C-tests revisited: back and forth with complexity

José Hernández-Orallo

DSIC, Universitat Politècnica de València, Spain
jorallo@dsic.upv.es

Abstract. We explore the aggregation of tasks by weighting them using a difficulty function that depends on the complexity of the (acceptable) policy for the task (instead of a universal distribution over tasks or an adaptive test). The resulting aggregations and decompositions are (now retrospectively) seen as the natural (and trivial) interactive generalisation of the C-tests.

Keywords: intelligence evaluation, artificial intelligence, C-tests, algorithmic information theory, universal psychometrics, agent response curve

1 Introduction

A first test using algorithmic information theory (AIT) was the C-test [9,2], where the goal was to find a continuation of a sequence of letters, as in some IQ tasks , and in the spirit of Solomonoff's inductive inference problems: "given an initial segment of a sequence, predict its continuation" (as quoted in [12, p.332]). Levin's Kt complexity (see, e.g., [12, sec.7.5]) was used to calculate the difficulty of a sequence of letters. The performance was measured as an aggregated value over a range of difficulties:

$$I(\pi) \triangleq \sum_{h=1}^{H} h^e \sum_{i=1}^{N} \frac{1}{N} \text{Hit}(\pi, x_{i,h}) \tag{1}$$

where π is the subject, the difficulties range from $h = 1$ to H and there are N sequences $x_{i,k}$ per difficulty h. The function hit returns 1 if π is right with the continuation and 0 otherwise. If $e = 0$ we have that all difficulties have the same weight. The N sequences per difficulty were chosen (uniformly) randomly.

This contrasts with a more common evaluation in artificial intelligence based on average-case performance according to a probability of problems or tasks:

$$\Psi(\pi) \triangleq \sum_{\mu \in M} p(\mu) \cdot \mathbb{E}[R(\pi, \mu)] \tag{2}$$

where p is a probability distribution on the set of tasks M, and R is a result function of agent π on task μ. Actually, eq. 2 can also be combined with AIT, in a different way, by using a universal distribution [14,12], i.e., $p(\mu) = 2^{-K(\mu)}$, where $K(\mu)$ is the Kolmogorov complexity of μ, as first chosen by [11].

The work in [11] has been considered a generalisation of [9,2], from static sequences (predicting a continuation of a sequence correctly) to dynamic environments. In this paper we challenge this interpretation and look for a proper generalisation of [9,2] using the notion of difficulty in the outer sum, as originally conceived and seen in eq. 1. The key idea is the realisation that for the C-test the task and the solution were the same thing. This meant that the difficulty was calculated as the size of the simplest program that generates the sequence, which is both the task and the solution. Even if the complexity of the task and the solution coincide here, it is *the complexity of the solution what determines the difficulty of the problem.*

However, when we move from sequences to environments or other kind of interactive tasks, the complexity of the policy that solves the task and the complexity of the environment are no longer the same. In fact, this is discussed in [7,6]: the complexity of the environment is roughly an upper bound of the complexity of the acceptable policies (any agent that reach an acceptable performance value), but very complex environments can have very simple acceptable policies. In fact, the choice of $p(\mu) = 2^{-K(\mu)}$ has been criticised for giving too much weight to a few environments. Also, it is important to note that the invariance theorem is more meaningful for Kolmogorov Complexity than for Algorithmic Probability, as for the former it gives some stability for values of K that are not very small, but for a probablity it is precisely the small cases that determine most of the distribution mass. In fact, for any computable distribution p there is a choice of a reference UTM that leads to a particular universal distribution that approximates p (to whatever required precision.This means that the choice of $p(\mu) = 2^{-K(\mu)}$ for Eq. 2 is actually a metadefinition, which leads to virtually any performance measure, depending on the Universal Turing Machine (UTM) that is chosen as reference.

By decoupling the complexity of task and policy we can go back to eq. 1 and work out a notion of difficulty of environments that depends on the complexity of the policy. While this may look retrospectively trivial, and the natural extension in hindsight, we need to solve and clarify some issues, and properly analyse the relation of the two different philosophies given by eq. 2 and eq. 1.

Section 2 discusses some previous work, introduces some notation and recovers the difficulty-based decomposition of aggregated performance. Section 3 introduces several properties about difficulty functions and the view of difficulty as policy complexity. Section 4 discusses the choices for the difficulty-dependent probability. Section 5 briefly deals with the role of computational steps for difficulty. Section 6 closes the paper with a discussion.

2 Background

AI evaluation has been performed in many different ways (for a recent account of AI evaluation, see [5]), but a common approach is based on averaging performance on a range of tasks, as in eq. 2.

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

Fig. 1. Several series of different difficulties 9, 12, and 14 used in the C-test [2].

In what follows, we will focus on the approaches that are based on AIT. As mentioned above, the first intelligence test using AIT was the so-called C-test [9,2]. Figure 1 shows examples of sequences that appear in this test. The difficulty of each sequence was calculated as Levin's Kt, a time-weighted version of Kolmogorov complexity K. Some preliminary experimental results showed that human performance correlated with the absolute difficulty (h) of each exercise and also with IQ test results for the same subjects ([9,2]). They also show a clear inverse correlation of results with difficulty (see Figure 2). HitRatio is defined as the inner sum of eq. 1:

$$\text{HitRatio}(\pi, h) \triangleq \sum_{i=1}^{N} \frac{1}{N} \text{Hit}(\pi, x_{i,h}) \tag{3}$$

An interesting observation is that by arranging problems by difficulty we see that HitRatio seems to be very small from a given difficulty value (in the figure this is 8, but it can be any other, usually small, value). This makes the estimation of the measure much easier, as we only need to focus on (the area of) a small interval of difficulties. In fact, this use of difficulty is common in psychometrics.

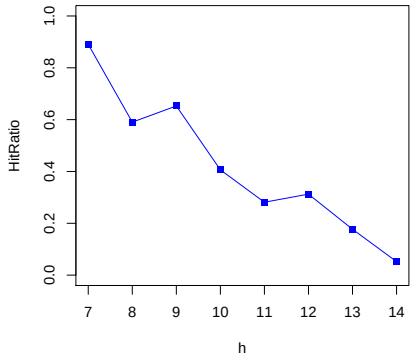

Fig. 2. Results obtained by humans on task of different difficulty in the C-test [2].

Several generalisations of the C-test were suggested (for "cognitive agents [...] with input/output devices for a complex environment" [9] where "rewards and penalties could be used instead" [4]) or extending them for other cognitive abilities [3], but not fully developed.

AIT and reinforcement learning were finally combined in [11], where all possible environments were considered in eq. 2, instantiated with a universal distribution for p, i.e., $p(\mu) = 2^{-K(\mu)}$, with $K(\mu)$ being the Kolmogorov complexity of

each environment μ. Some problems (computability, discriminating power, overweight for small environments, time, ...) were discussed with the aim of making a more applicable version of this appraoch by [10] and [7, secs. 3.3 and 4].

While the aim of all these proposals was to measure intelligence, many interesting things can happen if AIT is applied to cognitive abilities other than intelligence, as suggested in [3] for the passive case and hinted in [7, secs. 6.5 and 7.2] for the dynamic cases, which proposes the use of different kinds of videogames as environments (two of the most recently introduced benchmarks and competitions in AI are in this direction [1,13]).

We consider tests that are composed of tasks (also called environments or items) and are performed by agents (also called policies or subjects). The set of tasks is denoted by M. Its elements are usually denoted by μ. The set of agents is denoted by Π. Its elements are usually denoted by π. Both can be stochastic, i.e., they *can* use a probabilistic instruction or transition function. The length in bits of a string is denoted by $L(x)$. We can think of a proper encoding of tasks and agents as strings. Given a UTM U we define Kolmogorov complexity as $K_U(y) = \min_{x\,:\,U(x)=y} L(x)$. We will usually drop the subindex U. Finally, we have the expected value of the response, score or result of π in μ for a time limit τ as $\mathbb{E}[R(\pi,\mu,\tau)]$. The value of τ will be usually omitted. The R function always gives values between 0 and 1 and we assume it is always defined (a value of $R = 0$ is assumed for non-halting situations) We also define $\mathbb{E}[LS(\pi,\mu,\tau)] \triangleq L(\pi) + \log \mathbb{E}[S(\pi,\mu,\tau)]$. Logarithms are always binary.

It is actually in [8], where we can find a first connection between the schemas of eq. 1 and eq. 2. We adapt definition 14 in [8], which is a generalisation of eq. 2, by making the set M and the task probability p explicit as a parameters.

Definition 1. *The expected average result for a task class M, a distribution p and an agent π is:*

$$\Psi(\pi, M, p) \triangleq \sum_{\mu \in M} p(\mu) \cdot \mathbb{E}[R(\pi,\mu)] \tag{4}$$

And now we use proposition 4 in [8] that decomposes it. First, we define partial results for a given difficulty h as follows:

$$\Psi_h(\pi, M, p) \triangleq \sum_{\mu \in M, \hbar(\mu)=h} p(\mu|h) \cdot \mathbb{E}[R(\pi,\mu)] \tag{5}$$

Where $\hbar$ is a difficulty function $\hbar : M \to \mathbb{R}^+ \cup 0$. Note that this parametrises the result of eq. 4 for different difficulties. For instance, for two agents π_A and π_B we might have that $\Psi_3(\pi_A) < \Psi_3(\pi_B)$ but $\Psi_7(\pi_A) > \Psi_7(\pi_B)$. If we represent $\Psi_h(\pi, M, p)$ on the y-axis versus h on the x-axis we have a so-called agent response curve, much like Fig. 2.

If we want to get a single number from an agent response curve we can aggregate performance for a range of difficulties, e.g., as follows:

Proposition 1. *([8, prop. 4]) The expected average result $\Psi(\pi, M, p)$ can be rewritten as follows: in the particular case when $\hbar$ only gives discrete values:*

$$\Psi(\pi, M, p) = \sum_{h=0}^{\infty} p(h)\Psi_h(\pi, M, p) \tag{6}$$

where $p(h)$ is a discrete probability function for eq. 6. Note that equations 4, 5 and 6 are generalisations, respectively, of equations 2, 3 and 1.

3 Difficulty functions

Before setting an appropriate measure of difficulty based on the policy, in this section we will analyse which properties a difficulty function may have.

The decomposition in previous section suggests that we could try to fix a proper measure of difficulty first and then think about a meaningful distribution $p(h)$. Once this is settled, we could try to find a distribution for all environments of that difficulty $p(\mu|h)$. In other words, once we determine how relevant a difficulty is we ask which tasks to take for that difficulty. This is the spirit of the C-test [9,2] as seen in eq. 1. In fact, we perhaps we do not need a $p(h)$ that decays dramatically, as it is expectable to see performance to decrease for increasing difficulty, as in Figure 2.

To distinguish $p(h)$ and $p(\mu|h)$ we will denote the former with w and the latter with p_M. We will use any distribution or even a measure (not summing up to one, for reasons that we will see later on) as a subscript for Ψ. For instance, we will use the following notation $\Psi_{\mathcal{U}(h_{min},h_{max})}(\pi, M, p_M)$, where $\mathcal{U}(a, b)$ represents a uniform distribution between a and b. For instance, we can have two agents π_A and π_B such that $\Psi_{\mathcal{U}(1,10)}(\pi_A) > \Psi_{\mathcal{U}(1,10)}(\pi_B)$ but $\Psi_{\mathcal{U}(11,20)}(\pi_A) < \Psi_{\mathcal{U}(11,20)}(\pi_B)$. We will use the notation $\Psi_{\oplus}(\pi, M, p_M)$ when $w(h) = 1$ (note that this is not the uniform distribution for discrete h), which means that the partial aggregations for each difficulty are just added. In other words, $\Psi_{\oplus}(\pi, M, p_M) \triangleq \sum_{h=0}^{\infty} \Psi_h(\pi, M, p_M)$ for discrete difficulty functions . We will explore whether this (area under the agent response curve) is bounded.

Figure 3 shows approach A, which has already been mentioned, while approaches B and C will be seen in sections 4.1 and 4.2 respectively.

When we aggregate environments with different scales on R and different difficulties, we may have that an agent focusses on a few environments with high difficulty while another focusses on many more environments with small responses. Agent response curves in [8], which are inspired by item response curves in psychometrics (but inverting the view between agents and items), allow us to see how each agent performs for different degrees of difficulty. Looking at Figure 2 and similar agent response curves in psychometrics, we see that the notion of difficulty must be linked to R, i.e., how well the agents perform, and not about the complexity of the task, as in the previous section.

Another option is what is done in [6], as $\hbar(\mu) \triangleq \min_{\pi:\mathbb{E}[R(\pi,\mu)]=R_{max}(\mu)} L(\pi)$ where $R_{max}(\mu) = \max_{\pi} \mathbb{E}[R(\pi, \mu)]$. However, R_{max} may be hard to calculate

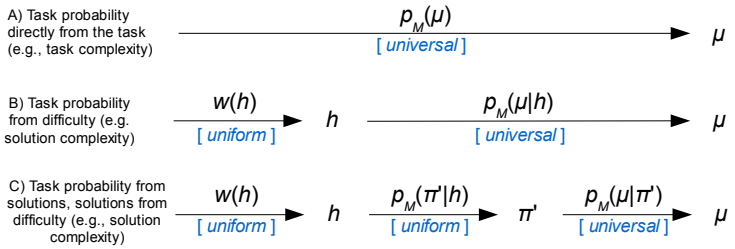

Fig. 3. Three approaches to aggregate the results for a set of tasks. Top (A) shows the classical approach of choosing a probability for the task, according to the properties of the task. Middle (B) shows the approach where we arrange tasks by difficulty, and the notion of difficulty is derived from the properties of the policy. Bottom (C) shows a variation of B where we derive acceptable policies for a given difficulty and then generate tasks for each policy. Between square brackets some choices we examine in this paper.

and even if it can be effectively calculated, any minor mistake or inefficiency in a very good agent will prevent the agent from reaching the optimal result, leading to a notion of difficulty linked to the complexity of the 'perfect' policy. In [6], a 'tolerance value' is considered and, instead of one policy, difficulty is linked to the probability of finding a policy under this tolerance.

We are going to consider this tolerance ϵ of acceptability.

$$\mathbb{A}^{[\epsilon]}(\pi, \mu) \triangleq \mathbf{1}(\mathbb{E}[R(\pi, \mu)] \geq 1 - \epsilon) \tag{7}$$

This returns 1 if the expected response is above $1 - \epsilon$ and 0 otherwise. If $\mathbb{A}^{[\epsilon]}(\pi, \mu) = 1$ we say that π is ϵ-acceptable. With this, we binarise responses. One can argue that we could have just defined a binary R, but it is important to clarify that it is not the same to have tolerance for each single R (or a binarised R) than to have a tolerance for the expected value $\mathbb{E}[R]$. The tolerance on the expected value allows the agent to have variability in their results (e.g., stochastic agents) provided the expected value is higher than the tolerance. Finally, even if we will be very explicty about the value of ϵ, and changing it will change the difficulty value of any environment, it is important to say that this value is not so relevant. The reason is that for any environment we can build any other environment where the responses are transformed by any function. In fact, we could actually consider one fixed threshold, such as 0.5, always.

And now we can just define a new version of eq. 5 using this new function:

$$\Psi_h^{[\epsilon]}(\pi, M, p_M) \triangleq \sum_{\mu \in M, \hbar(\mu)=h} p_M(\mu|h) \cdot \mathbb{A}^{[\epsilon]}(\pi, \mu) \tag{8}$$

We can just rewrite equations 6 accordingly:

$$\Psi_w^{[\epsilon]}(\pi, M, p_M) = \sum_{h=0}^{\infty} w(h) \Psi_h^{[\epsilon]}(\pi, M, p_M) \tag{9}$$

Given the above, we are now ready for a few properties about difficulty functions.

Definition 2. *A difficulty function $\hbar$ is strongly bounded in M if for every π there is a difficulty h such that for every $\mu \in M : \hbar(\mu) \geq h$ we have $\mathbb{A}^{[\epsilon]}(\pi, \mu) = 0$.*

Now we choose the difficulty function in terms of ϵ-acceptability, i.e.:

$$\hbar^{[\epsilon]}(\mu) \triangleq \min\{L(\pi) : \mathbb{E}[R(\pi, \mu)] \geq 1 - \epsilon\} = \min\{L(\pi) : \mathbb{A}^{[\epsilon]}(\pi, \mu) = 1\} \quad (10)$$

We can say a few words about the cases where a truly random agent (choosing actions at random) gives an acceptable policy for an environment. If this is the case, we intuitively consider the environment easy. So, in terms, of L, we consider random agents to be simple, and goes well with our consideration of stochastic agents and environments having access to some true source of randomness.

Figure 4 (left) shows the distribution of response according to $L(\pi)$, but setting $\epsilon = 0.9$. We see that the simplest ϵ-acceptable policy has $L = 12$.

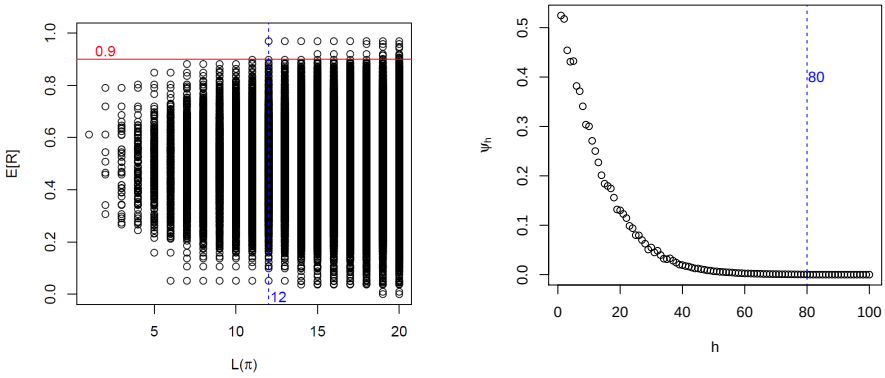

Fig. 4. Left: an illustrative distribution of responses of a population of agents for a single environment. If we set the threshold at $0.9 = 1 - \epsilon$, the simplest policy above this threshold is of 'complexity' $h = 12$. Right: an illustrative distribution of the result of Ψ_h considering a population of environments for a single agent, an *agent response curve*. There is no task π of difficulty above 80 for which $\mathbb{E}[R(\pi, \mu)] \geq 1 - \epsilon$, i.e., there is no task for which π is ϵ-acceptable, so Ψ_h is 0 from 80 on. If we were using the definition of $\hbar$ as for Eq. 10, this 80 would be $L(\pi)$. Also note on the right plot that all 'heaven' tasks (good results independently of what the agent does) are at $h = 1$, while all 'hell' tasks (bad response independently of what the agent does) are at $h = \infty$.

With the difficulty function in eq. 10 we have:

Proposition 2. *The difficulty function $\hbar^{[\epsilon]}$ in eq. 10 is strongly bounded.*

Proof. For every policy π, if a task μ has a difficulty $\hbar^{[\epsilon]}(\mu) > L(\pi)$, it means that π is not ϵ-acceptable, because otherwise the difficulty would be $L(\pi)$ and not h. Consequently, $\mathbb{A}^{[\epsilon]}(\pi, \mu) = 0$ for all μ of difficulty $\hbar^{[\epsilon]}(\mu) > L(\pi)$. It is sufficient to take $h > L(\pi)$ for every π to see that $\hbar$ is strongly bounded.

This is what we see in Fig. 4 (right), where $L(\pi) = 80$. With $\hbar^{[\epsilon]}$ in eq. 10, we can ensure that the values are going to be 0 from $h = 80$ on.

This may not be the case for other difficulty functions. We can imagine a situation where the curve never converges to zero. For instance, if the difficulty function is decoupled from resources (length and/or steps) of the acceptable policies or we do not use the notion of ϵ-acceptability then we cannot avoid that a very simple policy could eventually score well in a problem with very high difficulty. This would be counter-intuitive, as if there is a simple policy for a difficult problem, the latter should not be considered difficult any more.

4 Difficulty-conditional task probabilities

In the previous sections we have focussed on $w(h)$ and whether it is necessary or not. We have seen difficulty functions where just aggregating Ψ_h without $w(h)$ (or $w(h) = 1$) leads to a $\Psi_\oplus(\pi, M, p_M)$ that is bounded. The question now is how to choose the conditional probability $p_M(\mu|h)$. In the C-test, eq. 1, this was chosen as a uniform distribution. However, this is not possible in an interactive scenario if we consider all possible tasks, as the number of tasks for which there is an acceptable policy π of $L(\pi) = n$ can be infinite. Even if we cannot set a uniform distribution, we want a choice of $p_M(\mu|h)$ that keeps the task diversity (unless there is any special bias to choose the tasks).

4.1 Task probability depends on difficulty

The first thing we can do is to assume $p(\mu|h)$ in eq. 8 as $p(\mu|h) = \frac{2^{-K(\mu)}}{\nu(h)}$ if $\hbar^{[\epsilon]}(\mu) = h$ and 0 otherwise, where $\nu(h)$ is a normalisation term to make the mass of the distribution equal to 1, which can be formulated as $\nu(h) = \sum_{\mu:\hbar^{[\epsilon]}(\mu)=h} 2^{-K(\mu)}$.

And now we have:

$$\Psi_h^{[\epsilon]}(\pi, M, p_M) = \sum_{\mu\in M,\hbar^{[\epsilon]}(\mu)=h} p_M(\mu|h) \cdot \mathbb{A}^{[\epsilon]}(\pi, \mu) = \frac{1}{\nu(h)} \sum_{\mu\in M,\hbar^{[\epsilon]}(\mu)=h} 2^{-K(\mu)} \cdot \mathbb{A}^{[\epsilon]}(\pi, \mu)$$

From here, we can plug it into eq. 9 for the discrete case:

$$\Psi_w^{[\epsilon]}(\pi, M, p_M) = \sum_{h=0}^{\infty} w(h) \frac{1}{\nu(h)} \sum_{\mu\in M,\hbar^{[\epsilon]}(\mu)=h} 2^{-K(\mu)} \cdot \mathbb{A}^{[\epsilon]}(\pi, \mu) \qquad (11)$$

Note that the above is going to be bounded independently of the difficulty function if w is a probability distribution. Also notice that $\frac{1}{\nu(h)}$ is on the outer sum, and that $\nu(h)$ is lower than 1, so the normalisation term is actually greater than 1.

And if we use any of the difficulty functions in equations 10 we can choose $w(h) = 1$ and $\Psi_\oplus^{[\epsilon]}(\pi, M, p_M)$ is bounded.

4.2 Task probability depends on the policy probability

One of things of the use of equation 10 is that the number of acceptable policies per difficulty is finite. This is what happened in the C-test and that is the reason why a uniform distribution could be used for the inner sum. We could try to decompose the inner sum by using the policy and get the probability of the task given the policy.

The interpretation would be as follows: for each difficulty value we aggregate all the acceptable policies with size equal to that difficulty uniformly and for each of these policies all the environments where each policy is acceptable with a universal distribution. This extra complication with respect to eq. 11 can only be justified if we generate environments and agents and we check them as we populate $Pairs$, as a way of constructing a test more easily.

5 Using computational steps

As we mentioned in the introduction, the C-test [9,2] used Levin's Kt instead of K. We explore the use of Kt here. However, when working with interactive tasks and with stochastic tasks and agents, the number of steps must be in expected value. We extend the definition of LS given in section 2 for a tolerance ϵ:

$$LS^{[\epsilon]}(\pi, \mu) \triangleq \mathbb{E}[LS(\pi, \mu)] \text{ if } \mathbb{A}^{[\epsilon]}(\pi, \mu) = 1 \text{ and } \infty \text{ otherwise}$$

and we define a new difficulty function that considers computational steps:

$$\hbar^{[\epsilon]}(\mu) \triangleq \min_{\pi} LS^{[\epsilon]}(\pi, \mu)$$

This difficulty function is not bounded, as LS depends on μ, and we can always find a very short policy that takes an enormous amount of steps for a task with very high difficulty. This is an acceptable policy, but does not reduce the difficulty of the task, so it can always score non-zero beyond any limit. This means that for this difficulty function we would need to use equation eq. 9 with an appropriate $w(h)$ (e.g., a small decay or a uniform interval of difficulties).

If the testing procedure established a limit on the number of steps (total or per transition) we would have this new difficulty function would be strongly bounded. Alternatively, we could reconsider the inclusion the computational steps in the notion of acceptability. In this case, the approach in section 4.2 could not be used, as the probability of π given h would also depend on μ.

6 Discussion

We have gone from eq. 1 taken from C-test to eq. 9. We have seen that difficulties allow for a more detailed analysis of what happens for a given agent, depending on whether it succeeds at easy or difficult tasks. For some difficulty functions, we do not even need to determine the weight for each difficulty and just calculate

the area, as an aggregated performance for all difficulties, and cutting the tail at some maximum difficulty for practical reasons.

The important thing is that now we do not need to set an a priori distribution for all tasks $p(\mu)$, but just a conditional distribution $p(\mu|h)$. Note that if we set a high h we have the freedom to find simple task that creates that difficulty. Actually, the choice of $p(\mu|h)$ as a universal distribution still depends on the reference machine and can set most of the probability mass on smaller tasks, but as it is conditional on h, all trivial, dead or simply meaningless tasks have usually very extreme values of h (very low or infinite). That means that there is a range of *intersting* difficulties, discarding very small values of h and very large values of h. Figure 2 is a nice example of this, where only difficulties between 1 and 8 were used, and we see also that $h = 1$ and $h > 7$ are not really very discriminating. The bulk of the testing effort must be performed in this range.

Note that the middle (B) and bottom (C) decompositions in Figure 3 can be done in such a way that the original $p_M(\mu)$ is preserved, if $w(h)$ is not taken uniform but slowly decaying. But we can just start with option B or C directly. This is the alternative in this paper, which we think has several advantages in terms of agent evaluation, the construction of tests and AGI development, as we can focus on those tasks of appropriate difficulty and even define adaptive tests easily. Having said this, we have an infinite set for $p_M(\mu|h)$ and $p_M(\mu|\pi')$, and a universal distribution is the appropriate for both, so that Occam's razor is still very present. This means that both B and C (using a slowly decaying $w(h)$) would lead to a computable aggregated distribution $p_M(\mu)$, which can be approximated as a universal distribution, highlighting that universal intelligence is rather a schema for definitions rather than a specific definition.

Acknowledgements: This work has been partially supported by the EU (FEDER) and the Spanish MINECO under grants TIN 2010-21062-C02-02, PCIN-2013-037 and TIN 2013-45732-C4-1-P, and by Generalitat Valenciana PROMETEOII2015/013.

References

1. Bellemare, M.G., Naddaf, Y., Veness, J., Bowling, M.: The arcade learning environment: An evaluation platform for general agents. Journal of Artificial Intelligence Research 47, 253–279 (06 2013)
2. Hernández-Orallo, J.: Beyond the Turing Test. J. Logic, Language & Information 9(4), 447–466 (2000)
3. Hernández-Orallo, J.: Computational measures of information gain and reinforcement in inference processes. AI Communications 13(1), 49–50 (2000)
4. Hernández-Orallo, J.: On the computational measurement of intelligence factors. In: Meystel, A. (ed.) Performance metrics for intelligent systems workshop. pp. 1–8. National Institute of Standards and Technology, Gaithersburg, MD (2000)
5. Hernández-Orallo, J.: AI evaluation: past, present and future. arXiv preprint arXiv:1408.6908 (2014)
6. Hernández-Orallo, J.: On environment difficulty and discriminating power. Autonomous Agents and Multi-Agent Systems pp. 1–53 (2014), http://dx.doi.org/10.1007/s10458-014-9257-1

7. Hernández-Orallo, J., Dowe, D.L.: Measuring universal intelligence: Towards an anytime intelligence test. Artificial Intelligence 174(18), 1508–1539 (2010)
8. Hernández-Orallo, J., Dowe, D.L., Hernández-Lloreda, M.V.: Universal psychometrics: Measuring cognitive abilities in the machine kingdom. Cognitive Systems Research 27, 5074 (2014)
9. Hernández-Orallo, J., Minaya-Collado, N.: A formal definition of intelligence based on an intensional variant of Kolmogorov complexity. In: Proc. Intl Symposium of Engineering of Intelligent Systems (EIS'98). pp. 146–163. ICSC Press (1998)
10. Hibbard, B.: Bias and no free lunch in formal measures of intelligence. Journal of Artificial General Intelligence 1(1), 54–61 (2009)
11. Legg, S., Hutter, M.: Universal intelligence: A definition of machine intelligence. Minds and Machines 17(4), 391–444 (2007)
12. Li, M., Vitányi, P.: An introduction to Kolmogorov complexity and its applications (3rd ed.). Springer-Verlag (2008)
13. Schaul, T.: An extensible description language for video games. Computational Intelligence and AI in Games, IEEE Transactions on PP(99), 1–1 (2014)
14. Solomonoff, R.J.: A formal theory of inductive inference. Part I. Information and control 7(1), 1–22 (1964)