009]). That is certainly an option and perhaps it will be the best option once we get machine intelligence over human capabilities.But at the present moment, we prefer to think about a bottom-up approach. This can be understood in at least a couple of ways. One possibility is to consider a very simple agent, better than random, and improve it in some way such that its intelligence increases. The idea here is to find "improvements" in such a way that we can find a continuum up to a level of intelligence compared to a human being and beyond. If there is something that, the more we put on that thing to an agent it will get better results on the test, then we would have eventually solved the problem because "that thing" would be "operative intelligence". Inside this possibility we could include an evolutionary process/algorithm, since if we have a test, we have a fitness function and we could evolve agents into more intelligent ones. However, it is not clear how this process could be sped up (the evolution in human brains took millions of years). A clearer thing is that an intelligence test is a requirement in this approach.

But another possibility is to design an agent which is better than random and try to *combine* it with other agents of similar capabilities in order to get more and more intelligence. This approach is known as "emergent intelligence" [Huneman and Humphreys 2008]. Given a formal definition of intelligence and its test variant, we could try to formally prove that given two agents a_1 and a_2, if we combine them in some way, we could get better results. If we can do that without reaching a saturation point, we would have another path towards intelligence. Note the relevance of considering time here; otherwise we would be able to combine agents without any "extra" computation overload. Increasing intelligence by combining several parts usually must take into account the slowing overload that it might imply.

Turing himself [Turing 1950] uses the fact that an onion is merely made up of layers of onion skin to raise the possibility that intelligence is emergent. We have some evidence in favour of this, such as the human brain or swarm intelligence. The authors have earlier discussed [Dowe 2008a, sec. 0.2.7] the notion of quantifying the intelligence of a system of agents and endeavouring to quantify how much of this comes from the individual agents (in isolation) and how much comes from their communication. Note that as long as emergent intelligence is based on the cooperation of relatively independent agents, the notions of social intelligence and emergent intelligence are closely interwoven (as it is, e.g., in swarm intelligence). This is even more so when complex communication takes place, or some kind of proto-language emerges or it is hard-wired on agents.

An interesting related issue is the connection between prediction and induction, since the combination of several agents' predictions (if collaboration is made in this way) would appear to fit prediction better than induction. That would make Solomonoff prediction better suited than MML induction [Wallace and Dowe 1999a] [Wallace 2005 (sec. 10.1] [Dowe 2008a, sec 0.3.1] because the former combines several theories and the latter uses "one" theory and the agents should reach a consensus on the underlying model (not only on the predictions). So, whether combination through weighting or other more elaborated techniques is better than consensus is an open question in general, for which testing tools would also be useful.

7. Conclusions

This paper represents a very important challenge which might have strong and direct implications in many fields: artificial intelligence, psychometrics, comparative cognition, philosophy, ... We have developed a set of tests and, especially, an *anytime* intelligence test, which 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 get a rough approximation of the intelligence of an agent with a small amount of time and much better approximations with more time. Additionally, the term also originates

from the issue 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 quick intelligent agents, also incorporating these times into the measurement.

It is fair to recognise that the first anytime intelligence test in artificial intelligence is, precisely, the Turing Test. The Turing Test requires a human for its implementation, it is anthropomorphic and it tests humanity rather than intelligence, but it is anytime at least in the sense that the more time we allow the examiner 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 which are considered here is that intelligence is defined as an average which converges in the limit, i.e., for infinitely large environments. Many very complex environments have simple patterns and consequently the performance of an agent which 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, so 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 would not work).

Some other less determinant choices are our selection of environments. We have restricted environments to be observation and reward sensitive (without loss of generality, in our opinion, but it makes evaluation much more efficient) and the most important one, to be balanced, which changes rewards from a range between 0 and 1 into a range between –1 and 1, which, additionally 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, in [Hernandez-Orallo 2009b] we present a universal environment class which complies 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, collaborative platforms, social networks, psychometrics, animal comparative cognition, etc. In the mid-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 would provide enormous feedback information to improve intelligent systems. This would not only possible on general artificial 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), as those shown in some examples here.
- New generation of CAPTCHAs [von Ahn et al 2002, 2008] taking 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 mid-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.
- At a 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), which some (optimistic) researchers place around about twenty years ahead from now. This means that in about that time, we will require a battery of tests for different kinds of factors and environment classes, 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) will help to focus the ethical debate that will be generated near the moment of the singularity.

In other words, and 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 into 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 is possible around many different lines. Implementation and experimentation using the tests are on the top of the list. Experimentation on any kind of subjects (humans, non-human animals,

artificial intelligent systems) would bring valuable information from which we can learn lessons and improve, in case, 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 in [Hernandez-Orallo and Minaya-Collado 1998] and [Hernandez-Orallo 2000])) is certainly a challenge, but it is one of the combinations where we can obtain more useful information. But, implementations on restricted environment classes are also of utmost relevance in artificial intelligence. We now have a general theory on how to evaluate agents solving restricted problems such as maze learning, 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 to the environment we used in the example of section 5.3. In [Zatuchna and Bagnall 2009], for instance, they 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 a huge work) but are crucial for the real evaluation of the progress in artificial intelligence. But, 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 [Weyns et al. 2005]) 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 which are able to measure intelligence and speed at the same time, without a batch combination of tests as we suggested in the previous section.

There is also much theoretical work ahead. Some decisions 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 done here for random agents and naïve agents). In this regard, the completion of the taxonomy of environments, as it appears in [Legg 2008], is one of the most appealing things to do first, including the new environment definitions we have introduced here. A formalisation of the notion of social environment, their parametrisation and the inclusion in the taxonomy would also be necessary. And, finally, emergent intelligence and the analysis of its relation with multi-agent systems and ensemble methods (in machine learning), under the view of the tests (either experimentally or theoretically), is also a promising line of research.

Overall, the practical experimentation and the theoretical exploration of the notion of anytime universal intelligence test will tell whether some of the choices introduced in this paper can be improved, will help understand their implications and set up future research.

Acknowledgments

This work has benefited from discussions and suggestions made from many people from different areas. We want to 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 embark on this work. The early idea of anytime intelligence test and its applicability to areas out of artificial intelligence matured from insightful comments from María Victoria Hernández-Lloreda and Sergio España. The authors also have to thank a grant from the Spanish *Ministerio de Educación y Ciencia* (MEC) 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 TIN2009-06078-E, CONSOLIDER-INGENIO 26706 and TIN 2007-68093-C02, and GVA project PROMETEO/2008/051.

References

[Agrawal et al. 2004] Agrawal, M.; Kayal, N.; Saxena, N. "PRIMES is in P", Annals of Mathematics 160 (2004), no. 2, pp. 781–793.

[Dowe 2008a] Dowe, D.L. "Foreword re C. S. Wallace", Computer Journal, 51, 5, pp. 523 – 560, September, 2008.

[Dowe 2008b] Dowe, D.L. "Minimum Message Length and statistically consistent invariant (objective?) Bayesian probabilistic inference - from (medical) "evidence"", Social Epistemology, 22, 4, October – December, pp 433-460, 2008.

[Dowe and Hajek 1997a] Dowe, D.L. and Hajek, A.R. "A computational extension to the Turing Test", Proceedings of the 4th Conference of the Australasian Cognitive Science Society, Newcastle, NSW, Australia, September 1997.

[Dowe and Hajek 1997b] Dowe, D.L. and Hajek, A.R. "A computational extension to the Turing Test", Technical Report #97/322, Dept Computer Science, Monash University, Melbourne, 9pp, 1997. http://www.csse.monash.edu.au/publications/1997/tr-cs97-322-abs.html

[Dowe and Hajek 1998] Dowe, D.L. and Hajek, A.R."A non-behavioural, computational extension to the Turing Test", pp101-106, Proceedings of the International Conference on Computational Intelligence and Multimedia Applications (ICCIMA'98), Gippsland, Australia, February 1998

[Embretson 1998] Embretson, S. E. "A cognitive design system approach to generating valid tests: Application to abstract reasoning". Psychological Methods, 3, 300-396, 1998.

[Embretson and Mc Collam 2000] Embretson, S.E. amd McCollam, K.M.S. Psychometric approaches to understanding and measuring intelligence. En R.J. Sternberg (Ed.). Handbook of intelligence (pp. 423-444). Cambridge, UK: Cambridge University Press, 2000.

[Embretson and Reise 2000] Embretson, S. and Reise, S. Item response theory for psychologists. Mahwah, NJ: Erlbaum, 2000.

[Embretson 2004] Embretson, S. E. "Measuring human intelligence with artificial intelligence: Adaptive item generation". In R. J. Sternberg and J. Pretz (Eds.). Cognition and Intelligence. New York: Cambridge University Press. (http://psychology.gatech.edu/departmentinfo/faculty/bio-SEmbretson.html), 2004.

[Hernandez-Orallo and Minaya-Collado 1998] Hernández-Orallo, J.; Minaya-Collado, N.: "*A Formal Definition of Intelligence Based on an Intensional Variant of Kolmogorov Complexity*", Proceedings of the International Symposium of Engineering of Intelligent Systems (EIS'98) ICSC Press 1998, pp. 146-163.

[Hernandez-Orallo 2000a] Hernández-Orallo, J.: "Beyond the Turing Test. Journal of Logic, Language and Information" 9(4): 447-466 (2000).

[Hernandez-Orallo 2000b] Hernández-Orallo, J. "*On The Computational Measurement of Intelligence Factors*", appeared in A. Meystel "Performance Metrics for Intelligent Systems Workshop", National Institute of Standards and Technology, Gaithersburg, MD, USA, August 14-16, 2000, pp. 1-8, section XXI.

[Hernandez-Orallo 2009a] Hernández-Orallo, J.: "On Discriminative Environments, Randomness, Two-part Compression and MML", Technical Report, Available at http://users.dsic.upv.es/proy/anynt/

[Hernandez-Orallo 2009b] Hernández-Orallo, J.: "A (hopefully) Non-biased Universal Environment Class for Measuring Intelligence of Biological and Artificial Systems", Under Review, 2009. Available at http://users.dsic.upv.es/proy/anynt/

[Hernandez-Orallo 2009c] Hernández-Orallo, J.: "On Evaluating Agent Performance in a Fixed Period of Time", Under Review, 2009. Available at http://users.dsic.upv.es/proy/anynt/

[Herrmann et al 2007] Herrmann, E., Call, J., Hernández-Lloreda, M.V., Hare, B., Tomasell, M. "Humans Have Evolved Specialized Skills of Social Cognition: The Cultural Intelligence Hypothesis", Science, 7 September 2007, Vol. 317. no. 5843, pp. 1360 - 1366, DOI: 10.1126/science.1146282

[Hibbard 2009] Hibbard, B. "Bias and No Free Lunch in Formal Measures of Intelligence" Journal of Artificial General Intelligence, 54-61, vol. 1, no. 1, 2009.

[Huneman and Humphreys 2008] Huneman, P., Humphreys, P. "Dynamical Emergence and Computation: An Introduction", Special Issue in Minds and Machines, Volume 18 , Issue 4 (December2008)

[Hutter 2005] Hutter, M. "Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability", Springer, 2005.

[Hutter 2007] Hutter, M. "Universal algorithmic intelligence: A mathematical top→down approach". In Artificial General Intelligence, pages 227–290. Springer, Berlin, 2007.

[Hutter 2009] Hutter, M. "Open Problems in Universal Induction and Intelligence", Algorithms, vol. 3, no. 2, pp. 879-906, 2009 (http://arxiv.org/abs/0907.0746).

[Legg and Hutter 2005] Legg, S. and Hutter, M. "A universal measure of intelligence for artificial agents". In Proc. 21st International Joint Conf. on Artificial Intelligence (IJCAI-2005), pages 1509–1510, Edinburgh, 2005.

[Legg and Hutter 2007] Legg, S. and Hutter, M. "Universal Intelligence: A Definition of Machine Intelligence". In *Minds and Machines*, pages 391-444, volume 17, number 4, November 2007. http://www.vetta.org/documents/UniversalIntelligence.pdf

[Legg 2008] Legg, S. "Machine Super Intelligence", Ph.D. thesis, Department of Informatics, University of Lugano, Switzerland, June 2008. http://www.vetta.org/documents/Machine_Super_Intelligence.pdf

[Levin 1973] Levin, L.A. "Universal search problems" Problems Inform. Transmission, 9:265-266, 1973.

[Li and Vitányi 2008] Li, M.; Vitányi, P. "An Introduction to Kolmogorov Complexity and its Applications" 3rd Edition. Springer-Verlag, New York, 2008

[Lord 1980] Lord, F.M. Applications of item response theory to practical testing problems. Mahwah, NJ: Erlbaum, 1980.

[Pell 1993] Pell, B. *Strategy Generation and Evaluation for Meta-Game Playing.* Thesis dissertation, University of Cambridge, 1993. http://www.barneypell.com/papers/pell-thesis.pdf

[Pell 1994] Pell, B. "A Strategic Metagame Player for General Chesslike Games" AAAI 1994: 1378-1385

[Sanghi and Dowe 2003] Sanghi, P. and Dowe, D.L. "A computer program capable of passing I.Q. tests", Proc. 4th International Conference on Cognitive Science (and 7th Australasian Society for Cognitive Science Conference), ICCS ASCS 2003 (http://www.cogsci.unsw.edu.au), Univ. of NSW, Sydney, Australia, 13-17 July 2003, Vol. 2, pp. 570-575.

[Schmidhuber 2002] Schmidhuber, J. "The speed prior: A new simplicity measure yielding nearoptimal computable predictions". In *Proc. 15th Conf. on Computational Learning Theory (COLT'02)*, volume 2375 of *LNAI*, pages 216–228, Sydney, 2002. Springer, Berlin.

[Searle 1980] Searle, J. (1980), "Minds, Brains and Programs", Behavioral and Brain Sciences 3 (3): 417–457

[Shoham and Leyton-Brown 2008] Shoham, Y. and Leyton-Brown, K., *Multi-agent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*, Cambridge University Press, 2008

[Solomonoff 1964] Solomonoff, R.J. "A formal theory of inductive inference" inform. Contr. vol. 7, pp. 1-22, Mar. 1964; also, pp. 224-254, June 1964.

[Solomonoff 1986] Solomonoff, R.J. "The Application of Algorithmic Probability to Problems in Artificial Intelligence" in L.N.Karnal and J.F. Lemmer (eds.) *Uncertainty in Artificial Intelligence* (L.N. Karnal and J.F. Lemmer, eds.), Elsevier Science, pp. 473-491, 1986.

[Turing 1950] Turing, A. "Computing Machinery and Intelligence" Reprinted in "Minds and Machines", edited by Alan Ross Anderson, Englewood Cliffs, N.J., Prentice Hall 1964.

[Veness et al 2009] Joel Veness, Kee Siong Ng, Marcus Hutter, and David Silver. A Monte Carlo AIXI Approximation. CoRR, abs/0909.0801, 2009. informal publication (http://arxiv.org/abs/0909.0801)

[von Ahn et al 2002] von Ahn, L.; Blum, M.; Langford, J. "Telling Humans and Computers Apart (Automatically) or How Lazy Cryptographers do AI" Communications of the ACM, 2002. www.captcha.net.

[von Ahn et al 2008] von Ahn, L.; Maurer, B.; McMillen, C.; Blum, M. "reCAPTCHA: Human-Based Character Recognition via Web Security Measures" Science, Vol. 321, September 12th, 2008.

[Wainer et al. 2000] Wainer, H.; Dorans, N.J.; Flaugher, R.; Green, B.F.; Mislevy, R.J.; Steinberg, L.; Thissen, D.; Wainer, H. *Computerized Adaptive Testing: A Primer*, Lawrence Erlbaum 2000.

[Wallace 2005] Wallace, C.S. "Statistical and Inductive Inference by Minimum Message Length", Springer Verlag, 2005.

[Wallace and Boulton 1968] Wallace, C.S. and Boulton, D. M. "An Information Measure for Classification" *Computer Journal, 11, 2,* pp. 185-195, 1968.

[Wallace and Dowe 1999a] Wallace, C.S. and Dowe, D.L. "Minimum Message Length and Kolmogorov Complexity", Computer Journal, 4, 42, pp. 270-283, 1999.

[Washburn and Astur 2003] Washburn, D.A. and Astur, R.S. "Exploration of virtual mazes by rhesus monkeys (Macaca mulatta), Animal Cognition, Vol.6, No3, pp. 161-168, 2003.

[Weyns et al 2005] Danny Weyns, H.V.D. Parunak, F. Michel, Tom Holvoet, and J. Ferber. Environments for multi-agent systems, state-of-the-art and research challenges. In Environments for multi-agent systems, volume 3374 of Lecture Notes in Computer Science, pages 1-48. Held with the 3th Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS, Springer-Verlag, 2005.

[Zatuchna and Bagnall 2009] Zatuchna Z., Bagnall A. "Learning Mazes with Aliasing States: An LCS Algorithm with Associative Perception" Adaptive Behavior 17(1): 28-57, 2009.