A TURING Test Scenario to Estimate an AI System's Validity

Klaus P. Jantke[†], Rainer Knauf[‡] & Thomas Abel[‡]

[†]Hokkaido University
Meme Media Laboratory
Kita–13, Nishi–8, Kita-ku
Sapporo 060
Japan

[‡]Technische Universität Ilmenau
Fak. für Informatik und Automatisierung
Postfach 100 565
98684 Ilmenau
Germany

Abstract

There is an urgent need to develop, to implement and to apply methodologies resp. tools for complex system validation. The inability to evaluate complex systems may become a crucial limiting factor to future technological progress.

The authors are engaged in a quite comprehensive endeavour towards intelligent system validation. This paper deals with both the fundamentals of a so-called "TURING Test Methodology" for intelligent system validation and some ideas to make them practicable.

First, we survey several concepts of verification and validation. Our favoured concepts are lucidly characterized by the words that verification guarantees *to build the system right* whereas validation deals with *building the right system*.

Next, we critically inspect the thought-experiment called the TURING test. It turns out that, although this approach may not be sufficient to reveal a system's intelligence, it provides a suitable methodological background to certify a system's validity. The principles of our validation approach are surveyed.

The informal discussion leads to a collection of a few formalisms suitable for a systemtic approach to intelligent system validation.

A discussion of essential problems beyond the basic TURING test methodology reveals some problems in making it practicable. Even in very simple scenarios the problem of vagueness and uncertainty in the experts' knowledge has to be managed in a reasonable way. This is due to the fact that the real target system's behaviour is not available as a yardstick. Instead, experts' knowledge has to be taken as a basis for system validation. The problem of measuring a system's behaviour against some knowledge source as insecure as human beings is analyzed.

There is motivated, developed and illustrated a family of TURING test scenarios which provides the basis of further steps towards systematic intelligent system validation. The TURING test scenario presents some main ideas how to perform a (more or less) good approximation of the target knowledge by using some (more or less competent) experts' knowledge.

Based on the precise formal concepts developed before, individual steps of performing the TURING test scenario can be described in some detail.

The test results need to be evaluated to certify a given intelligent system's validity. Formal concepts can be utilized to estimate competence, local validity, and global validity.

Contents

1 Introduction **1**

2 Validation and Verification **1**
2.1 Basic Concepts of Validation and Verification related to Human Factor Issues 1
2.2 Validation of Existing XPS 2
2.3 Validation – Building the Right System 2

3 The TURING Test as a Methaphor **2**
3.1 Razing Down the TURING Test . . . 3
3.2 Analyzing Limits of the Approach . . 3

4 The TURING Test Revisited – Towards a Validation Methodology **4**
4.1 Validation by Interrogation 4
4.2 Basic Formalizations 4
4.3 Systematic System Validation 5

5 Essential Problems Beyond the Basic TURING Test Methodology **6**
5.1 Context Conditions 6
5.2 Validation in Open Loops 7
5.3 Integrated Man–Machine Systems . . . 8
5.4 Vagueness and Uncertainty 8
5.5 Improvements by Learning 9

6 Basic TURING Test Scenarios – An Overview **9**
6.1 The Elementary Scenario 9
6.2 Knowledge Sources Underlying the TURING Test Methodology 9
6.3 Advanced Test Scenarios 10

7 Performing the TURING Test **11**
7.1 Solving Test Cases 11
7.2 Mixing the Test Case Solutions and Removing Their Authorship 11
7.3 Rating the Test Case Solutions 12

8 Evaluating TURING Test Results **13**
8.1 Estimating Local Competence 13
8.2 Estimating Local Validity 15
8.3 Estimating Global Validity 15

9 Acknowledgments **15**

References **15**

1 Introduction

The present report deals with the fundamentals of some research program proposed by Rainer Knauf and very recently published in [KP96] which aims at the development and application of a methodology to validate intelligent systems.

A closer look at the overall approach exhibits some potentials for misunderstandings. This presentation is intended to make those stumbling-blocks explicit.

In his noteworthy publication [Hal87], Halpern provides a critical assessment of the essentials of Turing's approach in [Tur50] and its usage in recent AI. At a first glance, this criticism seems to raze Turing's approach to the ground. Nevertheless, we will rebuild it or, at least, use it as a corner stone of our own approach towards validating intelligent systems.

The purpose of the present endeavour is to set the stage for more technical research towards generating test sets for validation of intelligent systems as described in [AKG96], [KP96], [HJK97a], and [HJK97b], e.g.

2 Validation and Verification

This is a brief introduction into the underlying concepts of system validation and verification. The obvious difficulty is that there are orthogonal and even contradictory opinions about these concepts. "The subject of validity is complex, controversial, and peculiarly important in psychological and educational research. Here perhaps more than anywhere else, the nature of reality is questioned." (cf. [Ker64]) As a result, there are both misunderstandings and ignorance in the topical literature.

Usually, there are many ways to react to such a Babylonian situation in science. We decided to adopt a systematically very lucid approach found in the literature.

Here, we briefly refer some basic approaches and distinguish our favoured one which is used throughout the present paper.

2.1 Basic Concepts of Validation and Verification related to Human Factor Issues

In [WW93], the authors clearly point out that "the inability to adequately evaluate systems may become the limiting factor in our ability to employ systems that our technology and knowledge will allow us to design". This bears abundant evidence of the need for an integrated approach towards validation and verification of complex systems ranging from mathematically well-based formal approaches (like [ABB+91], e.g.) to high-level philosophical and psychological approaches focusing on human factor issues (like [Sta93], e.g.). In between, there is a huge amount of variants.

The state of the art is still quite unsatisfactory. Air traffic control is a favoured application domain (cf. [FWH96], for a typical paper on modeling problems, [CLEKR96], for a representative paper on planning, and [Har93], for a paper with a cognitive science background). [Har93] illustrates both the lack of knowledge and the rudimentary state of the theory.

Thus, it is no surprise that even fundamental concepts are used in contradictory ways. Scientists with some background in cognitive sciences frequently refer to standard references like [Reb85], for instance, where *verification* is defined to be "the process of determining the truth or correctness of a hypothesis", with the more specific definition in scientific contexts as "the controlled process of collecting objective data to assess the truth of a theory or a hypothesis", whereas *validation* is characterized as "the process of determining the formal logical correctness of some propositions or conclusions" with the more specific meaning ... as "the process of assessing the degree to which a test or other instrument of measurement does indeed measure what it purports to measure". Authors with some more formal background prefer another perspective. In the area of software technology, especially in the subarea of program specification and program verification, the term *verification* refers to the more formal attempt to prove programs correct, which leads to the particular notion of "deductive programming" (cf. [Bie85], [BB93], e.g.), and, in particular, to "proofs as programs" (cf. [BC85]).

An essential flaw in the area focusing on human factor issues is to completely ignore the formal approaches developed so far. This is nicely illustrated by the claim: "The paradox is the potential production of conclusions and recommendations about verification and validation which themselves are unverified and unvalidated." ([Hop93], p. 9). For program verification approaches based on theoretical computer science, this statement is simply wrong. For illustration, the interested reader may consult the logic-based verification approach ranging from [Hoa69] to [RS93] and [Rei95], e.g., which has its own verification in mathematical logic (cf. [BM79], e.g.).

In artificial intelligence research, fortunately, a community of authors tends to invoke those concepts which are formally well-based.

2.2 Validation of Existing XPS

There has been a quite comprehensive approach to the validation of knowledge-based systems within the ESPRIT–II project VALID during 1989–92 surveyed in [MP94]. This project's goal was to undertake a comprehensive approach to the problem of Validation for existing KBS. In order to do so, several methods for different validation issues were created, and different expert systems were considered. The project's main result is a validation environment in which different expert systems can be validated. To relate this to our present endeavour, we need to make the validation concept underlying the mentioned project explicit.

In [MP94], the authors characterize their approach as follows. "First, that the structure and behavior of a KBS, viewed as a program, would be represented by a model expressed in CCR (Common Conceptual Representation). Second, that the validation method would know about and work upon the KBS model in CCR. Essentially, the metalevel approach to validation is the view that a validation tool is a program working upon a model of another program, the KBS."

Two points are important to relate our present work to the project mentioned. First, VALID makes enormously strict assumptions about the object to be validated. We consider this a case for verification (cf. [Abe96] and see the following chapter, for a specification of concepts). The high degree of assumed formal knowledge is nicely illustrated in [Mes90], where a Petri net approach is invoked for validation within VALID. Second, after the completion of VALID it has been recognized that there is still a flaw in testing methodolodies: "It seems that testing is a mandatory step in the KBS validation. However, no substantiated testing methodology for KBS is available and often knowledge engineers are guideless in the testing phase." (cf. [MP94]) Adressing this problem area is exactly the aim of our forthcoming work for which the present paper is intended to set the stage.

2.3 Validation – Building the Right System

O'Keefe and O'Leary (cf. [OO93]) found a quite intuitive and systematic approach to characterize and to distinguish *verification* and *validation* by the two circumscriptions of

- *building the system right* and
- *building the right system*,

respectively. The first property relates a system to some specification which provides a firm basis for the question whether or not the system on hand is *right*. In contrast, the latter formulation asks whether or not some system is considered *right*, what somehow lies in the eye of the beholder. The essential difference is illustrated below.

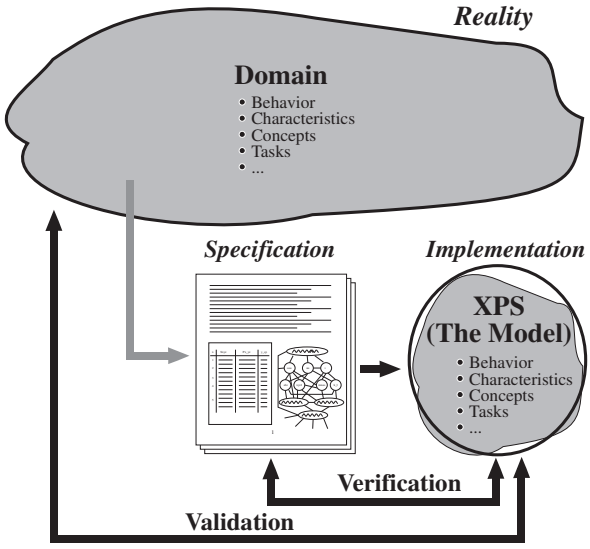

Figure 1: Relating Validation and Verification

In [Hop90], the author expresses a similar opinion by the words that "knowledge acquisition systems as well as knowledge engineers must ensure that knowledge is 'right' with respect to the users intention."

Throughout the present paper, we adopt this perspective and put the emphasis of our investigation on the validation issue. Our further work is focussing on the systematic development of test sets and on the development of some methodology for system validation by systematically testing.

3 The TURING Test as a Methaphor

First, the classical approach (cf. [Tur50], for the origin, and [Gin93], for an easy introduction), although well-known, is briefly outlined to be compared to the validation approach to which the present paper focusses. Very roughly, the TURING Test means a scenario in which an artifact, usually a computer system, is checked for "intelligent behaviour". In the basic setting, a human examiner is put into a dialogue situation with some remote partner not knowing whether or not this partner is another human or just a computer system. In case the human interrogater is convinced to be in contact with another human, this is understood as a sufficient criterion for calling the system under examination "intelligent". The discussion about the nature of intelligence is postponed, for a moment.

3.1 Razing Down the TURING Test

The idea seems acceptable. Faced to the absence of any sufficient consensus on the notion of intelligence, this concept is specified implicitly rather than explicitly. But a closer look exhibits that the question for the nature of intelligence is not only postponed, it is just missed.

Halpern's criticism quite convincingly developed in [Hal87] can be summarized as follows: The so-called TURING test does not reveal anything about the inner "intelligence" of an interrogated system, it might rather answer the question whether a computer can be programmed so as to fool human beings, with some regularity, into thinking that it is another human being. By numerous implementations intended to demonstrate instances of the TURING test, this has been exhibited, indeed. "The test is, in short, not an experiment but a trap" ([Hal87], p. 83).

There are some further investigations in the spirit of Turing's approach which are of some relevance to the issue discussed here. In [Kak96], e.g., the author asks for refined concepts of intelligence which describe the target phenomenon as a graded one. His approach is substantially based on findings on animal intelligence. His references to experiments exhibiting pigeons' classification skills are quite illuminating. Obviously, Turing did not make an attempt towards the gradation of intelligence.

To sum up, neither the TURING test reveals a system's intelligence nor seems an ungraded approach like this appropriate to deal with phenomena as complex as natural resp. machine intelligence.

3.2 Analyzing Limits of the Approach

In the remaining part of the present chapter, we want to analyze Turing's thought-experiment to identify substantial ideas which might survive the fundamental criticism cited above. It seems that the question whether or not this thought-experiment may be reasonably interpreted heavily depends on certain characteristics of the interrogated computer program.

For illustration, assume any conventional computer program solving routine tasks with an enormous precision and within a remarkably short time. Numerical computations in arithmetics provide the simplest and quite typical instances of this application domain. Two insights are essential. First, these are the cases in which a computer program is usually not understood to be "intelligent", although its computational power exceeds the intellectual power of every human being enormously. Second, human interrogators will normally quite easily identify the hidden partner to be a computer and not another human being. To sum up, there is a class of computer programs implementing straightforward computations to which the TURING test approach does not apply, by nature. Number crunching does not relate directly to artificial intelligence.

More explicitly, computer programs intended to perform straightforward deterministic computations were not called intelligent, if they would behave like human experts. Even worse, such a behaviour would be usually a strong indication of their incorrectness.

Similarly, it is quite unlikely that airplanes flying like birds, ships swimming (and occasionally diving) like dolphins, e.g., and trains behaving like a herd (or, more moderate, like a caravan) of camels are considered a success of science and technology.

The following seems to be the essence of the comparisons under investigation: In application domains where there exists a restricted and well-defined target behaviour to be implemented by some artifact (some computer system, in our case), nature rarely provides better solutions. Beyond those deterministic problem domains, artificial intelligence research and engineering aims at solving problems by means of computer systems that are less deterministic and less well-specified or, even worse, possibly unspecifiable. In many cases, humans deal with these problems quite adequately. Moreover, some humans are even called experts. There is not much hope to find a formal characterization of problem domains to which artificial intelligence approaches typically apply so as to separate them clearly from other domains.

However, for clarity of our subsequent investigations, we circumscribe a certain type of problems in a partially formal way. In typical application domains of artificial intelligence like classification, game playing, image understanding, planning, and speech recognition, input data frequently occur in a well-formalized form. They are attribute/value vectors, configurations on a chess board, pixel arrays, and so on. Generally, there are several acceptable ways to react to certain input data. Given symptoms usually have multiple interpretations, in most positions on a chess board, there are several reasonable moves, and so on. Thus, such a system's correct behaviour is more suitably considered a relation than a function. Humans rarely behave functionally. Thus, in its right perspective, programs implementing functions are inappropriate subjects of the TURING test, whereas the relational case might be revisited.

4 The TURING Test Revisited – Towards a Validation Methodology

Throughout the sequel, we assume that some desired target behaviour may be understood as a relation $\mathcal{R}$. With respect to some domain of admissible input data I and some space of potential outputs O, the system's behaviour is constrained by $\mathcal{R} \subseteq I \times O$ and by further problem specific conditions which might be difficult to express formally[1].

The validity of a system means some fitness with respect to $\mathcal{R}$. The crucial difficulty is that in most interesting application domains, the target behaviour $\mathcal{R}$, in contrast to a given specification underlying verification, is not directly accessible. It might even change

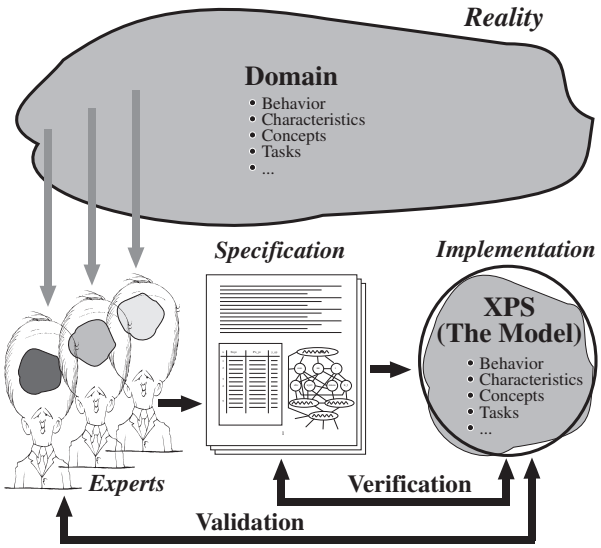

Figure 2: Relating Validation and Verification Taking Experts as the Basic Knowledge Source Into Account

over time. Consequently, the illustration of figure 1, which was intended to relate and to distinguish validation and verification, needs some refinement.

At a first glance, it seems that the concept of a system's validity refers to building the right system with respect to the needs of the application domain. This might be the ultimate goal, indeed. However, the real situation illustrated by the figure above suggests that there might be transmitters between reality and the AI system under investigation[2]. Consequently, one has to measure a system's performance against the experts' expectations. For this purpose, the TURING test approach will rise again like a phoenix from the ashes.

4.1 Validation by Interrogation

Assume some system implemented to meet some given target behaviour circumscribed by a relation $\mathcal{R}$ as introduced above. Assume furthermore that this system is put into a test scenario as proposed by Turing in [Tur50] and discussed above. To cap it all, assume that this system passes several interrogations by human experts successfully.

What does such a successful TURING test reveal?

If several human experts admit that the hidden probationer – in our case, the system under consideration – behaves like another expert, this obviously certifies the systems validity, at least up to the competence of the examining experts.

This insight is setting the stage for a TURING test approach to intelligent systems validation. The following chapter is intended to provide a few essentials of this approach.

4.2 Basic Formalizations

Within the present paper, we confine ourselves to an introduction of the basic approach. More details can be found in [AKG96], [KP96], and [HJK97b], e.g., and further publications are in preparation. The following chapter 5 will layout some necessary generalizations of the basics developed here.

An in-depth investigation of validation problems requires some minimal terminology. So far, even the problem to be attacked is not yet well-defined. What we need are a few formal concepts to characterize, at least partially, the target behavioural phenomena of the application domain, to relate the experts' knowledge to the domain, and to relate a knowledge-based system's behaviour via the experts and their knowledge to the target phenomenon.

In the minimal formal setting assumed so far, there are only two sets I and O. On these sets, there is somehow given a target relation $\mathcal{R} \subseteq I \times O$. One may assume that $\mathcal{R}$ is non-trivial, i.e. $\emptyset \subset \mathcal{R} \subset I \times O$ holds.

Under the assumption of some topology on I, one may assume that $\mathcal{R}$ is decomposable into finitely

[1] Game playing like chess provides an excellent illustration. There are only a finite number of possible positions on a chess board and, thus, there do exist formal expressions definitely telling whether any given move belongs to some winning strategy or not. Nevertheless, no formal condition like that is known so far. It might be rather complex and, therefore, unfeasible.

[2] There is a particular area of applications where certain AI systems directly interact with some target environment like in robotics, e.g. Under those circumstances, validation techniques must become similar to experimentation like in physics. This is beyond the scope of our work.

many functional components $\mathcal{R}_i^y$, where y ranges over some finite subset of O and i ranges over some finite index set, such that (i) $\mathcal{R} = \bigcup \mathcal{R}_i^y$, (ii) $\mathcal{R}_i^y \subseteq I \times \{y\}$ (for all i and y), and (iii) every $\mathcal{R}_i^y$ is convex.

Informally, (i) is stating that the set union of all $\mathcal{R}_i^y$ really exhausts the entire target relation $\mathcal{R}$. (ii) is specifying that every individual $\mathcal{R}_i^y$ is, so to speak, functional with the uniquely determined function value y. Finally, convexity is a topological property which requires that between any two elements of some given region $\mathcal{R}_i^y$ there do exist only elements of this particular region.

Although these conditions seem rather natural, are frequently assumed (in most classification tasks, e.g.), and allow for certain algorithmic concepts as in [AKG96], e.g., they are already more specific than required here. Therefore, we drop the requirement of convexity as well as the finiteness of the decomposition. As a minimal setting, we stick with conditions (i) and (ii), only.

An expert's knowledge should be consistent with the target phenomenon. This is an elementary requirement to characterize expertise. For illustration, if I comprises all admissible configurations on a chess board and elements of O are moves of one player, for every configuration $c \in I$, every expert's response $mv \in O$ should meet the condition $(c, mv) \in \mathcal{R}$.

On the other hand, everybody who is not providing any response at all (formally: $\mathcal{E} = \emptyset$) is always consistent. Thus, one needs another requirement of expertise to complement consistency. Informally, from the possibly large amount of answers to an admissible question, an expert should know at least one.

We formalize expertise as follows. An expert's knowledge about some target phenomenon $\mathcal{R}$ as introduced above is assumed to be a particular relation $\mathcal{E} \subseteq I \times O$ such that the following requirements of expertise are satisfied.

$$\mathcal{E} \subseteq \mathcal{R} \qquad \textbf{[Exp1]}$$

$$\pi_I(\mathcal{E}) = \pi_I(\mathcal{R}) \qquad \textbf{[Exp2]}$$

For notational convenience, we have introduced π_I to denote the projection of (sets of) pairs from $I \times O$ to the first argument. In some sense, **[Exp1]** is a condition of consistency and **[Exp2]** is a condition of completeness.

A team of n experts with their respective domain knowledge $\mathcal{E}_1, \ldots, \mathcal{E}_n$ is said to be competent, exactly if it meets the equation of competence:

$$\bigcup_{i=1}^{n} \mathcal{E}_i = \mathcal{R} \qquad \textbf{[Cmpt]}$$

Note that this very basic approach allows for a number of interesting further concepts like redundancy of experts, minimal bases of expert teams, and so on. However, this is out of the scope of the present paper.

Only one property is worth to be supplemented, as it might be useful when validating systems. An ith expert (or a system, lateron) is called omnipotent, exactly if it satisfies

$$\mathcal{E}_i = \mathcal{R} \qquad \textbf{[Omn]}$$

It might be usually unrealistic that, starting with some not explicitly given target phenomenon formally understood as some relation $\mathcal{R}$, one tries to find a competent team of experts. Vice versa, every team of experts is implicitly determining its own area of competence by the equation **[Cmpt]**. Practically, because of not having $\mathcal{R}$, we estimate $\mathcal{R}$ by **[Cmpt]**. That is, because we can't speak about "grey arrows" (cf. figure 2 above). In case some intended system's behaviour lies completely outside the knowledge of all available experts[3], there is no reasonable way to invoke this missing knowledge for system validation or to measure the system's validity against this piece of knowledge.

4.3 Systematic System Validation

Based on the minimal formalisms provided, we are now able to develop our focused validation scenario.

- There is assumed some application domain in which a system's desired target behaviour is implicitly given as a relation $\mathcal{R} \subseteq I \times O$.
- There is a team of n experts competent for the problem under consideration, i.e. meeting $\bigcup_{i=1}^{n} \mathcal{E}_i = \mathcal{R}$.
- There is some system to be validated. Its input/output relation is called $\mathcal{S}$.

Ideally, a system is omnipotent, i.e. meets

$$\circ\ \mathcal{S} = \mathcal{R} \qquad \textbf{[S–Omn]}$$

or, at least, satisfies the requirements of consistency and completeness.

$$\circ\ \mathcal{S} \subseteq \mathcal{R} \qquad \textbf{[S–Cons]}$$

$$\circ\ \pi_I(\mathcal{S}) = \pi_I(\mathcal{R}) \qquad \textbf{[S–Compl]}$$

[3] Again, chess playing illustrates the problem quite well. There are, for sure, many configurations on the chess board (formally: $c \in I$) such that some particular good move (formally: $mv \in O$ such that $(c, mv) \in \mathcal{R}$) is not known to any expert (formally: $(c, mv) \notin \bigcup_{i=1}^{n} \mathcal{E}_i$), so far. It does not make any sense to blame a chess playing computer program invalid, if it does not "know" this particular move.

But these three properties relating a given system's behaviour directly to the target phenomenon are usually undecidable or, at least, unfeasible. Thus, as motivated above and as illustrated by our figure 2, validation deals with relating the system's behaviour to the experts' knowledge.

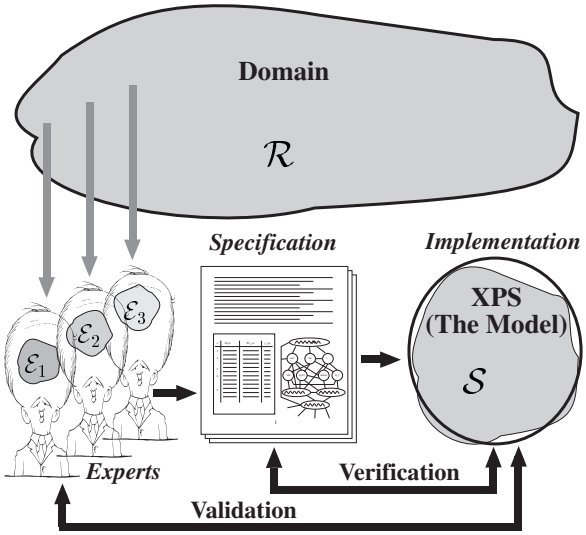

Figure 3: Relating Validation and Verification Based on a Minimal Setting of Formalized Human Expertise

Figure 3 is incorporating the minimal setting of formal concepts developed within the present paper.

The TURING test approach deals with a methodology to provide systematically some evidence for the validity of **[S–Cons]** and **[S–Compl]**.

Informally, the TURING test scenario for system validation is based on the following intuition.

1. Some team of experts meets **[Exp1]** and **[Exp2]**.
2. It's desirable, but unprovable, whether or not some system meets the corresponding conditions **[S–Cons]** and **[S–Compl]**.
3. If the system, when systematically interrogated, behaves like some team of experts (satisfying **[Exp1]** and **[Exp2]**), this is understood as some strong indication that the system is satisfying **[S–Cons]** and **[S–Compl]**, correspondingly.

Intuitively, nothing better could be imagined. In general, there are no proofs of properties involving $\mathcal{R}$, like **[S–Cons]** and **[S–Compl]**, e.g. The key reason is that the target phenomenon $\mathcal{R}$ might be far beyond formalizations, whereas $\mathcal{S}$ is completely formal.

The research to be undertaken concentrates on the derivation of suitable interrogation strategies. Especially, one needs to know "what questions to ask". This is – in formal terms – the question for suitable[4] test sets. Our further work as already described in [AKG96] is focussing on test set generation for use in a TURING test scenario as outlined here. The formal concepts introduced above will form a basis for the characterization of several types of automatically generated test sets.

5 Essential Problems Beyond the Basic TURING Test Methodology

Our paper deals with quite general problems of intelligent systems' validation. Therefore, it seemed methodologically reasonable to start with some oversimplified setting in which a system is assumed to simply implement some target relation. The simplicity of the approach allows for the lucid introduction of fundamental properties like consistency, completeness, and competence. These concepts occur in more complex settings as well, but are considerably more involved and more difficult to analyze.

The present chapter is intended to systematically introduce a few basic generalizations of the elementary approach above. Every issue is addressed by only a very short introduction which should be understood a launching pad for future work. We hope to extend the basic approach of the present paper towards covering the extensions sketched in the sequel.

Last but not least, this chapter – by pointing to obviously inevitable extensions of the elementary approach above – aims at stimulating some in-depth discussion about the intrinsic complexity of realistic validation concepts.

5.1 Context Conditions

Even in case we stick with the relational approach, to model expert knowledge appropriately it might be necessary to express certain context conditions. Within some relation $\mathcal{R}$, knowing or believing some $(x, y) \in \mathcal{R}$ might be usually coupled to know resp. believe some $(x', y') \in \mathcal{R}$, as well. Analogously, some $(x, y) \in \mathcal{E}_i$ might rule out some other knowledge (x', y') from the expert's perspective.

It seems promising to express context conditions by similarity concepts. This brings the fundamentals of our validation approach remarkably close to case-based reasoning.

[4] For instance, suitability of a set of test cases may refer to completeness properties as formally expressed by **[Exp2]** and **[S–Compl]**, respectively. But this is already beyond the scope of the present paper.

Here, we sketch only a very rough idea, for illustration. Assume any binary function σ called a similarity measure[5] with $\sigma : (I \times O) \times (I \times O) \to [0,1]$. In the simplest case, σ may be binary. A similarity measure can be used to rule out incompatibilities from an experts knowledge.

$$\sigma((x,y),(x',y')) = 0 \implies \neg(\ (x,y) \in \mathcal{E}_i \wedge (x',y') \in \mathcal{E}_i\ ) \qquad \textbf{[Incomp]}$$

This approach may be refined immediately, if one replaces the strong assumption $\sigma((x,y),(x',y')) = 0$ by the more moderate version $\sigma((x,y),(x',y')) < \varepsilon$ for some ε. Even more important, this suggests a dual version

$$\sigma((x,y),(x',y')) > \eta \implies (x,y) \in \mathcal{E}_i \Rightarrow (x',y') \in \mathcal{E}_i \qquad [\eta\textbf{–Coh}]$$

expressing some coherence of expert's knowledge. For the reader's convenience, we provide a sample interpretation of these formalisms. **[Incomp]** expresses that two knowledge pieces which are extraordinarily unsimilar (perhaps, even contradictory) can not simultaneously belong to one expert's knowledge. In contrast, the condition [η–**Coh**] of η–coherence states that knowing one fact implies the knowledge of every other one provided the similarity of them exceeds η.

To mention just a final idea, which is, as far as we know, a novelty in this area: One might measure an expert's confidence by the variation of σ over his base of knowledge, e.g.

There is an overwhelming number of further ideas, approaches, and even first results, which are postponed to a forthcoming investigation.

5.2 Validation in Open Loops

The crucial drawback of the approach developed so far is that it is based on a perspective under which an intelligent system's behaviour appears as time-independent and ahistorical.

In contrast, a historical perspective assumes that some action – either performed by a human or by a computer – may depend on the steps performed before, not only on the current situation. In particular, if some situation can been reached along different paths, the next step may be determined in dependence on the current path[6].

[5] In the future, we might prefer structural similarity concepts (cf. [Jan94]), because of their expressive power.

[6] In chess playing like in other games, a certain configuration may be reached by moves according to some player's strategy. This strategy may determine his next move. In another game, he might arrive at the same configuration from another point driven by another strategy. Thus, his next move might be different from the one in the earlier game.

To reflect phenomena like this in some formalism, we need to abandon or, at least, to substantially generalize the relational approach.

Essentially, the use of an intelligent system takes normally place in an open loop of environment–machine interactions. Sometimes, the environmental inputs are human-made, sometimes not. If the system under consideration is autonomously working remote from human beings, in particular, the number of interactions is unforeseeable and, thus, potentially infinite[7]. There is usually no way to estimate the number of interactions sufficiently precise. Thus, the following formalization seems reasonably simple and adequate.

Instead of $\mathcal{R}$, some target behaviour $\mathcal{B} \subseteq (\ I \times O\ )^*$ contains infinite sequences of action–response pairs. Even more specific, we formally assume $\mathcal{B} \subseteq (\ I \times O\ )^+$ to exclude the empty interaction sequence λ. Formally, for any set M, we have $M^+ = M^* \setminus \{\lambda\}$. As man–machine interaction will usually last only for a finite time, one need to reason about initial segments of those infinite sequences. By $\sqsubseteq$ we denote the prefix relation among words which is canonically extended to the case that the second argument belongs to $\mathcal{B}$. Furthermore, $A \sqsubseteq B$ abbreviates the case that for every element $a \in A$ there is some element $b \in B$ such that $a \sqsubseteq b$ holds. Finally, we need a generalization of the projection concept. Both for finite and for infinite sequences $s = (x_1,y_1)\ldots(x_n,y_n)$ resp. $s = (x_1,y_1),(x_n,y_n),\ldots$, the term $\pi_I(s)$ abbreviates $\{x_1,x_2,\ldots\}$, accordingly. π_I is extended to sets via $\pi_I(S) = \bigcup_{s\in S} \pi_I(s)$, as usual. Expert activities are assumed to be finite, i.e.

$$\mathcal{E} \subseteq \bigcup_{i=1}^{\infty} (\ I \times O\ )^i \qquad \textbf{[Fin]}$$

Based on this extension, consistency and completeness can be rewritten as follows.

$$\mathcal{E} \sqsubseteq \mathcal{B} \qquad \textbf{[Exp}^*\textbf{1]}$$

$$\pi_I(\mathcal{E}) = \pi_I(\mathcal{B}) \qquad \textbf{[Exp}^*\textbf{2]}$$

As before, teams of experts are of interest. Here, we refrain from rewriting the other formalizations discussed in chapter 4 of the present paper.

The reader is invited to ponder about a suitable combination of the extensions of chapter 5.1 and the present one. Even more cumbersome, he should be prepared to carry over the formalisms from these two chapters to the considerably more general perspective provided within the following one.

[7] As Gurevich says, "the infinite is a good approximation of the finite." [Gur92]

5.3 Integrated Man–Machine Systems

The behaviour of intelligent systems in use is not only determined by the functionality inherent in the engineering design. Most systems solve the problems towards their solution they have been designed within an interactive process between the operators and the system. It is not really a very recent insight that it is "possible to view an organization such as an air-defense, man-machine system as a single organism" and that it is "possible to study the behavior of such an organism" (cf. [Por64]). However, *viewing* in Porter's sense does not mean *formalizing* in the sense of our present paper.

From the viewpoint of cognitive psychology, there is paramount work both on an integral view of man–machine systems and on the cognitive processes of teams of human beings. [Hut95] is an interesting source, in some respect.

The necessity of further steps towards an integrated treatment of man–machine systems lies in the importance of those systems. Air traffic control, as mentioned earlier, is just one prototypical application area. [BCM96] nicely illustrates the degree of interaction in an integrated system composed of a human being working as an air traffic controller and a system for air traffic conflict resolution. The ultimate system's behaviour evolves as a result of the highly dovetailed activities of both – the human controller and the intelligent system. It is said that this interactiveness has been crucial in gaining a degree of acceptance.

To employ further computer science methods and results, one would need a considerably higher degree of formalization, somehow similar to our attempts in the preceding chapters. But this seems difficult: "What is needed is a methodology which accepts the impossibility of accurately predicting the future but which nevertheless seeks to permit rational decision-making." (cf. [Fos93])

Does this mean that the validation of integrated man–machine systems lies outside the reach of computer science methodologies?

We don't think so. Basically, there are at least two types of approaches. First, one might approach the target concept of validity by even weaker concepts understood to express a minimum of desirable properties. It seems that the idea of resilence (cf. [Fos93]) is of this type. Second, one may extend test scenarios like those proposed in the present paper to composite systems as considered here.

About resilence, we confine ourselves to a few words. A key intention of approaches like the one in [Fos93] is to identify "elements of resilence" which imply "the ability to accommodate change without catastrophic failure". System validation is than reduced to validating the presence of those elements of resilence.

The second approach – which, naturally, is our favoured one – may be implemented by developing something like a higher-level TURING test approach in which such a composite system, i.e. a pair of a human operator together with his mate which is an intelligent computer system, is put into some test scenario for systematic interrogation. This seems to offer a properly novel perspective.

We find it worth to reserve an in-depth discussion to a forthcoming paper.

5.4 Vagueness and Uncertainty

So far, the concepts investigated and formalized are basically deterministic. Only the idea of similarity concepts brought in some vagueness. Usually, there are several sources of vagueness and uncertainty. Here, we do not consider the problem of vagueness and uncertainty in application domains and, accordingly, in modelling. This is not our turn.

There is an intrinsic fuzzyness of the approach to validate intelligent systems against the opinion of some team of human experts. If a subject is difficult, humans rarely agree completely. This applies in particular to those complex application domains where AI systems are intended to work.

Even under the assumption of consistency and completeness of experts' knowledge (cf. **[Exp1]** and **[Exp2]** resp. **[Exp*1]** and **[Exp*2]**) it might happen that experts do not agree on several pairs of input and system's response. This is due to the fact that – in formal terms – the symmetric difference of some $\mathcal{E}_i$ and some $\mathcal{E}_j$ might be non-empty. Several methodologies like in [KP96] are intended to cope with this phenomenon.

For readability, we supplement the definition of the symmetric difference of any two sets A and B:

$$A \Delta B \;=\; (A \setminus B) \cup (B \setminus A) \qquad \textbf{[SymmDiff]}$$

Loosely speaking, the symmetric difference of two experts' knowledge is all what the one knows, but not the other.

From the TURING test perspective, to cope with vagueness and uncertainty in the experts' knowledge, one might arrange an interrogation setting in which statistical results are evaluated (cf. [KP96]).

5.5 Improvements by Learning

It might be considered the crucial limitation of our overall approach to intelligent systems validation that it aims at validity statements only within the limits of human expertise.

Indeed, all imaginable steps beyond these limits are steps beyond the nature of the TURING test idea which – in essence – relates a system's behaviour to the behaviour of human experts. As usual, making such a limit explicit is bringing us close to ways for crossing the border. As this exceeds the scope of the present paper, we confine ourselves to a single paragraph sketching one basic idea.

Experimentation[8] with a system under inspection may lead to the discovery of knowledge outside the competence (cf. **[Cmpt]**) of the available team of experts, i.e. – in formal terms – cases in the difference $\mathcal{R} \setminus \mathcal{E}$ resp. $\mathcal{B} \setminus \mathcal{E}$. Systematic knowledge discovery of this type requires a careful design of experiments as well as some methodology for analyzing the results of experimentation.

Throughout the remaining part of this chapter, we focus on learning within the present setting.

Basically, the TURING test approach is tailored towards the identification of a system's validity with respect to the available expertise of some team of human experts. In the simplest case, the test's result is either to certify validity or not. In a more realistic setting, validity is usually certified to some degree. Especially if the validity statement is a graded one, it is a highly desirable extension of the underlying approach to work towards improving a system's validity when examining it.

This idea is essentially based on the insight that system validation understood as the process of determining validity proceeds in steps over time. During this process, one might work towards improving the system's validity for passing the TURING test more successfully. The system is learning when being examined.

The reader may easily recognize that this properly extends the basic approach. It assumes some behaviour of the system under investigation and, thus, is far beyond the limits of validation. However, it is worth to be mentioned that the basic formalizations introduced above provide enough precision to clarify essential learning approaches.

[8] Chess playing shall provide us some final illustration. Computer programs playing chess can systematically try moves which human experts never tried, so far. In being sometimes successful, they may occasionally discover knowledge beyond the competence of present experts.

Systematically, there is some further learning problem related to the present investigation: Experts that learn during system validation to improve their competence and to do a better job in validation. As far as we know, this is a novelty, as well.

6 Basic TURING Test Scenarios – An Overview

As investigated in [JAK97] and briefly reported above, the TURING test scenario is assuming that there is someone resp. something sitting in a black box and being interrogated. Although this interrogation methodology might not be appropriate to reveal the "intelligence" of the one hidden in the black box, it might be quite appropriate as a basis for validation as discussed above. There are variants of a different sophistication in implementing these ideas.

6.1 The Elementary Scenario

If we take the discussion of the TURING test scenario literally, the key approach consists in placing the expert system to be validated into some black box and asking experts whether or not they evaluate the behaviour of the "expert" in the box as *competent*, i.e. the system as *valid*.

There are several psychological objections against such an elementary approach, although there might be sometimes no better solution than asking the one and only available expert for his opinion. Difficulties of that type can not be solved by computer scientists. We refrain from further discussions which lead into psychology, social sciences, or even politics.

Instead, we proceed with a brief critical assessment of the knowledge sources available for a validation procedure following the TURING test paradigm. This analysis presented within the sequel will be used as a guideline to develop a refined TURING test scenario in the closing part of this chapter.

For the subsequent investigation, the reader should recall the formalisms developed in chapter 4 above.

6.2 Knowledge Sources Underlying the TURING Test Methodology

Artificial Intelligence application fields are usually highly complex ones, where there is often no commonly accepted standard for the knowledge in these fields. That means there is no truly objective source of knowledge. Furthermore, there is usually no model which can be used to prove whether a given knowledge base is a correct representation of reality.

If there was such a model, it would have been used already for the system's design, thus obviating the need for a knowledge-based system.[9]

Formally speaking, it means that we don't have the knowledge $\mathcal{R}$. So the only way to achieve a nearly objective validation technique is to use the expert's knowledge, i.e. to approximate $\mathcal{R}$ by $\bigcup_{i=1}^{n} \mathcal{E}_i$.

This, of course, causes a bunch of problems, not only those mentioned in [JAK97]:

- It may happen that there is an $\mathcal{E}_i$ with $\mathcal{E}_i \setminus \mathcal{R} \neq \emptyset$, i.e. a certain expert's knowledge is really wrong. More formally, **[Exp1]** may be violated.

- It may happen that a certain expert's knowledge is wrong and the system incorporates the same "defect", i.e. this "defect" can't be detected by this expert.[10] This is formally covered by **[Cmpt]** reflecting the insight that a test methodology based on $\bigcup_{i=1}^{n} \mathcal{E}_i$ can not go beyond these limits, by nature.

- There may be contradictions between different experts. This problem is only partially and implicitly covered by **[Exp1]** and **[Cmpt]**.

It might even be the case that there are further essentials which can not be expressed by the formalisms developed so far. Thus, there might be some feedback resulting in an extension of the fundamentals provided in [JAK97] and briefly discussed above.

At a first glance, it appears useless to invoke the same source of knowledge for validation which already has been used for the system's design. However, there are some aspects which make this approach very useful nevertheless[11]. The deeper reasons are of a different type:

1. There are psychological reasons (cf. [PAG92] and [Gor92], e.g.) beyond the scope of the present paper. Just for illustration, as humans are usually not aware of their own limitations, they are not aware of their own potentials, as well. Thus, appropriately guided experts can provide much more support in system validation than they would expect of themselves.

2. Competence is, of course, a property of experts, which isn't distributed homogeneously in the field of their expertise. Furthermore, all experts are not equally competent in a certain problem solving task. In its right perspective, this is closely related to the phenomenon of graded intelligence (cf. [Kak96]).

3. Furthermore, there are usually disciplinary reasons in the respective application domain which determine the fact that even contradicting strategies of experts are acceptable. This is nicely illustrated in strategic games like chess, for instance (cf. [PR87]).

From these insights, one might draw several conclusions. First, there is an obvious need of further research covering all the domains of relevance. Second, this research, in particular, and the development of artificial intelligence, in general, seems to depend essentially on more interdisciplinary work than we are used to currently. All these are long-term projects.

For our present endeavour, we suggest to apply at least the following consequence when developing an approach to validate intelligent systems:

- to the maximum extent possible, involve experts with different views on the application field who may have contrary opinions and

- to the maximum extent possible, involve experts who were *not* involved in the design process of the knowledge-based system to be validated.

For the purpose of the present approach, we develop one scenario in some more detail.

6.3 Advanced Test Scenarios

Every scenario has some parameters which might be tuned in the one or the other way, thus, representing a family of closely related approaches. The suggested methodology is quite similar in concept to the TURING test. Let's have

- *one* expert system which is to be validated,

- n experts,

- a quasi–exhaustive or a smaller set of m "good" test cases[12],

- *two* ratings $\{0, 1\}$, in which 1 means to be "correct" and 0 means to be "incorrect", and

- *two* values of certainty $\{0, 1\}$, in which 1 means to be "sure" and 0 means to be "unsure". (A finer gradation is postponed, for a moment.)

[9] The *real* objective of separating knowledge and representing it in a problem oriented manner is to facilitate its change. And the *real* reason for needing to change the knowledge is always to make it more and more correct, i.e. it isn't correct yet.

[10] This is definitely the case, when a system's designer becomes a validator.

[11] The most convincing argument is, of course, that in most cases this seems to be the only way.

The idea of the TURING Test methodology as illustrated in figure 3 is divided into four steps:

1. solving of the test cases by the expert validation panel as well as by the expert system to be validated,

2. randomly mixing the test case solutions and removing their authorship,

3. rating all (anonymous) test case solutions, and

4. evaluating the ratings.

We refrain from the interesting discussion of concurring scenarios and focus on the simple one illustrated and sketched above.

Implementing any scenario always means to fix a lot of further details. Based on the formalisms developed in [JAK97] and introduced in the preceding chapter 4, we are going to explain the above steps in detail in the following chapter.

7 Performing the TURING Test

In chapter 4 we have introduced the fundamentals adopted from [JAK97]. Chapter 5 has been devoted to further questions intended to widen the horizons. However, the usage of the approaches of this chapter are still beyond the limit of the present more fundamental investigation. Chapter 6 has lead to a family of closely related basic scenarios of system validation. It remains to be discussed how to implement these ideas.

Implementing a test scenario as illustrated above means to make the system under validation running, to get the experts engaged, to set up an appropriate validation environment, to prepare for the evaluation of validation runs, and many further activities up to documentation.

Here, we deal neither with social problems nor with formal administrative arrangements. The focus of our investigation is on methodological problems, i.e. on arranging the intellectual setting. We have to clarify which data to acquire and how to process them.

The next chapter demonstrates how several ways of using the formalisms developed may result in validity statements which are the ultimate goal of our overall endeavour.

[12] Steps towards the generation of sets of "good" test cases are the subject of other papers published by the authors and their group like [AKG96], [HJK97a], and [HJK97b], e.g.

7.1 Solving Test Cases

Having m test cases, each test case has to be solved by both

- the n (human) experts $E_1, \ldots, E_n$ who realize the expertise

$$\bigcup_{i=1}^{n} \mathcal{E}_i = \mathcal{E} \subseteq I \times O$$

  and

- the *one* expert system E_{n+1} which realizes the system's knowledge

$$\mathcal{E}_{n+1} = \mathcal{S} \subseteq I \times O$$

  and which is being validated.

This leads to $m \cdot (n+1)$ *solved test cases*. Each *solved test case* contains

- the test case t_j $(1 \leq j \leq m)$ which is an input of both the expertise and the system $(t_j \in I)$,

- the test case solver E_i $(1 \leq i \leq n+1)$ who has share in the expertise $(\mathcal{E}_i \subseteq \mathcal{E})$, and

- the test case solution $s(E_i, t_j) = s_{ij}$ which is an output of both the expertise and the system $(s_{ij} \in O)$.

Thus, a *solved test case* can be represented by a triple

$$[t_j, E_i, s_{ij}]$$

which is an element of $I \times \{E_1, \ldots, E_{n+1}\} \times O$.

The test case solution s_{ij} is either a final conclusion of the system or a special test case solution value *unknown*. The latter solution value gives the experts an opportunity to express their own incompetence in solving a specific test case.

7.2 Mixing the Test Case Solutions and Removing Their Authorship

To ensure that the human experts are not aware of a solution's author (and especially which is the system's solution and which is their own), the solved test cases are mixed and presented to the (human) experts $E_1, \ldots, E_n$ in a random sequence [13] and without any information about the authorship.

[13] From a practical standpoint this mixing procedure should be carried out only within a considered test case; this gives the (human) experts the opportunity to compare the different solutions for a certain test case. But for each considered test case this, random sequence should be a different one.

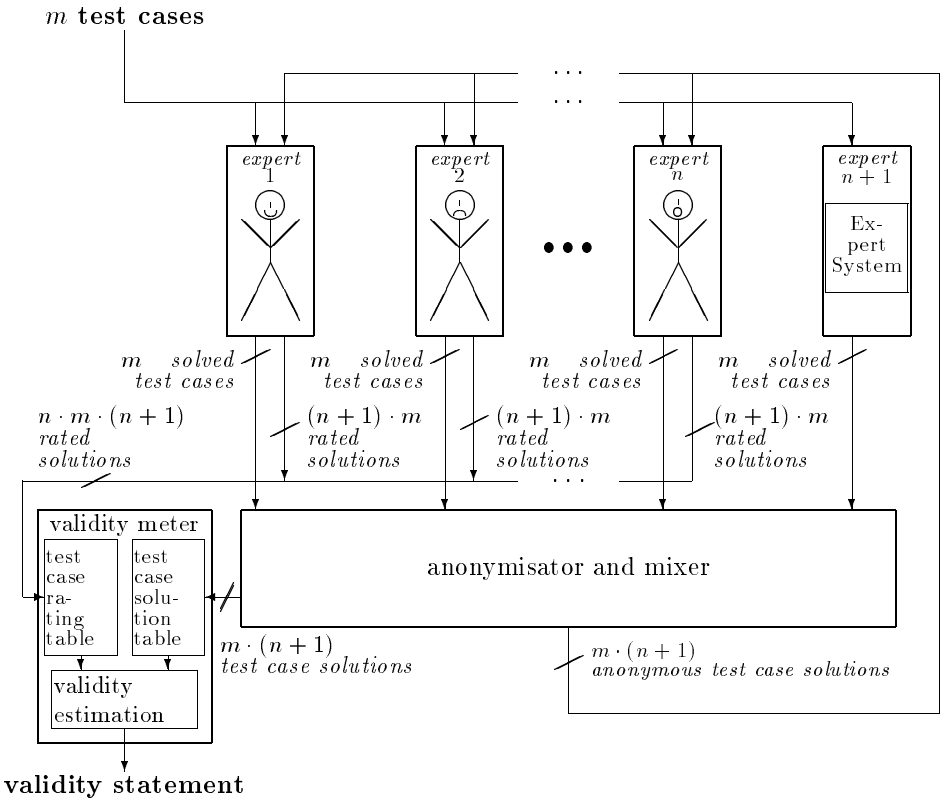

Figure 3: A survey of the TURING Test to estimate an AI System's validity

As a result of this procedure each of the (human) experts $E_1, \ldots, E_n$ gets $m \cdot (n+1)$ solved test cases – but without the information, who was the solver E_i. That means, they get $m \cdot (n+1)$ *anonymous test case solutions* which can be represented as pairs

$$[t_j, s_{ij}]$$

In addition to that, the assignment of solutions to their author (a certain expert $E_1, \ldots, E_n$ or the expert system E_{n+1} with $\mathcal{E}_{n+1} = \mathcal{S}$) should be kept for evaluation purposes. That's why this procedure also provides the complete solved test cases, i.e. the whole triples $[t_j, E_i, s_{ij}]$ *before* it begins with it's job of removing the authorship and mixing the solutions.

7.3 Rating the Test Case Solutions

The job of an expert (who is a validator now) is to evaluate the $n+1$ solutions for each of the m test cases separately and without knowing the solvers. Among these solutions are both, the system's as well as each validator's own solution. In this procedure each expert has two general ways to react to a given test case solution:

1. With the help of a rating $r \in \{0, 1\}$ and a certainty $c(r) \in \{0, 1\}$ an expert can express
   - his opinion about the solution ($r = 1$ for "correct" and $r = 0$ for "incorrect") and
   - his confidence in his opinion being valid ($c = 1$ for "sure" and $c = 0$ for "unsure").
2. Each expert has the opportunity to express lack of competence in his ability to evaluate a given solution. This can be expressed by the special rating *norating*, which always has the certainty $c(norating) = 0$.[14]

The result of the rating procedure is a set of $n \cdot m \cdot (n+1)$ rated solutions. Each rating

$$r(E_i, t_j, s(E_k, t_j)) = r(E_i, t_j, s_{kj}) = r_{ijk}$$

is assigned

- to a certain solution s_{kj} of a certain expert E_k $(1 \leq k \leq n+1)$
- of a certain test case t_j $(1 \leq j \leq m)$ and
- a certain evaluating (human) expert E_i (where $1 \leq i \leq n$).

[14] This is not because of any semantic reason. To define $c(norating) = 0$ is merely useful in making the formulas in section 8.1 simpler.

Assigned to each rating r_{ijk} there is a certainty

$$c(r(E_i, t_j, s_{kj})) = c(r_{ijk}) = c_{ijk}$$

A complete *rated test case solution* can be represented by a quintuple $[t_j, E_i, s_{kj}, r_{ijk}, c_{ijk}]$.

8 Evaluating TURING Test Results

This procedure is managed by the *validity meter*, which gets

1. a set of $m \cdot (n+1)$ *solved test cases* (stored in the *test case solution table*) and
2. a set of $n \cdot m \cdot (n+1)$ *rated test case solutions* (stored in the *test case rating table*).

The data in both tables will be used to compute some *validity statement* about the expert system. Note, that there is a special solution value called *unknown* and a special rating value called *norating*, which should be evaluated in a correct manner.

8.1 Estimating Local Competence

To make this explicit from the early beginning, this is – for not being misinterpreted – not a contribution to the cognitive science research area of expertise, of relating experts and novices, and so on. This chapter deals with technical termini within the formal setting adopted from [JAK97].

The first step towards a validity statement is to estimate the competence of each expert for each test case.

We prefer to do that for each expert and for each test case separately due to the fact that

- not all experts are equally competent for a given test case and
- a certain expert's competence is not equally for each test case.

The experts' competences depend on

- their different talents,
- their different educational background, and
- their different experiences.

Experts, therefore, are likely to have different regions of competence within the application field.

The best opportunity to estimate the competence of an expert E_i for a considered test case t_j is to use

1. his evaluation of his own competence, for a given test case, which is expressed by
   (a) the solution $s_{ij} = unknown$ or a real solution in the solving procedure and/or
   (b) the ratings of other experts' E_k solutions $r_{ijk} = norating$ or a real rating $r_{ijk} \in \{0, 1\}$ in the rating procedure,
2. his certainty values c_{ijk} of his ratings r_{ijk}, of other experts' $\mathcal{E}_k$ solutions s_{jk} in the rating procedure,
3. his consistency in the solving and the rating process[15],
4. his stability[16], and
5. the other experts' E_k $(k \neq i)$ ratings r_{kji} of his solution s_{ij}.

Each of these components will be graded with a number between 0 (which stands for "incompetent") and 1 (which stands for "competent").

8.1.1 ... by using his own opinion about being competent

An expert's competence can be revealed as the solution s_{ij} in the solving procedure and as the ratings r_{ijk} in the rating procedure.

The first component, the competence opinion while solving the test case t_j is simple to estimate: It is

- 0, iff the expert gave the "solution" *unknown* and
- 1, iff he gave a real solution.

The second component, the competence opinion while rating the solutions for the test case t_j (with the exception of the own solution, which is considered separately) can be estimated as the ratio between the number of *norating*s and the number of ratings altogether (which is n after excluding the rating of his own solution) for the considered test case t_j.

We did not find any reason to let one of these components be more important than the other. That's why we let them contribute the same portion of $\frac{1}{2}$ each. So the estimated "self-estimation" $slf_est(E_i, t_j)$ of an expert E_i to be competent for a test case t_j is[17]

$$slf_est(E_i, t_j) = \frac{1}{2} \cdot ord(s_{ij} \neq unknown) + \frac{1}{2}\frac{1}{n} \sum_{k=1, k\neq i}^{n+1} ord(r_{ijk} \neq norating)$$

[15] Does he/she give his/her own solution good marks?
[16] Is he/she certain while rating his/her own solution?
[17] $ord(Logic_Expr) = \begin{cases} 0 & \text{, iff } Logic_Expr = false \\ 1 & \text{, iff } Logic_Expr = true \end{cases}$

8.1.2 ...by using his certainty while rating other experts' solutions

The job of judging, whether a given solution is correct or not, is totally different from the job of finding a solution. So it may happen, e.g., that an expert on one hand isn't able to find a solution, but on the other hand he can definitely judge the validity of a given solution. The more often an expert is certain that a given solution is correct or not, the more he is considered to be competent. That's why this capability of an expert should be included in our competence estimation.

The competence of an expert E_i based on the certainty (the certainty-estimation $crt_est(E_i, t_j)$) while rating given solutions for a test case t_j can be estimated as the ratio between the number of certain ($c_{ij(n+1)} = 1$) ratings (with the exception of the one for his own solution[18]) and the number of ratings altogether (which is n after excluding the rating of the own solution):

$$crt_est(E_i, t_j) = \frac{1}{n} \sum_{k=1, k \neq i}^{n+1} c_{ijk}$$

8.1.3 ...by using his consistency

An expert behaves *consistently*, if he gives his own solution[19] good marks while rating it. An expert who behaves consistently can be said to be more competent than an expert who doesn't.

The consistency-based estimation of an expert's E_i competence for a test case t_j is

$$cns_est(E_i, t_j) = r_{iji}$$

8.1.4 ...by using his stability

An expert behaves *stably*, if he gives the rating for his own solution a certainty $c_{iji} = 1$ (regardless of whether it is consistent).

Of course, it may happen that he changes his opinion about the solution after analysing the solutions of others during the solution process, but that doesn't have any influence on his stability. Stability merely means that the expert is sure whether his own solution is correct or not in the rating process. An expert who behaves stably seems to be more competent than a one who doesn't.

The stability-based estimation of an expert's E_i competence for a test case t_j is

$$stb_est(E_i, t_j) = c_{iji}$$

[18] The rating of the own solution and its certainty is considered separately below.

[19] ..., without knowing that it is his own, ...

8.1.5 ...by using the other experts' ratings of his solution

Another component of the competence estimation of an expert E_i for a test case t_j are the ratings r_{kji} of his test case solutions s_{ij} by the other experts E_k ($k \neq i$). In case an expert E_k is uncertain whether E_i's solution is correct or not (which means, that $c_{kji} = 0$), his rating for s_{ij} shouldn't be included in our competence estimation.

So the external estimation $frgn_est(E_i, t_j)$ of the competence of an expert E_i by the other (external) experts E_k ($k \neq i$) for a test case t_j is the ratio between the number of certain ratings ($c_{kji} = 1$) "correct" ($r_{kji} = 1$) and the certain ratings altogether, which is the average rating of all certain ratings:

$$frgn_est(E_i, t_j) = \frac{1}{\left(\sum_{k=1, k \neq i}^{n} c_{kji} \right)} \sum_{k=1, k \neq i}^{n} (c_{kji} \cdot r_{kji})$$

8.1.6 ...by using all the five components

We believe that the components above can be divided into three main groups:

- self-estimation and certainty (slf_est, crt_est),
- consistency and stability (cns_est, stb_est), and
- external estimation of competence ($frgn_est$).

We did not find any reason to let one of these groups be more important than the others. That's why we use all three groups for estimating an expert E_i's competence $cpt(E_i, t_j)$ for a test case t_j and let them contribute equally to the final result[20] each with the same portion of $\frac{1}{3}$. For the same reason, the sources within a group contribute with equivalent portions, if there is more than one source.

$$\begin{aligned} cpt(E_i, t_j) = \\ & \tfrac{1}{6} slf_est(E_i, t_j) + \tfrac{1}{6} crt_est(E_i, t_j) + \\ & \tfrac{1}{6} cns_est(E_i, t_j) + \tfrac{1}{6} stb_est(E_i, t_j) + \\ & \tfrac{1}{3} frgn_est(E_i, t_j) \end{aligned}$$

[20] ..., which is a value for the expert's local competence, ...

8.2 Estimating Local Validity

Our next step is to come up with a statement about the (local) validity $v(t_j)$ of the system (which is the "expert" E_{n+1} in our scenario with $\mathcal{E} = \mathcal{S}$ in [JAK97]) for a test case t_j. To reach this objective, the human experts' $(E_1, \ldots, E_n)$ ratings $r_{ij(n+1)}$ $(1 \leq i \leq n)$ of the system's solution $s(E_{n+1}, t_j) = s_{(n+1)j}$ should be considered.

Knowing the fact, that the (human) experts E_i $(1 \leq i \leq n)$ have different local competences $cpt(E_i, t_j)$ for a considered test case t_j, their ratings for the system's solution should be weighted with their competences as a coefficient.

If an expert E_i is uncertain whether $E_{(n+1)}$'s solution is correct or not (which means, that $c_{ij(n+1)} = 0$), his rating for $s_{(n+1)j}$ should not be included in our estimation of the system's local validity.

Thus, we suggest a weighted average of all certainty ratings:

$$v(t_j) = \frac{1}{\sum_{i=1}^{n} \left(cpt(\mathcal{E}_i, t_j) \cdot c_{ij(n+1)} \right)} \cdot \sum_{i=1}^{n} \left(cpt(E_i, t_j) \cdot c_{ij(n+1)} \cdot r_{ij(n+1)} \right)$$

8.3 Estimating Global Validity

Now, the entire expert system's validity v can be estimated by the average local validity $v(t_j)$ for each test case t_j[21]:

$$v = \frac{1}{m} \sum_{j=1}^{m} v(t_j)$$

Of course, computing the average validity of all test cases is not sufficient in many cases. It may happen, that depending on some conclusion-related validation criteria (cf. [Abe97]) some test cases are seemed to be more important for establishing the validity of the system than others. One approach is to take that fact into consideration and to weight the local validities $v(t_j)$ with their conclusion-dependent validation criteria and to compute a weighted average.

In any case (with or without weighting the $v(t_j)$), the *expert system's validity* v gets a value between 0 and 1 (both inclusive), in which

- $v = 0$ means "the system is totally invalid" and
- $v = 1$ means "the system is totally valid".

[21] By the way, this formula to estimate the *system's* validity v_{n+1} can be used to estimate a (human) *expert's* "validity" v_i $(1 \leq i \leq n)$ as well. Whether this should be done, is a question for psychologists ;-)

Depending on the domain- and user- related validation criteria (cf. [Abe97]) each system is associated with a minimum validity v^{min}, which is a threshold value for the validity statement. That is, of course, the final objective of our research:

The system is called

- **valid**, iff $v \geq v^{min}$ and
- **invalid** otherwise.

Refinements of this perspective are left to forthcoming investigations.

9 Acknowledgments

The first author's work has been substantially supported by a 2 months visit to the Meme Media Laboratory of Hokkaido University, Sapporo, Japan. He especially acknowledges the fruitful scientific co-operation with colleagues and friends at the Kyushu University, Fukuoka, Japan. This co-operation made the present report possible. Furthermore, the authors gratefully acknowledge the sponsoring by the German Research Fund (DFG) within the project on Expert System Validation under grant Ph 49/3–1.

The third author's participation in FLAIRS–97, the Florida AI Research Symposium, Daytona Beach, FL, USA, May 1997, will be supported by a travel grant from the Stifterverband für die Deutsche Wissenschaft. This support is gratefully acknowledged. His paper [Abe97] to be presented at FLAIRS–97 is closely related to the present investigations.

It is worth to be mentioned that the authors' close cooperation with US American colleagues had an enormous impact on their work. They especially acknowledge the continuous discussions with Avelino Gonzalez, University of Central Florida. His outstanding practical experience forms a permanent yardstick of the authors' efforts.

Last but not least, the authors wish to express their sincere gratitude to all members of VAIS[22], the international and interdisciplinary Research Institute for Validation of AI Systems (cf. the internet appearance http://kiwi.theoinf.tu-ilmenau.de/vais/). Communications among the team members of the institute have always been very stimulating and influential.

References

[ABB+91] Andrejs Auziņš, Jānis Bārzdiņš, Jānis Bičevskis, Kārlis Čerāns, and Audris

[22] Note that in [AU97] VAIS is called *the first virtual research institute in the world wide web.*

Kalniņš. Automatic construction of test sets: Theoretical approach. In Jānis Bārzdiņš and Dines Bjørner, editors, *Baltic Computer Science*, volume 502 of *Lecture Notes in Artificial Intelligence*, pages 286–359. Springer–Verlag, 1991.

[Abe96] Thomas Abel. The pursuit of quality: Efforts in verification and validation of expert systems. VAIS–Report 02/96, Techn. Uni. Ilmenau, Fak. Informatik & Automatisierung, January 1996.

[Abe97] Thomas Abel. Utilizing criteria to reduce a set of test cases for expert system validation. In *FLAIRS–97, Proc. Florida AI Research Symosium, Daytona Beach, FL, USA, May 11–14, 1997*, 1997.

[AKG96] Thomas Abel, Rainer Knauf, and Avelino Gonzalez. Generation of a minimal set of test cases that is functionally equivalent to an exhaustive set, for use in knowledge-based system validation. In John H. Stewman, editor, *Proc. Florida AI Research Symposium (FLAIRS-96), Key West, FL, USA, May 20–22, 1996*, pages 280–284. Florida AI Research Society, 1996.

[AU97] Oksana Arnold and Andreas Ulrich. Towards an integral and unified appearance of virtual enterprises. In *IT Vision '97, The European Conference on Virtual Enterprises and Networked Solutions, April 7–10, 1997, Paderborn, Germany*, 1997.

[BB93] Wolfgang Bibel and Alan W. Biermann. Special issue: Automatic programming - foreword of the guest editors. *Journal of Symbolic Computation*, 15(5 & 6):463–465, 1993.

[BC85] Joseph L. Bates and Robert L. Constable. Proofs as programs. *ACM Transactions on Programming Languages and Systems*, 7(1):113–136, 1985.

[BCM96] Andrea Bonzano, Pádraig Cunningham, and Colin Meckiff. ISAC: A CBR system for decision support in air traffic control. In I. Smith and B. Faltings, editors, *Advances in Case-Based Reasoning, Proc., 3rd European Workshop on Case–Based Reasoning (EWCBR'96), November 14–16, 1996, Lausanne, Switzerland*, volume 1168 of *Lecture Notes in Artificial Intelligence*, pages 44–57. Springer-Verlag, 1996.

[Bie85] Alan W. Biermann. Automatic programming: A tutorial on formal methodologies. *Journal of Symbolic Computation*, 1(1):119–142, 1985.

[BM79] R. S. Boyer and J. S. Moore. *A Computational Logic*. Academic Press, 1979.

[CLEKR96] Ho-Jin Choi, Vassilis Liatsos, Amin El-Kholy, and Barry Richards. Recovering from failure in temporal planning. In Susanne Biundo, Thomas Dean, and Richard Waldinger, editors, *12th European Conference on Artificial Intelligence, ECAI'96, Workshop on Cross-Fertilization in Planning, August 12, 1996, Budapest, Hungary*, pages 31–35. European Coordinating Committee for Artificial Intelligence (ECCAI), 1996.

[Fos93] Harold D. Foster. Resilence theory and system evaluation. In John A. Wise, V. David Hopkin, and Paul Stager, editors, *Verification and Validation of Complex Systems: Human Factors Issues*, volume 110 of *NATO ASI Series, Series F: Computer and Systems Sciences*, pages 35–60. Springer–Verlag, 1993.

[FWH96] Bob Fields, Peter Wright, and Michael Harrison. Time, tasks and errors. *ACM SIGCHI Bulletin*, 283(2):53–56, 1996.

[Gin93] Matt Ginsberg. *Essentials of Artificial Intelligence*. Morgan Kaufmann, 1993.

[Gor92] Dallie E. Gordon. Implication of cognitive theory for knowledge acquisition. In Robert R. Hoffman, editor, *The Psychology of Expertise. Cognitive Research of Empirical AI*, pages 99–120. Springer-Verlag, 1992.

[Gur92] Yuri Gurevich. Zero-one laws. *Bulletin of the EATCS*, (46):90–106, 1992.

[Hal87] Mark Halpern. Turing's test and the ideology of artificial intelligence. *Artificial Intelligence Review*, 1:79–93, 1987.

[Har93] Kelly Harwood. Defining human-centered system issues for verifying and validating air traffic control systems. In John A. Wise, V. David Hopkin, and Paul Stager, editors, *Verification and Validation of Complex Systems: Human Factors Issues*, volume 110 of *NATO ASI Series, Series F: Computer and Systems Sciences*, pages 115–129. Springer–Verlag, 1993.

[HJK97a] Jörg Herrmann, Klaus P. Jantke, and Rainer Knauf. Towards using structural knowledge for system validation. Technical Report MEME–IMP–2, Hokkaido University Sapporo, Meme Media Laboratory, February/March 1997.

[HJK97b] Jörg Herrmann, Klaus P. Jantke, and Rainer Knauf. Using structural knowledge for system validation. In *FLAIRS–97, Proc. Florida AI Research Symosium, Daytona Beach, FL, USA, May 11–14, 1997*, 1997.

[Hoa69] C.A.R. Hoare. An axiomatic basis for computer programming. *Comm. of the ACM*, 12(10):576–583, 1969.

[Hop90] Thomas Hoppe. Hypotheses generation for knowledge validation. In Luigia Carlucci Aiello, editor, *European Conference on Artificial Intelligence, ECAI'90, Stockholm, August 1990*, pages 354–356. Pitman Publ., London, 1990.

[Hop93] V. David Hopkin. Verification and validation: Concepts, issues, and applications. In John A. Wise, V. David Hopkin, and Paul Stager, editors, *Verification and Validation of Complex Systems: Human Factors Issues*, volume 110 of *NATO ASI Series, Series F: Computer and Systems Sciences*, pages 9–33. Springer–Verlag, 1993.

[Hut95] Edwin Hutchins. *Cognition in the Wild*. The MIT Press, 1995.

[JAK97] Klaus P. Jantke, Thomas Abel, and Rainer Knauf. Fundamentals of a TURING test approach to validation. Technical Report MEME–IMP–1, Hokkaido University Sapporo, Meme Media Laboratory, February 1997.

[Jan94] Klaus P. Jantke. Nonstandard concepts of similarity in case-based reasoning. In Hans-Hermann Bock, Wolfgang Lenski, and Michael M. Richter, editors, *Information Systems and Data Analysis: Prospects – Foundations – Applications, Proceedings of the 17th Annual Conference of the GfKl, Univ. of Kaiserslautern, 1993*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 28–43. Springer-Verlag, 1994.

[Kak96] Subhash C. Kak. Can we define levels of artificial intelligence? *Journal of Intelligent Systems*, 6, 1996.

[Ker64] Fred N. Kerlinger. *Foundations of Behavioral Research*. Holt, Rinehart & Winston, 1964.

[KP96] Rainer Knauf and Ilka Philippow. Ein Turing Test zur Validitätsabschätzung von KI-Systemen. In Klaus P. Jantke and Gunter Grieser, editors, *4. Leipziger Informatik–Tage, 29./30. August 1996*, pages 147–152. Forschungsinstitut für Informations-Technologien Leipzig e.V., 1996.

[Mes90] Pedro Meseguer. A new method to checking rule bases for inconsistencies: A Petri net approach. In Luigia Carlucci Aiello, editor, *European Conference on Artificial Intelligence, ECAI'90, Stockholm, August 1990*, pages 437–442. Pitman Publ., London, 1990.

[MP94] Pedro Meseguer and Enric Plaza. The VALID project: Goals, development and results. *International Journal of Intelligent Systems*, 9(9):867–892, 1994.

[OO93] R. M. O'Keefe and D. E. O'Leary. Expert system verification and validation: A survey and tutorial. *Artificial Intelligence Review*, 7:3–42, 1993.

[PAG92] David S. Prerau, Mark R. Adler, and Alan S. Gunderson. Eliciting and using experiential knowledge and general expertise. In Robert R. Hoffman, editor, *The Psychology of Expertise. Cognitive Research of Empirical AI*, pages 137–148. Springer-Verlag, 1992.

[Por64] E. H. Porter. *Manpower Development*. Harper & Row, 1964.

[PR87] Chr. Posthoff and G. Reinemann. *Computerschach – Schachcomputer*. Akademie–Verlag, 1987.

[Reb85] A. S. Reber. *The Penguin Dictionary of Psychology*. Penguin Books, 1985.

[Rei95] W. Reif. The KIV–approach to software verification. In M. Broy and S. Jähnichen, editors, *KORSO: Methods, Languages, and Tools for the Construction of Correct Software – Final Report*, volume 1009 of *Lecture Notes in Computer Science*, 1995.

[RS93] W. Reif and K. Stenzel. Reuse of proofs in software verification. In R. Shyamasundar, editor, *Foundations of Software Technology and Theoretical Computer Science, Bombay, India, 1993*, volume 761 of *Lecture Notes in Computer Science*, 1993.

[Sta93] Paul Stager. Validation in complex systems: Behavioral issues. In John A. Wise, V. David Hopkin, and Paul Stager, editors, *Verification and Validation of Complex Systems: Human Factors Issues*, volume 110 of *NATO ASI Series, Series F: Computer and Systems Sciences*, pages 99–114. Springer–Verlag, 1993.

[Tur50] Alan M. Turing. Computing machinery and intelligence. *Mind*, LIX(236):433–460, 1950.

[WW93] John A. Wise and Mark A. Wise. Basic considerations in verification and validation. In John A. Wise, V. David Hopkin, and Paul Stager, editors, *Verification and Validation of Complex Systems: Human Factors Issues*, volume 110 of *NATO ASI Series, Series F: Computer and Systems Sciences*, pages 87–95. Springer–Verlag, 1993.