Chapter 1
Computational Intelligence: The Legacy of Alan Turing and John von Neumann

Heinz Mühlenbein

Abstract In this chapter fundamental problems of collaborative computational intelligence are discussed. The problems are distilled from the seminal research of Alan Turing and John von Neumann. For Turing the creation of machines with human-like intelligence was only a question of programming time. In his research he identified the most relevant problems concerning evolutionary computation, learning, and structure of an artificial brain. Many problems are still unsolved, especially efficient higher learning methods which Turing called initiative. Von Neumann was more cautious. He doubted that human-like intelligent behavior could be described unambiguously in finite time and finite space. Von Neumann focused on self-reproducing automata to create more complex systems out of simpler ones. An early proposal from John Holland is analyzed. It centers on adaptability and population of programs. The early research of Newell, Shaw, and Simon is discussed. They use the logical calculus to discover proofs in logic. Only a few recent research projects have the broad perspectives and the ambitious goals of Turing and von Neumann. As examples the projects Cyc, Cog, and JANUS are discussed.

1.1 Introduction

Human intelligence can be divided into *individual*, *collaborative*, and *collective intelligence*. Individual intelligence is always multi-modal, using many sources of information. It developed from the interaction of the humans with their environment. Based on individual intelligence, collaborative intelligence developed. This means that humans work together with all the available allies to solve problems. On the next level appears collective intelligence. It describes the phenomenon that families, groups, organizations and even entire societies seem to act as a whole living organism.

The importance of interactions between higher animals has been reinforced by the discovery of *mirror neurons*. These are neurons which fire both

Fraunhofer Institut Autonomous intelligent Systems Schloss Birlinghoven 53757 Sankt Augustin, Germany, e-mail: heinz.muehlenbein@online.de

when an animal acts and when the animal observes the same action performed by another (especially conspecific) animal. The neurons have been observed in primates and are believed to exist also in humans and in some birds. The function of the mirror system is still a subject of much speculation. To date no plausible neural or computational models have been developed to explain how mirror neurons support the cognitive functions.

In my opinion the most impressive collaborative computational intelligence examples developed so far are the search machine Google and Wikipedia. In both systems the interaction human-computer plays an important role. Google is a gigantic storage system with an impressive fast search engine. It remains the task of the user to filter out the important information from the search results.

Wikipedia tries to make the dream of the Age of Enlightenment become true, to develop an encyclopedia describing all human knowledge and making it accessible to all humans. Both systems use pure text driven search. More intelligent search methods have been not successful so far. Despite the many efforts no computational system is approaching the level of human intelligence.

Today computational intelligence is partitioned into many specialized and separate research areas. This was not always the case. The aim of this chapter is to recall the broader issues and research goals of computational intelligence. To this end the seminal research of Alan Turing and John von Neumann is reviewed in detail. Their proposals discuss many areas of computational intelligence necessary to create automata with human-like intelligence.

Right at the beginning of electronic computers researchers looked into nature for ideas to solve difficult problems or even create what is called today *artificial intelligence*. Because of the lack of understanding the functioning of natural systems, the research had to be largely experimental. This was already pointed out by John von Neumann (25).

Natural organism are, as a rule, much more complicated and subtle, and therefore much less well understood in detail, than are artificial automata. Nevertheless, some regularities, which we observe in the organization of the former may be quite instructive in our thinking and planning of the latter; and conversely, a good deal of our experiences and difficulties with our artificial automata can be to some extend projected on our interpretations of natural organisms.

In this chapter I will first review the work of Alan Turing, described in his famous seminal paper "Computing Machinery and Intelligence" (23) and in the not so well known paper "Intelligent Machinery" (24). Turing's thoughts about *learning, evolution, and structure of the brain* are described.

Then I will discuss the most important paper of John von Neumann concerning our subject "The General and Logical Theory of Automata" (25). John von Neumann's research centers about *artificial automata, computability, complexity, and self-reproduction*

All three papers have been written before the first electronic computers became available. Turing even wrote programs for paper machines. As third example I will describe the proposal of John Holland (10). The simplification of this proposal to optimization lead later to the famous genetic algorithm (11). The historical part ends with a discussion of the early research of Newell, Shaw and Simon.

I will first discuss this early research in detail, without commenting it using todays knowledge. Then I will try to evaluate the proposals in answering the following questions

- What are the major ideas for creating machine intelligence?
- Did the proposals lack important components we see as necessary today?
- What are the major research problems of the proposals and do exist solutions today?

Then two recent large projects are shortly summarized. The goal of the project Cyc is to specify in a *well-designed language common sense knowledge* The Cog project tried to build a *humanoid robot that acts like a human*. In addition the architecture of our *hand-eye robot JANUS* is described. It has a modular structure similar to the human brain.

This chapter is a tour de force in computational intelligence. It requires that the reader is willing to contemplate fundamental problems arising in building intelligent machines. Solutions are not given. I hope that the reader finds interesting research problems worthwhile to be investigated. This paper extends my research started in (15).

1.2 Turing and machine intelligence

The first sentences of the paper "Computing machinery and intelligence" have become famous.

I propose to consider the question "Can machines think?" This should begin with definitions of the meaning of the terms "machine" and "think"....But this is absurd. Instead of attempting such a definition I shall replace the question by another, which is closely related to it and is expressed in relatively unambiguous words. The new form of the question can be described in terms of a game which we call the imitation game.

Turing's definition of the imitation game is more complicated than what is normally used today. Therefore I describe it shortly. It is played with three actors, a man (A), a woman (B) and an interrogator (C). The object of the game for the interrogator is to determine which of the other two is the man and which is the woman. It is A's objective in the game to cause C to make the wrong identification. Turing then continues: "We now ask the question "What will happen when a machine takes the part of A in the game?" Will the interrogator decide wrongly as often when the game is played as this as he does when the game is played between a man and a woman? These questions will replace our original "Can machines think".

Why did Turing not define just a game between a human and a machine trying to imitate a human, as the Turing test is described today? Is there an additional trick in introducing gender into the game? There has been quite a lot of discussions if this game characterizes human intelligence at all. Its purely behavioristic definition leaves out any attempt to identify important components which together produce human intelligence. I will not enter this discussion here, but just state the opinion of Turing about the outcome of the imitation game.

I believe that in about fifty years' time it will be possible to programme computers with a storage capacity of about 10^9 bits to make them play the imitation game so well that an average interrogator will not have more than 70% chance of making the right identification after five minutes of questioning.

The very detailed prediction is funny: Why a 70% chance, why a duration of five minutes? In the next section I will discuss what arguments Turing used to support this prediction.

1.2.1 Turing's construction of an intelligent machine

In sections $2-6$ of (23) Turing mainly seeks to refute general philosophical arguments against the possibility of constructing intelligent machines. "The reader will have anticipated that I have no very convincing argument of a positive nature to support my views. If I had I should not have taken such pains to point out the fallacies in contrary views. Such evidence as I have I shall now give." What is Turing's evidence?

As I have explained, the problem is mainly one of programming. Advances in engineering will have to be made too, but it seems unlikely that these will not be adequate for the requirements. Estimates of the storage capacity of the brain vary from 10^{10} to 10^{15} binary digits.[1] *I incline to the lower values and believe that only a small fraction is used for the higher types of thinking. Most of it is probably used for the retention of visual impressions. I should be surprised if more than 10^9 was required for satisfactory playing of the imitation game. Our problem then is to find out how to programme these machines to play the game. At my present rate of working I produce about a thousand digits of programme a day, so that about sixty workers, working steadily through fifty years might accomplish the job, if nothing went into the wastepaper basket.*

The time to construct a machine which passes the imitation game is derived from an estimate of the storage capacity of the brain [2] and the speed of programming. Turing did not see any problems in creating machine intelligence purely by programming, he just found it too time consuming. So he investigated if there exist more expeditious methods. He observed:

"In the process of trying to imitate an adult human mind we are bound to think a good deal about the process which has brought it to the state that it is in. We may notice three components.

1. The initial state of the brain, say at birth.
2. The education to which it has been subjected.
3. Other experience, not to be described as education, to which it has been been subjected.

[1] At this time the number of neurons was estimated as being between 10^{10} to 10^{15}. This agrees with the estimates using todays knowledge.

[2] It was of course a mistake to set the storage capacity equal to the number of neurons! Von Neumann estimated the storage capacity of the brain to be about 10^{20}. But this affects in no way the logic of Turing's argument.

Instead of trying to produce a programme to simulate an adult mind, why not rather try to produce one which simulates the child's...Presumably the child brain is something like a notebook. Rather little mechanism, and lots of blank sheets. Our hope is that there is so little mechanism in the child brain that something like it can easily be programmed. The amount of work in the education we can assume, as a first approximation, to be much the same as for the human child."

1.2.2 Turing on learning and evolution

In order to achieve a greater efficiency in constructing a machine with human like intelligence, Turing divided the problem into two parts

- The construction of a child brain
- The development of effective learning methods

Turing notes that the two parts remain very closely related. He proposes to use experiments: teaching a child machine and see how well it learns. One should then try another and see if it learns better or worse. "There is an obvious connection between this process and evolution, by the identifications

- structure of the machine = hereditary material
- changes of the machine = mutations
- Natural selection = judgment of the experimenter

Survival of the fittest is a slow process of measuring advantages. The experimenter, by the exercise of intelligence, should be able to speed it up."

Turing then discusses learning methods. He notes ((23), p.454): "We normally associate the use of punishments and rewards with the teaching process...The machine has to be so constructed that events which shortly proceeded the occurrence of a punishment signal are unlikely to be repeated, whereas a reward signal increases the probability of repetition of the events which lead to it."

But Turing observes the major drawback of this method: "The use of punishments and rewards can at best be part of the teaching process. Roughly speaking, if the teacher has no other means of communicating to the people, the amount of information which can reach him does not exceed the total number of rewards and punishments applied."

In order to speed up learning Turing demanded that the child machine should understand some language. In the final pages of the paper Turing discusses the problem of the complexity the child machine should have. He proposes to try two alternatives: either to make it as simple as possible to allow learning or to include a complete system of logical inference. He ends his paper with the remarks: "Again I do not know the answer, but I think both approaches should be tried. We can see only see a short distance ahead, but we can see plenty there that needs to be done."

1.2.3 Turing and neural networks

In the posthumously published paper *Intelligent Machinery* (24) Turing describes additional details how to create an intelligent machine. First he discusses possible components of a child machine. He introduces *unorganized machines* of type A,B, and P. A and B are artificial neural networks with random connections. They are made up from a rather large number N of similar units, which can be seen as binary neurons. Each unit has two input terminals and one output terminal which can be connected to the input terminals of 0 (or more) other units. The connections are chosen at random. All units are connected to a central synchronizing unit from which synchronizing pulses are emitted. Each unit has two states. The dynamics is defined by the following rule:

The states from the units from which the input comes are taken from the previous moment, multiplied together and the result is subtracted from 1.

Thus a neuron is nothing else than a NAND gate. The state of the network is defined by the states of the units. Note that the network might have lots of loops, it continually goes through a number of states until a period begins. The period cannot exceed 2^N cycles. In order to allow learning the machine is connected with some input device which can alter its behavior. This might be a dramatic change of the structure, or changing the state of the network.

Maybe Turing had the intuitive feeling that the basic transition of the type A machine is not enough, therefore he introduced the more complex B-type machine. I will not describe this machine here, because neither for the A or the B machine Turing defined precisely how learning can be done.

A learning mechanism is introduced with the third machine, called a P-type machine. The machine is an automaton with a number of N configurations. There exist a table where for each configuration is specified which action the machine has to take. The action may be either

1. To do some externally visible act $A_1, \ldots A_k$
2. To set a memory unit M_i

The reader should have noticed that the next configuration is not yet specified. Turing surprisingly defines: If the current configuration is s, then the next configuration is the remainder of $2s$ or $2s + 1$ on division by N. These two configurations are called the alternatives 0 and 1. The reason for this definition is the learning mechanism Turing defines. At the start the description of the machine is largely incomplete. The entries for each configuration might be in five states, either U (uncertain), or T0 (try alternative 0), T1 (try alternative 1), D0 (definite 0) or D1 (definite 1).

Learning changes the entries as follows: If the entry is U, the alternative is chosen at random, and the entry is changed to either T0 or T1 according to whether 0 or 1 was chosen. For the other four states, the corresponding alternatives are chosen. When a pleasure stimulus occurs, state T is changed to state D, when a pain stimulus occurs, T is changed to U. Note that state D cannot be changed. The proposed learning method sounds very simple, but Turing surprisingly remarked:

I have succeeded in organizing such a (paper) machine into a universal machine.

Today the universal machine is called the *Turing Machine*. Turing even gave some details of this particular P-type machine. Each instruction consisted of 128 digits, forming four sets of 32 digits, each of which describes one place in the main memory.

1.2.4 Discipline and initiative

We now turn to the next important observation of Turing. Turing notes that punishment and reward are very slow learning techniques. So he requires:

If the untrained infant's mind is to become an intelligent one, it must acquire both discipline and initiative.

Discipline means strictly obeying the punishment and reward. But what is initiative? The definition of initiative is typical of Turing's behavioristic attitude. "Discipline is certainly not enough in itself to produce intelligence. That which is required in addition we call initiative. This statement will have to serve as a definition. Our task is to discover the nature of this residue as it occurs in man, and to try and copy it in machines."

With only a paper computer available Turing was not able to investigate the subject initiative further. Nevertheless he made the bold statement (24): "A great positive reason for believing in the possibility of making thinking machinery is the *fact that it is possible to make machinery to imitate any small part of a man*. One way of setting about our task of building a thinking machine would be to take a man as a whole and to try to replace all parts of him by machinery...Thus although this method is probably the 'sure' way of producing a thinking machine it seems to be altogether too slow and impracticable. Instead we propose to try and see what can be done with a 'brain' which is more or less without a body providing, at most organs of sight, speech, and hearing. We are then faced with the problem of finding suitable branches of thought for the machine to exercise its powers in."
Turing mentions the following fields as promising:

- Various games, e.g. chess, bridge
- The learning of languages
- Translation of languages
- Cryptography
- Mathematics

Turing remarks: "The learning of languages would be the most impressive, since it is the most human of these activities. This field seems however to depend rather too much on sense organs and locomotion to be feasible." Turing seems here to have forgotten that language learning is necessary for his imitation game!

1.3 Von Neumann's logical theory of automata

In 1938 Alan Turing was assistant of John von Neumann. But later they worked completely independent from each other, not knowing the thoughts the other had concerning the possible applications of the newly designed electronic computers. A condensed summary of the research of John von Neumann concerning machine intelligence is contained in his paper "The General and Logical Theory of Automata" (25). This paper was presented in 1948 at the Hixon symposium on: *Cerebral mechanism of behavior*. Von Neumann was the only computer scientist at this symposium. The reason was that von Neumann closely observed the theoretical research aimed to understand the brain in order to use the results for artificial automata.
Von Neumann notices three major limits of the present size of artificial automata

- The size of componentry
- The limited reliability
- The lack of a logical theory of automata

There have been tremendous achievements in the first two areas. Therefore I will concentrate on the theory problem. Here von Neumann predicted:

The logic of automata will differ from the present system of formal logic in two relevant respects.

1. *The actual length of "chains of reasoning", that is, of the chains of operations, will have to be considered.*
2. *The operations of logic will all have to be treated by procedures which allow exceptions with low but non-zero probabilities.*

...This new system of formal logic will move closer to another discipline which has been little linked in the past with logic. This is thermodynamics, primarily in the form it was received from Boltzmann, and is that part of theoretical physics which comes nearest in some of its aspects to manipulating and measuring information.

Von Neumann tried later to formalize probabilistic logic. His results appeared in (26). But this research was more or less a dead end, because von Neumann did not abstract from the hardware components and did consider time as part of the model. Probabilistic reasoning is now heavily used in artificial intelligence (17). The chains of operations are investigated in a branch of theoretical computer science called computational complexity (8).

1.3.1 McCulloch-Pitts theory of formal neural networks

In 1943 McCulloch and Pitts (13) had described the brain by a formal neural network, consisting of interconnected binary neurons. Von Neumann summarizes their major result follows:
"The "functioning" of such a network may be defined by singling out some of the inputs of the entire system and some of its outputs, and then describing what original stimuli on the former are to cause what ultimate

stimuli of the latter. McCulloch and Pitts' important result is that any functioning in this sense which can be defined at all logical, strictly, and unambiguously in a finite number of words can also be realized by such a formal system."

In modern terms: Any computable function can be realized by a sufficiently large McCulloch and Pitts network.
McCulloch and Pitts had derived this result by showing that their formal neural network connected to an infinite tape is equivalent to a Turing machine. But even given this result, von Neumann observes that at least two problems remain

1. Can the network be realized within a practical size?
2. Can every existing mode of behavior really be put completely and unambiguously into words?

Von Neumann informally discusses the second problem, using the problem of analogy. He remarks prophetically:

There is no doubt that any special phase of any conceivable form of behavior can be described "completely and unambiguously" in words.... It is, however an important limitation, that this applies only to every element separately, and it is far from clear how it will apply to the entire syndrome of behavior.

This severe problem has not been noticed by Turing. Using the example visual analogy von Neumann argues: "One can start describing to identify any two rectilinear triangles. These could be extended to triangles which are curved, whose sides are only partially drawn etc... We may have a vague and uncomfortable feeling that a complete catalogue along such lines would not only be exceedingly long, but also unavoidingly indefinite at its boundaries. All of this, however, constitutes only a small fragment of the more general concept of identification of analogous geometrical objects. This, in turn, is only a microscopic piece of the general concept of visual analogy." Thus von Neumann comes to the conclusion:

Now it is perfectly possible that the simplest and only practical way to say what constitutes a visual analogy consists in giving a description of the connections of the visual brain....It is not at all certain that in this domain a real object might not constitute the simplest description of itself.

Von Neumann ends the section with the sentence: "The foregoing analysis shows that one of the relevant things we can do at this moment is to point out the directions in which the real problem does not lie." In order to understand and investigate the fundamental problem, von Neumann identified an important subproblem. In nature obvious more complex beings have been developed from less complex ones. Is this also possible using automata? How much complexity is needed for automata to create more complex ones?

1.3.2 Complication and self-reproduction

Von Neumann starts the discussion of complexity with the observation that if an automaton has the ability to construct another one, there must be a decrease in complication. In contrast, natural organisms reproduce themselves, that is, they produce new organisms with no decrease in complexity. So von Neumann tries to construct a general artificial automata which could reproduce itself. The famous construction consists of the following automata:

1. A general constructive machine, A, which can read a description $\Phi(X)$ of another machine, X, and build a copy of X from this description:
$$A + \Phi(X) \leadsto X$$
2. A general copying machine, B. which can copy the instruction tape:
$$B + \Phi(X) \leadsto \Phi(X)$$
3. A control machine, C, which when combined with A and B, will first activate B, then A, link X to $\Phi(X)$ and cut them loose from A+B+C
$$A + B + C + \Phi(X) \leadsto X + \Phi(X)$$
Now choose X to be A+B+C
$$A + B + C + \Phi(A + B + C) \leadsto A + B + C + \Phi(A + B + C)$$
4. It is possible to add the description of any automaton D
$$\begin{aligned} A + B + C + \Phi(A + B + C + D) \leadsto\ & A + B + C + D \\ & + \Phi(A + B + C + D) \end{aligned}$$
Now allow mutation on the description $\Phi(A + B + C + D)$
$$\begin{aligned} A + B + C + \Phi(A + B + C + D') \leadsto\ & A + B + C + D' \\ & + \Phi(A + B + C + D') \end{aligned}$$

Mutation at the D description will lead to a different self-reproducing automaton. This might allow to simulate some kind of evolution as seen in natural organisms.

Von Neumann later constructed a self-reproducing automata which consisted of 29 states (27). This convinced von Neumann that complication can also be found in artificial automata. Von Neumann ends the paper with the remark:

This fact, that complication, as well as organization, below a critical level is degenerative, and beyond that level can become self-supporting and even increasing, will clearly play an important role in any future theory of the subject.

Von Neumann was well aware of the other two important evolutionary processes besides replication - namely variation and selection. He decided that knowledge about these two processes was not yet sufficient to incorporate them in his theory of automaton. “Conflicts between independent organisms lead to consequences which, according to the theory of natural selection, are believed to furnish an important mechanism of evolution. Our models lead to such conflict situations. The conditions under which this motive for evolution can be effective here may be quite complicated ones, but they deserve study."

Cellular automata have lead to great theoretical research. They can easily be extended to have the power of Turing machines. Nevertheless, the central problem of this approach remains unsolved: How can the automata evolve complex problem solving programs starting with fairly simple initial programs? This happened in biological evolution. Starting with small self-reproducing units complex problem solving capabilities have evolved, culminating in the human brain.

1.4 Holland's logical theory of adaptive systems

In the paper "Outline for a Logical Theory of Adaptive Systems" (10) John Holland tried to continue the scientific endeavor initiated by von Neumann. He wrote:

The theory should enable to formulate key hypotheses and problems particularly from molecular control and neurophysiology. The work in theoretical genetics should find a natural place in the theory. At the same time, rigorous methods of automata theory, particularly those parts concerned with growing automata should be used.

Holland's proposal is a very early attempt to work on a constructive theory of the evolution of automata. It tries to combine *being, acting, developing, and evolving*. This proposal is so important that I will describe it in detail. Holland's emphasis (like von Neumann's) is foremost on theories and systems, he does not claim to solve grand challenge applications with the proposed methods. This can be tried after the theories have been formulated and verified.

"Unrestricted adaptability (assuming nothing is known of the environment) requires that the adaptive system be able initially to generate any of the programs of some universal computer ... With each generation procedure we associate the population of programs it generates;... In the same vein we can treat the *environment as a population of problems*."

Now let us have a closer look at Holland's model. First, there is a finite set of generators (programs) $(g_1, \ldots, g_k)$. The generation procedure is defined in terms of this set and a *graph* called a *generation tree.* Each permissible combination of generators is represented by a vertex in the generation tree. Holland now distinguishes between auxiliary vertices and main vertices. Each auxiliary vertex will be labeled with two numbers, called the *connection* and *disconnection probabilities.* This technique enables to create new connections or to delete existing connections. Each main vertex is labeled with a variable referred to as *density*. The interested reader is urged to read the original paper (10).

Holland claims that from the generation tree and the transition equations of any particular generation procedure, one can *calculate the expected values of the densities of the main vertices as a function of time.* Holland writes: "From the general form of the transition equations one can determine such things as conditions under which the resulting generation procedures are *stationary* *processes.*" Thus Holland already tried to formulate a stochastic theory of program generation! This is an idea still waiting to be explored.

Holland's next extension of the system is similar in spirit to von Neumann's self-reproducing automata. Holland introduces *supervisory programs* which can construct *templates* which alter the probabilities of connections. Templates play the role of catalysts or enzymes. Thus program construction is also influenced by some kind of "chemical reactions."

The above process is not yet adaptive. Adaptation needs an environment posing problems. Therefore Holland proposes that the *environment is treated as a population of problems*. These problems are presented by means of a finite set of initial statements and an algorithm for checking whether a purported solution of the problem is in fact a solution. Hollland then observes the problems of partial solutions and subgoals. "When we consider the interaction of an adaptive system with its environment we come very soon to questions of partial solutions, subgoals etc. *The simplest cases occur when there is an a priori estimate of the nature of the partial solution and a measure of the closeness of its approach to the final solution*."

Holland then observes that a *rich environment* is crucial for the adaptation. "Mathematical characterization of classes of rich environments relative to a given class of adaptive systems constitutes one of the major questions in the study of adaptive systems. ... An adaptive system could enhance its rate of adaptation by somehow enriching the environment. Such enrichment occurs if the adaptive system can generate subproblems or subgoals whose solution will contribute to the solution of the given problems of the environment."

It is very interesting to note that Holland distinguished three kinds of programs – supervisory programs, templates, and programs for the problem solution. The supervisory programs use a probabilistic generation tree to generate programs, the templates are used as catalyst to "skew" the generation process. Holland perceived a hierarchy of programs (9):

1. productive systems – the generator system is able to produce other generators
2. autocatalytic systems – the generator system produces generators which are used in the construction
3. self-duplicating systems – the generator system produces duplicates of itself
4. general adaptive systems – has still to be defined

"The beginning of such a definition (of adaptive systems) lies in the following consideration: with the help of concepts such as autocatalytic and self-duplicating generator systems it is possible to define such concepts as steady-state equilibria and homeostasis for embedded automata... If the generator system for such an automaton has a hierarchical structure, then a small change in structure produces a small change in proportion to the "position" of the change in the hierarchy... By making changes first at the highest level and then at progressively lower levels of the hierarchy, it should be possible to narrow down rather quickly to any automaton in this category having some initially prescribed behavior."

I believe that Holland's proposal is a very good starting point for future research. It puts forward many ideas not yet contained in current research. After working several years on this theory Holland turned to a much simpler evolution model, in fact the Modern Synthesis mentioned before. The environment is hidden in a *fitness function*. Evolution then reduces to an optimization problem. This research lead to the famous *genetic algorithm*.

1.5 The beginning of artificial intelligence - the Logic Theorist

The term artificial intelligence was coined in the mid fifties. One of the first achievements was the logic theory machine, also called the Logic Theorist LT by Newell, Shaw and Simon (16). LT proved theorems in elementary symbolic logic, more precisely the sentential calculus. It consists of expressions built from combinations of basic symbols. *Principia Mathematica* from Russel and Whitehead lists five expressions as axioms for the sentential calculus. The first three are

$$
\begin{aligned}
(p \; or \; q) &\rightarrow p \\
p &\rightarrow (p \; or \; q) \\
(p \; or \; q) &\rightarrow (q \; or \; p)
\end{aligned}
$$

p and q are binary variables. Given any variable p we can form $(not\; p)$ Given any two variables we can form the expression $(p \; or \; q)$ or $p \rightarrow q$. From these axioms theorems can be derived.

When the LT found a simpler proof of proposition 2.85 of Principia Mathematica, Simon wrote to Russel: "We have accumulated some interesting experience about the effects of simple learning programs superimposed on the basic performance program. For example we obtain rather striking improvements in problem-solving ability by inducing the machine to remember and use the fact that particular theorems have proved in the past useful to it in the connection with particular proof methods.....In general, the machine's problem solving is much more elegant when it works with a selected list of strategic theorems than when it tries to remember and use all the previous theorems" ((20) ,p.208).

Russel answered: "I am delighted by your example of superiority of your machine to Whitehead and me...I am also delighted by your exact demonstration of the old saw that wisdom is not the same thing as erudition" ((20), p. 208).

Simon made serious attempts to interpret LT as a psychological theory of problem solving. But after analyzing thinking-aloud protocols he realized that LT did not yet fit at all the detail of human problem-solving revealed by the protocols. Newell and Simon identified the subjects principal problem solving tool. They called it *means-ends analysis*.

Means-ends analysis is accomplished by comparing the problem goal with the present situation and noticing one or more differences between them. The observed difference jogs memory for an action that might reduce or eliminate the differences. The action is taken, a new situation is observed, and if the goal has still not been reached, the whole process is repeated. Means-ends analysis is used today in many problem solving tools. In principle backpropagation in artificial neural networks can also be seen as means-ends analysis.

Means-ends analysis is the central component of the next AI system Newell, Shaw, and Simon developed. It was named the *General Problem Solver* GPS. It is an impressive system incorporating many important problem solving techniques, but difficult applications have not been reported.

The success of LT lead Simon and Newell in 1958 to their famous prediction : "I do not want to shock you, but there are now in the world machines that think, that learn, and that create. Moreover, their ability to do these things is going to increase rapidly until - in a visible future - the range of problems they can handle will be coextensive with the range to which the human mind has been applied (19)."

1.6 Discussion of the early proposals to create artificial intelligence by simulating evolution

I have reviewed only four of the early proposals which simulate natural systems to create machine intelligence. I have chosen the proposals of von Neumann and Turing because they have been the earliest. One observation strikes immediately: all researchers investigated the problem of machine intelligence on a very broad scale. The main emphasis of Turing was the design of efficient learning schemes. For Turing it was obvious that only by efficient learning of something like a child machine an intelligent machine could be developed. The attitude of Turing was purely that of a computer scientist. He firmly believed that machine intelligence equal to or surpassing human intelligence could eventually be created.

Von Neumann's approach was more interdisciplinary, using also results from the analysis of the brain. He had a similar goal, but he was much more cautious concerning the possibility to create an automaton with intelligence. He identified important problems which blocked the road to machine intelligence.

Both von Neumann and Turing investigated formal neural networks as a basic component of an artificial brain. This component was not necessary for the design, it was used only to show that the artificial automata could have a similar organization as the human brain. Both researchers ruled out that a universal theory of intelligence could be found, which would make it possible to program a computer according to this theory. So Turing proposed to use *learning* as the basic mechanism, von Neumann *self-reproducing automata*.

Von Neumann was sceptical about the creation of machine intelligence. He was convinced that learning leads to the *curse of infinite enumeration*. While every single behavior can be unambiguously described, there is obviously an infinite number of different behaviors. Turing also saw the limitations of teacher based learning by reward and punishment, therefore he required that the machine needs *initiative* in addition. Turing had no idea how the learning techniques initiative could be implemented. He correctly observed that it was necessary for creating machine intelligence by learning. Higher-level learning methods are still an open research problem.

The designs of Turing and von Neumann contain all components considered necessary today for creating machine intelligence. Turing ended his investigation with the problem of learning by initiative. Von Neumann invented as a first step self-reproducing cellular automata.

There is no major flaw in their designs. Von Neumann's question - can visual analogy be described in finite time and limited space, is still unsolved.

In order to make the above problem clear, let me formulate a conjecture: The computational universe can be divided into three sectors: *computable problems*; *non-computable problems* (that can be given a finite, exact description but have no effective procedure to deliver a definite result); and, finally, problems whose individual behaviors are, in principle, computable, but that, in practice, we are *unable to formulate in an unambiguous language* understandable for a Turing machine. Many non-computable problems are successfully approached by heuristics, but it seems very likely that the problem of visual analogy belongs to the third class.

Holland proposed a general scheme for breeding intelligent programs using the mechanisms of evolution. This was the most ambitious proposal using program generation by evolutionary principles to create intellgent machines. This proposal tried to circumvent Turing's problem to code all the necessary knowledge.

Let us try to contrast the approach of Turing with those of von Neumann and Holland. Turing proposed to programme the knowledge the humans have. In order to speed up the implementation he suggested to programme an automaton with only child like intelligence. The automaton child is then teached to become more intelligent.

Von Neumann was skeptical if all the components necessary for human like intelligence could be programmed in finite time and finite space. Therefore von Neumann started with the idea to automatically evolve automata. This idea was extended by Holland proposing an environment of problems to evolve the automata. On the first sight this seems to solve the programming problem. Instead of copying human like intelligence an environment of problems was used. But Holland overlooked the *complexity of programming the problems*. This seems to be not easier than to programme the knowledge humans have about the environment.

Holland's proposal to use stochastic systems, their steady-state equilibria and homeostasis is in my opinion still a very promising approach for a constructive evolution theory of automata. Holland itself never implemented his general model. It is still a theoretical design.

Later von Neumann's proposal has been extended insofar as both, the *problem solving programs and the problems evolve together* (14). This obviously happened in natural evolution. In a new research discipline called *artificial life* several attempts have been made to evolve automata and the environment together, but the evolution always stopped very early.

Newell, Shaw and Simon concentrated on the higher level problem solving capabilities of humans. Evolutionary principles or lower level structures like the human brain are not considered to be relevant. Instead a theory of problem solving by humans is used. Their research lead to cognitive science and to artificial intelligence research based on theories of intelligence. Despite their great optimism no convincing articial intelligence system has been created so far using this approach.

1.7 Cyc and Cog: Two large projects in the legacy of Alan Turing

Only very few big projects have been pursued in the spirit of Alan Turing. Two recent examples are the projects Cyc and Cog. Cyc is an attempt to assemble a comprehensive ontology and data base of everyday knowledge, with the goal of enabling the system human-like reasoning. The goal of the Cog project was to create a humanoid robot.

1.7.1 The Cyc project

The Cyc project was started in 1984 with the goal to specify in a *well-designed language common sense knowledge* (12; 6). Cyc attempts to assemble a comprehensive ontology and database of everyday common sense knowledge, with the goal of enabling AI applications to perform human-like reasoning. The original knowledge base is proprietary, but a smaller version of the knowledge base, intended to establish a common vocabulary for automatic reasoning, was released 2005 as OpenCyc under an open source license.

Typical pieces of knowledge represented in the database are "Every tree is a plant" and "Plants die eventually". When asked whether trees die, the inference engine can draw the obvious conclusion and answer the question correctly. The Knowledge Base (KB) contains over a million human-defined assertions, rules or common sense ideas. These are formulated in the language CycL, which is based on predicate calculus and has a syntax similar to that of the Lisp programming language.

Much of the current work on the Cyc project continues to be knowledge engineering, representing facts about the world by hand, and implementing efficient inference mechanisms on that knowledge. Increasingly, however, work at Cycorp involves giving the Cyc system the ability to communicate with end users in natural language, and to assist with the knowledge formation process via machine learning.

Currently (2007) the knowledge base consists of

- 3.2 million assertions (facts and rules)
- 280,000 concepts
- 12,000 concept-interrelating predicates

Cyc runs now for 32 years, it is the longest running project in the history of AI. But despite its huge effort its success is still uncertain. Up to now Cyc has not been successfully be used for any broad AI application. The system is far away to be used for a Turing test.

We remind the reader, that the coding of knowledge was considered by Turing as too inefficient. Von Neumann even doubted if the necessary knowledge for visual analogy could be specified in finite time. Today Cyc seems to be more a confirmation of von Neumann's doubt than a refutation.

1.7.2 The Cog project

The Cog project was started in 1993 with extreme publicity. The goal was to understand human cognitive abilities well enough to build a humanoid robot that develops and acts similar to a person (3; 4). One of the key ideas of the project was to build a robot with capabilities similar to a human infant. We have encountered this idea already in Turing's proposal.

"By exploiting a gradual increase in complexity both internal and external, while reusing structures and information gained from previously learned behaviors, we hope to be able to learn increasingly sophisticated behavior (4)." Cog was designed bottom-up (3). This lead to good success in the beginning. The big problems appeared later.

Brooks et al. wrote prophetically: *To date (1999), the major missing piece of our endeavor is demonstrating coherent global behavior from existing subsystems and sub-behaviors. If all of these systems were active at once, competition for actuators and unintended couplings through the world would result in incoherence and interference among the subsystems (4).*

During the course of the project a lot of interesting research has been done. But the problem of coherent or even intelligent behavior could not be solved. Therefore the project was stopped in 2002 without even entering the learning or development phase.

1.8 The JANUS hand-eye robot and the Pandemonium architecture

With my research group I have also tried two larger research projects in the spirit of Alan Turing and John von Neumann. The most spectacular was our hand-eye robot JANUS. With JANUS we bridged the gap between small-scale neural networks and real-world applicability. The robot had two eyes and two arms with which it observed and manipulated its environment. The robot learned from experience and self-supervision, initialized only with a few essential properties. JANUS also incorporated structural and functional medical knowledge of the brain.

The JANUS architecture was directly influenced by the top-level structure of the human brain and its hemispherical functional lateralization (7; 22). However the similarities end at that level and a great deal of freedom is permitted in lower-level neural networks. The name JANUS was chosen after the Roman god for a specific reason: The brain not only looks out and observes and weighs up its environment, but it also looks *inwardly* and is aware of its own processes. It has a *reflective architecture*.

The JANUS brain controls a physical robot that may exist in a changing environment. The robot can be affected by the environment either directly, through physical contact with objects, or indirectly by the thought and learning processes. The highest level description of the JANUS architecture is illustrated in figure 1.1.

The brain is divided in two halves laterally and two blocks vertically. All sensory signals from the left side of the robot pass directly to the right half of the brain, while those from the right side pass directly to the left half of the brain. The left half of the brain controls the motor outputs affecting the right side of the robot, and similarly the right half controls the left motor side. There exist an important connection between the two hemispheres (the *corpus callosum*) where information is exchanged.

The central concept of the JANUS architecture is the notion of *self-assessment* or self-supervision, within a hierarchy of adaptable network modules. The modules can modify themselves, and higher levels can act on other levels. In order that this might be possible, each module tries to estimate its limitations through self-assessment measures like confidence and reliability.

The JANUS project run from 1991 till 1997. It had to be abandoned because of lack of funding. The research progress was promising, but in 1997 JANUS was still far away to be used in a real application.The research has been published in the series GMD reports. The reports are out

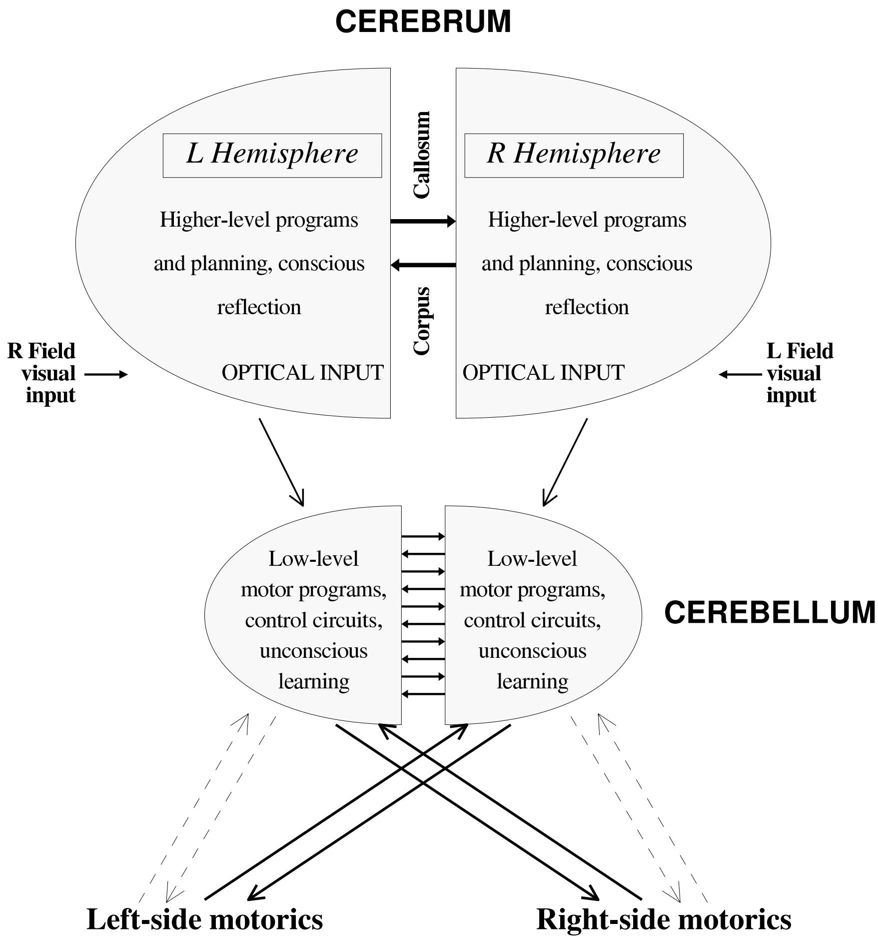

Fig. 1.1 The architecture of the JANUS brain

of print. The easiest access is via the WEB (http://citeseer.ist.psu.edu) or www.iais.fraunhofer.de/muehlenbein.html.

The low-level neural network architecture of JANUS has been investigated separately. We called it the PANDEMONIUM or MINOS architecture. Pandemonium had been originally proposed in 1958 by Selfridge (18). The idea is to divide adaptively a complex domain, through the use of specialized agents, working in parallel. All these agents, or daemons in Selfridge's words, process the same signal in parallel, and each provides an answer with a certain confidence. The daemon with the largest confidence will be chosen for classification.

Thus for a letter classification problem we might have 26 agents, each of which is specialized on recognizing a particular letter in all distortions. Each agent uses a number of filters. The *learning* method used by Selfridge was gradient descent for adapting the weights for each filter used.

We have taken this general idea and extended it to a modular system of neural networks. The central new idea is self-assessment by *reflection*. Each module observes its own behavior and produces information relating to the quality of its classification. The architecture was very successful in a number o classification tasks, but in the course of developing it more and more refinements had to be implemented. The interested reader is referred to (21; 1; 2).

1.9 Conclusion

Today computational intelligence is divided into many fields e.g. evolutionary computation, neural networks, fuzzy logic. These are further separated in a myriad of specialized techniques. In this paper I have recalled the fundamental research issues of machine intelligence by discussing the research of Alan Turing and John von Neumann. They represent two positions popular till today. For Turing the creation of machines with human-like intelligence was just a question of programming time. He estimated that sixty programmers had to work for fifty years. John von Neumann was more cautious. Using the example of visual analogy he doubted that human-like intelligent machines could be programmed in finite time and space. This lead him to the question if intelligent programs could automatically evolve by simulating evolution. While von Neumann solved the problem of self-reproducing automata, automata solving complex problems could not be yet obtained. I have identified the major peoblem of this approach: the programming of the environment seems to be as difficult as programming the human problem solving caoabilities.

In my opinion it is not yet clear if Turing will be ultimately right that automata with human like intelligence could be programmed. Up to now computational intelligence was successful in specialized applications only, automata passing the Turing test or understanding languages are not yet in sight.

References

[1] U. Beyer and F. J. Smieja. Data exploration with reflective adaptive models. *Computat. Statistics and Data Analysis*, 22:193–211, 1996.
[2] U. Beyer and F. J. Smieja. Learning from examples, agent teams and the concept of reflection. *International Journal of Pattern Recognition and Artificial Intelligence*, 10:251–272, 1996.
[3] R. Brooks. From earwigs to humans. *Robotics and Autonomous Systems*, 20:291–304, 1997.
[4] R. Brooks, C. Brezeal, M. Marjanovic, B. Scasselati, and M. Williamson. The Cog project: Building a humanoid robot. In C.A. Nehaniv, editor, *Computation for Metaphors, Analogy, and Agents*, Lecture Notes in AI 1562, pages 52–87. Springer, 1998.
[5] A.W. Burns. *Essays on Cellular Automata*. University of Illinois Press, Urbana, 1970.

[6] Y. Cai and J. Abascals. Common sense reasoning: From CYC to intelligent assistant. In C.A. Nehaniv, editor, *Ambient Intelligence in Everyday Life*, Lecture Notes in AI 3864, pages 1–31. Springer, 2006.
[7] J.C. Eccles. *The Human Mystery*. Springer, New York, 1979.
[8] M.R. Garey and D.S. Johnson. *Computers and intractability: a guide to the theory of NP-completeness*. Freeman, San Francisco, 1979.
[9] J.H. Holland. Iterative circuit computers. In *Essays on Cellular Automata* (5), pages 277–296.
[10] J.H. Holland. Outline for a logical theory of adaptive systems. In *Essays on Cellular Automata* (5), pages 296–319.
[11] J.H. Holland. *Adaptation in Natural and Artificial Systems*. Univ. of Michigan Press, Ann Arbor, 1975/1992.
[12] D.B. Lenat. Cyc: A large-scale investment in knowledge infrastructure. *Comm. ACM*, 38:33–38, 1995.
[13] W.S. McCulloch and W. Pitts. A logical calculus of the ideas immanent un nervous activity. *Bull. of Mathematical Biophysics*, 5:115–137, 1943.
[14] B. McMullin. John von Neumann and the evolutionary growth of complexity: Looking backward, looking forward... *Artificial Life*, 6:347–361, 2001.
[15] H. Mühlenbein. Towards a theory of organisms and evolving automata. In A. Menon, editor, *Frontiers of Evolutionary Computation*, pages 1–36. Kluwer Academic Publishers, Boston, 2004.
[16] A. Newell, J.C. Shaw, and H. Simon. Empirical explorations with the logic theory machine. *Proc. Western Joint Computer Conference*, 11:218–239, 1957.
[17] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufman, San Mateo, 1988.
[18] O.G. Selfridge. Pandemonium: a paradigm for learning. In *Mechanisation of Thought Processes*, pages 511–529. Her Majesty's Stationery Office, London, 1959.
[19] H. Simon and A. Newell. Heuristic problem solving: The next advance in operations research. *Operations Research*, 6:1–10, 1958.
[20] H. A. Simon. *Models of my Life*. MIT Press, Boston, 1991.
[21] F. J. Smieja. The pandemonium system of reflective agents. *IEEE Transact. on Neural Networks*, 7:193–211, 1996.
[22] S. Springer and G. Deutsch. *Left brain, right brain*. W.H. Freeman, New York, 1985.
[23] A.M. Turing. Computing machinery and intelligence. *Mind*, 59:433–460, 1950.
[24] A.M. Turing. Intelligent machinery. In B. Meltzer and D. Michie, editors, *Machine Intelligence 6*, pages 3–23. Oxford University Press, Oxford, 1969.
[25] J. von Neumann. The general and logical theory of automata. In *The world of mathematics*, pages 2070–2101. Simon and Schuster, New York, 1954.
[26] J. von Neumann. Probabilistic logics and the synthesis of reliable organs from unreliable components. In *Annals of Mathematics Studies 34*, pages 43–99. Princeton University Press, 1956.
[27] J. von Neumann. *Theory of Self-Reproducing Automata*. University of Illinois Press, Urbana, 1966.