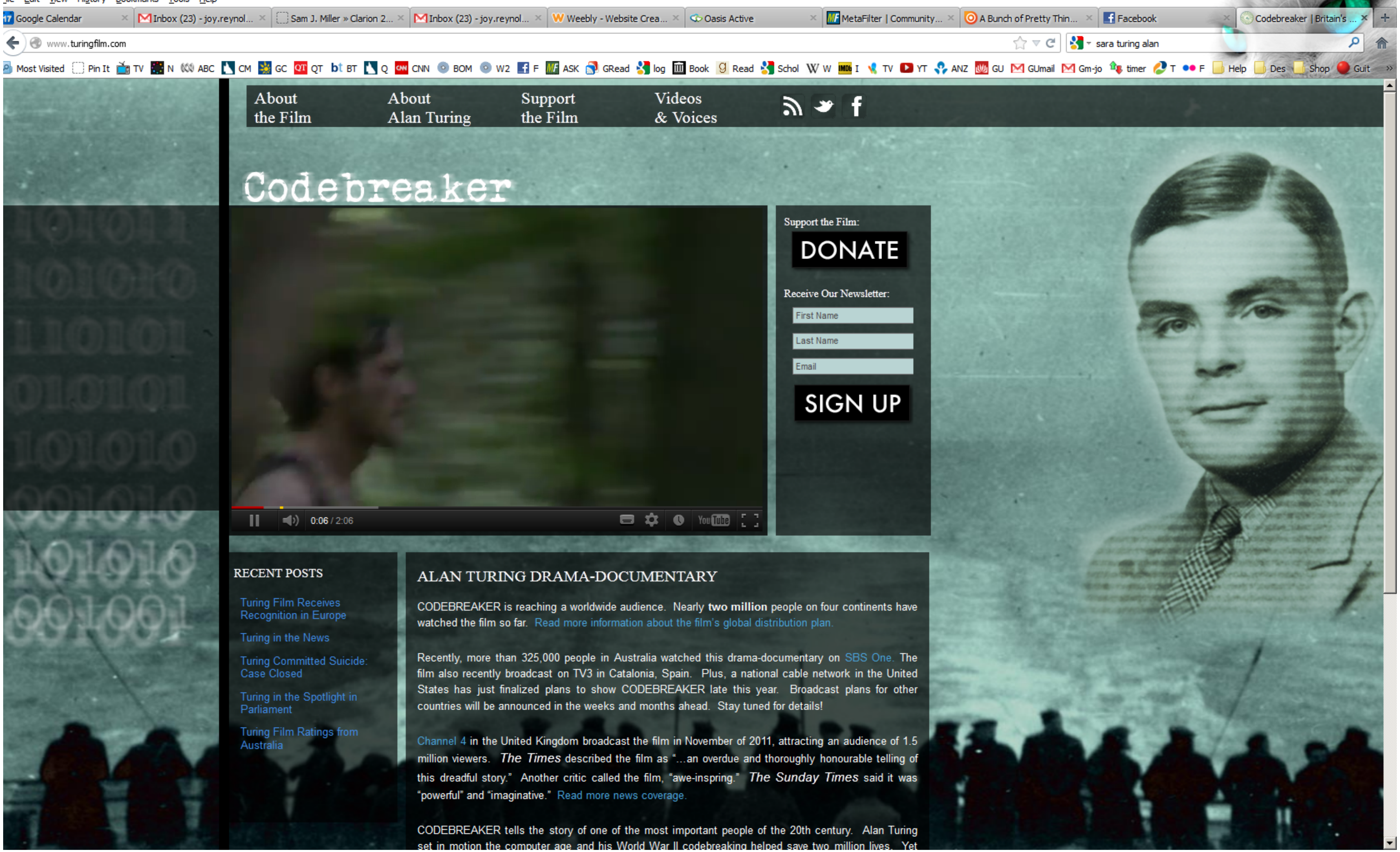
Turing computing machines,
Turing tests and intelligence

David L. Dowe
August 2012

Alan M. Turing
(23 June 1912 – 7 June 1954)

Said by some to be the greatest scientist of all time,
without whom the world might in so many ways be a different place.



RECENT POSTS

- Turing Film Receives Recognition in Europe
- Turing in the News
- Turing Committed Suicide: Case Closed
- Turing in the Spotlight in Parliament
- Turing Film Ratings from Australia

ALAN TURING DRAMA-DOCUMENTARY

CODEBREAKER is reaching a worldwide audience. Nearly **two million** people on four continents have watched the film so far. Read more information about the film's global distribution plan.

Recently, more than 325,000 people in Australia watched this drama-documentary on SBS One. The film also recently broadcast on TV3 in Catalonia, Spain. Plus, a national cable network in the United States has just finalized plans to show CODEBREAKER late this year. Broadcast plans for other countries will be announced in the weeks and months ahead. Stay tuned for details!

Channel 4 in the United Kingdom broadcast the film in November of 2011, attracting an audience of 1.5 million viewers. *The Times* described the film as "...an overdue and thoroughly honourable telling of this dreadful story." Another critic called the film, "awe-inspring." *The Sunday Times* said it was "powerful" and "imaginative." Read more news coverage.

CODEBREAKER tells the story of one of the most important people of the 20th century. Alan Turing set in motion the computer age and his World War II codebreaking helped save two million lives. Yet

www.TuringFilm.com

1936

A. M. Turing (1936), "On computable numbers with an application to the Entscheidungsproblem”

During WW II (1939-1945), led code-breakers at Bletchley Park to substantially influence outcome of war

1948

C. E. Shannon (1948), "A Mathematical Theory of Communication” birth of information theory, makes connection between probability and information

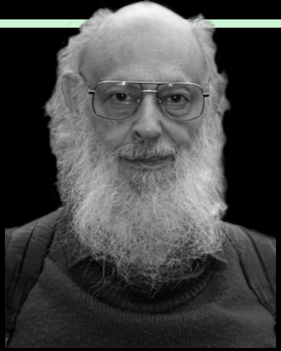
circa 1948 Turing writing chess algorithm

1950

A. M. Turing (1950), "Computing machinery and intelligence“ states the Imitation game, many now call this the Turing test

A. M. Turing (1952), "The chemical basis of morphogenesis”

1964



R. J. Solomonoff (1964a-b), “A formal theory of inductive inference, Part I”, “..., Part II” birth of algorithmic information theory and algorithmic probability, tells us how to use past data to probabilistically predict the future

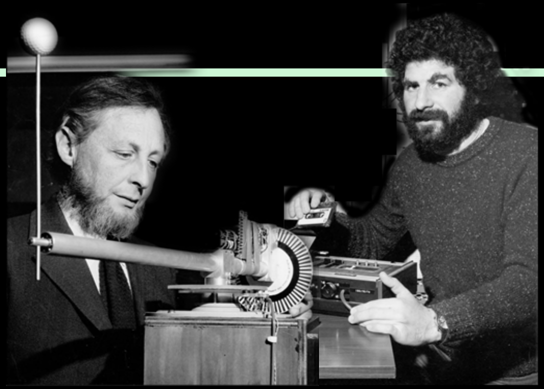
1965

A. N. Kolmogorov (1965), "Three approaches to the quantitative definition of information“ independent development of algorithmic information theory (also known as Kolmogorov complexity), but connection not made with probability

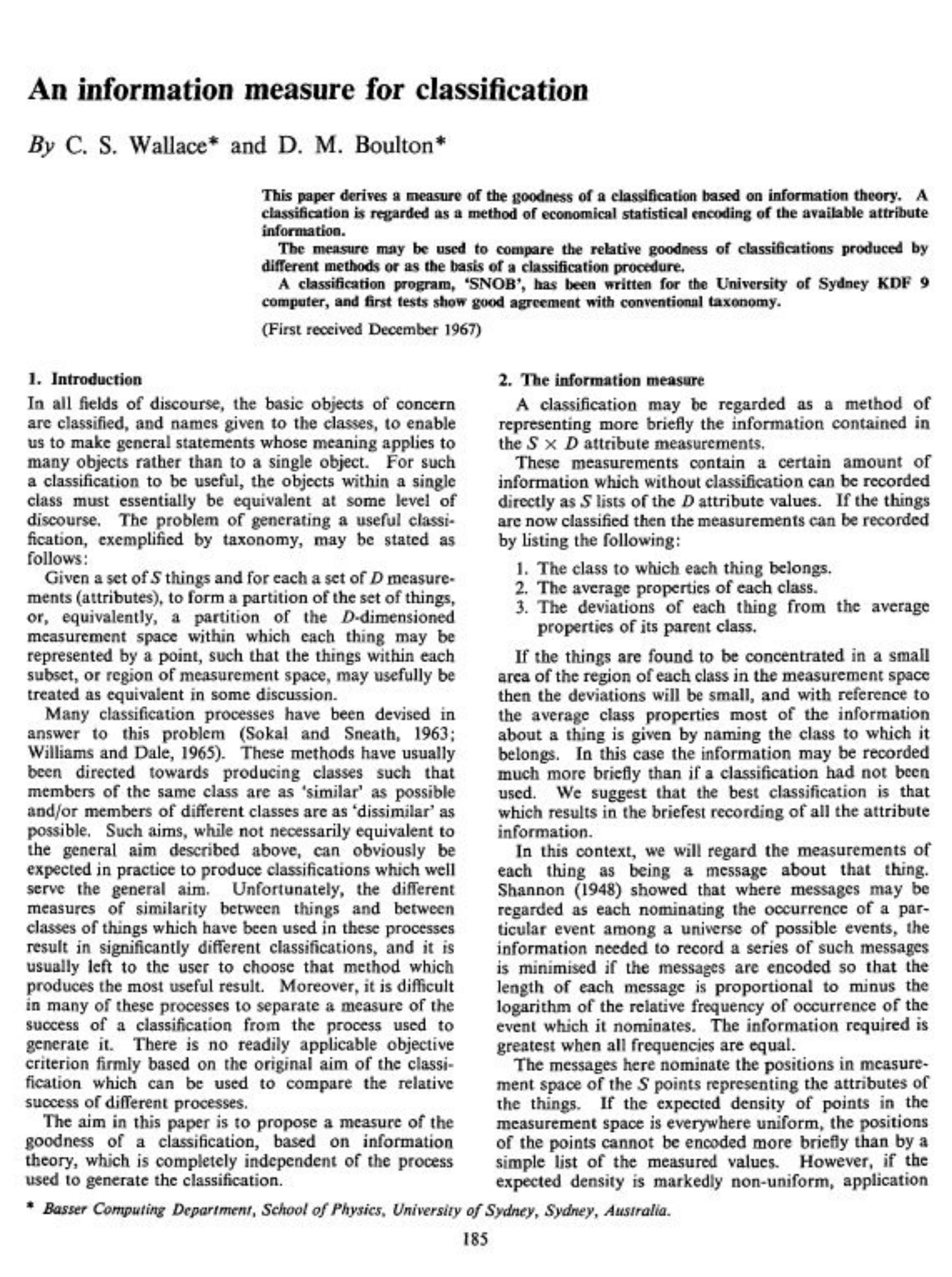
Later 1960s

G. J. Chaitin (1969, 1966) works on algorithmic information theory, connection also not made with statistics

1968



CS Wallace and DM Boulton (1968), "An information measure for classification” develops the Bayesian Minimum Message Length (MML) principle, shows how to use information theory and two-part compression to actually do statistical inference - initially with a clustering problem, and applies theory to a data-set of seal skull measurements [followed by Boulton and Wallace (1969, 1970, 1973a-b, 1975), Wallace and Boulton (1975) , etc.]



An information measure for classification

By C. S. Wallace* and D. M. Boulton*

This paper derives a measure of the goodness of a classification based on information theory. A classification is regarded as a method of economical statistical encoding of the available attribute information.

The measure may be used to compare the relative goodness of classifications produced by different methods or as the basis of a classification procedure.

A classification program, 'SNOB', has been written for the University of Sydney KDF 9 computer, and first tests show good agreement with conventional taxonomy.

(First received December 1967)

1. Introduction

In all fields of discourse, the basic objects of concern are classified, and names given to the classes, to enable us to make general statements whose meaning applies to many objects rather than to a single object. For such a classification to be useful, the objects within a single class must essentially be equivalent at some level of discourse. The problem of generating a useful classification, exemplified by taxonomy, may be stated as follows:

Given a set of S things and for each a set of D measurements (attributes), to form a partition of the set of things, or, equivalently, a partition of the D-dimensioned measurement space within which each thing may be represented by a point, such that the things within each subset, or region of measurement space, may usefully be treated as equivalent in some discussion.

Many classification processes have been devised in answer to this problem (Sokal and Sneath, 1963; Williams and Dale, 1965). These methods have usually been directed towards producing classes such that members of the same class are as 'similar' as possible and/or members of different classes are as 'dissimilar' as possible. Such aims, while not necessarily equivalent to the general aim described above, can obviously be expected in practice to produce classifications which well serve the general aim. Unfortunately, the different measures of similarity between things and between classes of things which have been used in these processes result in significantly different classifications, and it is usually left to the user to choose that method which produces the most useful result. Moreover, it is difficult in many of these processes to separate a measure of the success of a classification from the process used to generate it. There is no readily applicable objective criterion firmly based on the original aim of the classification which can be used to compare the relative success of different processes.

The aim in this paper is to propose a measure of the goodness of a classification, based on information theory, which is completely independent of the process used to generate the classification.

2. The information measure

A classification may be regarded as a method of representing more briefly the information contained in the $S \times D$ attribute measurements.

These measurements contain a certain amount of information which without classification can be recorded directly as S lists of the D attribute values. If the things are now classified then the measurements can be recorded by listing the following:

1. The class to which each thing belongs.
2. The average properties of each class.
3. The deviations of each thing from the average properties of its parent class.

If the things are found to be concentrated in a small area of the region of each class in the measurement space then the deviations will be small, and with reference to the average class properties most of the information about a thing is given by naming the class to which it belongs. In this case the information may be recorded much more briefly than if a classification had not been used. We suggest that the best classification is that which results in the briefest recording of all the attribute information.

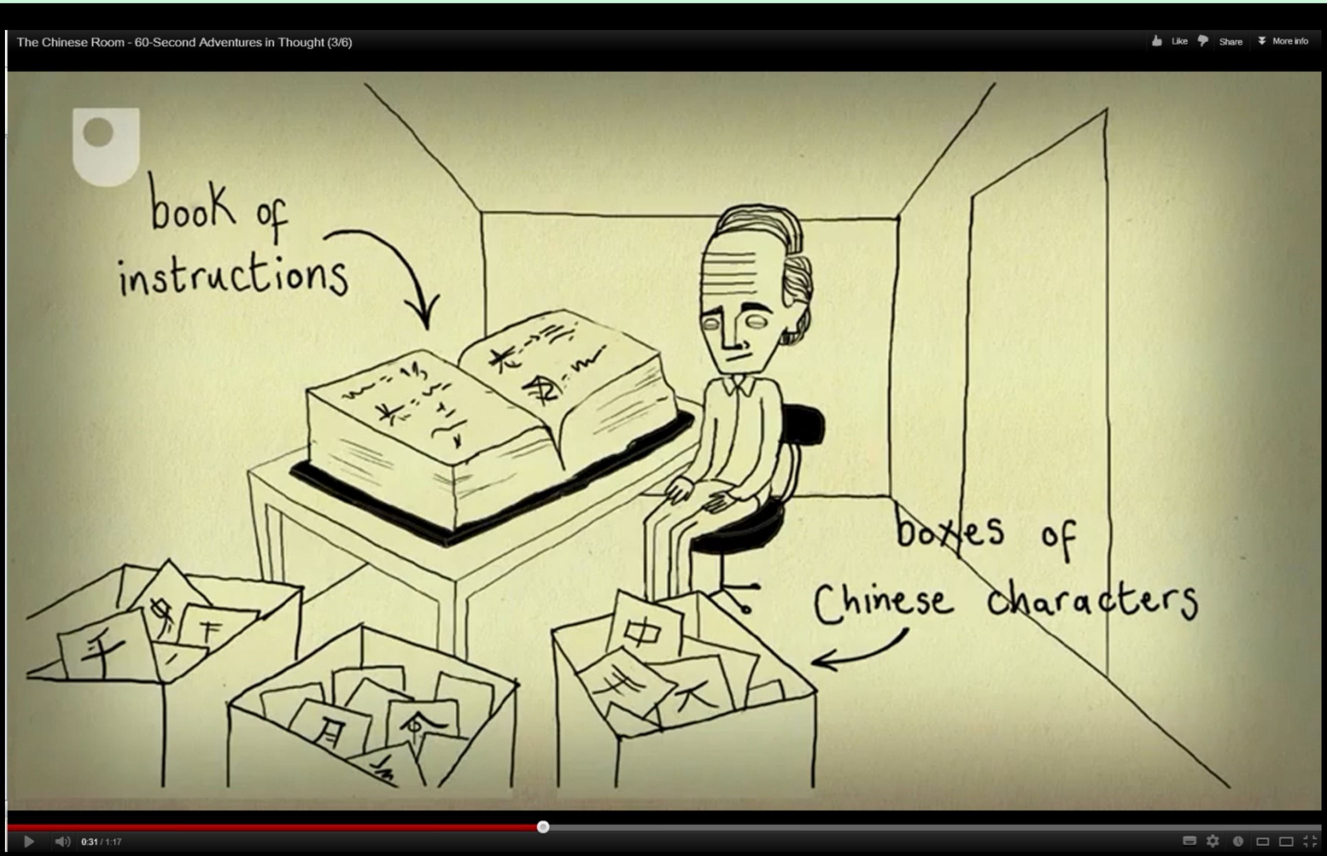
In this context, we will regard the measurements of each thing as being a message about that thing. Shannon (1948) showed that where messages may be regarded as each nominating the occurrence of a particular event among a universe of possible events, the information needed to record a series of such messages is minimised if the messages are encoded so that the length of each message is proportional to minus the logarithm of the relative frequency of occurrence of the event which it nominates. The information required is greatest when all frequencies are equal.

The messages here nominate the positions in measurement space of the S points representing the attributes of the things. If the expected density of points in the measurement space is everywhere uniform, the positions of the points cannot be encoded more briefly than by a simple list of the measured values. However, if the expected density is markedly non-uniform, application

* *Basser Computing Department, School of Physics, University of Sydney, Sydney, Australia.*

Image from *The Chinese Room*
60 seconds adventures in thought
The Open University *YouTube*

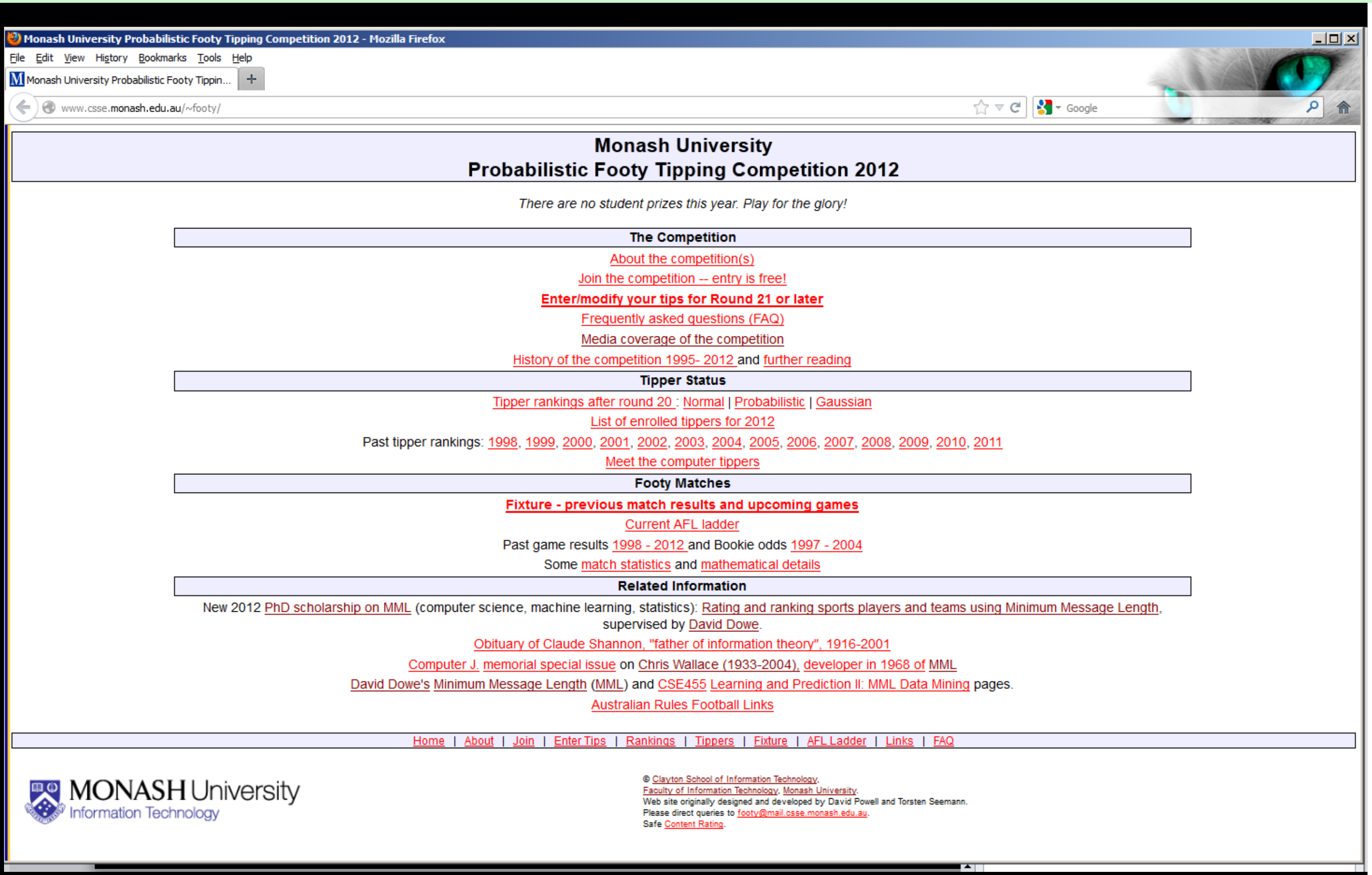
1980


The Chinese Room - 60-Second Adventures in Thought (3/6)
Like
Share
More info
book of instructions
boxes of Chinese characters
中
大
月
0:31 / 1:17

Searle's Chinese room thought experiment www.csse.monash.edu.au/~footy

1995

Start of world's longest running compression-based competition - applied to Australian AFL football


Monash University Probabilistic Footy Tipping Competition 2012 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Monash University Probabilistic Footy Tippin...
www.csse.monash.edu.au/~footy/
Google

Monash University
Probabilistic Footy Tipping Competition 2012

There are no student prizes this year. Play for the glory!

The Competition

About the competition(s)

Join the competition -- entry is free!

Enter/modify your tips for Round 21 or later

Frequently asked questions (FAQ)

Media coverage of the competition

History of the competition 1995- 2012 and further reading

Tipper Status

Tipper rankings after round 20 : Normal | Probabilistic | Gaussian

List of enrolled tippers for 2012

Past tipper rankings: 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011

Meet the computer tippers

Footy Matches

Fixture - previous match results and upcoming games

Current AFL ladder

Past game results 1998 - 2012 and Bookie odds 1997 - 2004

Some match statistics and mathematical details

Related Information

New 2012 PhD scholarship on MML (computer science, machine learning, statistics): Rating and ranking sports players and teams using Minimum Message Length, supervised by David Dowe.

Obituary of Claude Shannon, "father of information theory", 1916-2001

Computer J. memorial special issue on Chris Wallace (1933-2004), developer in 1968 of MML

David Dowe's Minimum Message Length (MML) and CSE455 Learning and Prediction II: MML Data Mining pages.

Australian Rules Football Links

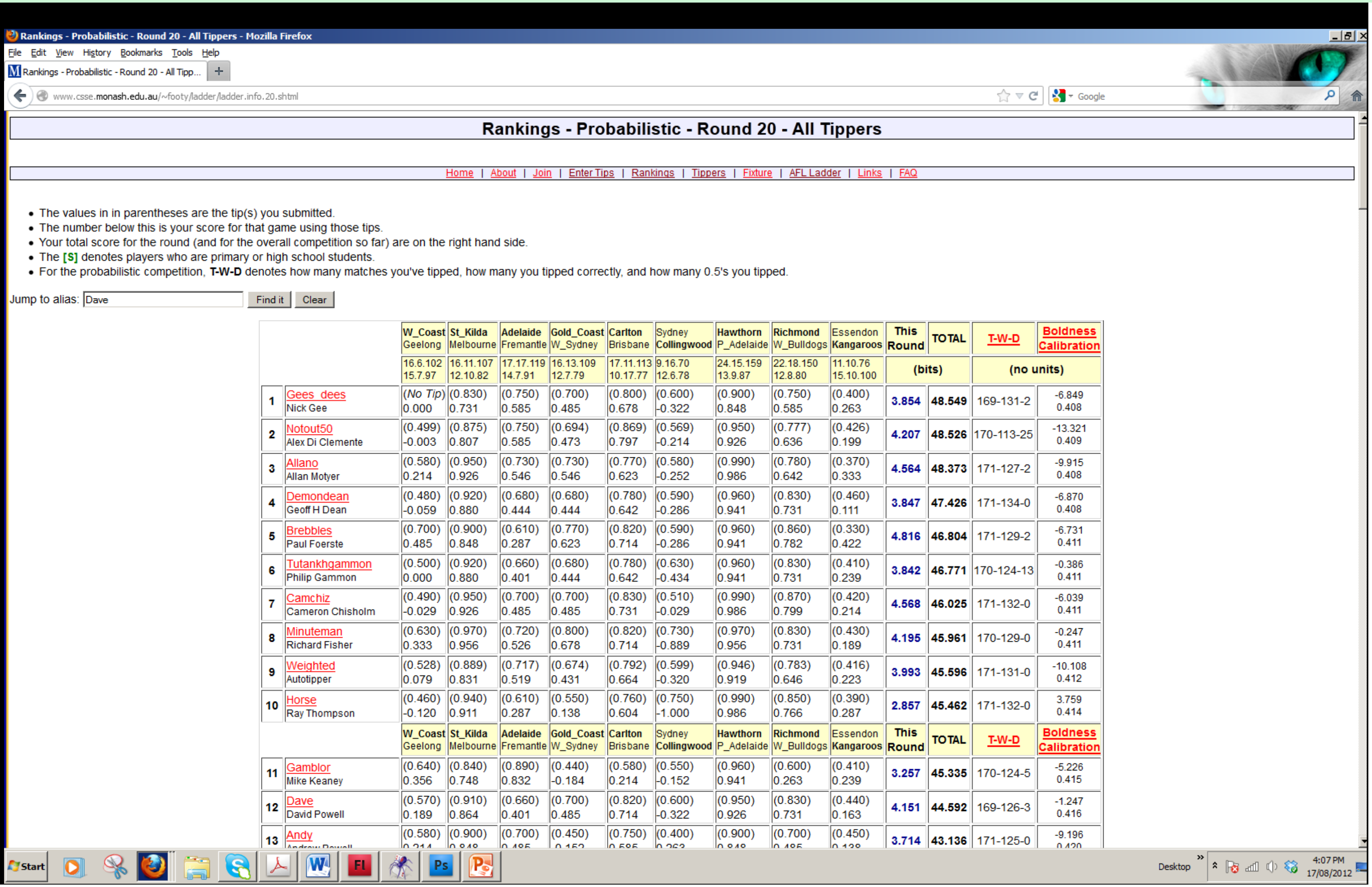
Home | About | Join | Enter Tips | Rankings | Tippers | Fixture | AFL Ladder | Links | FAQ

MONASH University
Information Technology

© Clayton School of Information Technology,
Faculty of Information Technology, Monash University.
Web site originally designed and developed by David Powell and Torsten Seemann.
Please direct queries to footy@mail.csse.monash.edu.au.
Safe Content Rating.

www.csse.monash.edu.au/~footy/ladder/ladder.info.20.shtml (from 2012 season)

1995


Rankings - Probabilistic - Round 20 - All Tippers - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Rankings - Probabilistic - Round 20 - All Tipp...
www.csse.monash.edu.au/~footy/ladder/ladder.info.20.shtml
Google
Start
Desktop
4:07 PM
17/08/2012

Rankings - Probabilistic - Round 20 - All Tippers

Home | About | Join | Enter Tips | Rankings | Tippers | Fixture | AFL Ladder | Links | FAQ

- The values in in parentheses are the tip(s) you submitted.
- The number below this is your score for that game using those tips.
- Your total score for the round (and for the overall competition so far) are on the right hand side.
- The **[S]** denotes players who are primary or high school students.
- For the probabilistic competition, **T-W-D** denotes how many matches you've tipped, how many you tipped correctly, and how many 0.5's you tipped.

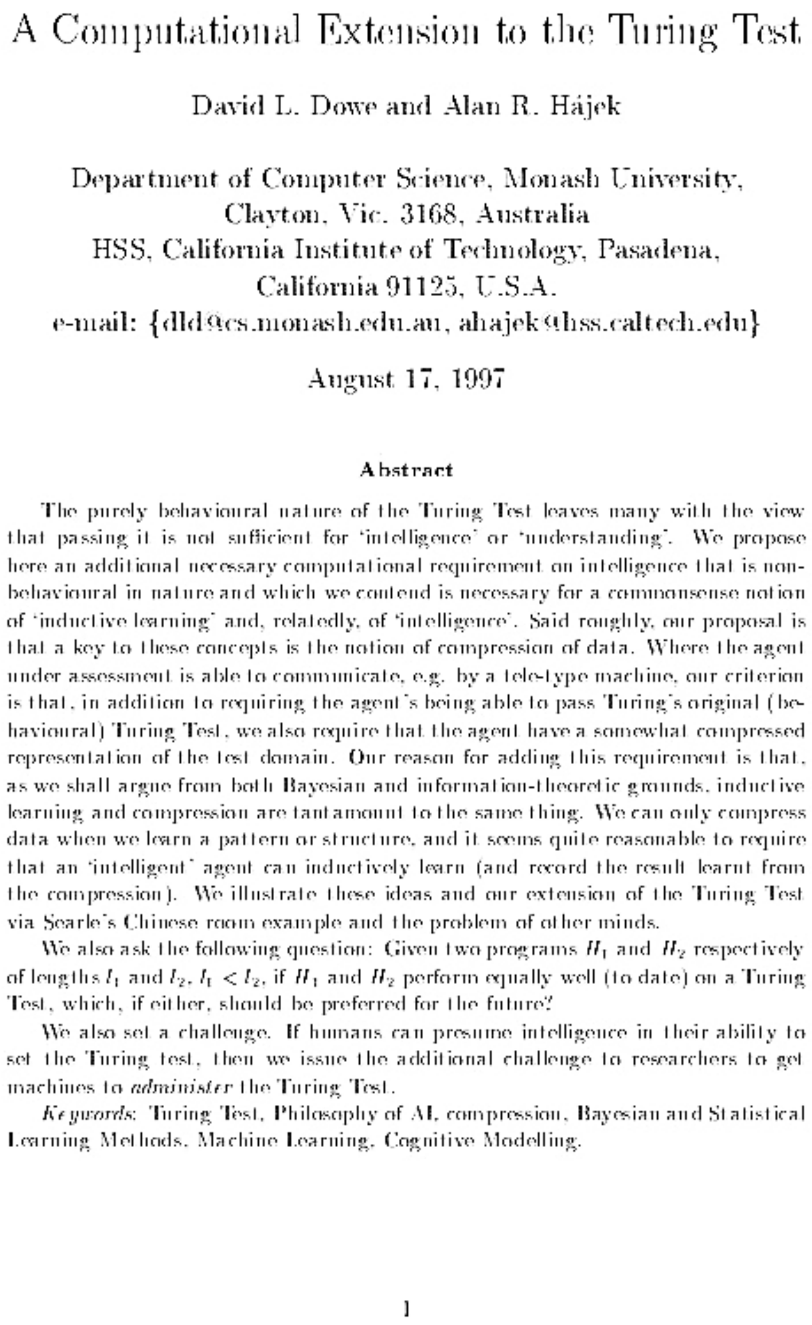
Jump to alias: Dave Find it Clear

		W_Coast Geelong	**St_Kilda** Melbourne	**Adelaide** Fremantle	**Gold_Coast** W_Sydney	**Carlton** Brisbane	Sydney **Collingwood**	**Hawthorn** P_Adelaide	**Richmond** W_Bulldogs	Essendon **Kangaroos**	**This Round**	**TOTAL**	**T-W-D**	**Boldness Calibration**
		16.6.102 15.7.97	16.11.107 12.10.82	17.17.119 14.7.91	16.13.109 12.7.79	17.11.113 10.17.77	9.16.70 12.6.78	24.15.159 13.9.87	22.18.150 12.8.80	11.10.76 15.10.100	**(bits)**		**(no units)**	
1	Gees_dees Nick Gee	*(No Tip)* 0.000	(0.830) 0.731	(0.750) 0.585	(0.700) 0.485	(0.800) 0.678	(0.600) -0.322	(0.900) 0.848	(0.750) 0.585	(0.400) 0.263	**3.854**	**48.549**	169-131-2	-6.849 0.408
2	Notout50 Alex Di Clemente	(0.499) -0.003	(0.875) 0.807	(0.750) 0.585	(0.694) 0.473	(0.869) 0.797	(0.569) -0.214	(0.950) 0.926	(0.777) 0.636	(0.426) 0.199	**4.207**	**48.526**	170-113-25	-13.321 0.409
3	Allano Allan Motyer	(0.580) 0.214	(0.950) 0.926	(0.730) 0.546	(0.730) 0.546	(0.770) 0.623	(0.580) -0.252	(0.990) 0.986	(0.780) 0.642	(0.370) 0.333	**4.564**	**48.373**	171-127-2	-9.915 0.408
4	Demondean Geoff H Dean	(0.480) -0.059	(0.920) 0.880	(0.680) 0.444	(0.680) 0.444	(0.780) 0.642	(0.590) -0.286	(0.960) 0.941	(0.830) 0.731	(0.460) 0.111	**3.847**	**47.426**	171-134-0	-6.870 0.408
5	Brebbles Paul Foerste	(0.700) 0.485	(0.900) 0.848	(0.610) 0.287	(0.770) 0.623	(0.820) 0.714	(0.590) -0.286	(0.960) 0.941	(0.860) 0.782	(0.330) 0.422	**4.816**	**46.804**	171-129-2	-6.731 0.411
6	Tutankhgammon Philip Gammon	(0.500) 0.000	(0.920) 0.880	(0.660) 0.401	(0.680) 0.444	(0.780) 0.642	(0.630) -0.434	(0.960) 0.941	(0.830) 0.731	(0.410) 0.239	**3.842**	**46.771**	170-124-13	-0.386 0.411
7	Camchiz Cameron Chisholm	(0.490) -0.029	(0.950) 0.926	(0.700) 0.485	(0.700) 0.485	(0.830) 0.731	(0.510) -0.029	(0.990) 0.986	(0.870) 0.799	(0.420) 0.214	**4.568**	**46.025**	171-132-0	-6.039 0.411
8	Minuteman Richard Fisher	(0.630) 0.333	(0.970) 0.956	(0.720) 0.526	(0.800) 0.678	(0.820) 0.714	(0.730) -0.889	(0.970) 0.956	(0.830) 0.731	(0.430) 0.189	**4.195**	**45.961**	170-129-0	-0.247 0.411
9	Weighted Autotipper	(0.528) 0.079	(0.889) 0.831	(0.717) 0.519	(0.674) 0.431	(0.792) 0.664	(0.599) -0.320	(0.946) 0.919	(0.783) 0.646	(0.416) 0.223	**3.993**	**45.596**	171-131-0	-10.108 0.412
10	Horse Ray Thompson	(0.460) -0.120	(0.940) 0.911	(0.610) 0.287	(0.550) 0.138	(0.760) 0.604	(0.750) -1.000	(0.990) 0.986	(0.850) 0.766	(0.390) 0.287	**2.857**	**45.462**	171-132-0	3.759 0.414
		W_Coast Geelong	**St_Kilda** Melbourne	**Adelaide** Fremantle	**Gold_Coast** W_Sydney	**Carlton** Brisbane	Sydney **Collingwood**	**Hawthorn** P_Adelaide	**Richmond** W_Bulldogs	Essendon **Kangaroos**	**This Round**	**TOTAL**	**T-W-D**	**Boldness Calibration**
11	Gamblor Mike Keaney	(0.640) 0.356	(0.840) 0.748	(0.890) 0.832	(0.440) -0.184	(0.580) 0.214	(0.550) -0.152	(0.960) 0.941	(0.600) 0.263	(0.410) 0.239	**3.257**	**45.335**	170-124-5	-5.226 0.415
12	Dave David Powell	(0.570) 0.189	(0.910) 0.864	(0.660) 0.401	(0.700) 0.485	(0.820) 0.714	(0.600) -0.322	(0.950) 0.926	(0.830) 0.731	(0.440) 0.163	**4.151**	**44.592**	169-126-3	-1.247 0.416
13	Andy Andrew Powell	(0.580) 0.214	(0.900) 0.848	(0.700) 0.485	(0.450) -0.152	(0.750) 0.585	(0.400) 0.263	(0.900) 0.848	(0.700) 0.485	(0.450) 0.138	**3.714**	**43.136**	171-125-0	-9.196 0.420

1997-8

Dowe and Hajek (1997a-b, 1998) : relevance of two-part compression and MML inductive inference to learning and intelligence

1997 Then World Chess Champion Garry Kasparov loses 3 ½ : 2 ½ to IBM Deep Blue.



A Computational Extension to the Turing Test

David L. Dowe and Alan R. Hájek

Department of Computer Science, Monash University,
Clayton, Vic. 3168, Australia
HSS, California Institute of Technology, Pasadena,
California 91125, U.S.A.
e-mail: {dld@cs.monash.edu.au, ahajek@hss.caltech.edu}

August 17, 1997

Abstract

The purely behavioural nature of the Turing Test leaves many with the view that passing it is not sufficient for 'intelligence' or 'understanding'. We propose here an additional necessary computational requirement on intelligence that is non-behavioural in nature and which we contend is necessary for a commonsense notion of 'inductive learning' and, relatedly, of 'intelligence'. Said roughly, our proposal is that a key to these concepts is the notion of compression of data. Where the agent under assessment is able to communicate, e.g. by a tele-type machine, our criterion is that, in addition to requiring the agent's being able to pass Turing's original (behavioural) Turing Test, we also require that the agent have a somewhat compressed representation of the test domain. Our reason for adding this requirement is that, as we shall argue from both Bayesian and information-theoretic grounds, inductive learning and compression are tantamount to the same thing. We can only compress data when we learn a pattern or structure, and it seems quite reasonable to require that an 'intelligent' agent can inductively learn (and record the result learnt from the compression). We illustrate these ideas and our extension of the Turing Test via Searle's Chinese room example and the problem of other minds.

We also ask the following question: Given two programs H_1 and H_2 respectively of lengths l_1 and l_2, $l_1 < l_2$, if H_1 and H_2 perform equally well (to date) on a Turing Test, which, if either, should be preferred for the future?

We also set a challenge. If humans can presume intelligence in their ability to set the Turing test, then we issue the additional challenge to researchers to get machines to *administer* the Turing Test.

Keywords: Turing Test, Philosophy of AI, compression, Bayesian and Statistical Learning Methods, Machine Learning, Cognitive Modelling.

1998-2000

Hernandez-Orallo and Minaya-Collado (1998), Hernandez-Orallo (2000): relate compression to intelligence and construct a compression-based test: the C-test

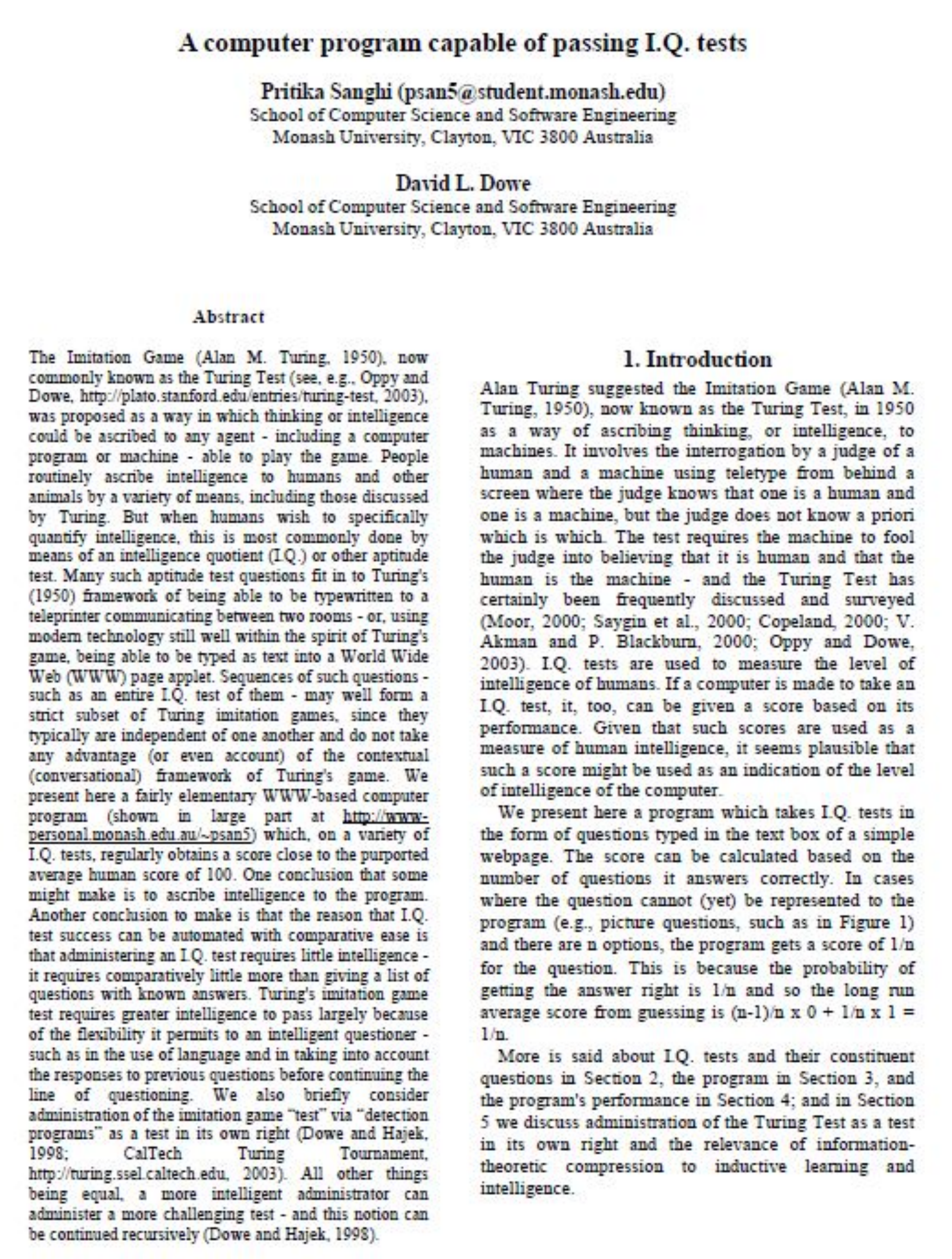
1999

M. Mahoney (1999) suggests text compression as a measure of intelligence

1999

C S Wallace and D L Dowe (1999) : "Minimum Message Length and Kolmogorov complexity" formally relates MML statistical inference to algorithmic information theory

2003



A computer program capable of passing I.Q. tests

Pritika Sanghi (psan5@student.monash.edu)
School of Computer Science and Software Engineering
Monash University, Clayton, VIC 3800 Australia

David L. Dowe
School of Computer Science and Software Engineering
Monash University, Clayton, VIC 3800 Australia

Abstract

The Imitation Game (Alan M. Turing, 1950), now commonly known as the Turing Test (see, e.g., Oppy and Dowe, http://plato.stanford.edu/entries/turing-test, 2003), was proposed as a way in which thinking or intelligence could be ascribed to any agent - including a computer program or machine - able to play the game. People routinely ascribe intelligence to humans and other animals by a variety of means, including those discussed by Turing. But when humans wish to specifically quantify intelligence, this is most commonly done by means of an intelligence quotient (I.Q.) or other aptitude test. Many such aptitude test questions fit in to Turing's (1950) framework of being able to be typewritten to a teleprinter communicating between two rooms - or, using modern technology still well within the spirit of Turing's game, being able to be typed as text into a World Wide Web (WWW) page applet. Sequences of such questions - such as an entire I.Q. test of them - may well form a strict subset of Turing imitation games, since they typically are independent of one another and do not take any advantage (or even account) of the contextual (conversational) framework of Turing's game. We present here a fairly elementary WWW-based computer program (shown in large part at http://www-personal.monash.edu.au/~psan5) which, on a variety of I.Q. tests, regularly obtains a score close to the purported average human score of 100. One conclusion that some might make is to ascribe intelligence to the program. Another conclusion to make is that the reason that I.Q. test success can be automated with comparative ease is that administering an I.Q. test requires little intelligence - it requires comparatively little more than giving a list of questions with known answers. Turing's imitation game test requires greater intelligence to pass largely because of the flexibility it permits to an intelligent questioner - such as in the use of language and in taking into account the responses to previous questions before continuing the line of questioning. We also briefly consider administration of the imitation game "test" via "detection programs" as a test in its own right (Dowe and Hajek, 1998; CalTech Turing Tournament, http://turing.ssel.caltech.edu, 2003). All other things being equal, a more intelligent administrator can administer a more challenging test - and this notion can be continued recursively (Dowe and Hajek, 1998).

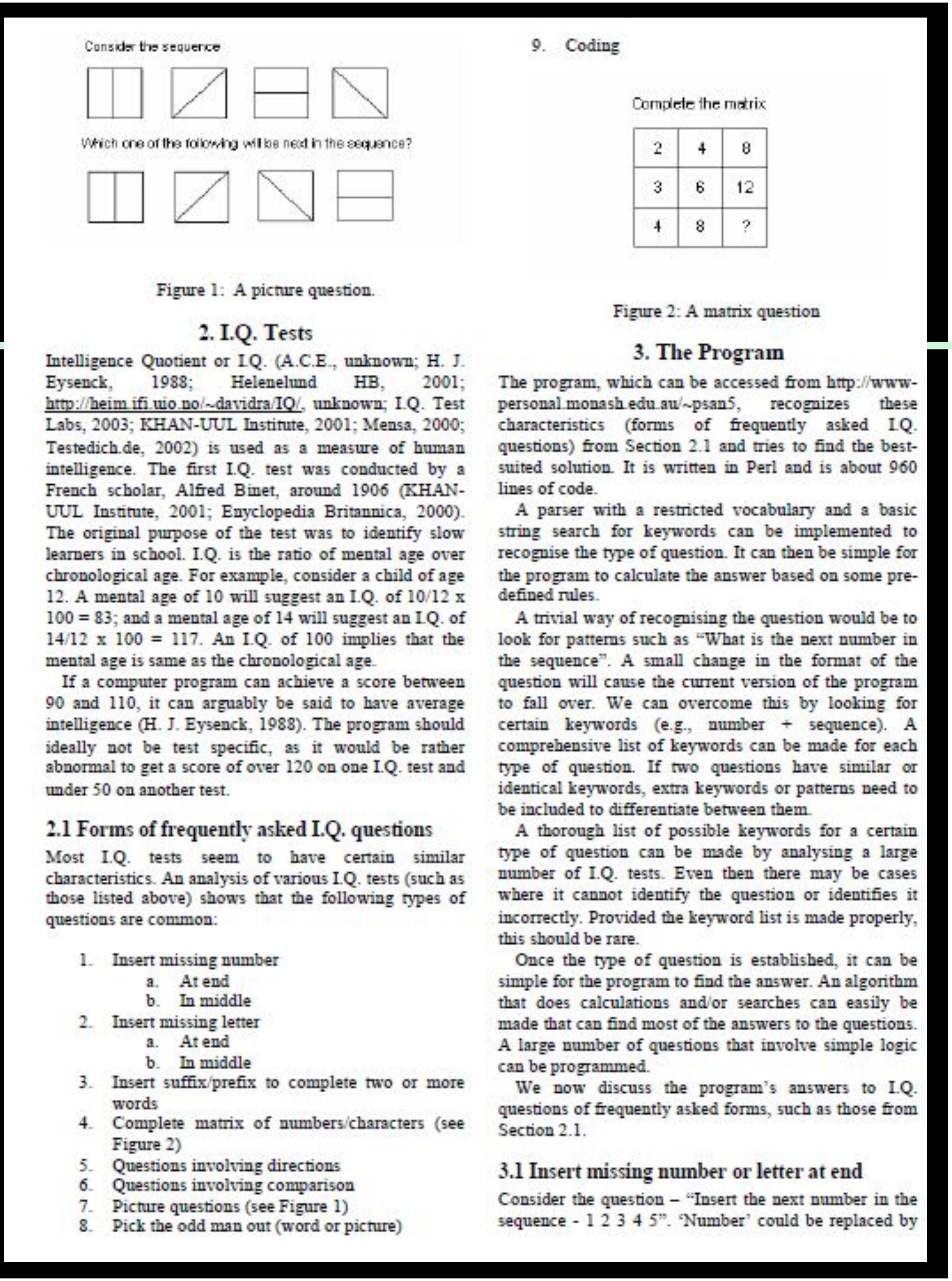
1. Introduction

Alan Turing suggested the Imitation Game (Alan M. Turing, 1950), now known as the Turing Test, in 1950 as a way of ascribing thinking, or intelligence, to machines. It involves the interrogation by a judge of a human and a machine using teletype from behind a screen where the judge knows that one is a human and one is a machine, but the judge does not know a priori which is which. The test requires the machine to fool the judge into believing that it is human and that the human is the machine - and the Turing Test has certainly been frequently discussed and surveyed (Moor, 2000; Saygin et al., 2000; Copeland, 2000; V. Akman and P. Blackburn, 2000; Oppy and Dowe, 2003). I.Q. tests are used to measure the level of intelligence of humans. If a computer is made to take an I.Q. test, it, too, can be given a score based on its performance. Given that such scores are used as a measure of human intelligence, it seems plausible that such a score might be used as an indication of the level of intelligence of the computer.

We present here a program which takes I.Q. tests in the form of questions typed in the text box of a simple webpage. The score can be calculated based on the number of questions it answers correctly. In cases where the question cannot (yet) be represented to the program (e.g., picture questions, such as in Figure 1) and there are n options, the program gets a score of 1/n for the question. This is because the probability of getting the answer right is 1/n and so the long run average score from guessing is (n-1)/n x 0 + 1/n x 1 = 1/n.

More is said about I.Q. tests and their constituent questions in Section 2, the program in Section 3, and the program's performance in Section 4; and in Section 5 we discuss administration of the Turing Test as a test in its own right and the relevance of information-theoretic compression to inductive learning and intelligence.

2003



Consider the sequence

Which one of the following will be next in the sequence?

Figure 1: A picture question.

2. I.Q. Tests

Intelligence Quotient or I.Q. (A.C.E., unknown; H. J. Eysenck, 1988; Helenelund HB, 2001; http://heim.ifi.uio.no/~davidra/IQ/, unknown; I.Q. Test Labs, 2003; KHAN-UUL Institute, 2001; Mensa, 2000; Testedich.de, 2002) is used as a measure of human intelligence. The first I.Q. test was conducted by a French scholar, Alfred Binet, around 1906 (KHAN-UUL Institute, 2001; Enyclopedia Britannica, 2000). The original purpose of the test was to identify slow learners in school. I.Q. is the ratio of mental age over chronological age. For example, consider a child of age 12. A mental age of 10 will suggest an I.Q. of 10/12 x 100 = 83; and a mental age of 14 will suggest an I.Q. of 14/12 x 100 = 117. An I.Q. of 100 implies that the mental age is same as the chronological age.

If a computer program can achieve a score between 90 and 110, it can arguably be said to have average intelligence (H. J. Eysenck, 1988). The program should ideally not be test specific, as it would be rather abnormal to get a score of over 120 on one I.Q. test and under 50 on another test.

2.1 Forms of frequently asked I.Q. questions

Most I.Q. tests seem to have certain similar characteristics. An analysis of various I.Q. tests (such as those listed above) shows that the following types of questions are common:

1. Insert missing number
   a. At end
   b. In middle
2. Insert missing letter
   a. At end
   b. In middle
3. Insert suffix/prefix to complete two or more words
4. Complete matrix of numbers/characters (see Figure 2)
5. Questions involving directions
6. Questions involving comparison
7. Picture questions (see Figure 1)
8. Pick the odd man out (word or picture)
9. Coding

Complete the matrix

2	4	8
3	6	12
4	8	?

Figure 2: A matrix question

3. The Program

The program, which can be accessed from http://www-personal.monash.edu.au/~psan5, recognizes these characteristics (forms of frequently asked I.Q. questions) from Section 2.1 and tries to find the best-suited solution. It is written in Perl and is about 960 lines of code.

A parser with a restricted vocabulary and a basic string search for keywords can be implemented to recognise the type of question. It can then be simple for the program to calculate the answer based on some pre-defined rules.

A trivial way of recognising the question would be to look for patterns such as "What is the next number in the sequence". A small change in the format of the question will cause the current version of the program to fall over. We can overcome this by looking for certain keywords (e.g., number + sequence). A comprehensive list of keywords can be made for each type of question. If two questions have similar or identical keywords, extra keywords or patterns need to be included to differentiate between them.

A thorough list of possible keywords for a certain type of question can be made by analysing a large number of I.Q. tests. Even then there may be cases where it cannot identify the question or identifies it incorrectly. Provided the keyword list is made properly, this should be rare.

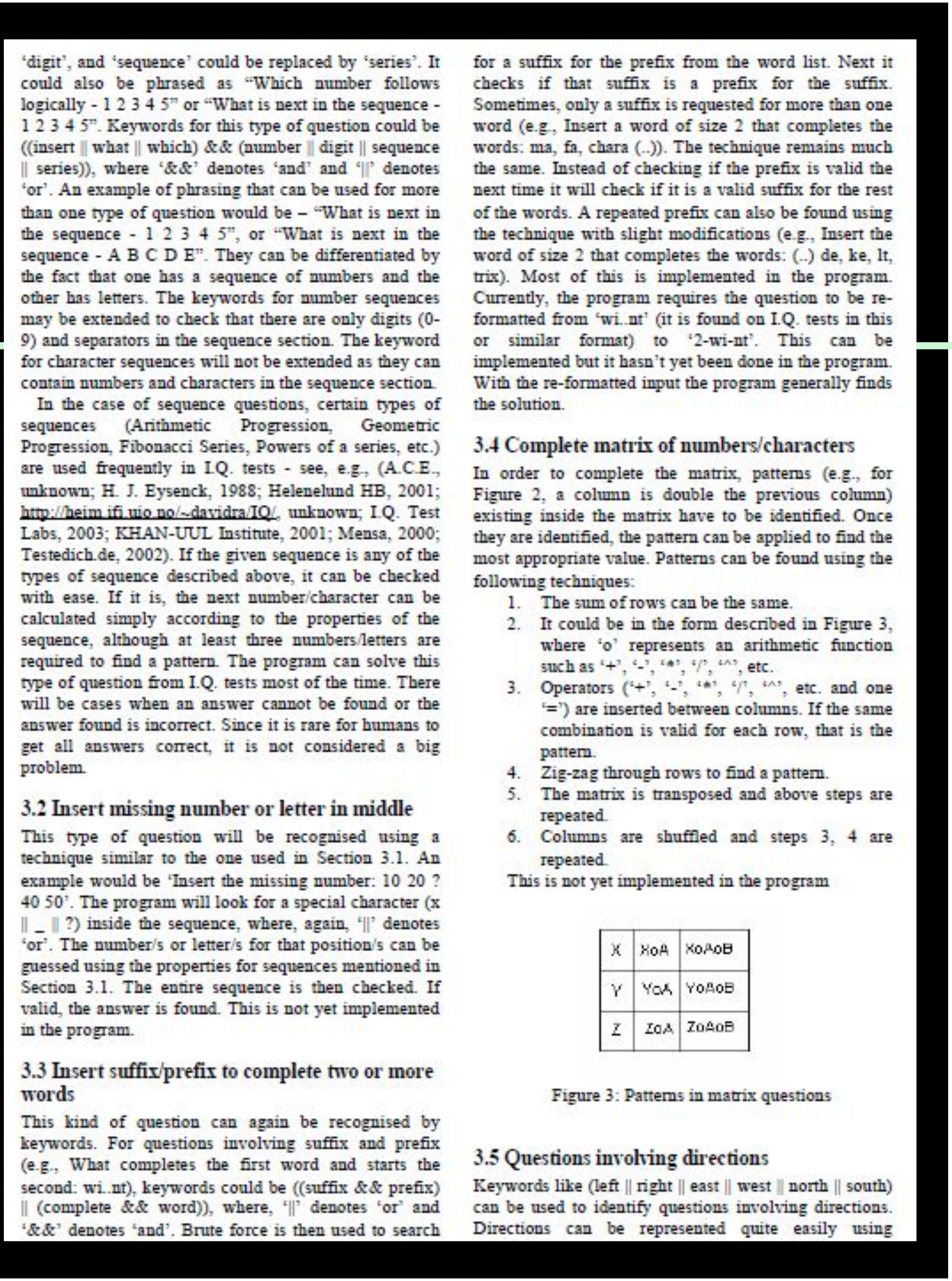
Once the type of question is established, it can be simple for the program to find the answer. An algorithm that does calculations and/or searches can easily be made that can find most of the answers to the questions. A large number of questions that involve simple logic can be programmed.

We now discuss the program's answers to I.Q. questions of frequently asked forms, such as those from Section 2.1.

3.1 Insert missing number or letter at end

Consider the question – "Insert the next number in the sequence - 1 2 3 4 5". 'Number' could be replaced by

2003



'digit', and 'sequence' could be replaced by 'series'. It could also be phrased as "Which number follows logically - 1 2 3 4 5" or "What is next in the sequence - 1 2 3 4 5". Keywords for this type of question could be ((insert || what || which) && (number || digit || sequence || series)), where '&&' denotes 'and' and '||' denotes 'or'. An example of phrasing that can be used for more than one type of question would be – "What is next in the sequence - 1 2 3 4 5", or "What is next in the sequence - A B C D E". They can be differentiated by the fact that one has a sequence of numbers and the other has letters. The keywords for number sequences may be extended to check that there are only digits (0-9) and separators in the sequence section. The keyword for character sequences will not be extended as they can contain numbers and characters in the sequence section.

In the case of sequence questions, certain types of sequences (Arithmetic Progression, Geometric Progression, Fibonacci Series, Powers of a series, etc.) are used frequently in I.Q. tests - see, e.g., (A.C.E., unknown; H. J. Eysenck, 1988; Helenelund HB, 2001; http://heim.ifi.uio.no/~davidra/IQ/, unknown; I.Q. Test Labs, 2003; KHAN-UUL Institute, 2001; Mensa, 2000; Testedich.de, 2002). If the given sequence is any of the types of sequence described above, it can be checked with ease. If it is, the next number/character can be calculated simply according to the properties of the sequence, although at least three numbers/letters are required to find a pattern. The program can solve this type of question from I.Q. tests most of the time. There will be cases when an answer cannot be found or the answer found is incorrect. Since it is rare for humans to get all answers correct, it is not considered a big problem.

3.2 Insert missing number or letter in middle

This type of question will be recognised using a technique similar to the one used in Section 3.1. An example would be 'Insert the missing number: 10 20 ? 40 50'. The program will look for a special character (x || _ || ?) inside the sequence, where, again, '||' denotes 'or'. The number/s or letter/s for that position/s can be guessed using the properties for sequences mentioned in Section 3.1. The entire sequence is then checked. If valid, the answer is found. This is not yet implemented in the program.

3.3 Insert suffix/prefix to complete two or more words

This kind of question can again be recognised by keywords. For questions involving suffix and prefix (e.g., What completes the first word and starts the second: wi..nt), keywords could be ((suffix && prefix) || (complete && word)), where, '||' denotes 'or' and '&&' denotes 'and'. Brute force is then used to search for a suffix for the prefix from the word list. Next it checks if that suffix is a prefix for the suffix. Sometimes, only a suffix is requested for more than one word (e.g., Insert a word of size 2 that completes the words: ma, fa, chara (..)). The technique remains much the same. Instead of checking if the prefix is valid the next time it will check if it is a valid suffix for the rest of the words. A repeated prefix can also be found using the technique with slight modifications (e.g., Insert the word of size 2 that completes the words: (..) de, ke, lt, trix). Most of this is implemented in the program. Currently, the program requires the question to be re-formatted from 'wi..nt' (it is found on I.Q. tests in this or similar format) to '2-wi-nt'. This can be implemented but it hasn't yet been done in the program. With the re-formatted input the program generally finds the solution.

3.4 Complete matrix of numbers/characters

In order to complete the matrix, patterns (e.g., for Figure 2, a column is double the previous column) existing inside the matrix have to be identified. Once they are identified, the pattern can be applied to find the most appropriate value. Patterns can be found using the following techniques:

1. The sum of rows can be the same.
2. It could be in the form described in Figure 3, where 'o' represents an arithmetic function such as '+', '-', '*', '/', '^', etc.
3. Operators ('+', '-', '*', '/', '^', etc. and one '=') are inserted between columns. If the same combination is valid for each row, that is the pattern.
4. Zig-zag through rows to find a pattern.
5. The matrix is transposed and above steps are repeated.
6. Columns are shuffled and steps 3, 4 are repeated.

This is not yet implemented in the program

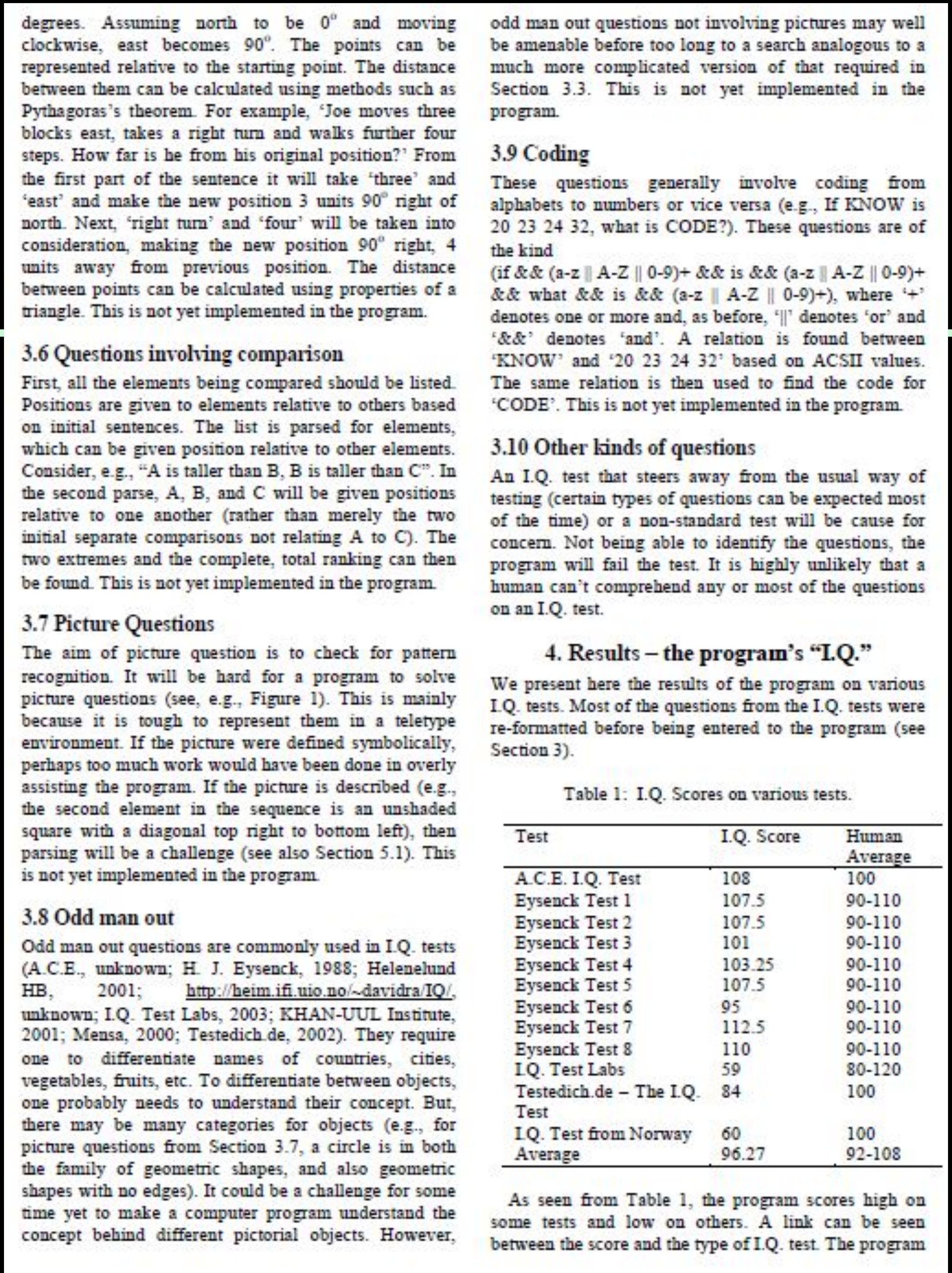
X	XoA	XoAoB
Y	YoA	YoAoB
Z	ZoA	ZoAoB

Figure 3: Patterns in matrix questions

3.5 Questions involving directions

Keywords like (left || right || east || west || north || south) can be used to identify questions involving directions. Directions can be represented quite easily using

2003



degrees. Assuming north to be 0° and moving clockwise, east becomes 90°. The points can be represented relative to the starting point. The distance between them can be calculated using methods such as Pythagoras's theorem. For example, 'Joe moves three blocks east, takes a right turn and walks further four steps. How far is he from his original position?' From the first part of the sentence it will take 'three' and 'east' and make the new position 3 units 90° right of north. Next, 'right turn' and 'four' will be taken into consideration, making the new position 90° right, 4 units away from previous position. The distance between points can be calculated using properties of a triangle. This is not yet implemented in the program.

3.6 Questions involving comparison

First, all the elements being compared should be listed. Positions are given to elements relative to others based on initial sentences. The list is parsed for elements, which can be given position relative to other elements. Consider, e.g., "A is taller than B, B is taller than C". In the second parse, A, B, and C will be given positions relative to one another (rather than merely the two initial separate comparisons not relating A to C). The two extremes and the complete, total ranking can then be found. This is not yet implemented in the program.

3.7 Picture Questions

The aim of picture question is to check for pattern recognition. It will be hard for a program to solve picture questions (see, e.g., Figure 1). This is mainly because it is tough to represent them in a teletype environment. If the picture were defined symbolically, perhaps too much work would have been done in overly assisting the program. If the picture is described (e.g., the second element in the sequence is an unshaded square with a diagonal top right to bottom left), then parsing will be a challenge (see also Section 5.1). This is not yet implemented in the program.

3.8 Odd man out

Odd man out questions are commonly used in I.Q. tests (A.C.E., unknown; H. J. Eysenck, 1988; Helenelund HB, 2001; http://heim.ifi.uio.no/~davidra/IQ/, unknown; I.Q. Test Labs, 2003; KHAN-UUL Institute, 2001; Mensa, 2000; Testedich.de, 2002). They require one to differentiate names of countries, cities, vegetables, fruits, etc. To differentiate between objects, one probably needs to understand their concept. But, there may be many categories for objects (e.g., for picture questions from Section 3.7, a circle is in both the family of geometric shapes, and also geometric shapes with no edges). It could be a challenge for some time yet to make a computer program understand the concept behind different pictorial objects. However, odd man out questions not involving pictures may well be amenable before too long to a search analogous to a much more complicated version of that required in Section 3.3. This is not yet implemented in the program.

3.9 Coding

These questions generally involve coding from alphabets to numbers or vice versa (e.g., If KNOW is 20 23 24 32, what is CODE?). These questions are of the kind
(if && (a-z || A-Z || 0-9)+ && is && (a-z || A-Z || 0-9)+ && what && is && (a-z || A-Z || 0-9)+), where '+' denotes one or more and, as before, '||' denotes 'or' and '&&' denotes 'and'. A relation is found between 'KNOW' and '20 23 24 32' based on ACSII values. The same relation is then used to find the code for 'CODE'. This is not yet implemented in the program.

3.10 Other kinds of questions

An I.Q. test that steers away from the usual way of testing (certain types of questions can be expected most of the time) or a non-standard test will be cause for concern. Not being able to identify the questions, the program will fail the test. It is highly unlikely that a human can't comprehend any or most of the questions on an I.Q. test.

4. Results – the program's "I.Q."

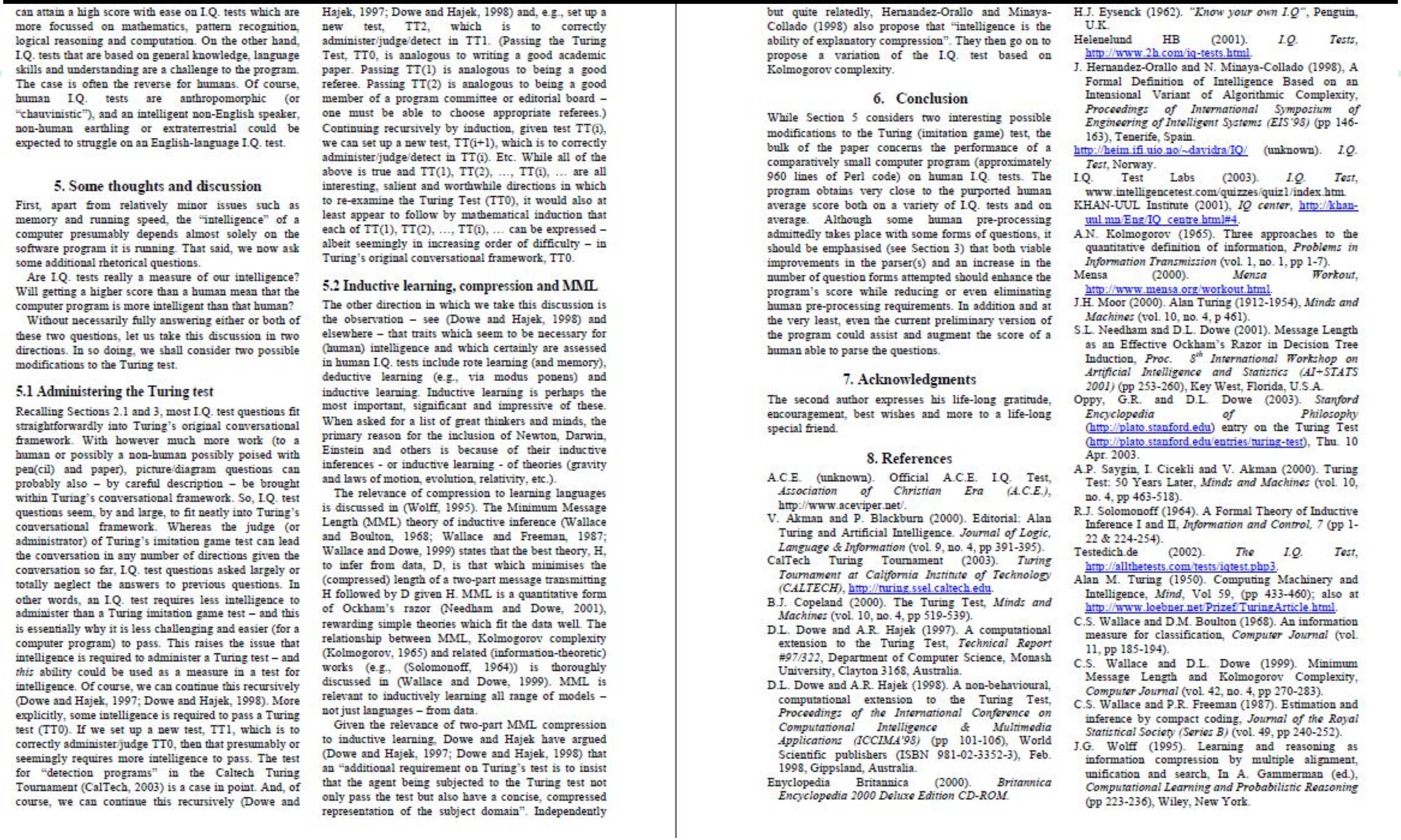
We present here the results of the program on various I.Q. tests. Most of the questions from the I.Q. tests were re-formatted before being entered to the program (see Section 3).

Table 1: I.Q. Scores on various tests.

Test	I.Q. Score	Human Average
A.C.E. I.Q. Test	108	100
Eysenck Test 1	107.5	90-110
Eysenck Test 2	107.5	90-110
Eysenck Test 3	101	90-110
Eysenck Test 4	103.25	90-110
Eysenck Test 5	107.5	90-110
Eysenck Test 6	95	90-110
Eysenck Test 7	112.5	90-110
Eysenck Test 8	110	90-110
I.Q. Test Labs	59	80-120
Testedich.de – The I.Q. Test	84	100
I.Q. Test from Norway	60	100
Average	96.27	92-108

As seen from Table 1, the program scores high on some tests and low on others. A link can be seen between the score and the type of I.Q. test. The program

2003



can attain a high score with ease on I.Q. tests which are more focussed on mathematics, pattern recognition, logical reasoning and computation. On the other hand, I.Q. tests that are based on general knowledge, language skills and understanding are a challenge to the program. The case is often the reverse for humans. Of course, human I.Q. tests are anthropomorphic (or "chauvinistic"), and an intelligent non-English speaker, non-human earthling or extraterrestrial could be expected to struggle on an English-language I.Q. test.

5. Some thoughts and discussion

First, apart from relatively minor issues such as memory and running speed, the "intelligence" of a computer presumably depends almost solely on the software program it is running. That said, we now ask some additional rhetorical questions.

Are I.Q. tests really a measure of our intelligence? Will getting a higher score than a human mean that the computer program is more intelligent than that human?

Without necessarily fully answering either or both of these two questions, let us take this discussion in two directions. In so doing, we shall consider two possible modifications to the Turing test.

5.1 Administering the Turing test

Recalling Sections 2.1 and 3, most I.Q. test questions fit straightforwardly into Turing's original conversational framework. With however much more work (to a human or possibly a non-human possibly poised with pen(cil) and paper), picture/diagram questions can probably also – by careful description – be brought within Turing's conversational framework. So, I.Q. test questions seem, by and large, to fit neatly into Turing's conversational framework. Whereas the judge (or administrator) of Turing's imitation game test can lead the conversation in any number of directions given the conversation so far, I.Q. test questions asked largely or totally neglect the answers to previous questions. In other words, an I.Q. test requires less intelligence to administer than a Turing imitation game test – and this is essentially why it is less challenging and easier (for a computer program) to pass. This raises the issue that intelligence is required to administer a Turing test – and *this* ability could be used as a measure in a test for intelligence. Of course, we can continue this recursively (Dowe and Hajek, 1997; Dowe and Hajek, 1998). More explicitly, some intelligence is required to pass a Turing test (TT0). If we set up a new test, TT1, which is to correctly administer/judge TT0, then that presumably or seemingly requires more intelligence to pass. The test for "detection programs" in the Caltech Turing Tournament (CalTech, 2003) is a case in point. And, of course, we can continue this recursively (Dowe and Hajek, 1997; Dowe and Hajek, 1998) and, e.g., set up a new test, TT2, which is to correctly administer/judge/detect in TT1. (Passing the Turing Test, TT0, is analogous to writing a good academic paper. Passing TT(1) is analogous to being a good referee. Passing TT(2) is analogous to being a good member of a program committee or editorial board – one must be able to choose appropriate referees.) Continuing recursively by induction, given test TT(i), we can set up a new test, TT(i+1), which is to correctly administer/judge/detect in TT(i). Etc. While all of the above is true and TT(1), TT(2), …, TT(i), … are all interesting, salient and worthwhile directions in which to re-examine the Turing Test (TT0), it would also at least appear to follow by mathematical induction that each of TT(1), TT(2), …, TT(i), … can be expressed – albeit seemingly in increasing order of difficulty – in Turing's original conversational framework, TT0.

5.2 Inductive learning, compression and MML

The other direction in which we take this discussion is the observation – see (Dowe and Hajek, 1998) and elsewhere – that traits which seem to be necessary for (human) intelligence and which certainly are assessed in human I.Q. tests include rote learning (and memory), deductive learning (e.g., via modus ponens) and inductive learning. Inductive learning is perhaps the most important, significant and impressive of these. When asked for a list of great thinkers and minds, the primary reason for the inclusion of Newton, Darwin, Einstein and others is because of their inductive inferences - or inductive learning - of theories (gravity and laws of motion, evolution, relativity, etc.).

The relevance of compression to learning languages is discussed in (Wolff, 1995). The Minimum Message Length (MML) theory of inductive inference (Wallace and Boulton, 1968; Wallace and Freeman, 1987; Wallace and Dowe, 1999) states that the best theory, H, to infer from data, D, is that which minimises the (compressed) length of a two-part message transmitting H followed by D given H. MML is a quantitative form of Ockham's razor (Needham and Dowe, 2001), rewarding simple theories which fit the data well. The relationship between MML, Kolmogorov complexity (Kolmogorov, 1965) and related (information-theoretic) works (e.g., (Solomonoff, 1964)) is thoroughly discussed in (Wallace and Dowe, 1999). MML is relevant to inductively learning all range of models – not just languages – from data.

Given the relevance of two-part MML compression to inductive learning, Dowe and Hajek have argued (Dowe and Hajek, 1997; Dowe and Hajek, 1998) that an "additional requirement on Turing's test is to insist that the agent being subjected to the Turing test not only pass the test but also have a concise, compressed representation of the subject domain". Independently but quite relatedly, Hernandez-Orallo and Minaya-Collado (1998) also propose that "intelligence is the ability of explanatory compression". They then go on to propose a variation of the I.Q. test based on Kolmogorov complexity.

6. Conclusion

While Section 5 considers two interesting possible modifications to the Turing (imitation game) test, the bulk of the paper concerns the performance of a comparatively small computer program (approximately 960 lines of Perl code) on human I.Q. tests. The program obtains very close to the purported human average score both on a variety of I.Q. tests and on average. Although some human pre-processing admittedly takes place with some forms of questions, it should be emphasised (see Section 3) that both viable improvements in the parser(s) and an increase in the number of question forms attempted should enhance the program's score while reducing or even eliminating human pre-processing requirements. In addition and at the very least, even the current preliminary version of the program could assist and augment the score of a human able to parse the questions.

7. Acknowledgments

The second author expresses his life-long gratitude, encouragement, best wishes and more to a life-long special friend.

8. References

A.C.E. (unknown). Official A.C.E. I.Q. Test, *Association of Christian Era (A.C.E.)*, http://www.aceviper.net/.

V. Akman and P. Blackburn (2000). Editorial: Alan Turing and Artificial Intelligence. *Journal of Logic, Language & Information* (vol. 9, no. 4, pp 391-395).

CalTech Turing Tournament (2003). *Turing Tournament at California Institute of Technology (CALTECH)*, http://turing.ssel.caltech.edu.

B.J. Copeland (2000). The Turing Test, *Minds and Machines* (vol. 10, no. 4, pp 519-539).

D.L. Dowe and A.R. Hajek (1997). A computational extension to the Turing Test, *Technical Report #97/322*, Department of Computer Science, Monash University, Clayton 3168, Australia.

D.L. Dowe and A.R. Hajek (1998). A non-behavioural, computational extension to the Turing Test, *Proceedings of the International Conference on Computational Intelligence & Multimedia Applications (ICCIMA'98)* (pp 101-106), World Scientific publishers (ISBN 981-02-3352-3), Feb. 1998, Gippsland, Australia.

Enyclopedia Britannica (2000). *Britannica Encyclopedia 2000 Deluxe Edition CD-ROM.*

H.J. Eysenck (1962). *"Know your own I.Q"*, Penguin, U.K.

Helenelund HB (2001). *I.Q. Tests*, http://www.2h.com/iq-tests.html.

J. Hernandez-Orallo and N. Minaya-Collado (1998), A Formal Definition of Intelligence Based on an Intensional Variant of Algorithmic Complexity, *Proceedings of International Symposium of Engineering of Intelligent Systems (EIS'98)* (pp 146-163), Tenerife, Spain.

http://heim.ifi.uio.no/~davidra/IQ/ (unknown). *I.Q. Test*, Norway.

I.Q. Test Labs (2003). *I.Q. Test*, www.intelligencetest.com/quizzes/quiz1/index.htm.

KHAN-UUL Institute (2001), *IQ center*, http://khan-uul.mn/Eng/IQ_centre.html#4.

A.N. Kolmogorov (1965). Three approaches to the quantitative definition of information, *Problems in Information Transmission* (vol. 1, no. 1, pp 1-7).

Mensa (2000). *Mensa Workout*, http://www.mensa.org/workout.html.

J.H. Moor (2000). Alan Turing (1912-1954), *Minds and Machines* (vol. 10, no. 4, p 461).

S.L. Needham and D.L. Dowe (2001). Message Length as an Effective Ockham's Razor in Decision Tree Induction, *Proc. 8th International Workshop on Artificial Intelligence and Statistics (AI+STATS 2001)* (pp 253-260), Key West, Florida, U.S.A.

Oppy, G.R. and D.L. Dowe (2003). *Stanford Encyclopedia of Philosophy* (http://plato.stanford.edu) entry on the Turing Test (http://plato.stanford.edu/entries/turing-test), Thu. 10 Apr. 2003.

A.P. Saygin, I. Cicekli and V. Akman (2000). Turing Test: 50 Years Later, *Minds and Machines* (vol. 10, no. 4, pp 463-518).

R.J. Solomonoff (1964). A Formal Theory of Inductive Inference I and II, *Information and Control, 7* (pp 1-22 & 224-254).

Testedich.de (2002). *The I.Q. Test*, http://allthetests.com/tests/iqtest.php3.

Alan M. Turing (1950). Computing Machinery and Intelligence, *Mind*, Vol 59, (pp 433-460); also at http://www.loebner.net/Prizef/TuringArticle.html.

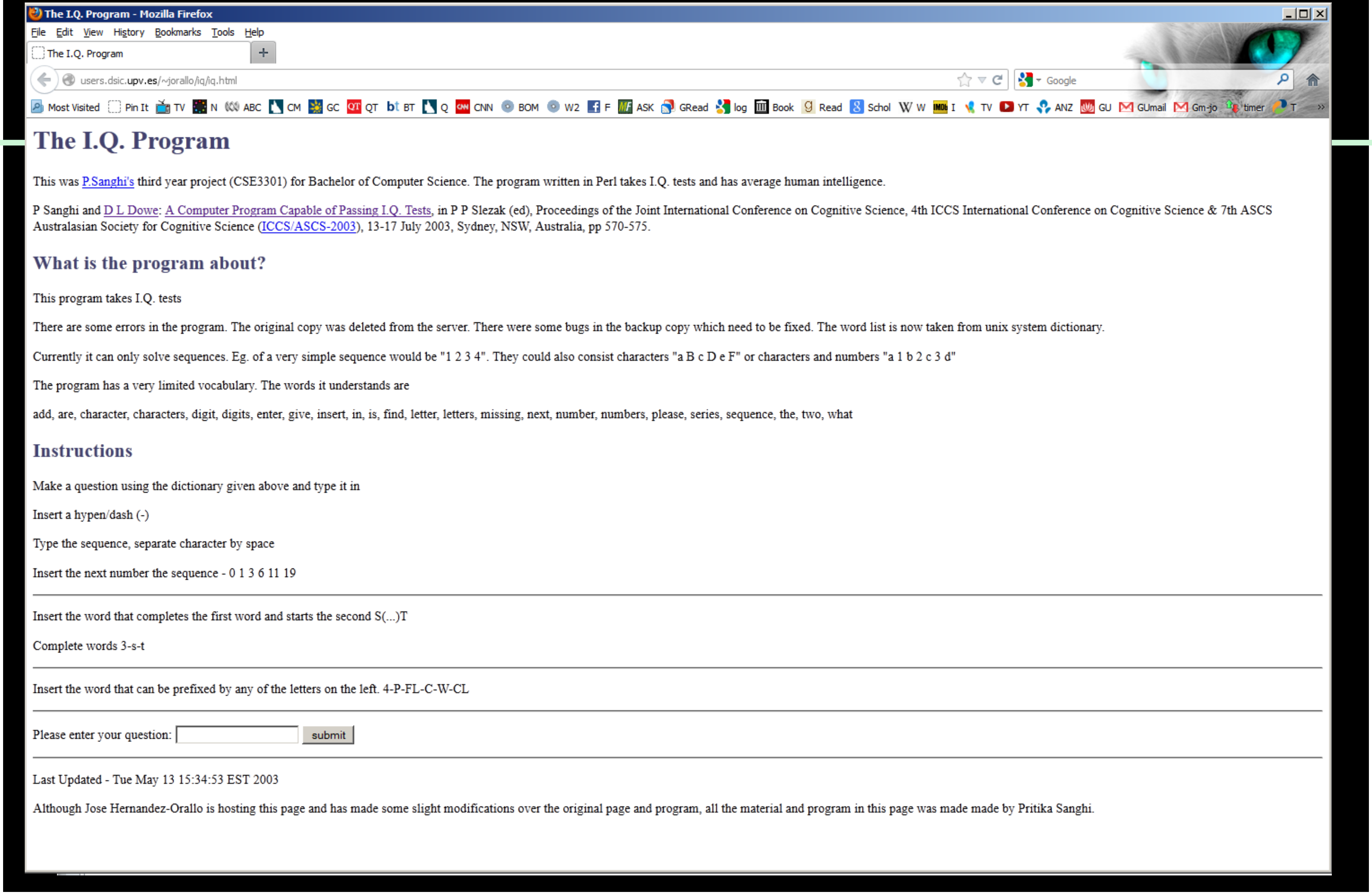
C.S. Wallace and D.M. Boulton (1968). An information measure for classification, *Computer Journal* (vol. 11, pp 185-194).

C.S. Wallace and D.L. Dowe (1999). Minimum Message Length and Kolmogorov Complexity, *Computer Journal* (vol. 42, no. 4, pp 270-283).

C.S. Wallace and P.R. Freeman (1987). Estimation and inference by compact coding, *Journal of the Royal Statistical Society (Series B)* (vol. 49, pp 240-252).

J.G. Wolff (1995). Learning and reasoning as information compression by multiple alignment, unification and search, In A. Gammerman (ed.), *Computational Learning and Probabilistic Reasoning* (pp 223-236), Wiley, New York.

2003


The I.Q. Program - Mozilla Firefox
File Edit View History Bookmarks Tools Help
The I.Q. Program
users.dsic.upv.es/~jorallo/iq/iq.html
Google
Most Visited Pin It TV N ABC CM GC QT BT Q CNN BOM W2 F ASK GRead log Book Read Schol W I TV YT ANZ GU GUmail Gm-jo timer T

The I.Q. Program

This was P.Sanghi's third year project (CSE3301) for Bachelor of Computer Science. The program written in Perl takes I.Q. tests and has average human intelligence.

P Sanghi and D L Dowe: A Computer Program Capable of Passing I.Q. Tests, in P P Slezak (ed), Proceedings of the Joint International Conference on Cognitive Science, 4th ICCS International Conference on Cognitive Science & 7th ASCS Australasian Society for Cognitive Science (ICCS/ASCS-2003), 13-17 July 2003, Sydney, NSW, Australia, pp 570-575.

What is the program about?

This program takes I.Q. tests

There are some errors in the program. The original copy was deleted from the server. There were some bugs in the backup copy which need to be fixed. The word list is now taken from unix system dictionary.

Currently it can only solve sequences. Eg. of a very simple sequence would be "1 2 3 4". They could also consist characters "a B c D e F" or characters and numbers "a 1 b 2 c 3 d"

The program has a very limited vocabulary. The words it understands are

add, are, character, characters, digit, digits, enter, give, insert, in, is, find, letter, letters, missing, next, number, numbers, please, series, sequence, the, two, what

Instructions

Make a question using the dictionary given above and type it in

Insert a hypen/dash (-)

Type the sequence, separate character by space

Insert the next number the sequence - 0 1 3 6 11 19

Insert the word that completes the first word and starts the second S(...)T

Complete words 3-s-t

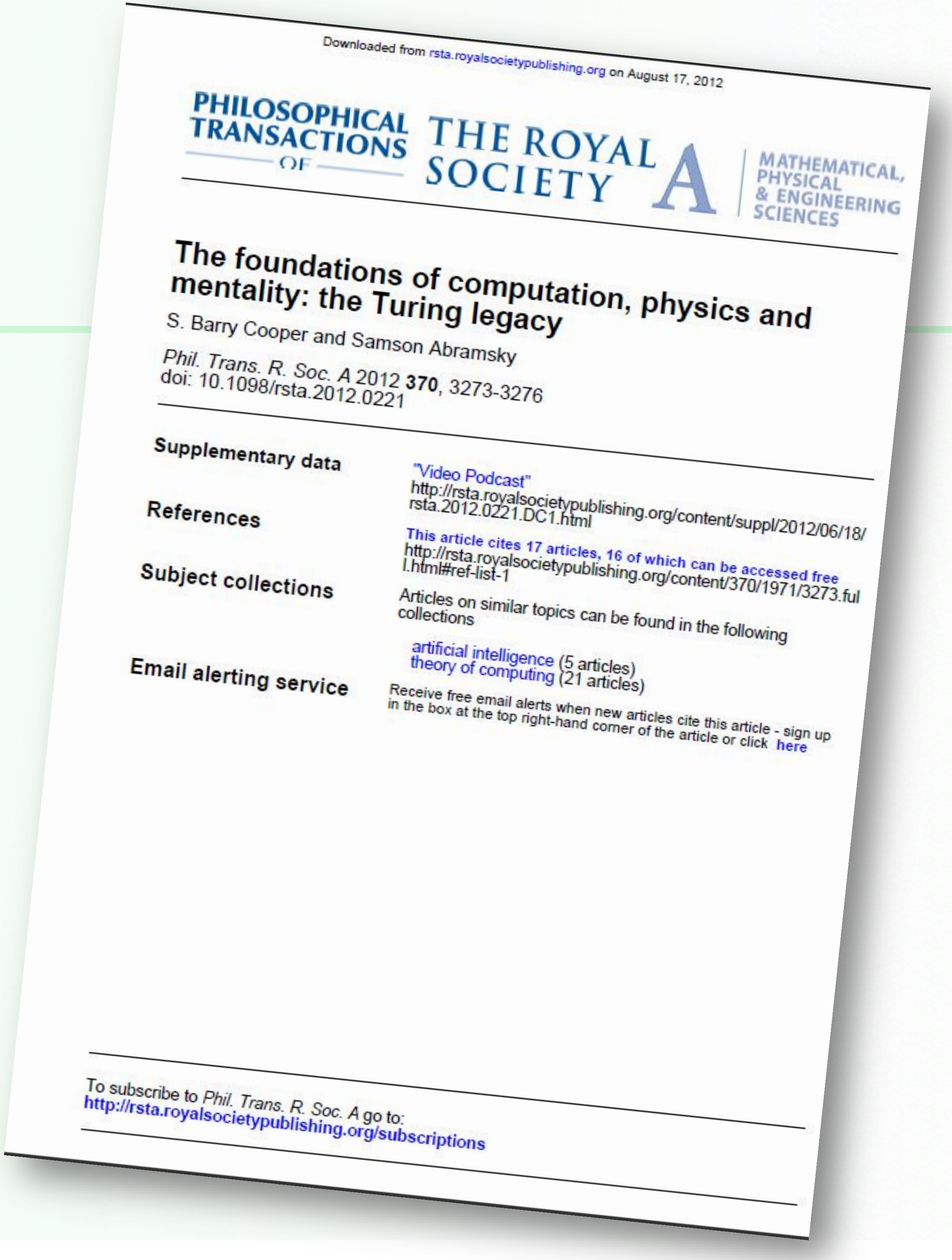
Insert the word that can be prefixed by any of the letters on the left. 4-P-FL-C-W-CL

Please enter your question: ▭ submit

Last Updated - Tue May 13 15:34:53 EST 2003

Although Jose Hernandez-Orallo is hosting this page and has made some slight modifications over the original page and program, all the material and program in this page was made made by Pritika Sanghi.

Turing Centenary June 2012



PHILOSOPHICAL TRANSACTIONS OF THE ROYAL SOCIETY A | MATHEMATICAL, PHYSICAL & ENGINEERING SCIENCES

The foundations of computation, physics and mentality: the Turing legacy

S. Barry Cooper and Samson Abramsky

Phil. Trans. R. Soc. A 2012 **370**, 3273-3276
doi: 10.1098/rsta.2012.0221

Supplementary data

"Video Podcast"
http://rsta.royalsocietypublishing.org/content/suppl/2012/06/18/rsta.2012.0221.DC1.html

References

This article cites 17 articles, 16 of which can be accessed free
http://rsta.royalsocietypublishing.org/content/370/1971/3273.full.html#ref-list-1

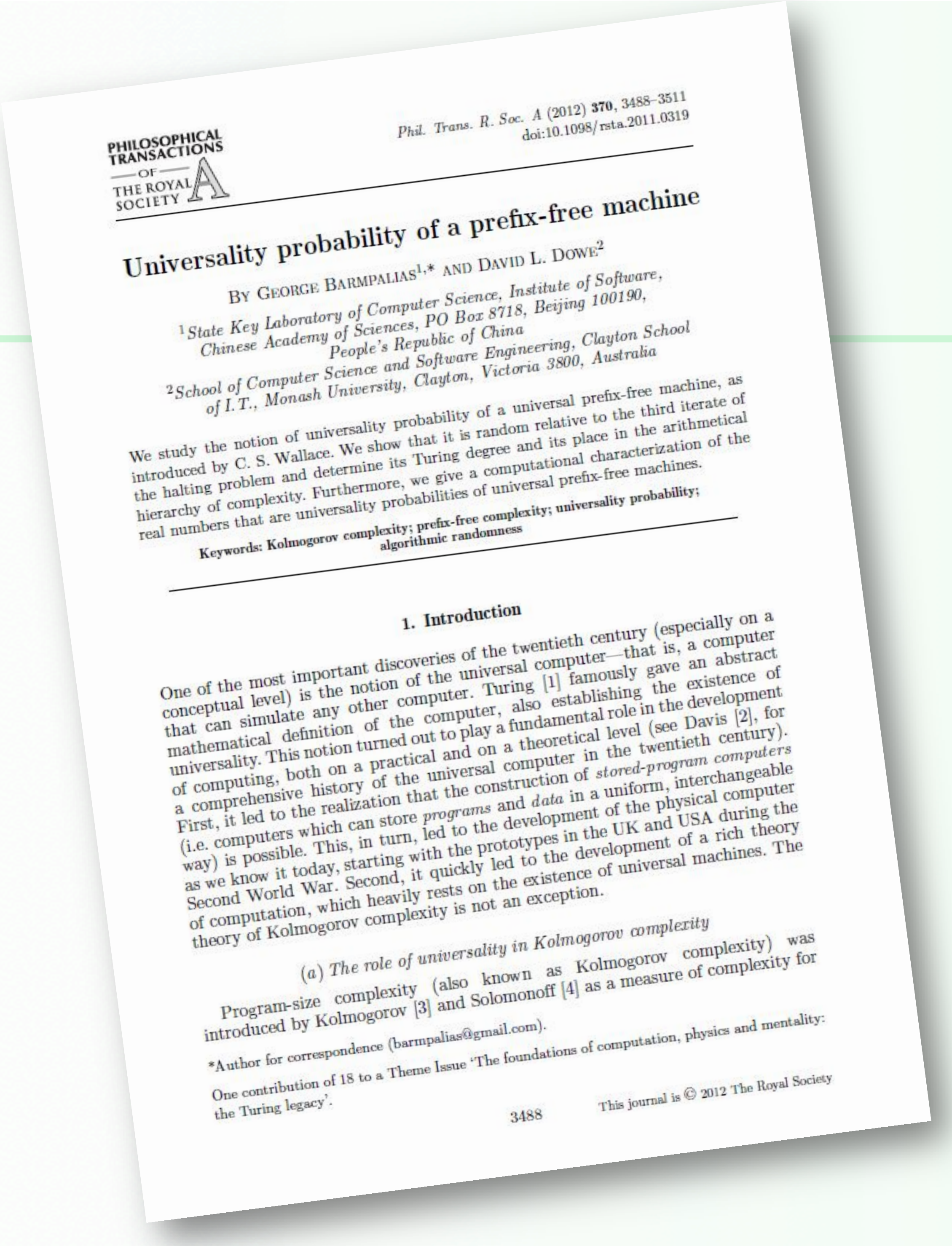
Subject collections

Articles on similar topics can be found in the following collections

artificial intelligence (5 articles)
theory of computing (21 articles)

Email alerting service

Receive free email alerts when new articles cite this article - sign up in the box at the top right-hand corner of the article or click here



PHILOSOPHICAL TRANSACTIONS OF THE ROYAL SOCIETY A

Phil. Trans. R. Soc. A (2012) **370**, 3488–3511
doi:10.1098/rsta.2011.0319

Universality probability of a prefix-free machine

By George Barmpalias[1,*] and David L. Dowe[2]

[1] *State Key Laboratory of Computer Science, Institute of Software,*
Chinese Academy of Sciences, PO Box 8718, Beijing 100190,
People's Republic of China

[2] *School of Computer Science and Software Engineering, Clayton School*
of I.T., Monash University, Clayton, Victoria 3800, Australia

We study the notion of universality probability of a universal prefix-free machine, as introduced by C. S. Wallace. We show that it is random relative to the third iterate of the halting problem and determine its Turing degree and its place in the arithmetical hierarchy of complexity. Furthermore, we give a computational characterization of the real numbers that are universality probabilities of universal prefix-free machines.

Keywords: Kolmogorov complexity; prefix-free complexity; universality probability; algorithmic randomness

1. Introduction

One of the most important discoveries of the twentieth century (especially on a conceptual level) is the notion of the universal computer—that is, a computer that can simulate any other computer. Turing [1] famously gave an abstract mathematical definition of the computer, also establishing the existence of universality. This notion turned out to play a fundamental role in the development of computing, both on a practical and on a theoretical level (see Davis [2], for a comprehensive history of the universal computer in the twentieth century). First, it led to the realization that the construction of *stored-program computers* (i.e. computers which can store *programs* and *data* in a uniform, interchangeable way) is possible. This, in turn, led to the development of the physical computer as we know it today, starting with the prototypes in the UK and USA during the Second World War. Second, it quickly led to the development of a rich theory of computation, which heavily rests on the existence of universal machines. The theory of Kolmogorov complexity is not an exception.

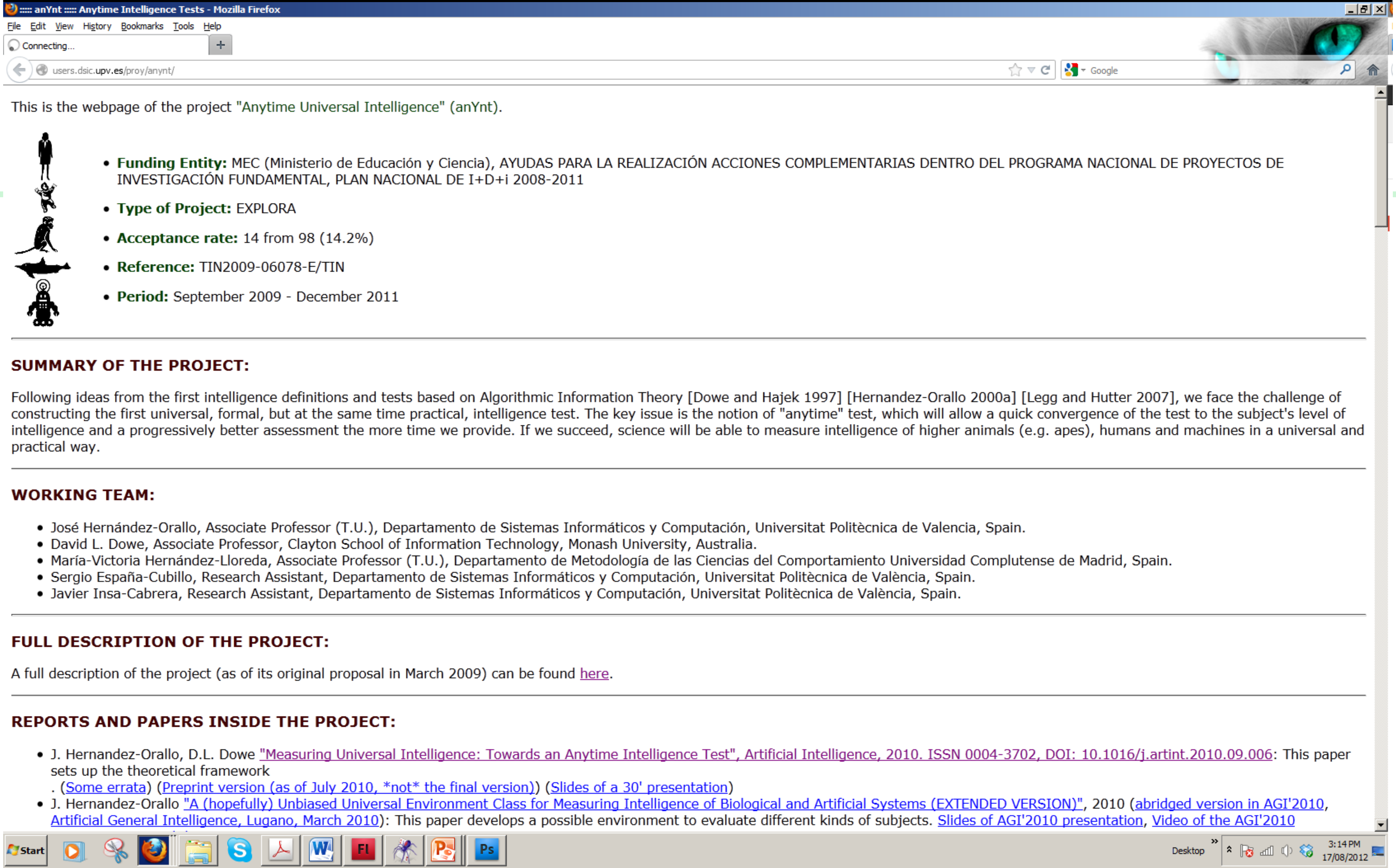
(*a*) *The role of universality in Kolmogorov complexity*

Program-size complexity (also known as Kolmogorov complexity) was introduced by Kolmogorov [3] and Solomonoff [4] as a measure of complexity for

*Author for correspondence (barmpalias@gmail.com).

One contribution of 18 to a Theme Issue 'The foundations of computation, physics and mentality: the Turing legacy'.

“Hockey, or Watching the Daisies Grow,” drawn by Sara Turing and sent to Miss Dunwall, matron at Hazelhurst School, in the spring of 1923. (King’s College, Cambridge)


:::: anYnt :::: Anytime Intelligence Tests - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Connecting...
users.dsic.upv.es/proy/anynt/
Google
Start
Desktop
3:14 PM
17/08/2012

This is the webpage of the project "Anytime Universal Intelligence" (anYnt).

- **Funding Entity:** MEC (Ministerio de Educación y Ciencia), AYUDAS PARA LA REALIZACIÓN ACCIONES COMPLEMENTARIAS DENTRO DEL PROGRAMA NACIONAL DE PROYECTOS DE INVESTIGACIÓN FUNDAMENTAL, PLAN NACIONAL DE I+D+i 2008-2011
- **Type of Project:** EXPLORA
- **Acceptance rate:** 14 from 98 (14.2%)
- **Reference:** TIN2009-06078-E/TIN
- **Period:** September 2009 - December 2011

SUMMARY OF THE PROJECT:

Following ideas from the first intelligence definitions and tests based on Algorithmic Information Theory [Dowe and Hajek 1997] [Hernandez-Orallo 2000a] [Legg and Hutter 2007], we face the challenge of constructing the first universal, formal, but at the same time practical, intelligence test. The key issue is the notion of "anytime" test, which will allow a quick convergence of the test to the subject's level of intelligence and a progressively better assessment the more time we provide. If we succeed, science will be able to measure intelligence of higher animals (e.g. apes), humans and machines in a universal and practical way.

WORKING TEAM:

- José Hernández-Orallo, Associate Professor (T.U.), Departamento de Sistemas Informáticos y Computación, Universitat Politècnica de Valencia, Spain.
- David L. Dowe, Associate Professor, Clayton School of Information Technology, Monash University, Australia.
- María-Victoria Hernández-Lloreda, Associate Professor (T.U.), Departamento de Metodología de las Ciencias del Comportamiento Universidad Complutense de Madrid, Spain.
- Sergio España-Cubillo, Research Assistant, Departamento de Sistemas Informáticos y Computación, Universitat Politècnica de València, Spain.
- Javier Insa-Cabrera, Research Assistant, Departamento de Sistemas Informáticos y Computación, Universitat Politècnica de València, Spain.

FULL DESCRIPTION OF THE PROJECT:

A full description of the project (as of its original proposal in March 2009) can be found here.

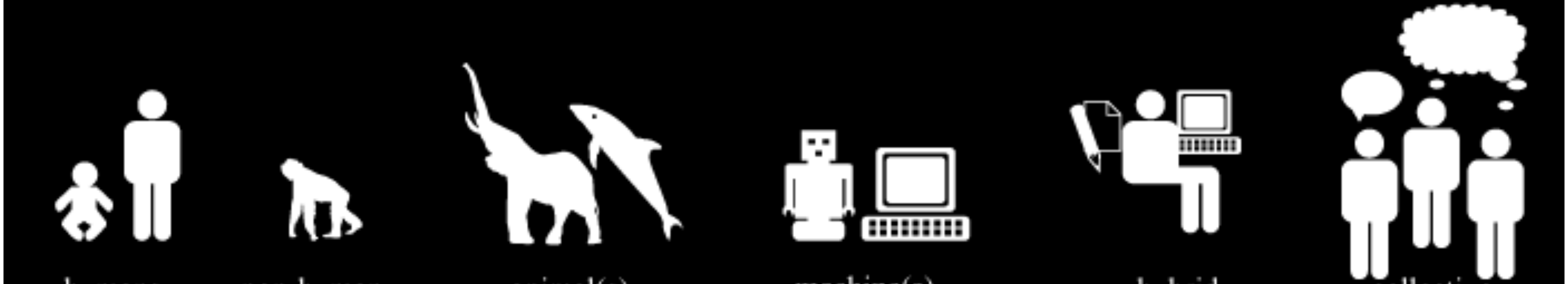
REPORTS AND PAPERS INSIDE THE PROJECT:

- J. Hernandez-Orallo, D.L. Dowe "Measuring Universal Intelligence: Towards an Anytime Intelligence Test", Artificial Intelligence, 2010. ISSN 0004-3702, DOI: 10.1016/j.artint.2010.09.006: This paper sets up the theoretical framework
  . (Some errata) (Preprint version (as of July 2010, *not* the final version)) (Slides of a 30' presentation)
- J. Hernandez-Orallo "A (hopefully) Unbiased Universal Environment Class for Measuring Intelligence of Biological and Artificial Systems (EXTENDED VERSION)", 2010 (abridged version in AGI'2010, Artificial General Intelligence, Lugano, March 2010): This paper develops a possible environment to evaluate different kinds of subjects. Slides of AGI'2010 presentation, Video of the AGI'2010

$k = 9$: a, d, g, j, … *Answer: m*

$k = 12$: a, a, z, c, y, e, x,… *Answer: g*

$k = 14$: c, a, b, d, b, c, c, e, c, d, … *Answer: d*



humans infant /adult

non-human ape

animal(s)

machine(s)

hybrid

collective

Measuring universal intelligence: Towards an anytime intelligence test

Artificial Intelligence 174 (2010) pp.1508–1539

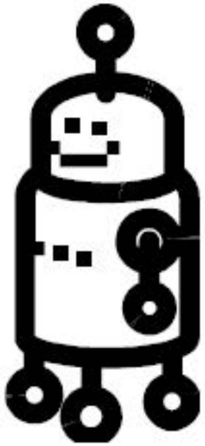
José Hernández Orallo[1], David L. Dowe[2]

1. *Departament de Sistemes Informàtics i Computació, Universitat Politècnica de València, Spain. jorallo@dsic.upv.es.*
2. *Computer Science & Software Engineering, Clayton School of I.T., Monash University, Clayton, Victoria, 3800, Australia.* david.dowe@infotech.monash.edu.au.

Outline

- Towards a universal intelligence test
- Precedents
- Addressing the problems of universal intelligence
- An anytime test
- Instances and implementation
- Conclusions and future work

Towards a universal intelligence test



Evaluating intelligence. Some issues:

1. Harder the less we know about the examinee.

2. Harder if the examinee does not know it is a test.

3. Harder if evaluation is not interactive (static vs. dynamic).

4. Harder if examiner is not adaptive.

Towards a universal intelligence test

State of the art: different subjects, different tests.

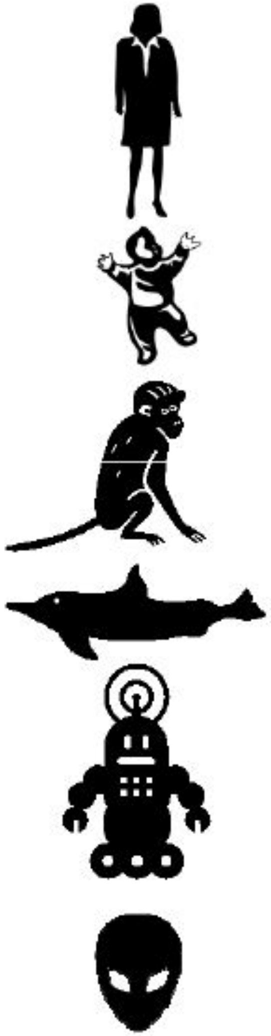
- IQ tests:
  1. Human-specific tests. Natural language assumed.
  2. The examinees know it is a test.
  3. Generally non-interactive.
  4. Generally non-adaptive (pre-designed set of exercises)
- Other tests exist (interviews, C.A.T.)

- Turing test:
  1. Held in a human natural language.
  2. The examinees 'know' it is a test.
  3. Interactive.
  4. Adaptive.
- Other task-specific tests exist.
  - Robotics, games, machine learning.

- Children's intelligence evaluation:
  1. Perception and action abilities assumed.
  2. The examinees do not know it is a test. Rewards are used.
  3. Interactive.
  4. Frequently non-adaptive (pre-designed set of exercises).

- Animal intelligence evaluation:
  1. Perception and action abilities assumed.
  2. The examinees do not know it is a test. Rewards are used.
  3. Interactive.
  4. Generally non-adaptive (pre-designed set of exercises).

Towards a universal intelligence test



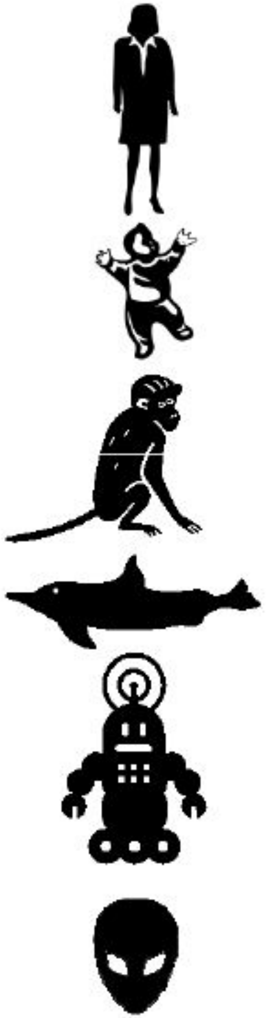
Can we construct a test for all of them?

- Without knowledge about the examinee,
- Derived from computational principles,
- Non-biased (species, culture, language, etc.)
- No human intervention,
- Producing a score,
- Meaningful,
- Practical, and
- **Anytime.**

Is this possible?

- No previous measurement or test of intelligence presented to date fulfils all of these requirements.

Towards a universal intelligence test



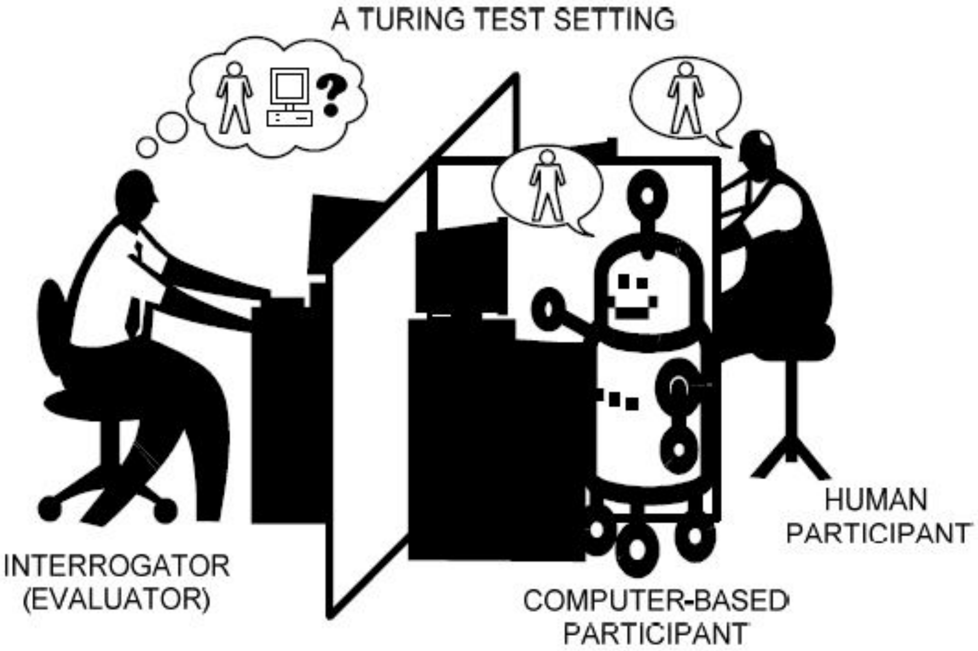
Project: **anYnt** (Anytime Universal Intelligence)

http://users.dsic.upv.es/proy/anynt/

- Any kind of system (biological, non-biological, human)
- Any system now or in the future.
- Any moment in its development (child, adult).
- Any degree of intelligence.
- Any speed.
- Evaluation can be stopped at any time.

Precedents

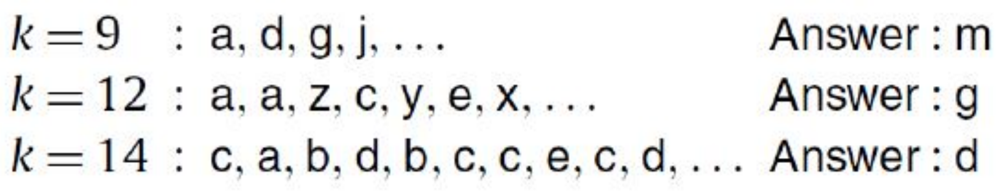
- Turing Test (Turing 1950): anytime and adaptive.



- It is a test of humanity, and needs human intervention.
- Not actually conceived to be a practical test to measure intelligence up to and beyond human intelligence.

Precedents

- Tests based on Kolmogorov Complexity (compression-extended Turing Tests, Dowe 1998) (C-test, Hernandez-Orallo 1998). Very much like IQ tests, but formal and well-grounded.
  - Exercises (series) are not arbitrarily chosen.
  - They are drawn and constructed from a universal distribution:



$k = 9$: a, d, g, j, … Answer : m

$k = 12$: a, a, z, c, y, e, x, … Answer : g

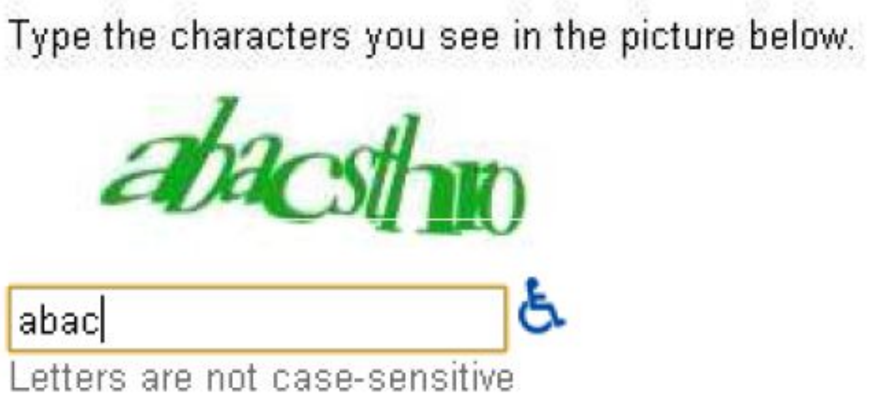
$k = 14$: c, a, b, d, b, c, c, e, c, d, … Answer : d

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

- However, some relatively simple agents can cheat on them (Sanghi and Dowe 2003) and they are static (no planning abilities are required).

Precedents

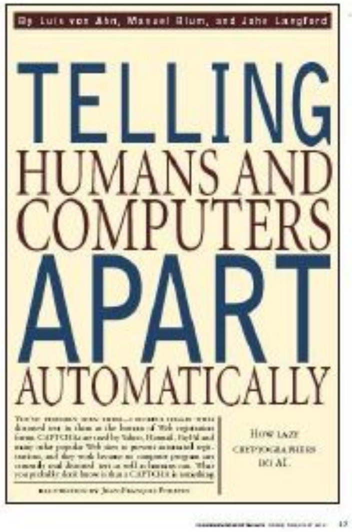
- Captchas (von Ahn, Blum and Langford 2002): quick and practical, but strongly biased. They soon become obsolete.



Type the characters you see in the picture below.

abac

Letters are not case-sensitive



By Luis von Ahn, Manuel Blum, and John Langford

TELLING HUMANS AND COMPUTERS APART AUTOMATICALLY

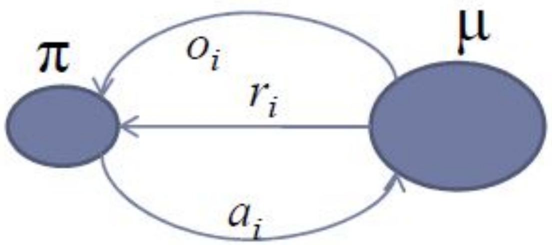
How lazy cryptographers do AI.

- A strong impact in real applications and in the scientific community.
- But...
  - They are not conceived to evaluate intelligence, but to tell humans and machines apart at the current state of AI technology.
  - It is widely recognised that CAPTCHAs will not work in the future.

Precedents

- Universal Intelligence (Legg and Hutter 2007): an interactive extension to C-tests from sequences to environments.

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



= performance over a universal distribution of environments.

- Obvious Problems:
  - *U* is a choice which defines the environment class.
  - The probability distribution is not computable.
  - There are two infinite sums (number of environments and interactions).
  - Time/speed is not considered for the environment or for the agent.
- Other less obvious problems.

Precedents

- A definition of intelligence does not ensure an intelligence test.

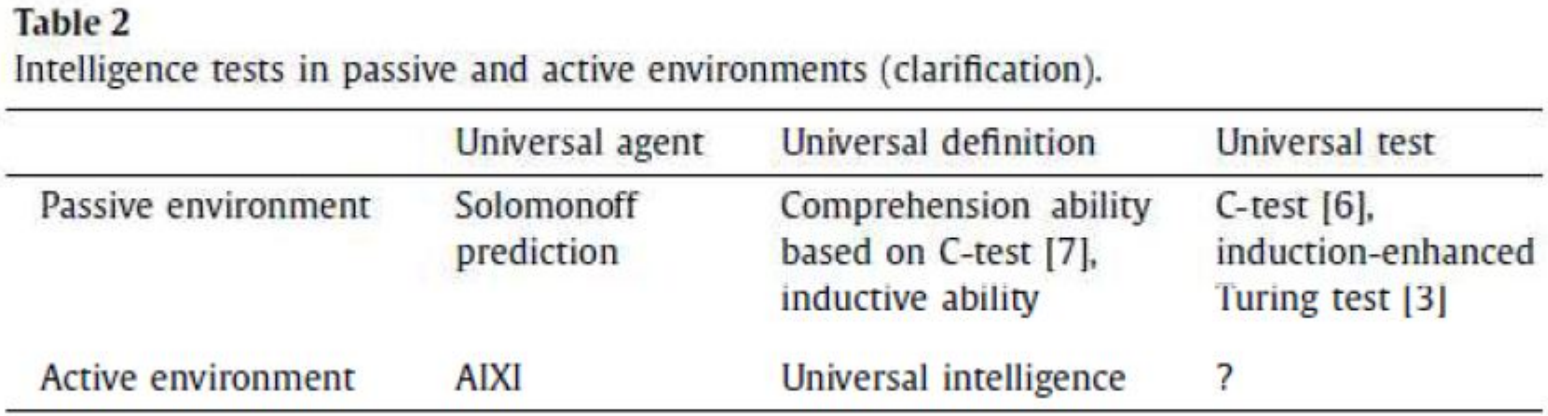


Table 2
Intelligence tests in passive and active environments (clarification).

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

- The C-test used Solomonoff's theory of inductive inference (predictive learning) to define an inductive inference test.
- Universal intelligence provides a definition which adds interaction and the notion of "planning" to the formula (so intelligence = learning + planning).
  - For "Universal Intelligence" we will have to "redefine" it, and then to think about how to use it to construct a feasible test.

Addressing the problems of universal intelligence

- On the difficulty of environments:
  - Very simple environments are given a very high probability

Definition 2 *(Kolmogorov complexity).*

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

Definition 3 *(Universal distribution).*

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

  - Most of the score will come from very simple environments.
    - □ E.g. The 256 environments with K ≤ 8 accumulate a probability of 0.996 (and hence weight, i.e., score) in the definition.

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

- Since we don't have any information about the examinee, we cannot set any limit (or *soften* the distribution).
  - one solution is to make the test adaptive.

Addressing the problems of universal intelligence

- Selecting discriminative environments:
  - Many environments will be completely useless to evaluate intelligence, because:
    - □ Rewards may be independent of agent actions.
    - □ There must be sequences of actions that lead to unrecoverable "states". We cannot assume environments to be ergodic.
    - □ Some environments may be highly benevolent (high expected rewards) and some others can be very malevolent (low expected rewards).
  - We introduce two constraints on environments:
    - □ Environments must be reward-sensitive: *an agent must be able to influence rewards at any point.*
    - □ Environments must be balanced: *a random agent must have an expected reward of 0 (with rewards ranging between -1 and 1).*

Addressing the problems of universal intelligence

- On practical interactions:
  - We have to consider that environments should react almost immediately. We modify the universal distribution as follows:

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

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

    - The use of a parameter *n* makes the definition computable.
  - From here, we re-define the distribution:

$$p_U^t(\mu) := 2^{-Kt_U^{\max}(\mu, n_i)}$$

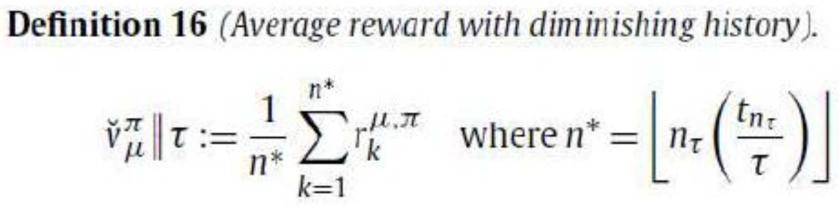
  - And now:
    - We create a finite sample of environments.
    - We also use a limited number of interactions for each environment.

Addressing the problems of universal intelligence

- Time and intelligence:
  - We must consider fast but unintelligent agents as well as slow and intelligent ones.
    - □ But we cannot make these two things independent.
      - □ Otherwise, intelligence would be computationally easier than it is.
  - A way to do that is to set a finite period of time for each environment instead of a "number of interactions".
    - □ Speed will be important because it will increase both exploration and exploitation possibilities.
    - □ In fact, agent's speed will be very relevant.
    - □ But, it is *crucial* to consider balanced environments.

Addressing the problems of universal intelligence

- Reward aggregation:
  - Can we use RL aggregation measures such as accumulated reward and general discounting?
    - □ We show they present important caveats when measuring agents:
      - □ with a finite (previously unknown) period of time,
    - □ Why?
      - □ Given an evaluation time ζ, a fast agent could act randomly and get a good accumulated score and then rest on its laurels.
      - □ These are called “stopping” policies in games.
  - We introduce [48] a new measure for aggregating rewards in a given time ζ, where “discounting” is made to be robust to delaying and stopping policies.

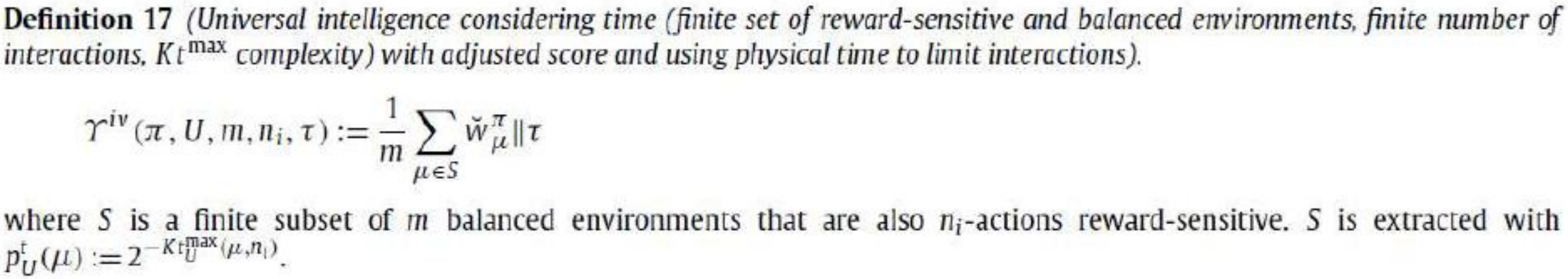


Definition 16 *(Average reward with diminishing history).*

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

An anytime test

- Given all the previous constraints and modifications we can give a definition, which is useful for a test.



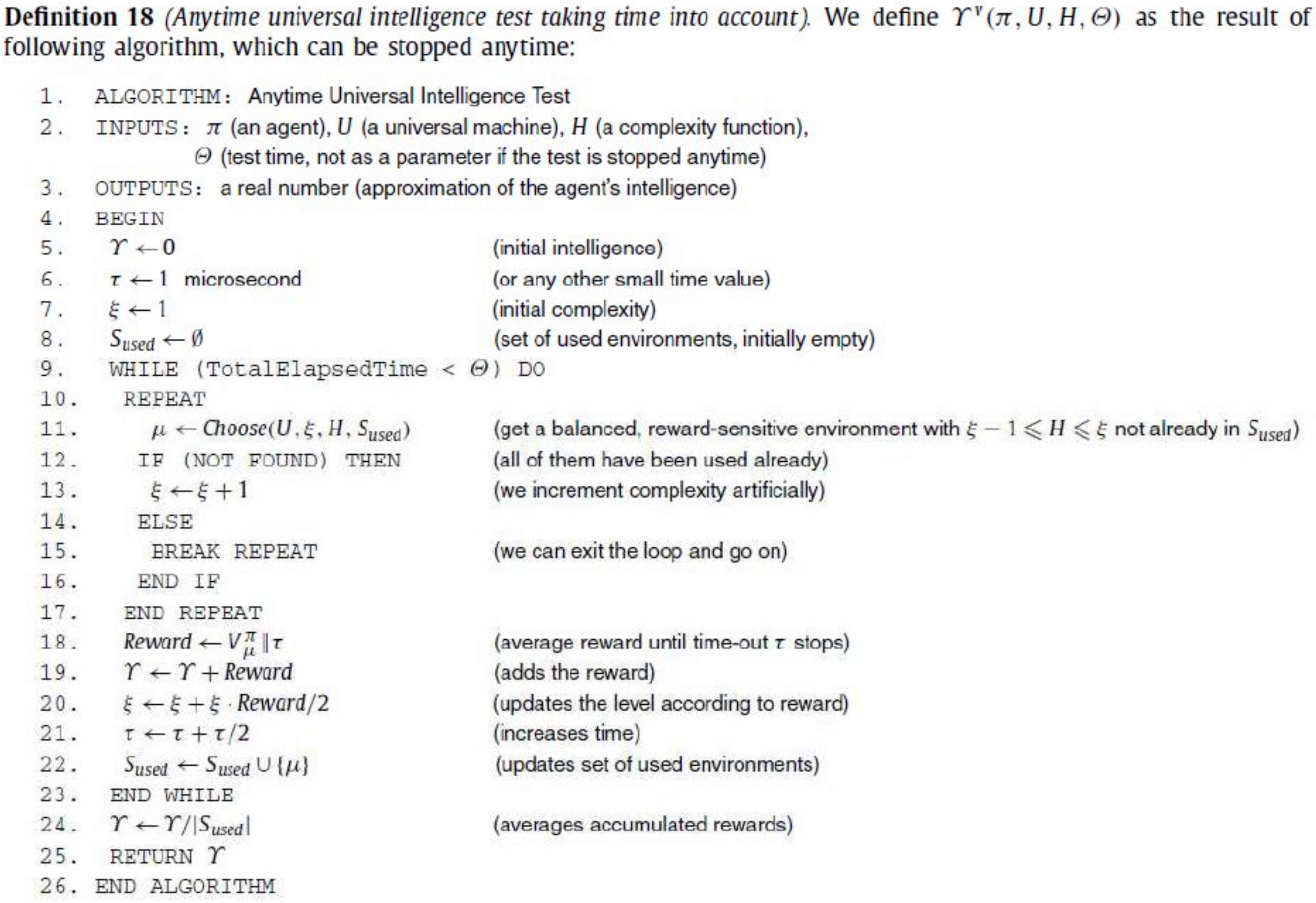
Definition 17 *(Universal intelligence considering time (finite set of reward-sensitive and balanced environments, finite number of interactions, $Kt^{\max}$ complexity) with adjusted score and using physical time to limit interactions).*

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

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

- The definition is parameterised by the number of environments *m* and the time limit for each of them ζ.
  - The higher *m* and ζ are, the better the assessment is expected to be.
  - For a new (unknown) agent, it is difficult to tell the appropriate *m* and ζ.

An anytime test



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

```
ALGORITHM: Anytime Universal Intelligence Test
INPUTS: $\pi$ (an agent), $U$ (a universal machine), $H$ (a complexity function),
          $\Theta$ (test time, not as a parameter if the test is stopped anytime)
OUTPUTS: a real number (approximation of the agent's intelligence)
BEGIN
  $\Upsilon \leftarrow 0$                            (initial intelligence)
  $\tau \leftarrow 1$ microsecond                  (or any other small time value)
  $\xi \leftarrow 1$                                (initial complexity)
  $S_{used} \leftarrow \emptyset$                   (set of used environments, initially empty)
  WHILE (TotalElapsedTime < $\Theta$) DO
    REPEAT
      $\mu \leftarrow Choose(U, \xi, H, S_{used})$   (get a balanced, reward-sensitive environment with $\xi - 1 \leqslant H \leqslant \xi$ not already in $S_{used}$)
      IF (NOT FOUND) THEN                (all of them have been used already)
        $\xi \leftarrow \xi + 1$                    (we increment complexity artificially)
      ELSE
        BREAK REPEAT                     (we can exit the loop and go on)
      END IF
    END REPEAT
    $Reward \leftarrow V_{\mu}^{\pi} \| \tau$              (average reward until time-out $\tau$ stops)
    $\Upsilon \leftarrow \Upsilon + Reward$              (adds the reward)
    $\xi \leftarrow \xi + \xi \cdot Reward/2$          (updates the level according to reward)
    $\tau \leftarrow \tau + \tau/2$                   (increases time)
    $S_{used} \leftarrow S_{used} \cup \{\mu\}$           (updates set of used environments)
  END WHILE
  $\Upsilon \leftarrow \Upsilon / |S_{used}|$             (averages accumulated rewards)
  RETURN $\Upsilon$
END ALGORITHM
```

Instances and implementation

- Implementation of the anytime test requires:
  - To define an environment class *U* (e.g., a Turing-complete machine), where all the environments are balanced and reward-sensitive (or define a computable, preferably efficient, sieve to select them).
  - A complexity function (e.g., Kt^{max})
- Several environment classes may determine general or specific *performance* tests:
  - In [53] we have presented a Turing-complete environment class Λ which is balanced and reward-sensitive .
  - Other specific classes can be used to evaluate subfields of AI:
    - If *U* is chosen to only comprise static environments, we can define a test to evaluate performance on sequence prediction (for machine learning).
    - If *U* is chosen to be *games* (e.g. using the Game Description Language in the AAAI General Game Playing Competition), we have a test to evaluate performance on game playing.
    - Similar things can be done with the reinforcement learning competition, maze learning, etc.

Conclusions and future work

- Since the late 1990s, we have derived several general intelligence tests and definitions with a precise mathematical formulation.
  - Algorithmic Information theory (a.k.a. Kolmogorov complexity) is the key for doing that.
- The most important conclusions of this work are:
  - We have shaped the question of whether it is possible to construct an intelligence test which is universal, formal, meaningful and anytime.
  - We have identified the most important problems for such a test:
    - the notion of environment complexity and an appropriate distribution,
    - the issue that many environments may be useless for evaluation (not discriminative),
    - a proper sample of environments and time slots for each environment,
    - computability and efficiency,
    - time and speed for both agent and environment,
    - evaluation (reward aggregation) in a finite period of time,
    - the choice of an unbiased environment.

Conclusions and future work

- This proposal can obviously be refined and improved:
  - The use of balanced environments and the character of the anytime test suggest that for many (Turing-complete) environment classes, the measure is convergent, but this should be shown theoretically or experimentally.
  - Kt^{max} needs a parameter to be computable. Other variants might exist without parameters (e.g., using the speed prior).
  - The probability of social environments (other intelligent agents inside) is almost 0. A complexity measure including other agents could be explored.
- Implementation:
  - Currently implementing an approximation to the test using the environment class Λ.
  - Also considering implementing an approximation using the GDL (Game Description Language) as environment class.
- Experimentation:
  - On AI agents (e.g. RL Q learning, AIXI approximations, etc.), humans, non-human animals, children.

On more realistic environment distributions for defining, evaluating and developing intelligence

José Hernández-Orallo[1] David L. Dowe[2] Sergio España-Cubillo[3]
M.Victoria Hernández-Lloreda[4] Javier Insa-Cabrera[1]

[1] DSIC, Universitat Politècnica de València, Spain. {jorallo, jinsa}@dsic.upv.es
[2] Clayton School of Information Technology, Monash University, Australia.
david.dowe@monash.edu
[3] ProS Research Center, Universitat Politècnica de València, Spain.
sergio.espana@pros.upv.es
[4] Departamento de Metodología de las Ciencias del Comportamiento, Universidad
Complutense de Madrid, Spain. vhlloreda@psi.ucm.es

Abstract. One insightful view of the notion of intelligence is the ability to perform well in a diverse set of tasks, problems or environments. One of the key issues is therefore the choice of this set and the probability of each individual, which can be formalised as a 'distribution'. Formalising and properly defining this distribution is an important challenge to understand what intelligence is and to achieve artificial general intelligence (AGI). In this paper, we agree with previous criticisms that a universal distribution using a reference universal Turing machine (UTM) over tasks, environments, etc., is perhaps much too general, since, e.g., the probability of other agents appearing on the scene or having some social interaction is almost 0 for most reference UTMs. Instead, we propose the notion of Darwin-Wallace distribution for environments, which is inspired by biological evolution, artificial life and evolutionary computation. However, although enlightening about where and how intelligence should excel, this distribution has so many options and is uncomputable in so many ways that we certainly need a more practical alternative. We propose the use of intelligence tests over multi-agent systems, in such a way that agents with a certified level of intelligence at a certain degree are used to construct the tests for the next degree. This constructive methodology can then be used as a more realistic intelligence test and also as a testbed for developing and evaluating AGI systems.

Keywords: Intelligence, Evolutionary Computation, Artificial Life, Social Intelligence, Intelligence Test, Universal Distribution.

1 Introduction

Understanding what intelligence is (and is not) plays a crucial role in developing truly general intelligent machines. Apart from the many informal definitions from psychology, philosophy, biology, artificial intelligence and other disciplines (see an account in [16]), there have been some definitions which include the notion of compression, Kolmogorov Complexity or related concepts such as

On more realistic environment distributions for defining, evaluating and developing intelligence

José Hernández Orallo[1] , David L. Dowe[2], Sergio España-Cubillo[1], M.Victoria Hernández-Lloreda[3], Javier Insa-Cabrera[1]

1. *Departament de Sistemes Informàtics i Computació, Universitat Politècnica de València, Spain.*
2. *Computer Science & Software Engineering, Clayton School of I.T., Monash University, Clayton, Victoria, 3800, Australia.*
3. *Departamento de Metodología de las Ciencias del Comportamiento, Universidad Complutense de Madrid, Spain*

AGI'2011 - Mountain View, California, 3-7 August 2011

Outline

- General performance, general distribution
- Generating more social, ‘natural’ environments
- Darwin-Wallace Distribution
- Approximations
- Discussion

General Performance, General Distribution

Intelligence as performance in a wide range of tasks

- Artificial (Specific) Intelligence focusses on *specific* tasks.
  - The development of successful agents in these domains usually entails a specialised approach.
  - Problem repositories for each domain are used to evaluate these agents or algorithms (pattern recognition, machine learning, games, natural language, robotics, etc.).
  - There are very few cases in the literature where the set of problems is obtained by a *problem generator* from a specific distribution.

General Performance, General Distribution

- Artificial General Intelligence must focus on *general* tasks.
  - We can construct a general set of tasks by aggregating several problems which humans face everyday.
    - Arbitrary approach (how many of these, how many of those, ...)
    - Makes it difficult to know what "intelligence" really means.
  - But we *can* formally define a general distribution and generate tasks or environments from it.

General Performance, General Distribution

- Let us choose the most general one: a *universal distribution* over tasks or environments.

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

  - Where *K* is a measure of complexity (Kolmogorov complexity, or any computable approximation, Levin's *Kt*, Schmidhuber's Speed Prior, etc.)

- This approach has been explored in many ways:
  - Compression-extended Turing Tests (Dowe & Hajek 1997a-b, 1998).
  - Formal definition of intelligence, C-test (Hernandez-Orallo 1998, 2000).
  - Compression tests (Mahoney't text compression test 1999, Jim Bowery's Cprize 2005, Hutter's Prize, 2006).
  - Universal Intelligence (Legg & Hutter 2007).
  - Anytime Intelligence Tests (Hernandez-Orallo & Dowe 2010).

General Performance, General Distribution

- A universal distribution.
  - Advantages:
    - We can assign probabilities to an infinite number of tasks.
    - Universal distributions “dominate” all other possible distributions.
    - Sound results (Solomonoff’s theory of prediction, Hutter’s AIXI, etc.).
    - Simple environments frequent ⇨ Tasks easier to generate and use.
  - Disadvantages:
    - The arbitrary choice of the reference machine is still important.
      - □ This can be minimised by using background knowledge or using simplest UTMs (Wallace 2005, Dowe 2008a).
    - Any environment of interest (e.g. multi-agent system) has a very low probability for almost every reference machine.
      - □ Performance in social, natural environments, including other (intelligent) agents will not be measured.

Generating more social, 'natural' environments

- But intelligence is all about *social* cognition!

The Social Cognition / Cultural Intelligence Hypothesis

[Herrmann et al. 2007]

- Alternative proposals:
  - More realistic (but simplified) worlds, not using a universal distribution:
    - Social, natural, embodied environments... (e.g. AGI preschool [Goertzel 2009])
  - Choose a very particular reference machine, keeping a universal distribution:
    - Games (Hernandez-Orallo & Dowe 2010).
  - "Alter" a universal distribution:
    - Include other agents.
    - *Evolve* the distribution.

Darwin-Wallace Distribution

- We define a distribution over *multi-agent* environments (not including the agents):

$$p_E(\mu) := 2^{-K_{U_e}(\mu)}$$

- We define a distribution over agents (a "mind distribution"):

$$p_A(\pi) := 2^{-K_{U_a}(\pi)}$$

  - We assume all the agents are physically equal.
    - This is important and very different to natural evolution.
    - We only care about their "minds".

- We combine these two distributions...

Darwin-Wallace Distribution

- The probability of the *start-up* multi-agent environment σ is:

$$p_S(\sigma) = p_S(\langle \mu, \pi_1, \pi_2, \ldots, \pi_m \rangle) := p_E(\mu) \times \prod_{j=1}^{m} p_A(\pi^j)$$

- And now we evolve this in the following way:
  - Agent survival depends on a function *d*, related to their average rewards.
    - Dead agents are replaced by new agents.
  - The environment can be replaced by any other environment in p_E with a rate of replacement of *c*.
    - Agents do not specialise in *one* environment. They adapt to changing environments.

- The Darwin-Wallace distribution for *d*, *c* at iteration *i* is given by:

$$p_{d,c,i}(\sigma) = p_i(\langle \mu, \pi_1, \pi_2, \ldots, \pi_m \rangle) := p_E(\mu) \times \prod_{j=1}^{m} q_{(d,c,i)}(\pi^j)$$

  - Where *q(d,c,i)* is the agent probability at iteration *i*.

Darwin-Wallace Distribution

- What does this family of distributions mean?
  - It just assigns probabilities to multi-agent environments.
  - Complex agents with complex/adaptive behaviour are much more likely in this distribution, for large values of *i*.
  - The distribution is completely different for low and high values of *i*.
    - Highly social agents may be unsuccessful in environments with very simple agents, where co-operation and language are useless.
      - □ As a single human on an island, in the Precambrian period or on Mars.
  - Social adaptability instead of adaptation to one single environment.

Previous definitions and tests of intelligence using a universal distribution could be re-understood with a Darwin-Wallace distribution.

Approximations

- Appealing as an abstract concept.
  - Problems for using it in practice:
    - The definition is a product of other distributions, which are not necessarily independent (it would require a normalisation).
    - The distribution is uncomputable (with *K* being Kolmogorov Complexity) or clearly intractable using computable variants of *K*.
    - Some evolution "accelerators" have been ruled out (mutations, cross-over, genotype, ...).
      - □ We cannot wait some billion years.
  - But...

Nobody is saying that we have to wait until the agents are "naturally" created by evolution.

Approximations

- Approximation through *testing*:
  - Research-driven evolution instead of natural evolution.
    - Agents can be created artificially (by AGI researchers) but assessed in an independent way.
  - The “intelligence”/”adaptability” of agents can be assessed for different values of *i*.
    - We certify agents at lower levels of *i*, before including them in the testbed.

- This (competitive) process can foster the development of more and more (socially) intelligent systems.

Discussion

- The Darwin-Wallace distribution is not a distribution of "life forms"
  - A distribution of 'life forms' gives higher probability to bacteria and cockroaches.

- The Darwin-Wallace is a distribution of (social) "mind forms".

  - There are three features which make this distinction:
    - i) Physical traits do not matter (no body).
      - Focus is placed on behaviour.
    - ii) There is no genotype, cross-over, mutation, etc.,
      - Selection does not work for genes or species, but for individuals.
    - iii) Environments are replaced.
      - Avoids specialisation in a single environment.
      - Instead, adaptability to a wide range of environments (i.e., intelligence) is the only fitness function for selection.

Discussion

The Darwin-Wallace distribution assigns probabilities to agents depending on their success on a variety of environments with a variety of other agents.

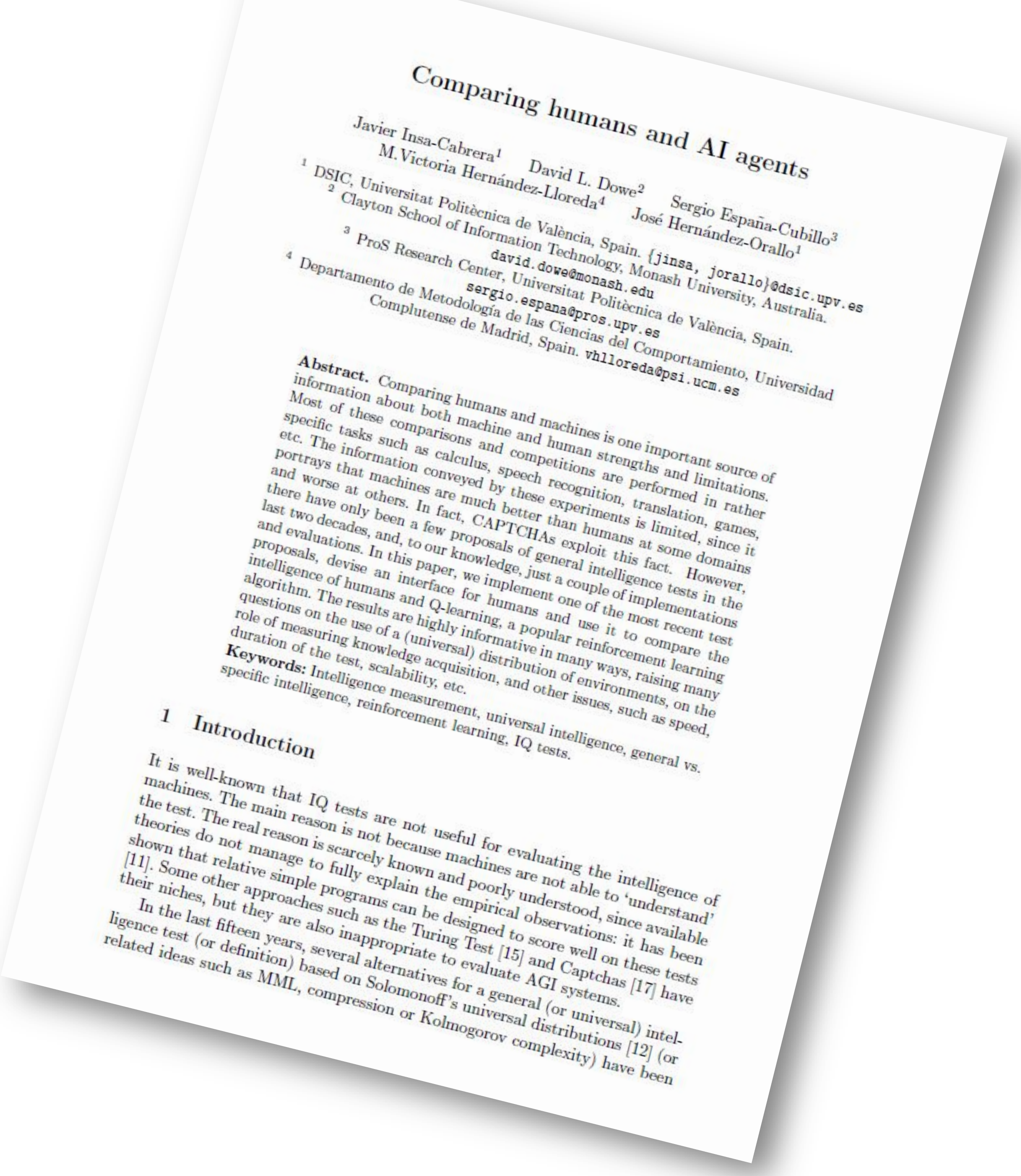
- It relates intelligence to evolution, without abandoning the context of universal distributions.
  - This, of course, raises more questions than it answers, but...
    - □ It can help understand why universal distributions may be "too general" and unrealistic for worlds where intelligence has developed.
    - □ It can help suggest ways to link intelligence definitions with evolution, adversarial learning, competition and collaboration.

Thank you!

Some pointers:

- Project: **anYnt** (Anytime Universal Intelligence)

  http://users.dsic.upv.es/proy/anynt/



Comparing humans and AI agents

Javier Insa-Cabrera[1] David L. Dowe[2] Sergio España-Cubillo[3]
M.Victoria Hernández-Lloreda[4] José Hernández-Orallo[1]

[1] DSIC, Universitat Politècnica de València, Spain. {jinsa, jorallo}@dsic.upv.es
[2] Clayton School of Information Technology, Monash University, Australia.
david.dowe@monash.edu
[3] ProS Research Center, Universitat Politècnica de València, Spain.
sergio.espana@pros.upv.es
[4] Departamento de Metodología de las Ciencias del Comportamiento, Universidad
Complutense de Madrid, Spain. vhlloreda@psi.ucm.es

Abstract. Comparing humans and machines is one important source of information about both machine and human strengths and limitations. Most of these comparisons and competitions are performed in rather specific tasks such as calculus, speech recognition, translation, games, etc. The information conveyed by these experiments is limited, since it portrays that machines are much better than humans at some domains and worse at others. In fact, CAPTCHAs exploit this fact. However, there have only been a few proposals of general intelligence tests in the last two decades, and, to our knowledge, just a couple of implementations and evaluations. In this paper, we implement one of the most recent test proposals, devise an interface for humans and use it to compare the intelligence of humans and Q-learning, a popular reinforcement learning algorithm. The results are highly informative in many ways, raising many questions on the use of a (universal) distribution of environments, on the role of measuring knowledge acquisition, and other issues, such as speed, duration of the test, scalability, etc.

Keywords: Intelligence measurement, universal intelligence, general vs. specific intelligence, reinforcement learning, IQ tests.

1 Introduction

It is well-known that IQ tests are not useful for evaluating the intelligence of machines. The main reason is not because machines are not able to 'understand' the test. The real reason is scarcely known and poorly understood, since available theories do not manage to fully explain the empirical observations: it has been shown that relative simple programs can be designed to score well on these tests [11]. Some other approaches such as the Turing Test [15] and Captchas [17] have their niches, but they are also inappropriate to evaluate AGI systems.

In the last fifteen years, several alternatives for a general (or universal) intelligence test (or definition) based on Solomonoff's universal distributions [12] (or related ideas such as MML, compression or Kolmogorov complexity) have been

Comparing Humans and AI Agents

Javier Insa-Cabrera[1], David L. Dowe[2], Sergio España-Cubillo[1],
M.Victoria Hernández-Lloreda[3], José Hernández Orallo[1]

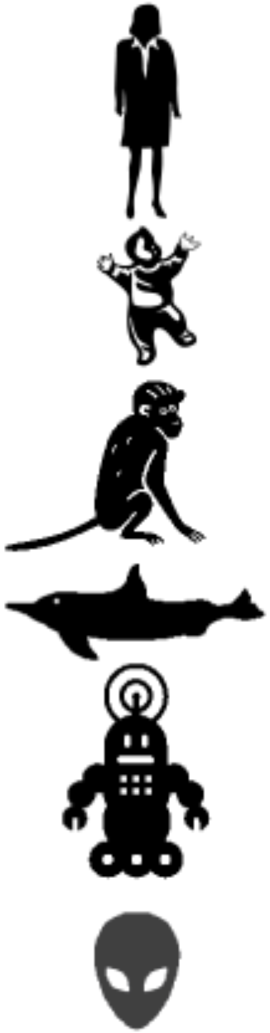
1. *Departament de Sistemes Informàtics i Computació, Universitat Politècnica de València, Spain.*
2. *Computer Science & Software Engineering, Clayton School of I.T., Monash University, Clayton, Victoria, 3800, Australia.*
3. *Departamento de Metodología de las Ciencias del Comportamiento, Universidad Complutense de Madrid, Spain*

AGI'2011 - Mountain View, California, 3-7 August 2011

Outline

- Measuring intelligence universally
- Precedents
- Test setting and administration
- Agents and interfaces
- Results
- Discussion

Measuring intelligence universally



- Can we construct a ‘universal’ intelligence test?

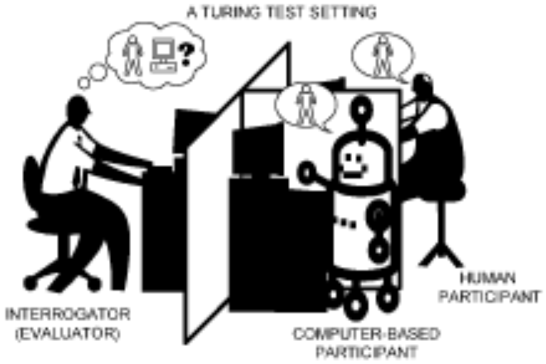
Project: **anYnt** (Anytime Universal Intelligence)

http://users.dsic.upv.es/proy/anynt/

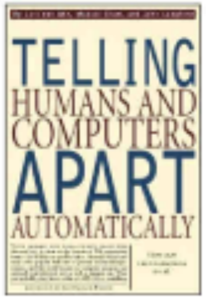
- Any kind of system (biological, non-biological, human)
- Any system now or in the future.
- Any moment in its development (child, adult).
- Any degree of intelligence.
- Any speed.
- Evaluation can be stopped at any time.

Precedents

- Imitation Game "Turing Test" (Turing 1950):
  - It is a test of *humanity*, and needs human intervention.
  - Not actually conceived to be a practical test for measuring intelligence up to and beyond human intelligence.



- CAPTCHAs (von Ahn, Blum and Langford 2002):
  - Quick and practical, but strongly biased.
  - They evaluate *specific* tasks.
  - They are not conceived to evaluate intelligence, but to tell humans and machines apart at the current state of AI technology.
  - It is widely recognised that CAPTCHAs will not work in the future (they soon become obsolete).



TELLING HUMANS AND COMPUTERS APART AUTOMATICALLY

Type the characters you see in the picture below.

abac

Letters are not case-sensitive

Precedents

- Tests based on Kolmogorov Complexity (compression-extended Turing Tests, Dowe 1997a-b, 1998) (C-test, Hernandez-Orallo 1998).
  - Look like IQ tests, but formal and well-grounded.
  - Exercises (series) are not arbitrarily chosen.
  - They are drawn and constructed from a universal distribution, by setting several 'levels' for *k*:

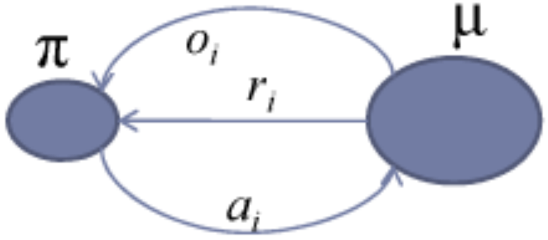
$$\begin{array}{lll} k=9 & : \text{a, d, g, j, } \ldots & \text{Answer : m} \\ k=12 & : \text{a, a, z, c, y, e, x, } \ldots & \text{Answer : g} \\ k=14 & : \text{c, a, b, d, b, c, c, e, c, d, } \ldots & \text{Answer : d} \end{array}$$

- However...
  - Some relatively simple algorithms perform well in IQ-like tests (Sanghi and Dowe 2003).
  - They are static (no planning abilities are required).

Precedents

- Universal Intelligence (Legg and Hutter 2007): an *interactive* extension to C-tests from sequences to environments.

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



= performance over a universal distribution of environments.

- Universal intelligence provides a definition which adds interaction and the notion of "planning" to the formula (so intelligence = learning + planning).
  - This makes this apparently different from an IQ (static) test.

Precedents

- A definition of intelligence does not ensure an intelligence test.

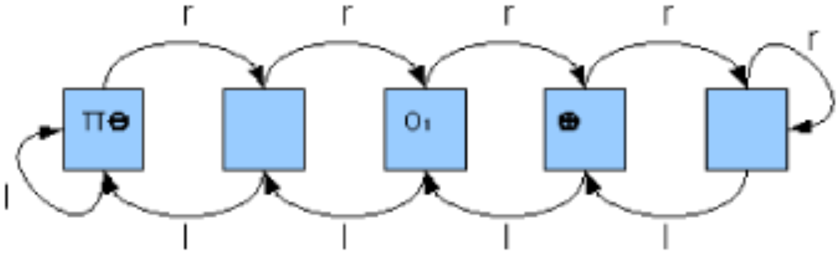
- Anytime Intelligence Test (Hernandez-Orallo and Dowe 2010):
  - An interactive setting following (Legg and Hutter 2007) which addresses:
    - □ Issues about the difficulty of environments.
    - □ The definition of discriminative environments.
    - □ Finite samples and (practical) finite interactions.
    - □ Time (speed) of agents and environments.
    - □ Reward aggregation, convergence issues.
    - □ Anytime and adaptive application.

- An environment class Λ (Hernandez-Orallo 2010) (AGI-2010).

In this work we perform an implementation of the test and we evaluate humans and a reinforcement learning algorithm with it, as a proof of concept.

Test setting and administration

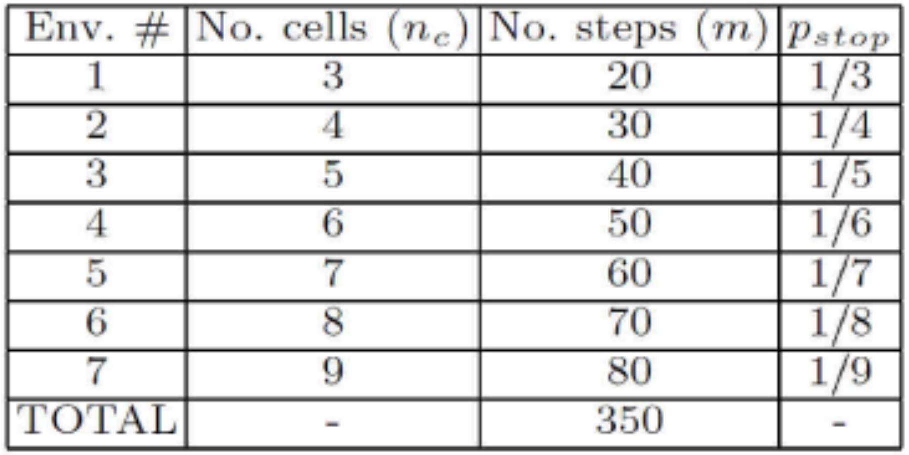
- Implementation of the environment class :
  - Spaces are defined as fully connected graphs.
- Actions are the arrows in the graphs.
- Observations are the ‘contents’ of each edge/cell in the graph.



  - Agents can perform actions inside the space.
  - Rewards:
- Two special agents *Good* (⊕) and *Evil* (⊖), which are responsible for the rewards. Symmetric behaviour, to ensure balancedness.

Test setting and administration

- We randomly generated only 7 environments for the test:
  - Different topologies and sizes for the patterns of the agents Good and Evil (which provide rewards).
  - Different lengths for each session (exercise) accordingly to the number of cells and the size of the patterns.



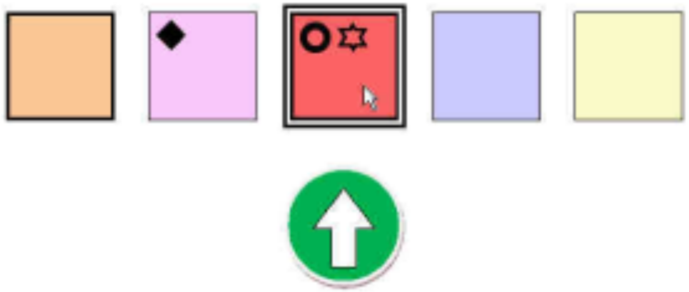
Env. #	No. cells (n_c)	No. steps (m)	p_{stop}
1	3	20	1/3
2	4	30	1/4
3	5	40	1/5
4	6	50	1/6
5	7	60	1/7
6	8	70	1/8
7	9	80	1/9
TOTAL	-	350	-

  - The goal was to allow for a feasible administration for humans in about 20-30 minutes.

Agents and interfaces

- An AI agent: Q-learning
  - A simple choice. A well-known algorithm.

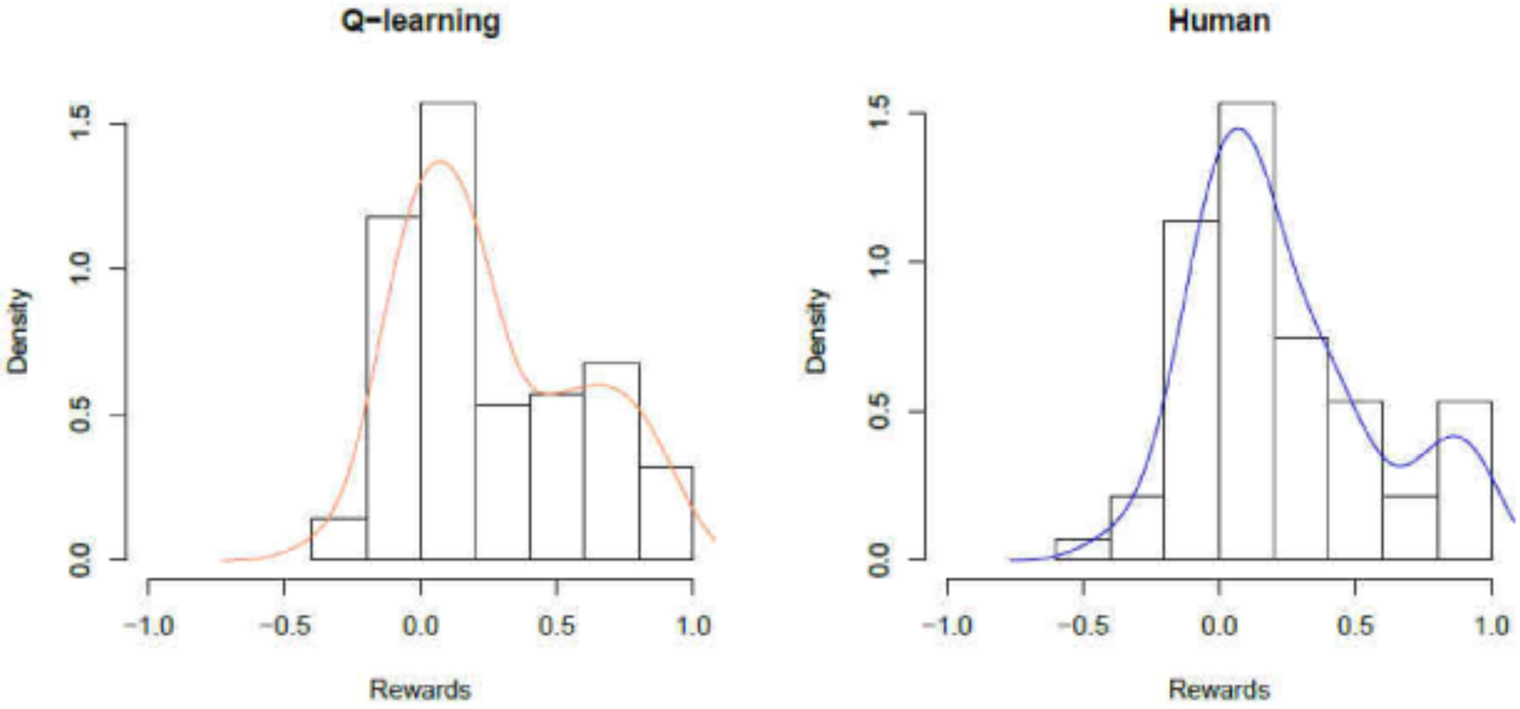
- A biological agent: humans
  - 20 humans were used in the experiment
  - A specific interface was developed for them, while the rest of the setting was equal for both types of agents.



  - http://users.dsic.upv.es/proy/anynt/human1/test.html

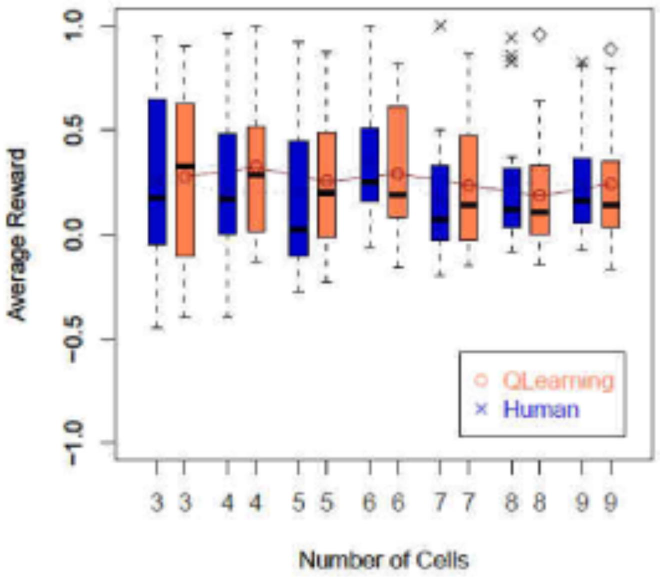
Results

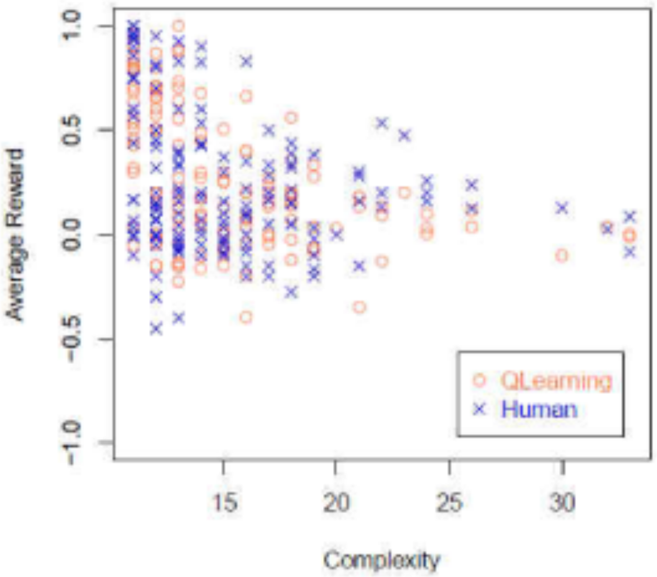
- Experiments were paired.
  - Results show that performance is fairly similar.



Results

- Analysis of the effect of complexity :
  - Complexity is approximated by using LZ (Lempel-Ziv) coding to the string which defines the environment.





  - Lower variance for exercises with higher complexity.
  - Slight inverse correlation with complexity (difficulty ↑, reward ↓).

Discussion

- Not many studies comparing human performance and machine performance on non-specific tasks.
  - The environment class here has not been designed to be anthropomorphic.
  - The AI agent (Q-learning) has not been designed to address this problem.

  - The results are consistent with the C-test (Hernandez-Orallo 1998) and with the results in (Sanghi & Dowe 2003), where a simple algorithm is competitive in regular IQ tests.

Discussion

- The results show *this is not a universal intelligence test*.
  - The use of an interactive test has not changed the picture from the results in the C-test.
- What may be wrong?
  - A problem of the current implementation. Many simplifications made.
  - A problem of the environment class. Both this and the C-test used an inappropriate reference machine.
  - A problem of the environment distribution.
  - A problem with the interfaces, making the problem very difficult for humans.
  - A problem of the theory.
    - Intelligence cannot be measured universally.
    - Intelligence is factorial. Test must account for more factors.
    - Using algorithmic information theory to precisely define and evaluate intelligence may be insufficient.

Thank you!

Some pointers:

- Project: **anYnt** (Anytime Universal Intelligence)

  http://users.dsic.upv.es/proy/anynt/

- Have fun with the test

  http://users.dsic.upv.es/proy/anynt/human1/test.html

Turing Tests with Turing Machines

José Hernández-Orallo[1], Javier Insa-Cabrera[1], David L. Dowe[2] and Bill Hibbard[3]

[1] DSIC, Universitat Politècnica de València, Spain.
jorallo@dsic.upv.es, jinsa@dsic.upv.es
[2] Clayton School of Information Technology, Monash University, Australia.
david.dowe@monash.edu
[3] Space Science and Engineering Center, University of Wisconsin - Madison, USA.
test@ssec.wisc.edu

Abstract

Comparative tests work by finding the difference (or the absence of difference) between a reference subject and an evaluee. The Turing Test, in its standard interpretation, takes (a subset of) the human species as a reference. Motivated by recent findings and developments in the area of machine intelligence evaluation, we discuss what it would be like to have a Turing Test where the reference and the interrogator subjects are replaced by Turing Machines. This question sets the focus on several issues that are usually disregarded when dealing with the Turing Test, such as the degree of intelligence of reference and interrogator, the role of imitation (and not only prediction) in intelligence, its view from the perspective of game theory and others. Around these issues, this paper finally brings the Turing *Test* to the realm of Turing *machines*.

Keywords: Turing Test, Turing machines, intelligence, learning, imitation games, Solomonoff-Kolmogorov complexity, game theory, human unpredictability, matching pennies.

Contents

1 Introduction

The Turing Test [26] is still the most popular test for machine intelligence. However, the Turing Test, as a measurement *instrument* and not as a philosophical argument, is very different to

Turing Tests with Turing Machines

José Hernández Orallo
DSIC, Universitat Politecnica de Valencia, Spain
jorallo@dsic.upv.es

David L. Dowe
Monash University, Australia
david.dowe@monash.edu

Javier Insa Cabrera
DSIC, Universitat Politecnica de Valencia, Spain
jinsa@dsic.upv.es

Bill Hibbard
University of Wisconsin - Madison, USA
test@ssec.wisc.edu

The comparative approach

Intelligence Evaluation:

- Intelligence has been evaluated by humans in all periods of history.
- Only in the XXth century, this problem has been addressed *scientifically*:
  - Human intelligence evaluation.
  - Animal intelligence evaluation.

What about machine intelligence evaluation?

Turing Test:

- The *imitation game* was not really conceived by Turing as a *test*, but as a compelling argument.
- Problems of using the imitation game as a test of intelligence.

Is there an alternative principled way of measuring intelligence?

Computational measurement of intelligence

During the past 15 years, there has been a discreet line of research advocating for a formal, computational approach to intelligence evaluation.

- Issues:
  - Humans cannot be used as a reference.
    - No arbitrary reference is chosen. Otherwise, comparative approaches would become circular.
  - Intelligence is a gradual (and most possibly factorial) thing.
    - It must be graded accordingly.
  - Intelligence as performance on a diverse tasks and environments.
    - Need to define these tasks and environments.
  - The difficulty of tasks/environments must be assessed.
    - Not on populations (psychometrics), but from computational principles.

Computational measurement of intelligence

Problems this line of research is facing at the moment.

- Most approaches are based on tasks/environments which represent patterns that have to be discovered and correctly employed.
- These tasks/environments are not representative of what an intelligence being may face during its life.

> (Social) intelligence is the ability to perform well in an environment full of other agents of similar intelligence

This idea prompted the definition of a different distribution of environments:

- **Darwin-Wallace distribution** (Hernandez-Orallo et al. 2011): environments with intelligent systems have higher probability.
  - It is a *recursive* (but not circular) distribution.
  - While resembles artificial evolution, it is guided and controlled by intelligence tests, rather than selection due to other kind of fitness.

Reunion: bridging antagonistic views

The setting of the Darwin-Wallace distribution suggests:

- Comparative approaches may not only be useful but necessary.
- The Turing Test might be more related to social intelligence than other kinds of intelligence.

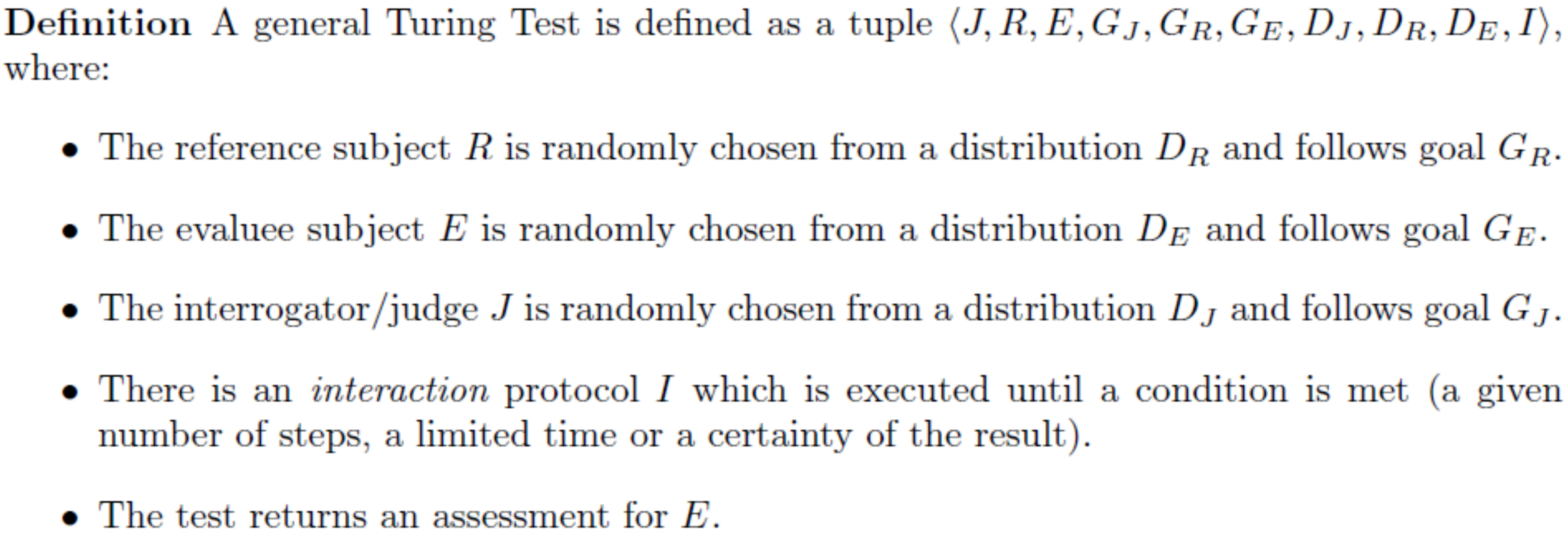
This motivates a reunion between the line of research based on computational, information-based approaches to intelligence measures with the Turing Test.

- However, this reunion has to be made without renouncing to one of the premises of our research: the elimination of the human reference.

Use (Turing) machines, and not humans, as references.

Make these references meaningful by recursion

Generalisation of the Turing Test



Definition A general Turing Test is defined as a tuple $\langle J, R, E, G_J, G_R, G_E, D_J, D_R, D_E, I\rangle$, where:

- The reference subject R is randomly chosen from a distribution D_R and follows goal G_R.
- The evaluee subject E is randomly chosen from a distribution D_E and follows goal G_E.
- The interrogator/judge J is randomly chosen from a distribution D_J and follows goal G_J.
- There is an *interaction* protocol I which is executed until a condition is met (a given number of steps, a limited time or a certainty of the result).
- The test returns an assessment for E.

Turing Test for Turing Machines

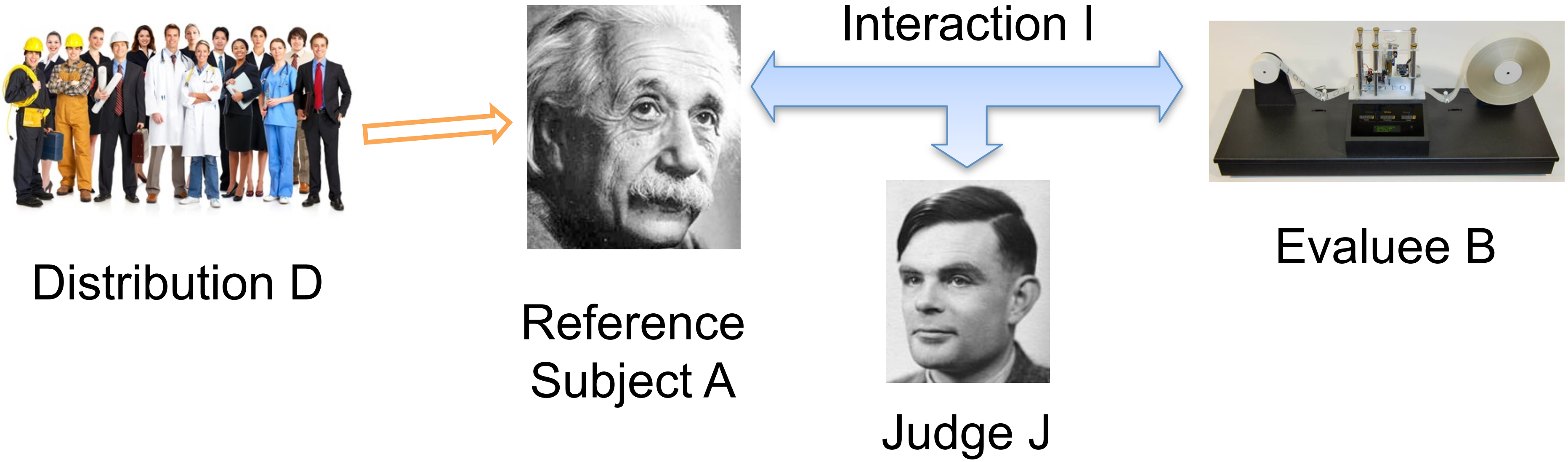
The Turing Test makes some particular choices:

- Takes the human reference from a distribution: adult homo sapiens.
- Takes the judges from a distribution (also adult homo sapiens) but they are also instructed on how to evaluate.

But other choices can be made.

- Informally?
  - A Turing Test for Nobel laureates, for children, for dogs or other populations?
- Formally? Generally?
  - Nothing is more formal and general than a Turing Machine.

The Turing Test for Turing Machines



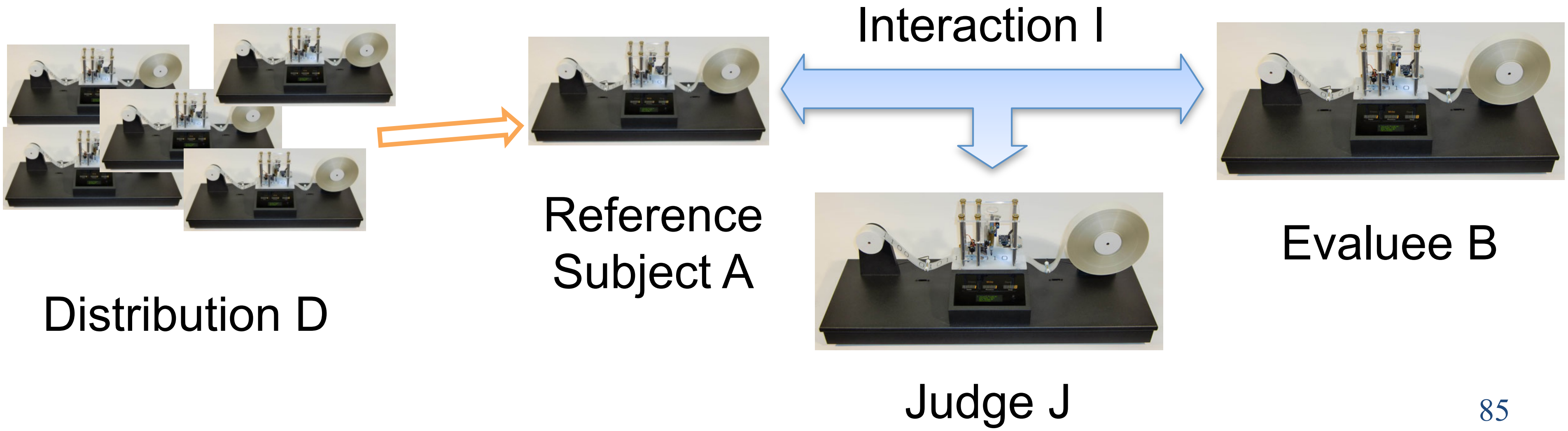
Distribution D

Reference Subject A

Interaction I

Judge J

Evaluee B



Distribution D

Reference Subject A

Interaction I

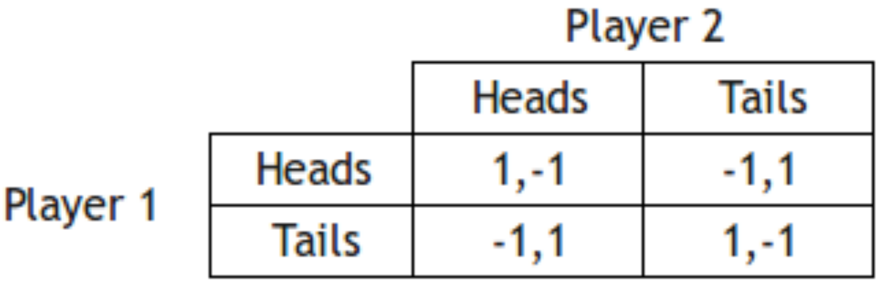
Judge J

Evaluee B

The Turing Test for Turing Machines

The simplest adversarial Turing Test:

- Symmetric roles:
  - Evaluee B tries to imitate A. It plays the *predictor* role.
  - Reference A tries to evade B. It plays the *evader* role.
- This setting is exactly the matching pennies problem.
  - Predictors win when both coins are on the same side.
  - Evaders win when both coins show different sides.



		Player 2	
		Heads	Tails
Player 1	Heads	1,-1	-1,1
	Tails	-1,1	1,-1

The Turing Test for Turing Machines

Interestingly,

- Matching pennies was proposed as an intelligence test (adversarial games) (Hibbard 2008, 2011).

The distribution of machines D is crucial.

- Machines with very low complexity (repetitive) are easy to identify.
- Machines with random outputs have very high complexity and are impossible to identify (a tie is the expected value).

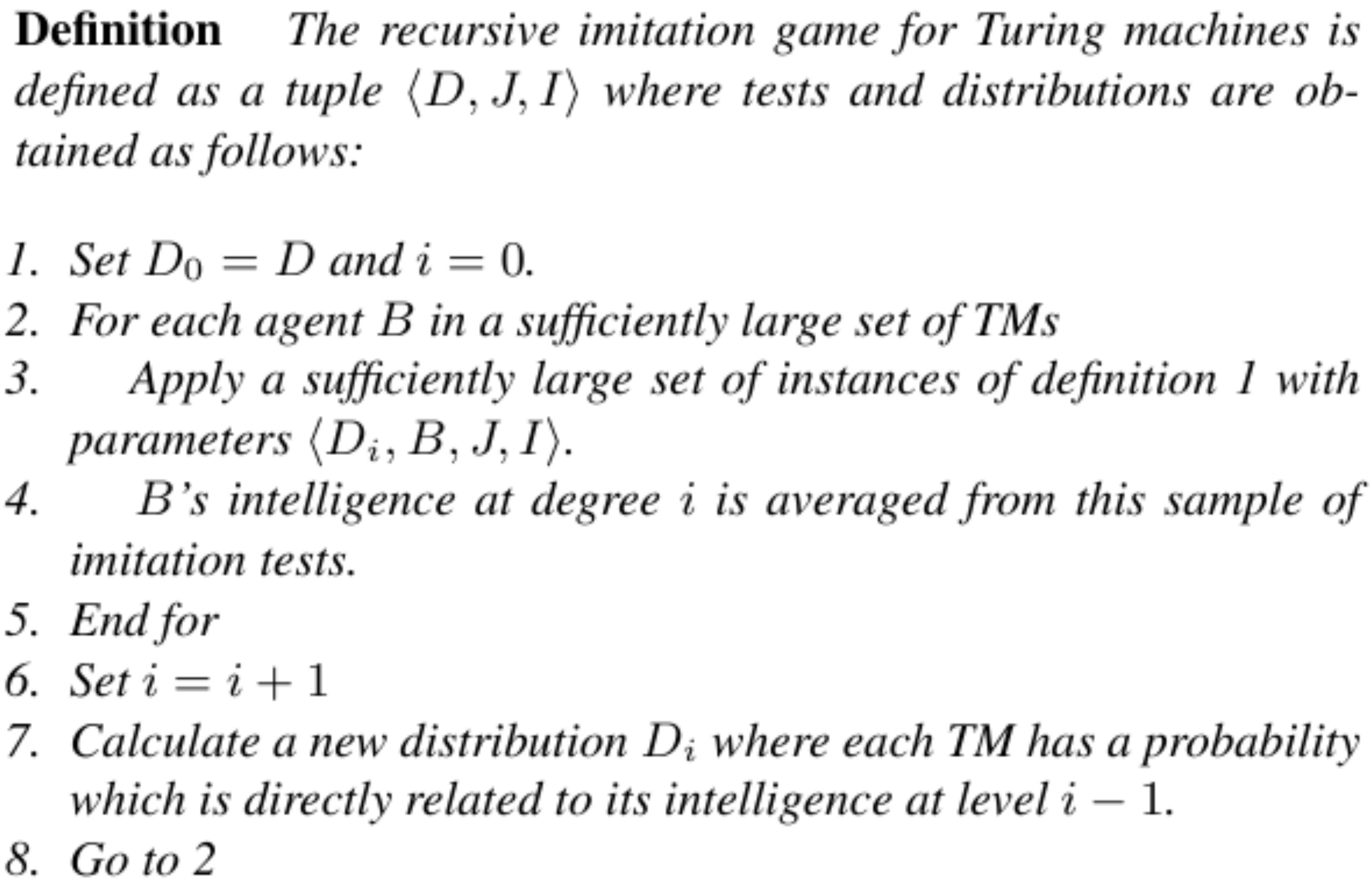
Can we derive a more realistic distribution?

Recursive TT for TMs

The Turing Test can start with a base distribution for the reference machines.

- Whenever we start giving scores to some machines, we can start updating the distribution.
  - Machines which perform well will get higher probability.
  - Machines which perform badly will get lower probability.
- By doing this process recursively:
  - We get a distribution with different levels of difficulties.
  - It is meaningful for some instances, e.g., matching pennies.

Recursive TT for TMs



Definition *The recursive imitation game for Turing machines is defined as a tuple $\langle D, J, I \rangle$ where tests and distributions are obtained as follows:*

```
1. Set $D_0 = D$ and $i = 0$.
2. For each agent $B$ in a sufficiently large set of TMs
3.     Apply a sufficiently large set of instances of definition 1 with
       parameters $\langle D_i, B, J, I \rangle$.
4.     B's intelligence at degree $i$ is averaged from this sample of
       imitation tests.
5. End for
6. Set $i = i + 1$
7. Calculate a new distribution $D_i$ where each TM has a probability
   which is directly related to its intelligence at level $i - 1$.
8. Go to 2
```

Recursive TT for TMs

The previous definition has many issues.

- Divergent?
- Intractable.

But still useful conceptually.

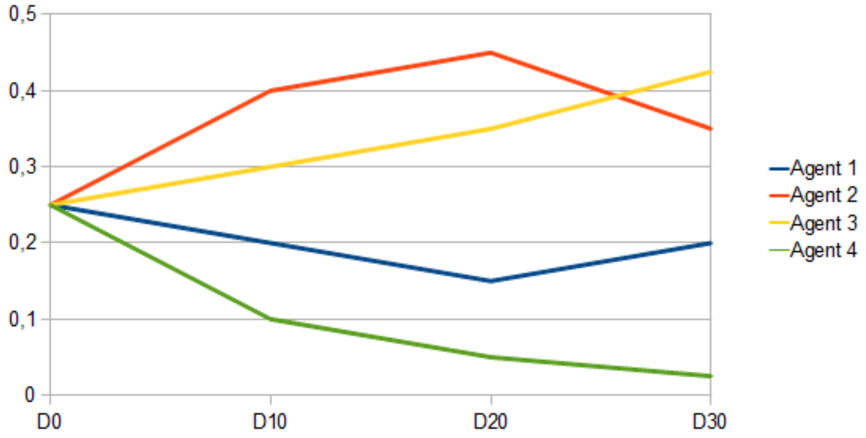
In practice, it can be substituted by a (sampling) ranking system:

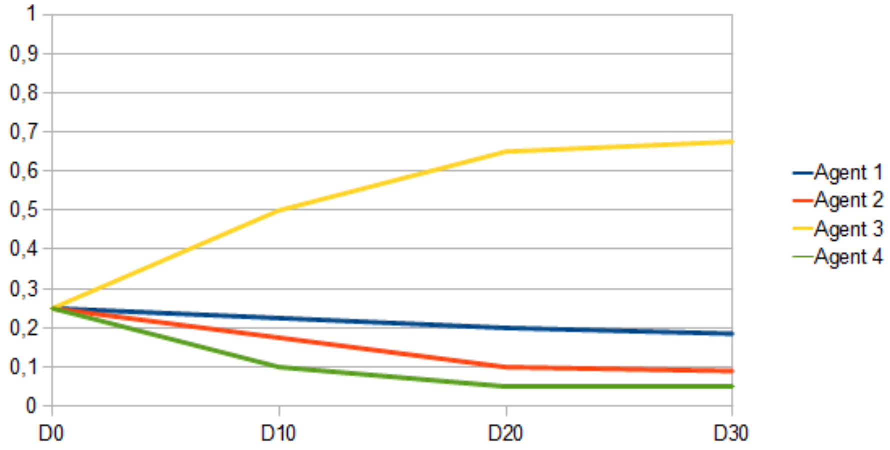
- (e.g.) Elo’s rating system in chess.

Given an original distribution, we can update the distribution by randomly choosing pairs and updating the probability.

Possible resulting distributions

Depending on the agents and the game where they are evaluated, the resulting distribution can be different.





Conclusions

- The notion of Turing Test with Turing Machines is introduced as a way:
  - To get rid of the human reference in the tests.
  - To see very simple social intelligence tests, mainly adversarial.
- The idea of making it recursive tries to:
  - escape from the universal distribution.
  - derive a different notion of difficulty.
- The setting is still too simple to make a feasible test, but it is already helpful to:
  - Bridge the (until now) antagonistic views of intelligence testing using the Turing Test or using computational formal approaches using Kolmogorov Complexity, MML, etc.
  - Link intelligence testing with (evolutionary) game theory.

Turing Machines and Recursive Turing Tests

José Hernández Orallo[1], Javier Insa-Cabrera[1],
David L. Dowe[2], Bill Hibbard[3],

1. *Departament de Sistemes Informàtics i Computació, Universitat Politècnica de València, Spain.*
2. *Computer Science & Software Engineering, Clayton School of I.T., Monash University, Clayton, Victoria, 3800, Australia.*
3. *Space Science and Engineering Center, University of Wisconsin - Madison, USA*

CQRW2012 - AISB/IA-CAP 2012 World Congress, July 4-5, Birmingham, UK

Outline

- The Comparative Approach
- Computational Measurement of Intelligence
- Reunion: bridging antagonistic views
- Base case: the TT for TMs
- Recursive TT for TMs
- Discussion

The comparative approach

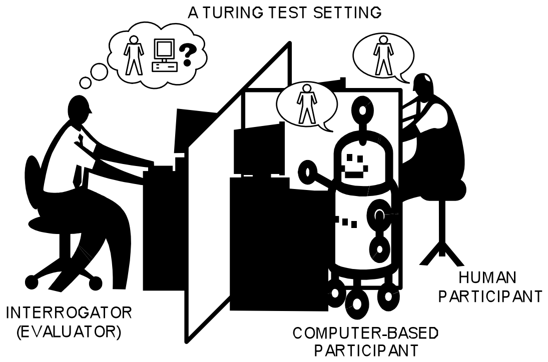
- Intelligence Evaluation:
  - Intelligence has been evaluated by humans in all periods of history.
  - Only in the XXth century, this problem has been addressed *scientifically*:
    - Human intelligence evaluation is performed and studied in psychometrics and related disciplines.
    - Animal intelligence evaluation is performed and studied in comparative cognition and related disciplines.

What about machine intelligence evaluation?

  - We only have partial approaches in some AI competitions and, of course, some variants and incarnations of the Turing Test.

The comparative approach

- Turing Test:
  - The *imitation game* was not really conceived by Turing as a *test*, but as a compelling argument.



- Problems of using the imitation game as a test of intelligence.
  - Humanity (and not intelligence) is taken as a reference.
  - Evaluation is subjective: evaluators are also humans.
  - Too focussed on (teletype) *dialogue*.
  - Not based on reproducible tasks but on particular, unrepeatable conversations.
  - Not really scalable far below or beyond human intelligence.
  - Not clear how it behaves for collective intelligence (with one teletype communicator).

Is there an alternative principled way of measuring intelligence?

Computational measurement of intelligence

- During the past 15 years, there has been a discreet line of research advocating for a formal, computational approach to intelligence evaluation.
  - Issues:
    - Humans cannot be used as a reference.
      - No arbitrary reference is chosen. Otherwise, comparative approaches would become circular.
    - Intelligence is a gradual (and most possibly factorial) thing.
      - It must be graded accordingly.
    - Intelligence as performance on a diverse tasks and environments.
      - Need to define these tasks and environments.
    - The difficulty of tasks/environments must be assessed.
      - Not on populations (psychometrics), but from computational principles.

Computational measurement of intelligence

- Problems this line of research is facing at the moment.
  - Most approaches are based on tasks/environments which represent patterns that have to be discovered and correctly employed.
  - These tasks/environments are not representative of what an intelligence being may face during its life.
  - Environments lack on evaluate some skills that discriminates better between different systems.

(Social) intelligence is the ability to perform well in an environment full of other agents of similar intelligence

Computational measurement of intelligence

- This definition of Social intelligence prompted the definition of a different distribution of environments:
  - **Darwin-Wallace distribution** (Hernandez-Orallo et al. 2011): environments with intelligent systems have higher probability.
    - It is a *recursive* (but not circular) distribution.
    - Use agents' intelligence to create new social environments.
    - While resembles artificial evolution, it is guided and controlled by intelligence tests, rather than selection due to other kind of fitness.

Reunion: bridging antagonistic views

- The setting of the Darwin-Wallace distribution suggests:
  - Comparative approaches may not only be useful but necessary.
  - The Turing Test might be more related to social intelligence than other kinds of intelligence.
- This motivates a reunion between the line of research based on computational, information-based approaches to intelligence measures with the Turing Test.
  - However, this reunion has to be made without renouncing to one of the premises of our research: the elimination of the human reference.

Use (Turing) machines, and not humans, as references.

Make these references meaningful by recursion

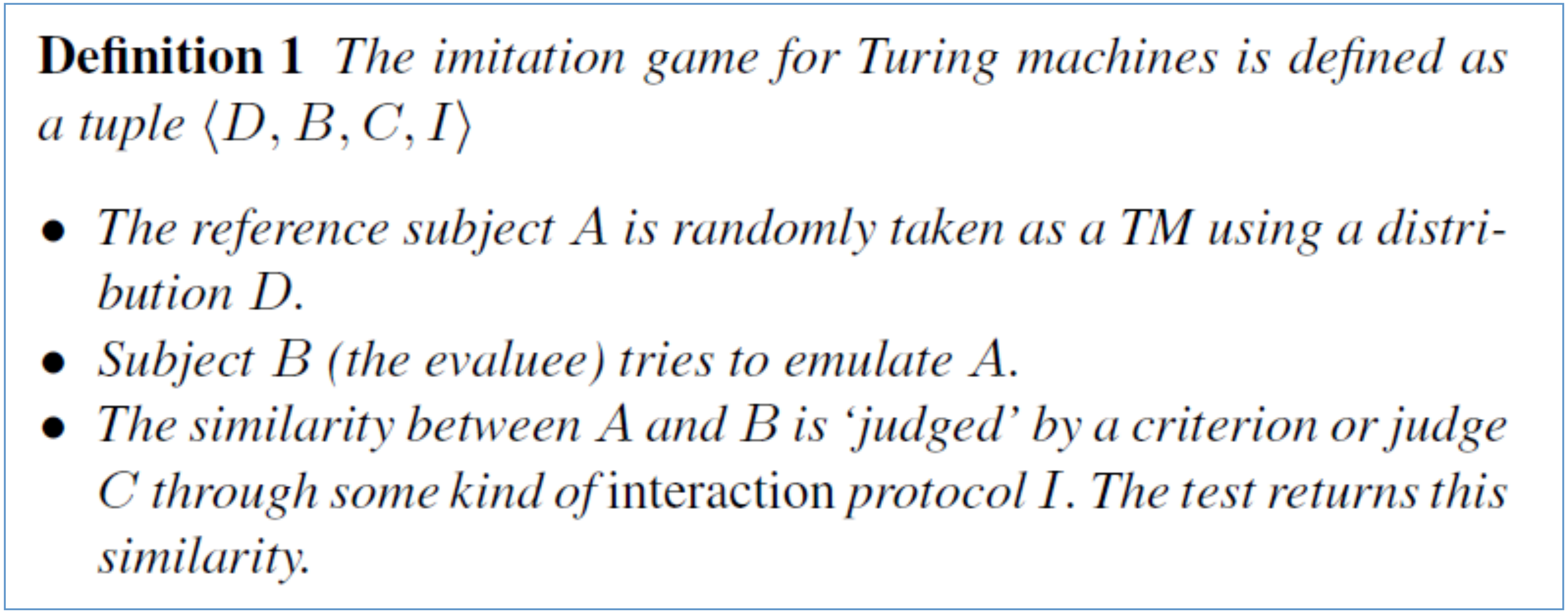
Base case: the TT for TMs

- The Turing Test makes some particular choices:
  - Takes the human reference from a distribution: adult homo sapiens.
  - Takes the judges from a distribution (also adult homo sapiens) but they are also instructed on how to evaluate.

- But other choices can be made.
  - Informally?
    - A Turing Test for Nobel laureates, for children, for dogs or other populations?
  - Formally? Generally?
    - Nothing is more formal and general than a Turing Machine.

Base case: the TT for TMs

- Let us generalise the TT with TMs:



Definition 1 *The imitation game for Turing machines is defined as a tuple* $\langle D, B, C, I \rangle$

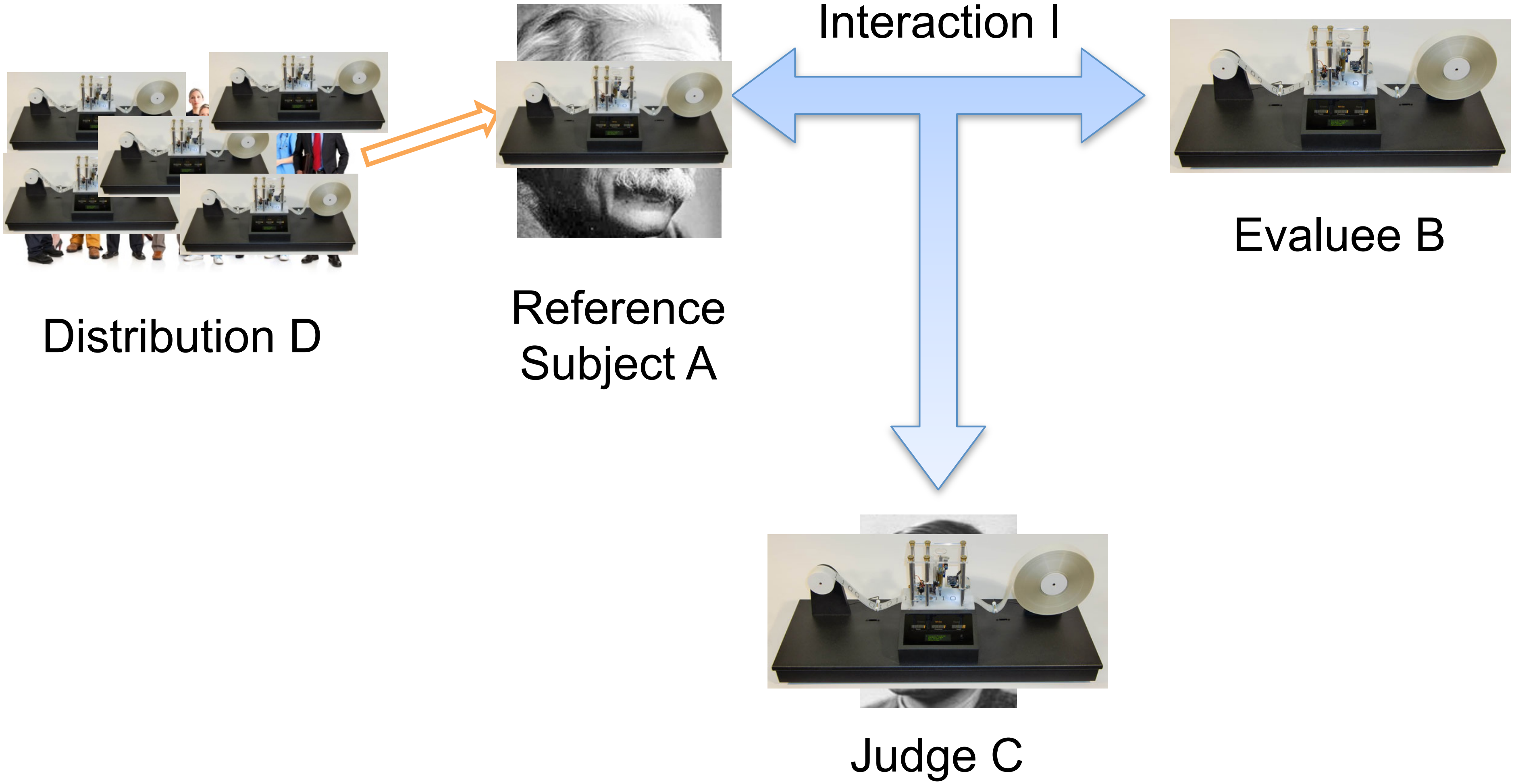
- *The reference subject* A *is randomly taken as a TM using a distribution* D.
- *Subject* B *(the evaluee) tries to emulate* A.
- *The similarity between* A *and* B *is 'judged' by a criterion or judge* C *through some kind of* interaction *protocol* I. *The test returns this similarity.*

Base case: the TT for TMs

- The use of *Turing* machines for the reference is relevant:
  - We can actually define formal distributions on them (this cannot be done for humans, or animals or “agents”).
- It is perhaps a convenience for the judge.
  - Any formal mechanism would suffice.

- It is not exactly a generalisation, because in the TT there is an *external reference*.
  - the judge compares both subjects with his/her knowledge about human behaviour.

Base case: the TT for TMs



Interaction I

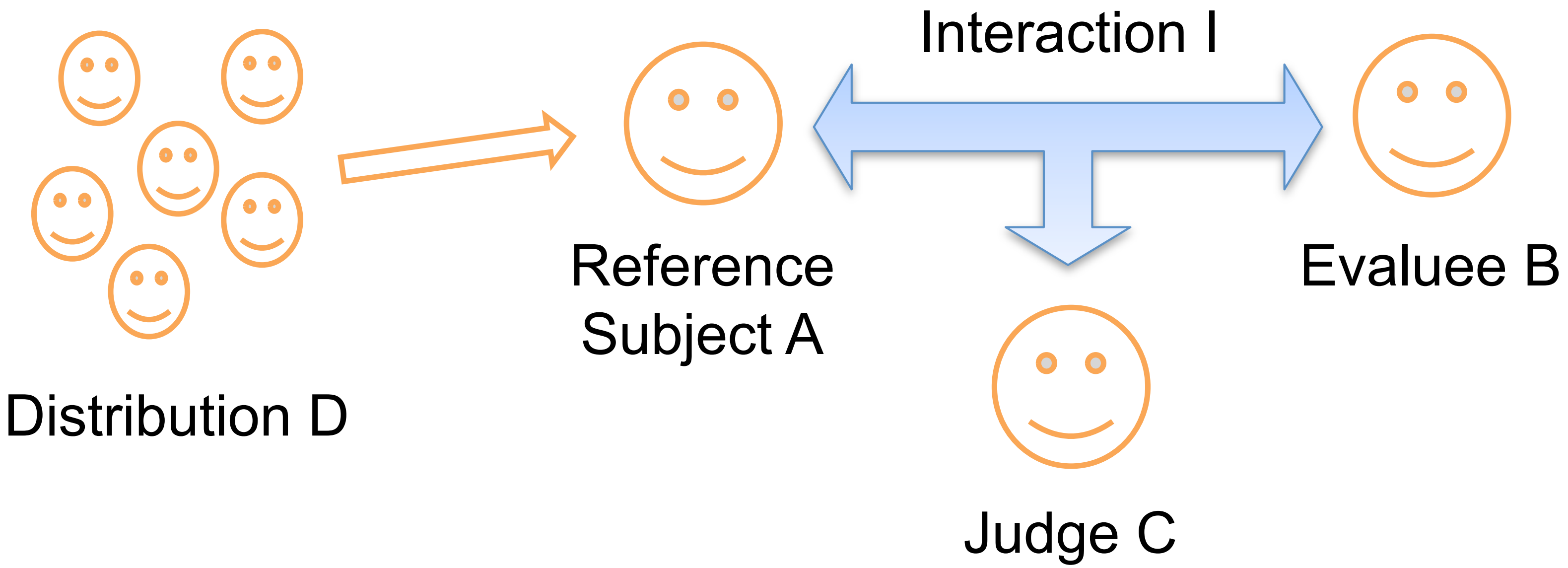
Evaluee B

Distribution D

Reference Subject A

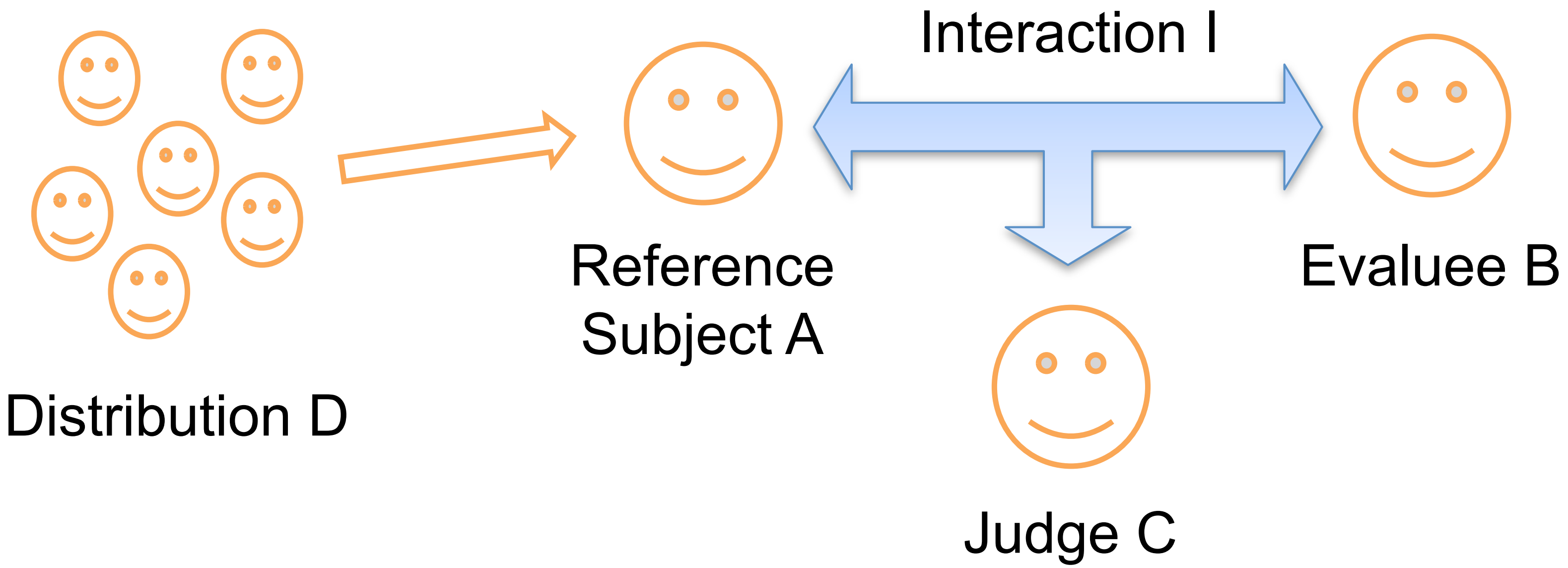
Judge C

Base case: the TT for TMs



- The C-test can be seen as a special case of the TT for TMs:
  - The reference machines have no input (they are static)
  - The distribution gives high probability to sequences of a range of difficulty (Levin’s Kt complexity).
  - The judges/evaluation just look for an exact matching between the reference outputs and the evaluee.

Base case: the TT for TMs



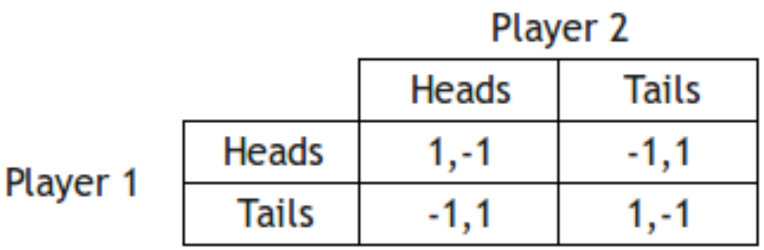
- Legg & Hutter’s Universal Intelligence can be seen as a special case of the TT for TMs:
  - The reference machines are interactive and issue rewards.
  - The distribution gives high probability to TMs with low Kolmogorov complexity.
  - The judges/evaluation just look for high rewards.

Base case: the TT for TMs

- Other more ‘orthodox’ versions could be defined:
  - Question-answer setting:
    - Judges just issue questions from a distribution (they are string-generating TM).
    - Reference A is another TM which receives the input and issues an output.
    - The evaluee learns from the input-outputs over A and tries to imitate.
  - However, the original version of the TT was adversarial.
    - Reference subjects were instructed to play against the evaluee (and vice versa). Both wanted to be selected as *authentic*.
      - However, we do not have an external reference.

Base case: the TT for TMs

- The simplest adversarial Turing Test:
  - Symmetric roles:
    - Evaluee B tries to imitate A. It plays the *predictor* role.
    - Reference A tries to evade B. It plays the *evader* role.
  - This setting is exactly the matching pennies problem.
    - Predictors win when both coins are on the same side.
    - Evaders win when both coins show different sides.



		Player 2	
		Heads	Tails
Player 1	Heads	1,-1	-1,1
	Tails	-1,1	1,-1

Base case: the TT for TMs

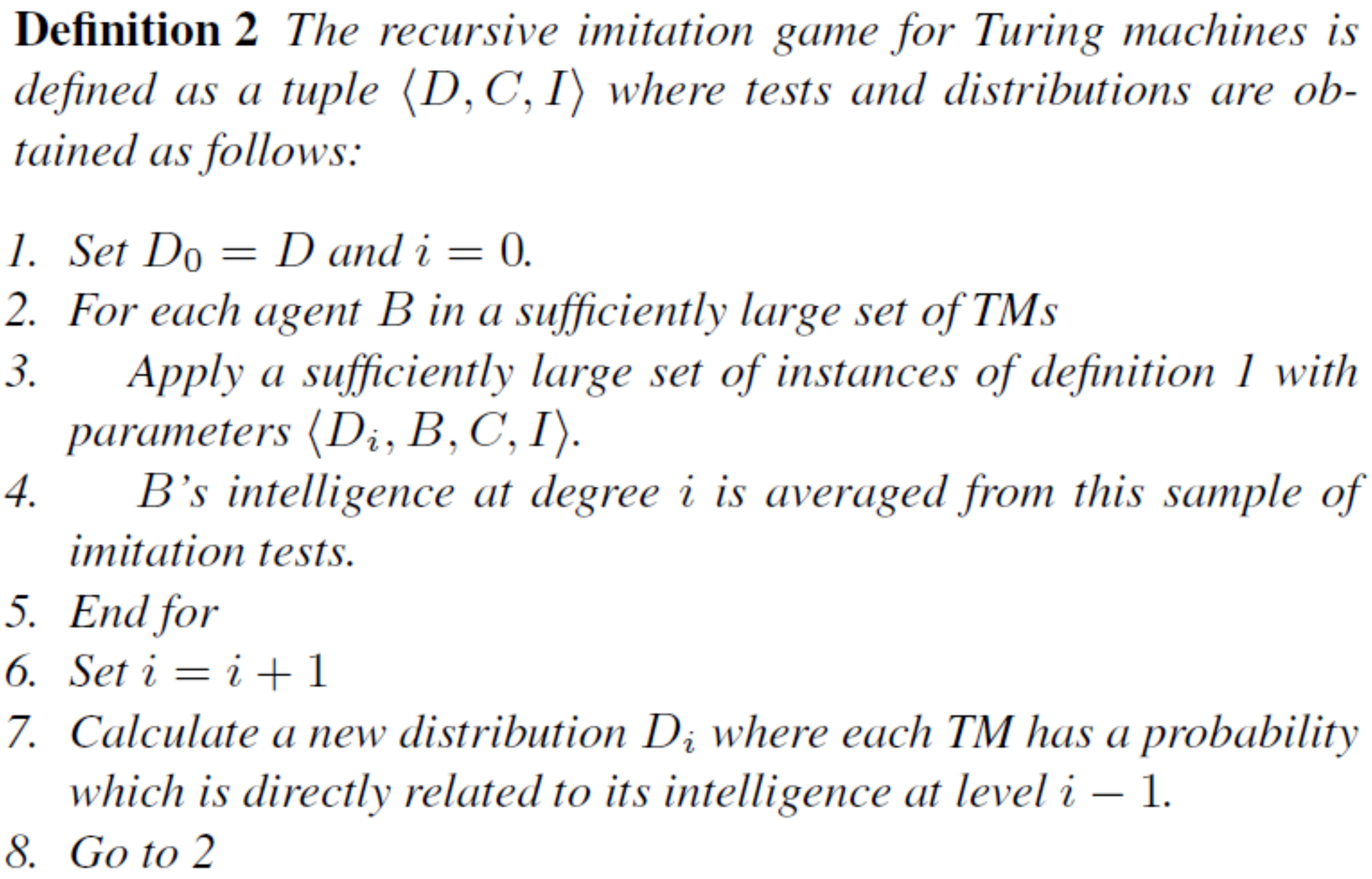
- Interestingly,
  - Matching pennies was proposed as an intelligence test (adversarial games) (Hibbard 2008, 2011).
- Again, the distribution of machines D is crucial.
  - Machines with very low complexity (repetitive) are easy to identify.
  - Machines with random outputs have very high complexity and are impossible to identify (a tie is the expected value).

Can we derive a more realistic distribution?

Recursive TT for TMs

- The TT for TMs can start with a base distribution for the reference machines.
  - Whenever we start giving scores to some machines, we can start updating the distribution.
    - Machines which perform well will get higher probability.
    - Machines which perform badly will get lower probability.
  - By doing this process recursively:
    - We get a controlled version of the Darwin-Wallace distribution.
    - It is meaningful for some instances, e.g., matching pennies.

Recursive TT for TMs



Definition 2 *The recursive imitation game for Turing machines is defined as a tuple $\langle D, C, I \rangle$ where tests and distributions are obtained as follows:*

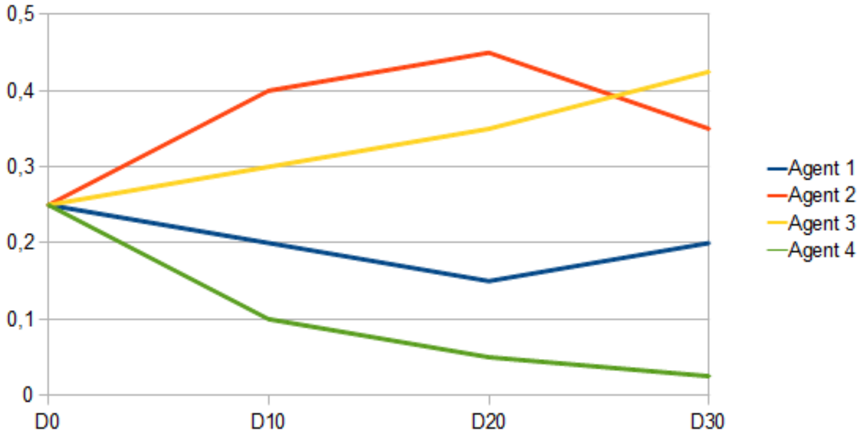
```
1. Set $D_0 = D$ and $i = 0$.
2. For each agent $B$ in a sufficiently large set of TMs
3.     Apply a sufficiently large set of instances of definition 1 with parameters $\langle D_i, B, C, I \rangle$.
4.     $B$'s intelligence at degree $i$ is averaged from this sample of imitation tests.
5. End for
6. Set $i = i + 1$
7. Calculate a new distribution $D_i$ where each TM has a probability which is directly related to its intelligence at level $i - 1$.
8. Go to 2
```

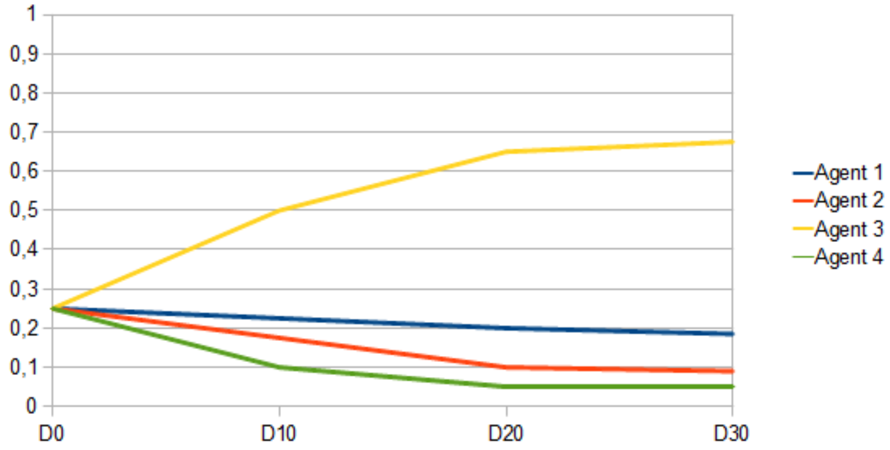
Recursive TT for TMs

- The previous definition has many issues.
  - Divergent?
  - Intractable.
- But still useful conceptually.
- In practice, it can be substituted by a (sampling) ranking system:
  - (e.g.) Elo’s rating system in chess.
- Given an original distribution, we can update the distribution by randomly choosing pairs and updating the probability.

Possible resulting distributions

- Depending on the agents and the game where they are evaluated, the resulting distribution can be different.





Discussion

- The notion of Turing Test with Turing Machines is introduced as a way:
  - To get rid of the human reference in the tests.
  - To see very simple social intelligence tests, mainly adversarial.

- The idea of making it recursive tries to:
  - escape from the universal distribution.
  - derive a different notion of difficulty.

Discussion

- The setting is still too simple to make a feasible test, but it is already helpful to:
  - Bridge the (until now) antagonistic views of intelligence testing using the Turing Test or using computational formal approaches using Kolmogorov Complexity, MML, etc.
  - Link intelligence testing with (evolutionary) game theory.

Thank you!

Some pointers:

- Project: **anYnt** (Anytime Universal Intelligence)

  http://users.dsic.upv.es/proy/anynt/