A Formal Definition of Intelligence
Based on an Intensional Variant of Algorithmic Complexity

JOSE HERNANDEZ-ORALLO

Universitat Politècnica de València, Departament de Sistemes Informàtics i Computació
Camí de Vera 14, Aptat. 22.012 E-46071, València, Spain
E-mail: `jorallo@dsic.upv.es`

NEUS MINAYA-COLLADO

Universitat de València, Facultat de Ciències Biològiques
Doctor Moliner s/n E-46100, Burjassot, València, Spain
E-mail: `ploro@uv.es`

After a brief discussion about the counter-intuitive results of a first approach like "*intelligence as the ability of universal compression*" we present a formal definition of *exception-free* description. Then we introduce a variant E of algorithmic complexity that we name *intensional complexity* with the only additional property that it allows no 'exceptions', making formal this *deep* notion in the very theory. Once defined this variant and a time-weighted version Et that we name *explanatory complexity* we retake the analogy and formulate the idea that "intelligence is the ability of *comprehension*" understanding this last term in a formal way: *explanatory* compression. Finally, we devise a test based on k-*incomprehensible* strings and we present an effective algorithm for generating them.

To dissipate its ethereal appearance, we devote most of the paper to present some computational, philosophical, psychological, psychometric and other experimental facts that support such a definition. We 'unfold' the definition to fill the gap between the formalities and the classical IQ tests. This relation to psychometrics leads us immediately to propose a framework to evaluate artificial —or not— intelligent systems. In the end we take a step forward from the evaluation into the understanding and design of intelligent systems.

0 Apologies

First, we must apologise for the title because it 'forces' to pay more attention than we are deserving. Secondly, we apologise for some unorthodox treatment, oversimplifications and partial views of the related areas we discuss. And last, we apologise for the haste we are taking for presenting it without the observation of some specialists of the different fields this article deals about. Right or wrong, we think there is no reason to keep back the current state of our theory for quick reproof or further improvement.

In the end, we think we have an intriguing view of 'intension' and what follows is, for the moment, the most we have been able to do from its interpretation.

1 Introduction

To face up a formal definition of intelligence may seem preposterous in the current complexity of the sciences which such a definition might influence. Also, there is now a considerable history of Artificial Intelligence to suggest this work is nonsense. Just beginning with the first words of its title and the authors' skill, we have, by the crudest probability, all the nays to this article.

Maybe this proposal seems more suitable in the fifties, as a response against the subjectivism and human-chauvinism of Alan Turing's Intelligent Test [Turing 1950]. Since no convincing *objectivistisc* response was given, it became a general belief for computer scientists, philosophers and psychologists that non-human intelligence is not measurable in any other way. The article also fits better in the atmosphere of the sixties, with the first works on computers making analogies and problems extracted from IQ tests. It even could have been seen in the frame of the seventies as a need to characterise Extraterrestrial Intelligence in a non-chauvinistic way [Sagan 1973].

Nevertheless the idea of an objective definition of intelligence reappeared in 1982. Also it did in its *adequate* context. One of the parents of modern information theory, Gregory C. Chaitin, proposed some directions for future research using algorithmic information theory. One of the proposals was the following one [Chaitin 1982]:

> (f) Develop formal definitions of intelligence and measures of its various components; apply information theory and complexity theory to AI.

The second part of this claim began time ago [Wallace & Boulton 1968] but it was Rissanen's MDL [Rissanen 1978] which promoted its current use. Later, other parent of algorithmic information theory [Solomonoff 1986], proposed explicitly to address directly AI using "algorithmic probability". We have not found (to our limited bibliographical knowledge) any work related to the first part of the claim (except Wolff's program that we will comment later). Considering that a formal definition of intelligence may be so influential, we suppose those who undertook it found counterintuitive results with the straightforward definition: "*intelligence as the ability of compression*".

About 1990, AI broke up its branches in different fields. One of the reasons that motivated this disruption was, in [Chandrasekaran 1990] words, that "*there is no broad agreement on the essential nature or formal basis of intelligence and the proper theoretical framework for it*". If there is a thing that has restrained the advent of artificial intelligent systems is the thought that intelligence is something completely subjective.

But precisely, and according to [Dietrich 1990], the goal of AI is to tell us what kind of computing intelligence is. In this paper we pretend precisely so. We are not giving a definition of human-like intelligence, we just pretend to give a formal definition of a computational ability essential to show intelligent behaviour. We agree with Minsky that any study of intelligence must take account of both human and artificial intelligence. Accordingly, most of the work of this paper is devoted to justify a relative short formal definition with many psychological, neurological, computational and even philosophical issues.

The links will show to be so close that we will be able to devise a human intelligence test based on our definition and —with all the cautions that a single psychometric study is considered— it correlates with the results that the Homo Sapiens obtains on IQ tests. In other words, we give a formal counterpart to *g* factor. If this is accurate, how to make operative this principle inside intelligent systems will be an important topic for research. In this sense, we do not want to stop only on the cognitive view of AI, we will revise some topics on applied AI, of how to judge the progress of intelligent systems and even how we can make them 'smarter'.

2 Previous Theories of Intelligence

Many theories have been proposed to define intelligence from many fields, e.g. philosophy, psychology, biology and, lately, computer science, and it is the capital key for many other problems of cognitive science and artificial intelligence. Most of them coincide in one thing: intelligence is an ability. All of them are informal in some extent and very different in kind and purposes. How can we define formally a thing that is so many other things at the same time?

The most practical approach is psychological: "*intelligence is the ability measured by the IQ (Intelligence Quotient) test, known as the g factor*" [Sternberg 1977]. Owing to the fact that IQ tests are made by psychologists and the philosophical popularity of the Turing test, it has been a generalised opinion that "*the intelligence of a system can only be judged by another intelligent system*".

Although IQ tests are used to corroborate analytic techniques, should we be satisfied with a definition like "intelligence is what is measured by IQ"?. Following [Eysenck 1979]'s analogy, we are not less satisfied with "temperature is what is measured by thermometers". But this is a trap because the question arises again: "What is actually measured by IQ tests?".

Apart from Chaitin's proposal others believe that an objective characterisation is possible [Hofstadter 1979]:

> It would be nice if we could define intelligence in some other way than "that which gets the same meaning out of a sequence of symbols as we do". [...] This in turn would support the idea of meaning being an inherent property.

We will in fact give inherent (intrinsic) definitions in section 4. Classical definitions lack a methodological basis that could be applicable to *compare* and *develop* artificial intelligent systems. Our proposal allows at least the first of these two things and we will see that, except Turing's conception, our definition agrees with all the other definitions of intelligence.

3 Compression and Intelligence

3.1 Notation

We will work with finite or infinite strings of *consecutive* elements from some (possibly unspecified) alphabet Ω with at least two elements. With $l(x)$ we denote the length of string x. The ith element of some string x is represented by x_i. The string x without its d last elements is denoted (awkwardly) as x_{-d}. By ϕ we denote some kind of universal function (a universal computer, a Turing Machine, etc.) and $\phi(p)$ is the result of executing the program p by ϕ. We use the word *bias* to refer to a *specific* universal computer, i.e., some descriptive framework β which is constituted by an alphabet Ω_β, some operations Θ_β, and a machine ϕ_β with some resources. x^* (or x^+) express any program for x in an unspecified *bias*. The empty string is ε. We say that y is a substring of x iff $\exists z,w$: $x=zyw$.

3.2 Algorithmic Complexity

Algorithmic Complexity (commonly referred as Kolmogorov Complexity) was independently introduced (with different reasons and directions) by R.J. Solomonoff, A.N. Kolmogorov and G.J. Chaitin. Since we will just give the only necessary definitions, see [Chaitin 1992], [Watanabe 1992] or [Li & Vitányi 1997] if you are not familiar with the theory. The term Algorithmic Complexity is generally associated with its 'purest' variant, based on Minimum Length Encoding (MLE) or Minimal Description Length (MDL), i.e., the shortest string that, taken as an algorithm, produces exactly the original string. Formally,

DEFINITION 3.1

$C_\phi(x) = \min \{ l(p): \phi(p) = x \}$

Since any universal machine ϕ_A can emulate any other universal machine ϕ_B, the value of $C(x)$ is said to be machine-independent to a small constant, that is usually expressed as O(1).

DEFINITION 3.2

$C_\phi(x|y) = \min \{ l(p): \phi(p,y) = x \}$

Straightforwardly we have $C(x) = C(x|\varepsilon)$, $C(x|x) = O(1)$ and $C(x|y) \leq C(x) + O(1)$. This is the *conditional* version and it will be of great importance later to clarify some intrinsic features of interdependence and separability in information.

DEFINITION 3.3

$K_\phi(x) = \min \{ l(p): \phi(p) = x \}$

is similar to definition 3.1 but forcing ϕ to be a *partial recursive prefix function*, i.e., if $\phi(x) < \infty$ and $\phi(y) < \infty$, then x is not a proper prefix of y.

K has some additional advantages over C, and is often considered the *standard* algorithmic complexity or Kolmogorov Complexity [Li & Vitányi 1997].

3.3 Universal Compressors

A universal compressing algorithm is a compressor that gives the minimal description for every string in a given descriptive mechanism. Usual compressing algorithms are not universal compressing algorithms. The reason is not only that C and K are not computable: it is formally impossible even to approximate to the optimal compression attainable by computers [Kieffer & Yang 1996] which suggests compression is very dependant and restricted to the bias used.

The general idea can be still profited if we limit the time in a *computable* universal compression algorithm.

DEFINITION 3.4

Enumeration Algorithm:

Given a bias B and an input string to compress x run all the possible programs p over B selected in length increasing order with a time limit t. Stop when we have $\phi_B(p) = x$.

It can easily be proved that this algorithm can always find the program with length equal to the time-bound complexity $K(x,t)$ (see [Li & Vitányi 1997]). Depending on the limit time t of the description, the algorithm varies from intractability to *tremendous* intractability.

3.4 Intelligence as Compression

Compression —or the broader sense of principle of economy— is omnipresent in different fields of computer science and cognition. This has suggested a new trend, seeing "computing as compression", led by Wolff's SP theory [Wolff 1995].

> All kinds of computing and formal **reasoning** may usefully be understood as information compression by pattern matching, unification and search.

Despite the fact that principles of compression of information are present in brains and nervous systems, artificial neural networks, knowledge systems, machine learning, etc., the essay "intelligence as the ability of universal compression" yields soon many problems. It does not account either for some classical definitions of intelligence (e.g. ability of explanation) and it does not bring hints for smarter systems. Just the idea "*statistical induction as compression*" has been successful [Li & Vitányi 1997].

3.5 Induction, Learning and MDL

Traditionally, reasoning is seen as the trait that distinguishes us from the rest of the animal world. Since Aristotle, deduction was granted the prominence (when no the exclusivity) of reasoning. The works of the philosophers Hume, Kant and Bacon gave an important role to inductive reasoning. More recently the works of [Popper 1968] and [Kuhn 1970] have brought some new ideas in the context of philosophy of science. Recently, deduction has been recognised by some authors as a special case of induction. Learning is the common and broader view of induction.

Induction, in the sense we will use throughout this paper, is the process of theory abstraction from facts. Since this is the problem usually faced by a scientist, induction has been known as the "logic of discovery".

Broad ended his book "The Philosophy of Francis Bacon" [Broad 1926] with the famous words: "*Inductive Reasoning... the glory of Science... the scandal of*

Philosophy". The things are not going better until the principle of economy has come strenuously into play. The role of simplicity is clearly represented in many works in this century [Barker 1957]. But it is attributed to William of Ockham 1290?-1349? this recurrent theme in philosophy of science and induction:

Occam's Razor Principle: "if there are alternative explanations for a phenomenon, then, ***all other things being equal*** [1]*, we should select the simplest one".*

This principle was rejected by Karl Popper because he said that it had no sense because there *is* no objective criterion for simplicity. He proposed [Popper 1968] instead the concept of *verisimilitude* as the level of agreement with the facts. A theory t_1 has more verisimilitude that t_2 if t_1 implies as many true observational sentences as t_2 and has less false observational sentences (exceptions). But algorithmic complexity $K(x)$ is an objective criterion for simplicity. This is precisely what R.J.Solomonoff proposed as a 'perfect' theory of induction, in [Li & Vitányi 1997] words. Algorithmic Complexity inspired J. Rissanen in 1978 to use it as a general modelling method, giving the popular MDL principle [Rissanen 1978]:

Minimum Description Length (MDL) principle*:*
The best theory to explain a set of data is the one which minimises the sum of: the length, in bits, of the description of the theory; and ,the length, in bits, of data when encoded with the help of the theory.
Then, we enclose the exceptions, if any.

This principle is not computable in general and it can only be used if we restrict the descriptive mechanism. The first justified reason to use the MDL principle is to avoid over-generalisation. But MDL is used mainly for another reason, "*the shorter the hypothesis the more predictable it is*". This has been proved for *statistical* prediction.

An oversimplification of the bias has self-defeating effects. For instance, if we select a very restricted bias, the MDL principle will be easy to apply, but it will require that the bias had been adequately selected by a person to operate for a specific problem class. If we want more flexibility in the bias, it has to be extensible, introducing new *synthesised* concepts.

3.6 Machine Learning and Prediction

Automated Learning has provided some techniques to obtain rules from experimental data. From these rules a theory is built up to predict future experiences. Some of the formal models gather some of the ideas about compression we have presented. Soon [Solomonoff 1964], it was recognised that the unsupervised learning of a grammar from raw data may be understood as information compression. The general field of machine learning has its more computational and formal part in the area of COmputational Learning Theory (COLT).

With the ideas of compression and with the aim of learning and prediction there were presented different models of learning: identification in the limit [Gold 1967], PAC model [Valiant 1984], Query-Learning [Angluin 1988]. Although the other models have brought learning to tractability, it is identification in the limit "*the model that embodies all the salient features of learning by example, and nothing else*" [Freivalds et al. 1995]. Identification in the limit is based on Popper's conception of the growth of knowledge. The idea is just obtaining the MDL in the limit, and then the enumeration algorithm we have presented can be used. The model presented in [Shapiro 1981] does precisely so, except that it reuses the old hypothesis instead of starting from scratch (to make the idea feasible).

Some recent different approaches [Rivest and Sloan 1994] are beginning to question that the preceding models were sufficient to comprise some learning tasks. The most interesting objection is the view that learning is an *incremental* process [Abe 1997]. The new goal is not necessarily to achieve the highest accuracy at the end of a learning session, but to maximise the cumulative benefits obtained throughout all the session. But is there any principle to achieve the maximum reliability during incremental learning?

It is said that the shorter the hypothesis, the better the prediction. So, MDL seems the first candidate (and so it has been used), but we can show that it is not the better option for incremental learning (although it is a good one) because after a counterexample n, the MDL nth hypothesis generally remakes to the same n−1th hypothesis just adding the exception. In this way, the hypothesis gets patched for a time until the length of coding the exceptions forces the appearance of a new radical hypothesis. In conclusion, the more tolerant we are with exceptions the less the accuracy in incremental learning. This is Kuhn's view of changing paradigms in philosophy of Science. MDL has another problem, it is usually not suitable for short finite strings, because exception-based descriptions are shorter. Let us see these three problems in an example:

EXAMPLE 3.1

Given the finite sequence: x = 1,2,3,5,7

we can guess some short hypotheses or descriptions for x, simply "the sequence 1,2,3,5,7", or "the first four odd numbers and the number two, ordered", or "the first three natural numbers and

[1] Some of the following sections originate on our thought that the following MDL principle ignores this.

the number 5 and 7", or "begin with numbers 1,2. The following three are the sum of their preceding two, but decrementing the last one" or "the first five numbers which are dividable only by 1 and itself". Almost everybody would select the last hypothesis as the more explanatory. But it is not the shortest one. But it is the shortest *exception-free* hypothesis. In fact, it is able to *predict reliably* the following value (11) because using that description, the right half of x reinforces the hypothesis while the others are weakened.

An immediate critique to this example is that the probability of 11 as being the correct answer is the same (or less according to the MDL) and it is only the assumption that the concept of prime is well known by humans which increases *our* probability of guessing 11. We think this is not accurate in this case because the shortest description "the first four odd numbers and the number 2" has an exception which 'perverts' the hypothesis, although it would be the 'elegant' program in Chaitin's LISP. In general, MDL has been used successfully because the strings are long enough or the bias has been selected to ban *exceptions*.

Exceptions are useful to memorise, to describe, to predict statistical problems, but they are not suitable for a *robust explanation*. So, how can we eliminate exceptions? But wait a moment, *what is an exception*?

4 Intensional Complexity

In mathematics, there are two kinds of descriptions: by extension and by intension (or by comprehension). But how can we know if a definition by intension 'hides' some extensionally defined information? Is it possible to know whether a definition is intensionally pure? Now, that is the question.

4.1 Natural Partitions

Algorithmic Complexity talks about the complexity of given strings, but who splits up these sequences?

EXAMPLE 4.1

Given the sequence: $x = 1,2,3,4,...,m,333,..^{\text{n times}}..,3$

we *intuitively* see that there is no *relation* between the *first* and the *second* part.

We say informally that y is a subsequence of x iff it is *easily recognisable* from x. Formally,

DEFINITION 4.1

y is a subsequence of (or easily recognisable from) x, denoted $y \angle x$, iff $C(y|x) < C(y)$.

Being $C(y)$ the algorithmic complexity of y (the length of its shortest description) and $C(y|x)$ the conditional complexity of y, given x.

The meaning of subsequence is much more than a sequence that is easily extractable in a string. It means any string that is easily described as a part —or a simple function of a part— of another. Since the concept of subsequence is rather lax, we normally understand y as easily recognisable from x when $C(y|x) << C(y)$. For instance, given the string x = "abcbcdcdedef" we have the subsequences "abcbcd", "bbdddf", "bcdcdedefefg" but not "abcd" or "xazabb".

DEFINITION 4.2

A *covering* or *recoverable* partition s of a string x is a set of subsequences $\{x_{s1} ; x_{s2} ; x_{s3} ; ... ; x_{sp}\}$ holding $C(x|s) \leq O(\log(l(x)))$

Since a partition is constituted by subsequences, they are easily recoverable from x. Inversely, since it is a covering partition, x is easily recoverable from s. There is no limit in the number p of subsequences or the length of the subsequences, so there are always infinite many covering subsequences. If we restrict also $l(s) \leq l(x)$ we have the notion of *low-redundant covering partition*. Every string has at least a *low-redundant covering partition* $s = \{x\}$. The proof is straightforward, since $C(x|s) \leq O(\log(l(x)))$ and $l(x) \leq l(x)$.

A partition should be better made using the prefix-free version of Algorithmic Complexity, i.e., $K(x)$. With this, we avoid the need to include some additional information to separate the strings of a partition (which is represented by a ';' in our notation).

The previous example 4.1 has at least two *low-redundant covering* partitions $s = \{ x \}$ and $s' = \{3,3,3..^{n \text{ times}}..,3 ; 1,2,3,4,..,m \}$. We have that $l(s') \leq l(x)$ and $C(x|s) \leq l$("take the second string of s' and then the first string") $= 1 < O(\log(l(x)) = O(\log(m+n))$.

The notion of low-redundant covering is not still suitable for our purposes. We have derived [Hernández-Orallo 1997] a *natural* partition from it that it is, to some extent, equivalent to the following definition.

4.2 Exception-free Descriptions

The next approach tries to avoid exceptions identifying them *in the description mechanism*. Informally, "*an exception is something we can take apart from a string so leaving its description much simpler respect the magnitude of the length of the elements removed*". An easy solution could be just not allowing some kind of *quoted exceptions* in the description mechanism, but as we saw in Example 4.1 there are no exceptions, only an 'arbitrary' concatenation. A deeper observation of this leads us to an *uncomfortable* definition:

DEFINITION 4.3

A description p of a string x is *exception-free* denoted $\Delta(p)$ if and only if:

$$\neg\exists y, y \angle x \wedge \exists p' : p' = subp(p) \wedge \phi(p') = y \wedge$$
$$\wedge\, l(p') < gain(p', p, x)$$

and being $subp(p)$ a subprogram of p and,

$$gain(p', p, x) = l(p) - \frac{\left[l(x) - l(y) + C(y/x)\right]}{r}$$

I.e., a description is *exception-free* if it does not exist a subdescription that produces almost the whole string, i.e., there is not a reduction in the *description* that could be greater that the corresponding reduction in the *strings*. The relevance of being subdescriptions is that every *long* string has subsequences easily compressible.

The hardness resides on the checking of whether p' is a subprogram of p since this depends on the descriptive mechanism ϕ. Also, it is difficult (e.g. is "the first n natural numbers" a subdescription of "the first 2^n numbers"?). A more formal definition of *subp* could be just definition 4.1. Now we realise that we are talking about descriptions of descriptions, and we discover the scaling of "abstractions of abstractions". It would be more accurate to use r to determine the "*exception degree*" or inversely "purity" of a description. For the moment we will use the a threshold r=1 to adjust a 'syntactical' approximation for a specific bias.

With the previous definition you can check[2] that:

EXAMPLE 4.1 (REVISITED)

Given the string $x = 1,2,3,4,...,m,333,..^{\text{n times}}.,3$

The description p = "the first m natural numbers followed by n 3's" is not *exception-free* (excluding the case $m = n$).

EXAMPLE 4.2

Given the string x=1,2,3,4,5, ..., m

The description p="the first m natural numbers" is *exception-free* for each *arbitrary m* (see next one).

EXAMPLE 4.3

Given the string x=a,a,..^{m}.,a being $m = 2^n + 2^l$

The description "$2^n + 2^l$ a's" is exception-free only when n and l are relatively far.

EXAMPLE 4.4

Given the string x = 1,2,3, ... , n, m.

The description "the n first natural numbers followed by number m" is obviously not exception-free when $m < n$.

EXAMPLE 4.5

Given the string x = 1,2,3,5,7

is exception-free for a bias B_1 where the concept prime is defined (or definable) and not exception-free for a bias B_2 where the concept prime is not.

The last example shows the difference between restricted bias and constructive bias where the system could handle every possible *definible* concept.

After the example, if we take a look at definition 4.3, we see that it *requires the comparison between programs and the data they describe. Since a jump between levels is required, only high-order bias will be able to detect exceptions —in its broad sense—.*

The preceding definition expresses a more important idea: whenever we can find an exception-free description we see that is intensional, i.e. it is the one that would be selected if we think that it must have a *sense*[3].

4.3 Intensional Complexity

DEFINITION 4.4

The Shortest Intensional Description p of a string x, denoted SID(x), is defined as follows:

SID(x) = argmin { $l(p) : \phi(p) = x \wedge \Delta(p)$ }

With this definition we can formulate the:

Shortest Intensional Description (SID) principle:

The best theory to explain a set of data is the one that minimises the length, in bits, of the description of the theory; and contains no implicit or explicit exceptions.

This idea of *implicit* exception tries to emphasise that the concept of exception is deep, and it follows from the preceding definition 4.3. Due to the non-computability of SID (as the MDL principle) it may be understood as a limit to come close. Let us see a hint of application:

EXAMPLE 4.6

Given x = aaaaaaabaaaaaaaaaaaaaabaaa

and two descriptions.

p_1= Output 26 a's.

Two exceptions: b at 8 and b at 23

p_2= let n = 7 . Repeat up to position 26

[2] See [Hernández-Orallo 1997a] for proofs.

[3] Regarding example 4.3 we may think that the string seems a whole but the length is an important trait to consider its 'separation' into two strings.

Output n a's and 1 b. Let $n = n \cdot 2$

p_1 is shorter (better according to MDL) but p_2 is intensionally purer (better according to SID).

Comparably we have,

DEFINITION 4.5

The *Intensional Complexity* of a string x, denoted $E(x)$, is defined as follows:

$$E(x) = l(\mathrm{SID}(x)) = \min \{ l(p) : \phi(p) = x \wedge \Delta(p) \}$$

Informally, this definition comes to say that the explanatory complexity of a string x is the shortest description which has no exceptions. There are strings that seem to have exceptions, e.g. x = 1, 2, 3, 4, 5, 6, 7, 342, 8, 9, 10 but it cannot be proved that a very long explanation would ever be found to give *placidly* the string x. Then $E(x)$ is not computable (like C) and it can be greater than $l(x)$ (not like C).

We also can affirm that "*the SID principle is the right principle to achieve Hypothesis Reliability during Exact Incremental Learning using an arbitrary and unknown probability distribution*". Informally, the nth hypothesis (given n+1 examples) is k-reliable iff the hypothesis has remained unchanged during the m last examples, being k=m/n. For n great, the use of exception-free hypotheses gives more probability that the last hypothesis had been older because they are sought sooner since no exceptions are allowed. This is, in the end, a formal view of Popperian methodology.

4.4 Explanatory Complexity

Most definitions we have been giving in this section are not computable. The reason is that there can be short descriptions whose computational cost would be so high, and other descriptions, taken as algorithms, do not terminate. The known non-halting problem makes that we cannot discern between both cases. The usual solution to make them computable is to restrict some resource of the description (time or space required for the program). We introduce a variant weighted on time.

DEFINITION 4.6

The *Explanatory Complexity* of a string x, denoted $Et_\beta(x)$, is defined as follows:

$$Et_\beta(x) = \min \{ LT_\beta(p) \colon \phi_\beta(p) = x \wedge \Delta(p) \}$$

We have chosen $LT_\beta(p) = l_\beta(p) + \log \mathrm{cost}_\beta(p)$ —the same weighing as [Levin 1973]'s Kt— because it provides a good compromise between space and time, but another relation could be tuned.

But Et_β is not computable in general, either. Thus, has definition 4.6 any utility? We could have used a time-bounded version to make it computable but the result is then a partial function. There are good reasons to choose a time-weighted definition of the best explanation. The intuitive view of explanation entails that the hypothesis can be *explained* to others. Winston [Winston 1992] says: "*if you want to understand a concept, try explaining it to someone else*". At the moment a system has to *tell* or communicate the explanation to other system (or internally work with it), there are two important topics: the space of the discourse and the time the system will need to relate it.

Accordingly, from definition 4.6 we could define SED (Shortest Explanatory Description) instead of SID. Obviously, $Et_\beta(x) = l_\beta(\mathrm{SED}_\beta(x))$. People, and Science, expect that nature has underlying mechanisms that emerge "quickly" in our observations. In this sense, SED is preferable over SID and over MDL.

If we regard the difficulty of finding an explanation for an information (to comprehend it), it would be more accurate to talk about levels of comprehensibility:

DEFINITION 4.7

A string x is *k-hard* (or *k-incomprehensible*) in a bias β iff k is the least positive integer such that $Et_\beta(x) \leq k \cdot \log l(x)$.

This will be the formal counterpart to our notion of hard-to-learn good explanations. In our sense, a *k-incomprehensible* string with a high k (difficult to comprehend) is different (harder) than a *k-compressible* string (difficult to learn) [Li & Vitányi 1997] and different from classical computational complexity (slow to compute). Calculating the value of k for a given string is not computable in general. Fortunately, the converse, i.e., given an arbitrary k, calculating whether a string is *k-comprehensible* is computable. We just generate the descriptions of x that have only $LT_\beta \leq k \cdot \log l_\beta(x)$, and check if there are at least one that is explanatory. This is the reason why we say that our definition of intelligence of section 5 is computable, or more exactly, the generator.

4.5 Induction, Complexity and Intrinsic Explanations

Algorithmic Complexity and its corresponding MDL principle would yield the explanatory descriptions if the sample is long enough (or even infinite) to make worthy the program with long concepts (e.g. the description of prime). There have been proposed many variants of Algorithmic Complexity to account for a fact: Kolmogorov Complexity measures the amount of information but not the complexity to understand them. They usually express better the complexity of physical systems *per se* but not the difficulty to model them. For instance, none of the variants has been assumed for systems theory as a formal definition for the complexity of a system. One of the factors that [Flood 1987] points out in the complexity of modelling a system is "no holonomicity" (holo= all, nomic= law), which means that not all the parts follow the rule. This view of complexity, although it is not formalised under Algorithmic Complexity, penalises exceptions, and it is similar to our unbounded version E.

Also, Algorithmic Complexity would not answer the following question to Mill [Mill 1843]:

> Why is a *single* instance, in some cases, sufficient for a complete induction, while in others myriads of concurring instances, without a single exception known or presumed, go such a very little way towards establishing a universal proposition?

An easy solution for this is that induction is a subjective matter that depends on which is the previous knowledge of the system. To respond to this rationale, we will see again our famous example:

x = 1,2,3,5,7,11,13,17,19,23

Using the MDL principle it will be described usually as a series of odd numbers with exceptions (+2, -9, -15, -21) because the description would be too long if we have to include our notion of prime inside it. Let us recall that since we have not seen much compression, the string is considered random, and therefore, not learnable, a result that is usual for short strings.

Using SED, we select the description x^* = "the first 10 primes" (being prime a large concept), even though it is longer. But now the length of this SED implies that the string x is quite hard, that is hard-to-learn. In fact, people which cannot multiply and divide would never *comprehend* the series. But if we have the subconcept of prime, the learnability of x is easy. This can explain why some subconcepts are necessary or useful to make some hard-to-learn concepts easier to learn.

Given the sequence x = "aabbbccdddee", 'f' is usually predicted. Followers of MDL will answer quickly: it can be compressed. But a short string is only compressible if the bias is prepared for some *kinds* of strings. Even universal biases cannot compress this string because it is not *worthy*. This is due to the fact that the term O(1) which appears in K and C is a short constant but it is greater than $l(x)$ or the information of everyday problems. It is like normal compressors, that do well on some strings but consequently wrong on others. Thinking in MDL is assuming bias rigidity.

Next, to the question of selecting the bias, many philosophers and scientists would have argued that this sequence has some patterns that are similar to those common in nature —'shaping' their bias— and this is simply the reason why people usually predict f. But this *empirical* point of view is not correct because we have given a function SED that will give us that as the best hypothesis. This hypothesis would be somehow similar to "Beginning with a, (1) print twice this symbol, print three times the next symbol and goto (1) till position $l(x)$". This can be modified to "Beginning with a, (1) print twice this symbol, print three times the next symbol and goto (1) till position $l(x) + 1$". And, as you can see in the appendix it is easy to implement an algorithm that gives an answer for it.

Summing up, [Whewell 1847] coined the term *consilience* to comprise the relevant basics in scientific theories: prediction, explanation and unification of fields. In this sense, we have given a formal view of Thagard's article [Thagard 1978].

4.6 Discontinuity and Stability

Another question is whether we can say that two explanations are *akin*. We give a definition for this:

DEFINITION 4.10

An explanation e_1 is *akin* to explanation e_2 on the bias β if and only if e_1 is a subprogram of e_2 or e_2 is a subprogram of e_1.

We have said in previous subsections that a simple exception "perverts" the hypothesis and forces to reformulate another one. That is to say, most of the times a simple change in the input surely produces a great change in the description. This means that all the definitions we have been given are *extremely* discontinuous. Immediately it poses the question of how we can tackle its radical discontinuity. We are beginning to realise that our definition are —in some sense and independently to other classical computational complexity hierarchies— the *hardest* problem of interest that we can imagine. In this sense they are much harder than universal compression because although original algorithmic complexity is discontinuous, it is more or less locally continuous, i.e., a change in the input generally changes in a coding of this as an exception. Here, usual methods of search, like "hill climbing or gradient descent" and "beam search" can be useful to some extent.

But with very discontinuous problems the only techniques that have shown useful are those based on "parallel combinatorial search" maybe with techniques from evolutionary computation. Fortunately, if we use some genetic search, we can profit a characteristic of this exception-free nature. When hypotheses fail (in a generation), we try to construct other (for the next generation) beginning with the data that was not covered in most of the previous hypotheses. In fact, this resembles how people look for explanations: an anomaly originates new hypotheses precisely from it.

If our definitions are so sensitive to any change in the inputs, they are less discontinuous to the length.

DEFINITION 4.8

A string x is *m-stable on the right* on the bias β iff

$\forall\, d,\ 1 \le d \le m$: $\text{SED}_\beta(x_{-d})$ is *akin* to $\text{SED}_\beta(x)$

Informally, a string x is *m-stable on the right* if taking at most m elements from the right, renders the same *kind* of explanation. Note also that the length of the explanations does not have to be decreasing with m

EXAMPLE 4.8

Given the sequence x = "1,2,3,…n" being $n = 2^p$.

has $\text{SED}_\beta(x)$ = "the first 2^p natural numbers". We have that has $\text{SED}_\beta(x_{-d})$ = "the first $2^p - d$ natural numbers" is a superstring of $\text{SED}_\beta(x)$.

It is also notable to realise that the longer the string the more probable that it would be more stable. This is why the longer the string the more $C(x)$ approximates $E(x)$.

5 A Formal Definition of Intelligence

First, the informal definition: "*Intelligence is the* ***ability*** *of explanatory compression*". So, let's measure it.

5.1 Assumptions

We establish the following reasonable assumptions: We cannot say that a system is or is not intelligent just like we cannot say that a person is or is not intelligent, as Turing Test does. Also, we think there is no 'meta-intelligence' since there is no discontinuity from the reactions of our pets, the sometimes smart behaviour of primates and dolphins to the smartest Homo Sapiens. So, it is gradual from birth to adolescence. Accordingly, most definitions of intelligence begin with the word 'ability'. Every ability is wont to be measured. But in this case it requires the collaboration of the subject. So the intelligence of a system S can only be measured by a test made to S. We also presuppose that the system *understands* the aim of (or at least it is well *programmed* for or it has been given an order to do) the test. It also *understands* the bias or language used for the test.

5.2 The Definition Is —Of Course— a Test

We are ready to give a formal definition of intelligence as a value resulting from the following test:

DEFINITION 5.1

Let us select an *expressible* and *fair* bias β and a *wide* range $1..K$. For each $k = 1..K$ we choose randomly p sequences $x^{k,p}$, such that are k-incomprehensible and *d-stable* with $d \ge r$.

We measure the intelligence of a pretended intelligent system S in the following way. We first give (or programme) to S the alphabet Ω_β and the operations Θ_β with their corresponding *costs*.

We give these $K{\cdot}p$ strings without its $d - r$ last elements ($x^{k,p}_{-d+r}$) to S and ask for the following element under some *short* 'explanation' that combines the elements of β (Ω_β and Θ_β). We give S a time t and record its answers: $guess(S, x^{k}_{-d+r+1})$.

Then we give the result for the *C-test*.

$$I(S) = \sum_{k=1..K} k^e \cdot \sum_{i=1..p} hit\left[x^{k,i}_{-d+r+1}, guess(S, x^{k,i}_{-d+r+1})\right]$$

being hit a difference function (usually $hit(a,b) = 1$ and 0 otherwise). r indicates the 'redundant' values we give in order to make easier (and reliable) the answer. e is an exponent to weight more or less the difficult questions. If we choose $e = 0$ all the questions have the same value.

Informally, "*the test measures the ability of finding the shortest explanation (exception-free description) for some strings of different difficulty (comprehensibility) in a fixed time*". The way it can be tuned and implemented is discussed in [Hernández-Orallo 1997b] but point 5.4 and the appendix could suggest many of the answers. We have called this test C-test (*C* from *Comprehension*) just to differentiate it from IQ-test.

5.3 $E(x)$ vs. $C(x)$

Maybe every system that is able of exception-free compression is able of normal compression (we doubt the converse), and all the definitions can be simplified a great deal, using the more generalised and comfortable C. But the latter is not practical for measuring this ability for short strings, because they usually render descriptions with exceptions that are not *intuitive*. The strings used in the tests would have to be longer and this will make the tests impracticable. The only objective view of eliminating some 'arbitrary' descriptions is by using our definition. Also, any kind of system that is good in E is good in C because it is able to detect exceptions and to code them *interestingly*. As we will see, the contrary is not the case.

5.4 The Generator

We have said that calculating the *k-comprehensibility* of an arbitrary string is not computable in general. But given an arbitrary k, the *k-comprehensibility* of a string is computable. This does not render the things easy. The process of finding a random string that would be k-incomprehensible goes as follows.

ALGORITHM FOR STRING GENERATION

We begin with K empty sets of pairs $<x, x^*>$ and we want to obtain p pairs for every k-set.

Beginning with $k=K$, and skipping it if the *k-set* is full, choose randomly a description x^* (combining operators and alphabet) such that $\Delta(x^*)$ and:

$$(k-1) \cdot \log l(x) \leq LT_\beta(x^*) \leq k \cdot \log l(x).$$

We have to look for all the explanations x^+ for x, with $LT_\beta(x^+) \leq k \cdot \log l(x)$. If there is not such an explanation (the less probable) we have that x is k-incomprehensible and we know its $SED_\beta(x)$, putting $<x, x^*>$ in the *k-set*. If there is such an explanation with $LT_\beta(x^+) \leq k' \cdot \log l(x)$ we put down in the *k' set* the pair $<x, x^+>$.

The more feasible option seems to *select a bias which does only allow explanatory descriptions maintaining expressibility.* But this is difficult because it is hard to detect whether some program is *hiding* an exception. For instance, we can design a bias that does not allow the operation "*Print x*" to avoid descriptions like "*Print* 1, *Print* 3, *Print* 5" but it can be cheated with the program *Push* 5, *Push* 3, *Push* 1, *Pop*, *Pop*, *Pop*. Although one usually does not hide exceptions like these without necessity the bias could do it in 'subtle' ways. We have precisely introduced all the section 4 for this reason.

Another approach is restricting memory (or stack) resources, so reducing the possibilities of hiding exceptions. The best practical solution we have found is using programs that are not separable (only one low-redundant covering $s=\{x\}$). For instance, if we use assembler-like programs we can force them to finish in a loop (the greater the final part of the string the loop covers the better). Curiously, similar restrictions appear in the early essays of [Simon & Kotovsky 1963].

It is left to see the stability of the strings. Once we have filled in each k-set, we pass the following sifting.

ALGORITHM TO SIEVE THE D-STABLE STRINGS.

Given a pair $<x, x^*>$ look for all the $SED_\beta(x_{-i})$ for all $i=1...d$ and check that all are akin to x^*. (Seek the $SED_\beta(x_{-i})$ only up to $LT \leq k \cdot \log l(x) + C$)[4].

We have again a problem of efficiency. Fortunately with the approximation we have commented it is easy since "*loop until position p arrived*" is stable for some q generally near to p.

[4] If we do not found one and only one for each d it is not valid either.

5.5 IQ-tests and C-tests

We have presented some guidelines to make the C-test. Most of them are restrictions to turn the production of the problems for the test more efficient —they originally are computable—. Non tractability is not so grave in this case because the generator can be executed on a speedy machine in a preparatory stage to store a good collection for future tests. Appendix A presents a test we have designed using these considerations.

The test-based nature of the definition presented suggests an analogy with the usual tests that measure intelligence, the *Intelligence Quotient* (*IQ*) tests. But, *will IQ tests correlate with C-tests?* We think that the result is far from conclusive to guarantee or discredit our theory, but according to [Brand 1996] correlation with IQ tests is a necessary (but not sufficient) condition for a good measurement of intelligence.

6. A Formal Definition of Analogy

The *test* nature of definition 5.1 gives the impression that is not so formal. In this section we show that it serves for giving formal definitions of many reasoning processes, like analogy. Complex recognition tasks are extremely well performed by animals (a dog which recognises his owner's smell or some spoken orders, an elephant which never forgets your face, etc.). They are not by far considered intelligent for this (more often they are considered intelligent for other different things). Humans are distinguished for a higher level of recognition, usually known as *analogy*. [Holland et al. 1986] regard analogy as a special case of induction: "*a subtle, powerful inductive process, often viewed as a mysterious fount of creativity*" and assert that analogy is a morphism between two mental models.

6.1 Analogy and Fluid Minds

Hofstadter's key insight is that machines will never satisfactorily pass the Turing test until they achieve true mental fluidity. In his view [Hofstadter 1985], fluid analogies "are not game for rigid minds" and are "so clearly subjective"**.** However, he does not think that this is not attainable, but that it requires a very complex and "fluid" system with many experiences (and evolution-originated schema) to give these answers. But he goes further when he asserts that "*there is no fixed mathematical recipe for reconciling all the different forces pushing and pulling you in analogies*".

To study the problem, he proposes analogies like "abc is to abd" as "ccc is to ?" and many more complex ones. With a convincing justification of why a simplified domain like this is meaningful and models real aspects of intelligence, he and the Fluid Analogies Research

Group [Hofstadter et al. 1995] present programs to give the same kind of answers as humans in these simple-domain complex-analogy problems. In fact, they give no formalism after their solutions.

6.2 A Formal View of Analogy

Different paradigms for solving analogies have been essayed since [Kling 1971] to [Winston 1992] to give the same solution as people to simple analogy problems. Given a "A is to B" as "C is to X" Winston presents a matching based on the 'similarities' between the mapping from *A* to *B* and *C* to *X*. It works quite well for the blocks world but his model cannot explain many other more 'elaborate' problems, like simply "abc is to abd" as "xyz is to ?". Why does it give intuitively the result *wyz* and not *xyd*? Using Winston's schema, we get the answer *xyd*. According to common sense, what can we do to obtain the answer *wyz*?

Common sense is to us the contrary to arbitrariness. Our solution is similar, apparently, but we define our similarity based on intensional complexity:

DEFINITION 6.1

$$\text{Analogy Plausibility} = -\alpha \cdot C(A) - \beta \cdot E(Map_{A\leftrightarrow B}) - \gamma \cdot E(Map_{A\leftrightarrow C}) - \delta \cdot E(Map_{AB\leftrightarrow CX \rightarrow A\leftrightarrow C})$$

With $E(Map_{A\leftrightarrow B})$ we mean the Intensional Complexity of the mapping between A and B (in both senses). The last term gives the commonalties of low and high level. If we make δ great enough compared to the other coefficients, we see how it works. For example "abc is to abd" as "xyz is to ?":

Analogy Plausibility ("wyz") =

$-\alpha \cdot C(A) = l($"the first 3 letters of the alphabet"$)$.

$-\beta \cdot E(Map_{A\leftrightarrow B}) = l($"inc. the greatest element"$)$

$-\gamma \cdot E(Map_{A\leftrightarrow C}) = l($"change orders"$)$

$-\delta \cdot E(Map_{A\leftrightarrow C \rightarrow AB\leftrightarrow CX}) = l($""$)$

You can also check out with the preceding definition why you may say "acg" to the question "Given *abc* to *abd*, what does *ace* go to?" instead of "acf" or "ade".

Obviously, as you can guess, this formulation requires some kind of high order language (or bias) to talk about maps of maps. Also it requires to compute *E* (or *Et*) or, at least, some approximation.

It is not casual that we have chosen analogy because it is the more common mechanism by which we can 'project' structure across levels. In our opinion the best approach in this line is the system FOL [Weyhrauch 1980].

In our view, the most remarkable result from this theory of analogy is to see 'beyond' and realise that fluidity is the ability of reducing *E* of a problem and the associated context. Most architectures essayed in the field of AI (and even more in expert systems) are 'corrupt' from the beginning because they fix (and then limit) the reasoning operations the system can do.

7 Intelligent Cognitive Agents

Just from the generator presented in section 5, we can program a machine to make the C-test. So we have done. If executed in a speeder machine, it would give moderate good scores. The question arises quickly. Must we call this machine intelligent? Our opinion is that the answer is no. We, at most, can call the machine *potentially* intelligent. The reason is that when we talk about an intelligent system we implicitly mean a *cognitive agent* with *intelligent abilities*.

The main difference is in knowledge. We must recall that the intelligence of a system is an ability that is stable during life while its knowledge is something that is in constant evolution and remaking. Also the different knowledge of the individuals is what *characterises* and *differentiates* them, much more than their intelligence. The relation between intelligence and knowledge must begin with Habermas' statement: "*there is no knowledge without interest*". This is the reason for using the word *agent*.

Well, we have been harping on the virtues of *exception-free* descriptions. But there are some circumstances where a leeway of exceptions can be profitable to make a description much shorter. The information that finds some suitable *transformation* to the *interesting* topics is considered *relevant* and gets the *attention* of the system. Tolerance on exceptions varies radically on the different purposes: memorising, recognising, learning, reasoning or explaining something. Without more discussion, we just give an informal and maybe inaccurate conception of an intelligent cognitive agent.

An intelligent cognitive agent is an interested system with input/output devices for a complex environment and a contextual knowledge organisation and competing concepts governed by principles of short explanations and purpose related relevance.

We identify a context with a *bias* that is selected for a problem to look for explanations. That is, when a purpose or a simple strange phenomenon triggers the attention to explain some fact, we construct "a virtual bias" in working memory composed of the concepts related to the problem. The contextualisation is usually seen in a more static idea of 'distance' to the huge information of long-term memory, but to some problems *Et*(*x*) would construct better contexts (by analogy).

It is well known in psychology that only a small number of items —six according to [Hamilton 1859] or seven according to [Miller 1956]— can be considered at a time. There have been given some rationales for this limit but the most reasonable one seems to be in, in Kotovsky words: "*people's limited working memory capacity controls their ability to think about problems and plan moves*". This may be the reason why sometimes a person is not prepared to learn some concept because it would require more than, say, seven concepts at a time. It is after the learning of a new concept that comprises a few ones that a person is able to learn some difficult concept or solve some problem. This limitation forces us to learn more abstract concepts to comprehend the world. Definition 4.6. avoids low-level descriptions when we have high-level descriptions.

7.1 Some Models of Explanatory Reasoning

We have talked about COLT (COmputational Learning Theory) to justify the introduction of our theory. But there are still many models and interesting work based on more classical knowledge representation in AI around the fashionable term "explanatory reasoning".

Inductive Logic Programming (ILP) has been paid a big attention as a framework of supervised learning in first order logic [Muggleton 1991] [Lavrac & Dzeroski 1994]. It has been said that: "regression gives numeric prediction, but little explanation. ILP gives explanation, but no numeric prediction" [Srinivasan & Camacho 1997]. Our interest about ILP is that it has made plain the necessity of inventing new predicates as bias shift operation [Stahl 1995]. Also, it has differentiated between useful and necessary predicates. The first allow the hypotheses to be shorter, but the generalisation is not strictly necessary (they are, in some way, extensional). The second are recursive ones that are strictly necessary to learn a concept (they are, to some extent, intensional). Since first-order seems exhausted for more ambitious purposes, [Lloyd 1995] has proposed a new inductive system called *Escher* to jump to Higher-Order Logic, based on Church's theory of types [Church 1940], a work which resembles high-order logic reasoning systems like HOL.

Abduction (Sherlock Holmes' intelligence [Josephson & Josephson 1994]) is a notion introduced by Peirce (1839-1914). He asserted [Peirce 1867/1960] that neither deduction nor induction can help us to unveil the internal structure of *meaning*. Abduction is the process of making assumptions to explain some facts. It is described as "inference to the best explanation" [Harman 1965]. A counterpart of ILP, ALP (Abductive Logic Programming) [Kakas et al. 1993] and other frameworks like EBL and Abductive Explanation Based Learning (AEBL) have appeared as more specific cases of abductive reasoning. [Michalski 1987] expresses perfectly our point of view inside all this diversity: "*inductive inference was defined as a process of generating descriptions that imply original facts in the context of background knowledge. Such a general definition includes inductive generalization and abduction as special cases*". The only subtle difference between abduction and induction originates mainly from the aim and utility. The following example shows *all* the difference we make in our C-tests.

Induction Problem:	1, 3, 5, 7, ?
Abduction Problem:	1, ?, 5, 7, 9

Finally, case-based reasoning appeared to face the problem of everyday abductive explanation [Leake 1992], [Shank et al. 1994]. The difference with the preceding methods is in the non-constructive character. Every explanation is matched (using schema) on a similar case on prior episodes from memory. Despite the limitation of *pure* case-based reasoning (in our view), one remarkable thing is the clarification of the issue *when to explain*. In [Leake 1995] words: the "explanation process is triggered when *anomalies* arise". But "*anomalies not only provide guidance about when to explain, but of what to explain as well*".

7.2 The Representation Problem

In AI, representation has been always the most debated topic. We will also make some considerations. It is known that (1) restricted representation limits expressiveness but usually favours tractability or adequacy for some kind of problems, (2) between two representations with the same expressive power, C(x) spirits away any reason to choose one over the other .

In our opinion, representation is not only a question of easiness to people or tractability for certain problems. Some of the discussions on representation could clarify a great deal if we think in terms of plain evidence of exceptions and easiness of separability. It is not casual that these two ideas have led from structured programming to object-oriented programming.

The other question about representation is which representation is more suitable for intelligence? Intuitively, for instance, high order logic seems more suitable than Turing machines because separability seems easier. While this would we a conjecture, one should use the bias more suitable for implementation.

7.3 The Implementation Problem

In reaction to traditional AI which used symbolic manipulation, exemplified by the list processing approaches in Expert Systems and the Logic

Programming paradigm, a modern AI, known as soft computing, entered the scene to deal with uncertainty. Artificial Neural Networks (ANN) (or the broader term connectionism) are the most prominent technology in this tendency. Its intrinsic parallel character makes them very efficient and suitable for recognition tasks and similar problems. But, in our opinion, the pending matter for ANN is to address higher-level tasks.

In Calvin's words: "*tying a necktie or hair ribbon required lots of conscious attention in the beginning, but once established (perhaps as a subcortical level) we can do it better if we don't try to think about it. What is, initially, consciously mediated can become subconscious with practice*" [Calvin 1996]. A lot of mixed systems have been essayed to cope the 'higher' side of the coin, reasoning. We see neuro-statistical, neuro-genetic, neuro-rule, neuro-fuzzy, neuro-chaos and the like. Also, recent algorithms for "symbolic representation of neural networks" [Gallant 1993], [Towell & Shavlik 1993] help to bridge them.

One of the great misconceptions in this area is to think that fuzzy and numerical approaches are just sufficient to fluid categories (e.g., assimilate a pen with a keyboard or the meaning of "head" in different contexts). We do not mean that fuzziness should be banned. Conversely, many times it is useful in cognition too. We think that, at the higher levels, fuzziness must be *voluntary*, guided by interest.

From all these examples and many psychological studies we are rather convinced that the way people think is rather different from how people recognise, control and even learn many routine problems. It is algorithmic plan construction, time-dependent mechanisms and symbolic reasoning which make the distinction between us and *animal intelligence*. Accordingly, recent work is centred on learning classical computational machines on neural networks [Giles et al. 1990], [Wiklicky 1994], [Zeng et al. 1994], [Giles et al. 1997]. Naturally, many classical problems in computational learning theory are reappearing here.

7.4 Detecting Anomalies and Selection

Event-related potential (ERP) has been used to study the selection of hypotheses in higher-level information processing. There have been also essays with semantic incongruity [Kutas & Van Petten 1988] and temporal sequence of language-processing events. In the latter, subjects were presented phrases like "*The pizza was too hot to* ... " but completed with three kinds of ending: most expected sentence completions (e.g., "*eat*"), unrelated anomalous completions (e.g., "*cry*") and related anomalous completions (e.g., "*drink*"). During the interval 200-600 ms, the result showed a flat waveform for the expected sentence completion, a pronounced negative waveform for unrelated anomalous completions and an irregular intermediate result for the related anomalous completion. This suggested that brain activity is higher and longer the greater the anomaly [Caryl & Harper 1996]. Since in this stage myriads of possible explanations are generated, selection must be done. Much recent work is precisely there, in evolutionary recurrent networks [Angeline et al. 1994] [Batali 1995]. But the selection rule is not clear. Usually a combination of interest, utility, and complexity is used, where MDL is the more popular candidate in the later case. We suggest SED instead or implementation-oriented approximations.

In our opinion, the next step is a re-encounter between "soft-computing" and a new generation of "symbolic approaches". They will have to collaborate to find similarities sometimes —frequently at the low level— in order to be able to find anomalies, exceptions and differences for making hypotheses —at a high level—.

8 Evaluating Intelligent Systems

We think that Turing Test had a lot of negative influences in AI research because, accordingly, the only way to measure progress in the field was comparing the skill of people in certain tasks with the skill of machines in the same tasks. This is partial and subjective. Our proposal is to use objective tests derived from our theory. At the current state of the art in artificial intelligence we cannot expect good results (vs. people scores) for these tests —if made with broader contexts than the toy machine we have used in appendix A—. The tests we propose to construct will provide a range of magnitudes large enough to remark this difference. Also this could be used someday to measure systems beyond the range of human intelligence —since IQ-tests would not have any reference in order to do this—.

This detailed scale of measuring provides a means for evaluating the progresses in learning systems or other intelligent systems. It is well known that libraries or corpus of problems used to evaluate systems in an area tend to produce *specialised* systems which attain better scores on that kind of problems while being completely dull at others. This has been the case even in TPTP [Suttner & Sutcliffe 1996], a complete and varied fair corpus used to compare Automatic Reasoning systems or ATP devices.

The idea of making feasible more objective measures for intelligent systems is not new. There are prediction contests based on chaotic strings [Ditto & Munakata 1995]. We hope that if our C-tests are confirmed flawless, a similar contest could be held in the future to compare artificial intelligent systems.

9 Conclusions

After all this work we pose the question of how much (if any) we have reduced the validity of Winograd's view of the state of the art in 1990: "*there is no reason but hubris to believe that we are any closer to understanding intelligence than the alchemists were to the secrets of nuclear physics*" [Winograd 1986].

We think we have effectively reflected the aim of the title. We are conscious that we leave moot points and many open questions to work on, but we have presented an intrinsic and human-independent formal definition of intelligence. It is compliant with the more common definitions of intelligence and it is also compliant with IQ tests. But this definition has to be conceived as a means for the discernment and measurement of a main ability that an intelligent cognitive system *must* have.

Conversely, and more remarkable, this article may be seen as an information-based justification of why IQ —currently based mostly on induction and analogy— actually measures intelligence. With all, the correlation between IQ-tests and C-tests cannot be considered as a support to our theory. More experiments may bring confidence. Some day, this theory could be used to ease the construction of IQ-tests and to justify why people see some problems harder than others.

From the more theoretical or philosophical point of view, Algorithmic Complexity $C(x)$ has a great, ignored, problem for modelling finite things. It leaves the notion of exception to the bias, which is outside the theory. Intensional complexity distinguishes the purity of an intensional mathematical definition (by comprehension). More philosophically, it can give some light into the dark question of "what is meaning" which is also a central question in AI.

Back on the classical view of computers, this definition, in some sense, proves that only the three basic operations required to construct any recursive function: composition, primitive recursion and minimisation [Boolos & Jeffrey 1989] are sufficient (finitely combined under the closure property) to implement intelligence. The question is how.

In this sense, we think that some practical questions may also be helped by the perspective of intensional complexity. First, we have given a definition of analogy. Secondly, we can determine which representation can be more suitable (and discard formally first-order approaches) and then help to make more creative and inventive machines.

Nonetheless, many informal ideas are not new (e.g. "explanatory coherence" [Thagard 1989], Wolff's proposal [Wolff 1995], "concept learning by complexity regularisation" [Lugosi & Zeger 1996] and many others). Maybe we are not the first who criticise the MDL principle; we know it is a good operative principle for tasks like recognition, classification, statistical prediction and approximations but it is not a justification principle useful for high-level rationales. It does not distinguish noise from exceptions.

In conclusion we think that the most reliable result of this work is to serve as a start point for a framework for measuring the *intelligence* of pretended artificial intelligent systems, a question that has not been addressed successfully to date in AI and cognitive science. As in any other field of science, a great advance of a discipline happens when some of the topics relevant to the theory can be effectively and justifiably measured, contrasted and compared.

9.1 Epilogue

When you do not have a direction, you follow somebody else's way. In AI we have been copying the brain (psychologically or neurologically) with the implicit objective of resembling human behaviour. We are not able to evaluate the admittance that our proposal can have in many fields, but we are convinced that a unified and formal framework is possible. Maybe it is not strictly necessary for making intelligent systems —we are sceptical but it could happen as an emergent property on some kind of evolutionary system— but we think it is the only way if we pretend to understand intelligence. We hope you agree that sooner or later we will have to walk alone.

Appendix A. An Example of C-Test

The problem of selecting a good bias for generating *k-hard* strings depends on many factors. The objective is to maintain expressiveness, to ease the problem of finding explanatory descriptions and to limit the combinatorial *explosion*. The final choice we present is an oversimplified abstract machine that is easily extensible to work as a Turing machine.

A.1 A Toy Memory-less Abstract Machine

Due to the current technology of the computers we can use, we have chosen an extremely abridged emulation of the machine that will *effectively* run the programs, instead of more proper languages, like λ-calculus (or LISP). We have adapted the "toy RISC" machine of [Hernández & Hernández 1993] with two remarkable features inherited from its object-oriented coding in

C++: it is easily tunable for our needs, and it is *efficient*. We have made it even more reduced, removing any operand in the instruction set, even for the loop operations. We have only three registers which are AX (the accumulator), BX and CX. The operations Θ_β we have used for our experiment are in Table 1:

LOOPTOP	Decrements CX. If it is not equal to the first element jump to the program top.
LOOPS	Same as LOOPTOP but it jumps n (for the tests n=4) instructions backward.
LOOPM	Same as LOOPTOP but it jumps m (for the tests m=7) instructions backward.
SUCC	Increments the accumulator.
PRED	Decrements the accumulator.
WRITE	Writes into the output and moves fwd.
BREAD[2]	Moves back and reads from the output.
FREAD[2]	Moves fwd and reads from the output.
MOV A,B[1]	Copy register BX into AX
MOV B,A[1]	Copy register AX into BX
MOV A,C	Copy register CX into AX
MOV C,A	Copy register AX into CX
ROTR[3]	Rotates 45° to the right.
ROTL[3]	Rotates 45° to the left.

Table 1. Instruction Set

The operations with no superscript are present in all the subsets. Operations marked with (1) are present in the 'professional' version of the machine, the operations with (2) are present in the *Turing-like* version and those with (3) are present in the *logo* version where the output is bidimensional. This sparseness of only 10 operations will be clearly justified later. We have essayed with many different alphabets but for this test we will use the professional version and a circular alphabet Ω_β = $\{a,b,c,d,...,z\}$, that is, incrementing 'z' yields 'a' and decrementing 'a' yields 'z'. Since the first element is an inflexion point for the loops, it is presented to the subjects as "*a critical element*".

This configuration *still* produces many programs that are not robust because programs can be often split into subprograms. The solution for these cases comes from another restriction: the programs must be comprised wholly inside a loop. This leaves a good approximation to explanatory programs. The rest to do is to avoid repetitions of patterns like "abcabcabcabc" and take apart the strings where an important part is explained by a shorter program. We think that the bias is not all the expressible we would like but it allows the generation of strings of certain complexity. Also we think it is *fair* because it does not relate on arithmetic (like *cryptarithmetics* tests) or any other preceding knowledge, except the order of the alphabet.

A.2 The Generation of *k*-Hard Strings

The algorithm we have used to generate a set of different *k*-incomprehensible strings is very similar to the one we presented in section 5.4. Having 10 operations, we have that usually only about a 20% of the programs of any size are explanatory. This means that trying to know if a randomly generated program of, say, size 15, will need the checking of more than 2,222,222,222,222 programs. And this is the case if the computational cost of x^* is slow, contrariwise (if x^* is a *costly* program) we will have to check longer programs.

We have used some optimisations and heuristics in order to make the great amount of programs to check more tractable. Some examples of questions are:

Prediction style:

*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'

Abduction style:

*k*8:	a, _, a, z, a, y, a, …	Answer: 'a'
*k*10:	a, x, _, v, w, t, u, …Answer: 'y'	
*k*13:	a, y, w, _, w, u, w, u, s, …	Answer: 'y'

A.3 The Tests

Four tests were devised to measure prediction, abduction, g-factor and similarity. The prediction test is composed of 19 exercises generated with the following *k*-hardness distribution (2 *k*7, 1 *k*8, 2 *k*9, 3 *k*10, 3 *k*11, 3 *k*12, 2 *k*13 and 1 *k*14), redundancy r = 2 and the less 'akin' as possible. The abduction test is composed of 15 exercises using the same generator and redundancy. The distribution was (2 *k*7, 2 *k*8, 1 *k*9, 2 *k*10, 1 *k*11, 3 *k*12 and 4 *k*13). In these two tests, the incorrect options were generated randomly but relative near to the solution and the letters appearing in the string. The IQ test we used was the European IQ test simply because it is a culture-fair test, devised for 20 minutes, ensuring a reasonable range (75-174) of values and available on the Internet. The similarity test is composed of 8 exercises generated with binary strings of different length and different levels of edit errors (insertion, deletion or change). The strings were generated and checked by dynamic programming to ensure that they did not have a better correction path. The purpose of this test was to measure the ability of compression by *trivial* pattern matching.

A.4 Subjects and Administration

Subjects were selected from two different groups: the first group was composed by 48 high-school students with ages comprised between 14 and 18 years. The second group was composed by 17 subjects of a mixed sample of undergraduate and postgraduate university students with ages comprised between 22 and 32 years.

All the tests were passed in the same session. The times were, without including instructions, 10 min. for the prediction test, 5 min. for the abduction test, 5 min. of break, 20 min. for the IQ test and 3 min. for the similarity test.

A.6 Results

We evaluated the test without penalising the errors, i.e, the function *hit* evaluated the same for blanks that for mistakes. We chose *e*=0, i.e. all questions with the same value. IQ-correlations are illustrated in Table 2.

	Pred.	Abd.	Induct.	Simil.
High-School	0.31	0.38	0.42	0.39
University	0.51	0.42	0.56	0.35
Both Groups	0.73	0.68	**0.77**	0.50

Table 2. Correlations with EIQ test

The correlation for induction (prediction + abduction) is of the same order as the usual correlation for induction tests made by psychologists. The correlation between the abduction and prediction tests was 0.61, less than expected, which suggests that even problems constructed by the same generator can be more or less difficult depending on its presentation (abductive or predictive). The correlation between induction and similarity was 0.51 which supports the thesis that "the ability of compression" is different from "the ability of comprehension". Finally, we think that an analogy test based on our theory would surely round off the study.

With these data and our amateur methods we are not in conditions to assert more things about the relation between C-tests and IQ-tests. There is only a thing that has no discussion in the light of the results, the *k*-hardness matches fairly well with the difficulty people found on them, as it is seen in Figure 1:

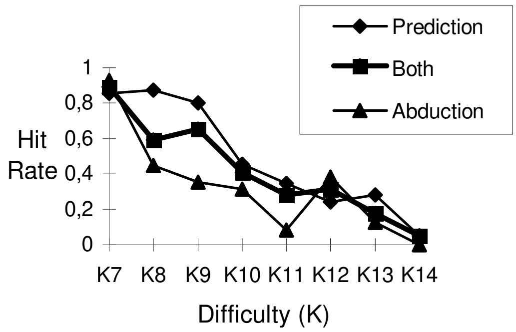

Figure 1. Hit Rate per Difficulty

Acknowledgements

We are much obliged to Rafael Beneyto and the 'fluidity' of his postgraduate sessions about intelligent systems and, in general, to all the Department of Logic and Philosophy of Science of the University of Valencia. We specially thank Enrique Hernández and Ismael García for their useful comments on the 'weighty' report for this paper [Hernández-Orallo 1997b] and for all their moral support. Finally, we agree the inestimable collaboration of Enrique Fueyo for administering the tests at the "I.B. Gil y Carrasco" at Ponferrada and all the students who made them.

References

[Abe 1997] Abe, N. "Towards Realistic Theories of Learning" New Generation Computing, 15, 1997

[Angeline et al. 1994] Angeline, P.J.; Saunders, G.M.; Pollac, J.B. "An Evolutionary Algorithm That Constructs Recurrent Neural Networks", IEEE Trans. Neur. Nets, Vol. 5, no. 1, pp 54-65, 1994.

[Baker & Anderson 1982] Baker, L.; Anderson, R. "Effects of inconsistent information on text processing: evidence for comprehension monitoring" *Reading Research Qtly.*, 17, 281-294, 1982.

[Barker 1957] Barker, S.F. *Induction and Hypothesis* Ithaca, 1957

[Barlow 1969] Barlow, H.B. "Trigger Features, Adaptation and Economy of Impulses" in K. N. Leibovic (ed.) *Information Processing in the Nervous System*, Springer, pp. 209-230, 1969.

[Batali 1995] Batali, J. "Recurrent Neural Networks, Context-Free Grammars, and Evolution" Department of Cognitive Science, University of California at San Diego, March 1995.

[Bertalanffy 1971] Bertalanffy, L.V. "Teoria generale dei sistemi", Instituto Librario Internationales, Milano 1971.

[Blum 1967] Blum, M. "A machine-independent theory of the complexity of recursive functions" J. ACM 14, 4, 322-6, 1967.

[Blumer et al. 1987] Blumer, A.; Ehrenfeucht, A.; Haussler, D.; Warmuth, M. K. "Occam's razor" Inf.Proc.Lett. 24, 377-380, 1987.

[Blumer et al. 1989] Blumer, A.; Ehrenfeucht, A.; Haussler, D.; Warmuth, M. "Learnability and the Vapnik-Chervonenkis Dimension" Journal of ACM, 36, pp. 929-965, 1989.

[Board & Pitt 1990] Board, R.; Pitt, L. "On the necessity of Occam algorithms" in Proc., 22nd ACM Symp. Theory of Comp., 1990.

[Boolos & Jeffrey 1989] Boolos, G.; Jeffrey, R. "Computability and Logic" 3rd Edition, Cambridge University Press 1989.

[Brand 1996] Brand, C. "The g Factor: General Intelligence and its implications" Wiley 1996

[Broad 1926] Broad, C.D. *The Philosophy of Bacon*, Cambrdge, 1926

[Calvin 1996] Calvin, W.H. "The Cerebral Code. Thinking a Thought in the Mosaics of the Mind" The MIT Press 1996

[Caryl & Harper 1996] Caryl, P.G.; Harper, A. "ERPs in Elementary Cognitive Tasks Reflect Task Difficulty and Task Threshold" Intelligence 22, 1-22, 1996.

[Chaitin 1982] Chaitin, G.J. "Gödel's Theorem and Information" Int. Jnal. of Theoretical Physics, vol.21, no.12, pp. 941-954, 1982.

[Chaitin 1992] Chaitin, G.J. "Algorithmic Information Theory", fourth printing, Cambridge University Press, 1992.

[Chandrasekaran 1990] Chandrasekaran, B. "What kind of Information Processing is Intelligence?" in Partridge,D.; Wilks, Y., *Foundations of AI: A Source Book* Cambridge Univ. Press, 1990.

[Church 1940] Church, A. "A formulation of the simple theory of types" *Journal of Symbolic Logic*, 5:56-68, 1940.

[Dietrich 1990] Dietrich, E. "Programs in the Search for Intelligent Machines: The Mistaken Foundations of AI" in Partridge & Yorick, *Foundations of AI: A Source Book*, Cambridge Univ. Press, 1990.

[Ditto & Munakata 1995] Ditto, W.; Munakata, T. "Principles and Applications of Chaotic Systems" Comm. ACM, v.38, n. 11, 1995.

[Eysenck 1979] Eysenck, H.J. "The Structure and Measurement of Intelligence", Springer-Verlag 1979.

[Flach 1995] Flach, P. "Abduction and Induction: Syllogistic and Inferential Perspectives" INFOLAB, Tilburg University 1995.

[Flood 1987] Flood, R.L. "Complexity: a definition by constructing a conceptual framework" System Rsrch, v 4, n.3., pp. 177-185, 1987.

[Freivalds 1990] Freivalds, R. "Inductive inference of minimal size programs", in M. Fulk and J. Case (eds) "Proceedings of the third Annual Workshop on Computational Learning Theory", pp. 1-20, Morgran Kaufman, San Mateo, CA, 1990.

[Freivalds et al. 1995] Freivalds, R.; Kinber, E.; Smith, C.H. "On the Intrinsic Complexity of Learning" Inf. & Control 123, 64-71, 1995.

[Gallant 1993] Stephen I.G. "Neural Network Learning and Expert Systems" MIT 1993

[Giles et al. 1990] Gile, C.L.; Sun, G.Z. and Chen, H.H.; Lee, Y.C.; D. Chen, D. "Higher Order Recurrent Networks & Grammatical Inference" in D.S.Touretzky "Advances in Neural Information Processing Systems, pp. 380-387, Morgan Kaufmann Pubs 1990.

[Giles et al. 1997] Giles, C.L.; Lin, T.; Horne, B.G. "Remembering the Past: The Role of Embedded Memory in Recurrent Neural Network Architectures" in "Neural Networks for Signal Processing VII", Proc.1997 IEEE Workshop, IEEE Press 1997.

[Giles et al. 1994] Giles, C.L.; Kuhn, G.M.; Williams, R.J. "Dynamic Recurrent Neural Networks: Theory and Applications" IEEE Transactions on Neural Networks, Vol. 5, no. 2, 1994.

[Gold 1967] Gold, E. M. "Language Identification in the Limit", Inform and Control, 10, pp. 447-474, 1967.

[Hamilton 1859] Hamilton, Sir W. "Lectures on metaphysics and logic", vol. 1. Blackwood, Edinburgh 1859.

[Herken 1994] Herken, R. "The universal Turing machine: a half-century survey" Oxford University Press, 1994.

[Hernández-Orallo 1997a] Hernandez-Orallo, J. "Intensional Complexity. An Exception-Free Variant of Algorithmic Complexity" .

[Hernández-Orallo 1997b] Hernandez-Orallo, J. "A Formal Definition of Intelligence. A Thoroughful Discussion".

[Hernández & Hernández 1993] Hernández-Orallo, E.; Hernández-Orallo, J. "Programación en C++" Paraninfo 1993.

[Hintikka 1970] Hintikka, J.; Suppes, P. "Surface Information and Depth Information" in Hintikka, Jaakko; Suppes, Patrick "Information and Inference", D. Reidel Publishing Company 1970.

[Hofstadter 1979] Hofstadter, D.R. "Gödel, Escher, Bach: An eternal golden braid" New York: Basic Books, 1979.

[Hofstadter 1985] Hoftadter, D.R. "Metamagical Themas. Questing for the Essence of Mind and Pattern" Basic Books, Inc., 1985.

[Hofstadter et al. 1995] Hofstadter, D.R.; Fluid Analogies Research Group "Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought" Basic Books, 1995.

[Holland et al. 1986] Holland, J.H.; Holyoak, K.J.; Nisbett, R.E.; Thagard, P.R. "INDUCTION, Processes of Inference, Learning and Discovery" The MIT Press 1986.

[Hopcroft & Ullman 1979] Hopcroft, J.E. and Ullman, J.D. "Introduction to Automata Theory, Languages, and Computation" Addison-Wesley Publishing Company, Inc. 1979.

[Josephson & Josephson 1994] Josephson, J.R.; Josephson, S.G. "Abductive Inference. Computation, Philosophy, Technology" Cambridge University Press, New York 1994.

[Kass 1986] Kass, A. "Modifying explanations to understand stories" in *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, pp. 691-696, Cognitive Science Society 1986.

[Kearns 1989] Kearns, Michael J. "The Computational Complexity of Machine Learning" An ACM Dist. Dissert. The MIT Press 1989.

[Kieffer & Yang 1996] Kieffer, J.C.; Yang, E. "Sequential Codes, Lossless Compression of Individual Sequences, and Kolmogorov Complexity" IEEE Trans. on Inf. Theory, vol. 42, no. 1, Jan. 1996.

[Kling 1971] Kling, R.E. "A paradigm for reasoning by analogy" Artificial Intelligence, 2:147-178, 1971.

[Klir 1985] Klir, George J. "Complexity, Some General Observations" Systems Research, no. 2, pp. 131-140, 1985.

[Kolmogorov 1965] Kolmogorov, A.N. "Three Approaches to the Quantitative Definition of Information" Problems Inform. Transmission, 1(1):1-7, 1965.

[Kotovsky and Simon 1990] Kotovsky, K.; Simon, H.A. "Why are some problems really hard: explorations in the problem space of difficulty". Cognitive Psychology, 22, 143-183 1990.

[Kuhn 1970] Kuhn, T.S. "The Structure of Scientific Revolution", University of Chigago 1970.

[Kutas & Petten 1988] Kutas, M.; Van Petten, C. "Event-related brain potential studies of language" in P.Ackles, J.R.Jennings, M. Cles *Advances in Psychophysiology*, Greenwich, JAI Press. 1988.

[Lakatos 1976] Lakatos, I. "Proofs and Refutations. The Logic of Mathematical Discovery" Cambridge University Press, 1976.

[Lavrac & Dzeroski 1994] Lavrac, N.; Dzeroski, S. "Inductive Logic Programming: Techniques and Applications" Ellis Horwood, 1994.

[Leake 1992] Leake, D.B."Evaluating Explanation: A Content Theory" Lawrence Erlbaum Associates, Hillsdale, NJ, 1992.

[Leake 1995] Leake, D.B "Abduction, Experience, and Goals: A Model of Everyday Abductive Explanation" The Journal of Experimental and Theoretical Artificial Intelligence 1995.

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

[Lloyd 1995] Lloyd, J.W. "Declarative programming in Escher" TR CSTR-95-013, Dep. of Comp. Science, Univ. of Bristol, 1995.

[Li & Vitányi 1997] Li, M.; Vitányi, P. "An Introduction to Kolmogorov Complexity and its Applications" 2nd Ed. Springer-Verlag 1997.

[Lugosi & Zeger 1996] Lugossi; Zeger "Concept Learning Using Complexity Regularization" IEEE T.Inf.Theory,v.42,n.1,Jan 1996.

[Lucas 1962] Lucas, J.R. "Minds, Machines, and Gödel" Reprinted in Anderson, A. (ed) "Minds and Machines", Prentice Hall, 1962.

[McKoon & Ratcliff 1992] McKoon, G.; Ratcliff, R. "Inference during reading" *Psychological Review*, 99 (3), 440-466.

[Michalski 1987] Michalski, R.S. "Concept Learning" in S.C. Shapiro (ed). "Encyclopedia of A.I." 185-194, John Wiley, 1987.

[Mill 1843] Mill, J.S."System of logic ratiocinative and inductive" 1843.

[Miller 1956] Miller, G. "The magic number seven, plus or minus two: Some limits on our capacity for processing information" *Psychol. Rev.* 63. 81-97, 1956.

[Muggleton 1991] Muggleton, S. "Inductive Logic Programming" *New Generation Computing, 8, 4*, pp. 295-318, 1991.

[Newell 1990] Newell, A. *Unified Theories of Cognition*, Cambridge, Mass.: Harvard University Press, 1990.

[Peirce 1867/1960] Peirce, C.S. "Collected papers of Charles Sanders Peirce" Cambridge. Harvard University Press 1960.

[Plotkin 1970] Plotkin G. "A note on inductive generalization" Machine Intelligence, Vol. 6, Edinburgh Univ. Press, 1970.

[Popper 1968] Popper, K.R. "Conjectures and Refutations: The Growth of Scientific Knowledge", Harper Torch Books, New York, 1968.

[Rissanen 1978] Rissanen, J. "Modelling by the shortest data description" Automatica-J.IFAC, 14:465-471, 1978.

[Rivest & Sloan 1994] Rivest, R.L.; Sloan, R. "A Formal Model of Hierarchical Concept Learning" Inf. and Comp. 114, 88-114, 1994.

[Rozenber & Salomaa 1994] Rozenberg, G.; Salomaa, A. "Cornerstones of Undecidability" Prentice Hall 1994.

[Sagan 1973] Sagan, C. (ed). "Communication with Extraterrestrial Intelligence" Cambridge, Mass.: MIT Press, 1973.

[Sanford 1990] Sanford A. "On the nature of text-driven inference" in Balota D., d'Arcais, G.F. & Rayner, K. (Eds.) *Comprehension processes in reading*, chap. 24. Lawrence Erlbaum, 1990.

[Schank et al. 1994] Schank, R.; Riesbeck, C.; Kass, A. (Eds.) "Inside Case-Based Explanation" Lawrence Erlbaum Assoc., 1994.

[Shannon and Weaver 1949] Shannon, C.E. and Weaver, W., "The Mathematical Theory of Communication", Univ. Illin. Press, 1949.

[Shapiro 1981] Shapiro, E. "Inductive Inference of Theories from Facts" RR 192, D. Comp. Science, Yale Univ., 1981, in Lassez, J.; Plotking, G. (eds.) "Computational Logic" The MIT Press 1991.

[Shavlik 1994] Shavlik, J.W. "Combining Symbolic and Neural Learning" Machine Learning, Vol. 14, no. 3, pp. 321-331, 1994.

[Simon & Kotovsky 1963] Simon, H.; Kotovsky, K. "Human acquisition of concepts for sequential patterns" Psych. Review 70, 534-46, 1963.

[Smullyan 1992] Smullyan, R.M. "Gödel's Incompleteness Theorems" Oxford University Press 1992.

[Solomonoff 1964] Solomonoff, R.J. "A formal theory of inductive inference" Inf. Control.. vol. 7, 1-22, Mar., 224-254, June 1964.

[Solomonoff 1986] Solomonoff, R.J. "The Application of Algorithmic Probability to Problems in AI" in L.N. Karnal; J.F. Lemmer(eds) *Uncertainty in AI*, Elsevier Science, pp.473-91, 1986.

[Spearman 1904] Spearman, C. "'General Intelligence' objectively determined and measured" Amer. J. of Psych-, 15, 201-293, 1904.

[Srinivasan & Camacho 1997] Srinivasan, A.; Camacho, R.C. "Experiments in numerical reasoning with ILP" J. of LP, 1997.

[Stahl 1995] Stahl, I. "The appropiateness of predicate invention as bias shift operation in ILP" *Machine Learning*, 20:95-117, 1995.

[Sternberg 1977] Sternberg, R.J. "Intelligence, Information Processing, and Analogical Reasoning" John Wiley & Sons 1977

[Suttner & Sutcliffe 1996] Suttner, C.B.; Sutchliffe, G. "The TPTP Problem Library", Tech. Univ. Munich, Germany, 1996

[Thagard 1978] Thagard, P. "The best explanation: Criteria for theory choice" Journal of Philosophy, 75, 76-92, 1978.

[Thagard 1989] Thagard, P. "Explanatory coherence" *The Behavioural and Brain Sciences*, 12 (3), 435-502, 1989.

[Towell & Shavlik 1993] Towell, G.G.; Shavlik, J.W. "Extracting Refined Rules from Knowledge-Based Neural Networks" *Machine Learning*, Vol. 13, no. 1, pp. 71-101, Oct. 1993.

[Turing 1950] Turing, A.M. "Computing Machinery and Intelligence" Mind 59: 433-460, 1950.

[Valiant 1984] Valiant, L. "A theory of the learnable". Communication of the ACM 27(11), 1134-1142, 1984.

[Wallace and Boulton 1968] Wallace, C.S.; Boulton, D.M. "An Information Measure for Classification" *Computer J. 11, 2,* 1968.

[Watanabe 1972] Watanabe, S. "Pattern Recognition as Information Compression" in Watanabe (ed.) *Frontiers of Pattern Recognition* New York: Academic Press, 1972.

[Watanabe 1992] Watanabe, O. (ed.) "Kolmogorov complexity and computational complexity" Monographs on TCS, Springer 1992.

[Weyhrauch 1980] Weyhrauch, R.W. "Prolegomena to a theory of formal reasoning" Artificial Intelligence 13, 133-170, 1980.

[Watrous & Kuhn 1992] Watrous, R.L.; Kuhn, G.M. "Induction of Finite-State Languages Using Second-Order Recurrent Networks" *Neural Computation* 4, 469-490.

[Whewell 1847] Whewell W. "The Philosophy of the Inductive Science" New York: Johnson Reprint Corp. 1847,

[Wiklicky 1994] Wiklicky, H. "On the Non-Existence of a Universal Learning Algorithm for Recurrent Neural Networks" in Advances in Neural Inf. Proc.Systems 6, Morgan Kaufmann, pp. 431-6, 1994.

[Winograd 1986] Winograd, T. "Thinking Machines: Can There Be? Are We?" in Winograd & Fernando Flores "Understanding Computers and Cognition: A New Foundation for Design" Norwood, 1986.

[Winston 1992] Winston, P.H. "Artificial Intelligence" Third edition, Addison-Wesley Pusblishing Company 1992

[Wolff 1995] Wolff, J.G. "Computing as Compression: An Overview of the SP Theory and System" New Gen. Computing 13, 187-214, 1995.

[Zeng et al. 1994] Zeng, Z.; Goodman, R.M.; Smyth, P. "Discrete Recurrent Neural Networks for Grammatical Inference", IEEE Transactions on Neural Networks, Vol. 5, no. 2, 320-330, 1994.