The Child Machine vs the World Brain

Claude Sammut
School of Computer Science and Engineering, The University of New South Wales, Sydney, Australia
claude@cse.unsw.edu.au

Keywords: Turing, Machine Learning

Received: December 17, 2012

Machine learning research can be thought of as building two different types of entities: Turing's Child Machine and H.G. Wells' World Brain. The former is a machine that learns incrementally by receiving instruction from a trainer or by its own trial-and-error. The latter is a permanent repository that makes all human knowledge accessible to anyone in the world. While machine learning began following the Child Machine model, recent research has been more focussed on "organising the world's knowledge"

Povzetek: Raziskovanje strojnega učenja je predstavljeno skozi dve paradigmi: Turingov Child Machine in H.G. Wellsov World Brain.

1 Encountering Alan Turing through Donald Michie

My most immediate knowledge of Alan Turing is through many entertaining and informative conversations with Donald Michie. As a young man, barely out of school, Donald went to work at Bletchley Park as a code breaker. He became Alan Turing's chess partner because they both enjoyed playing but neither was in the same league as the other excellent players at Bletchley. Possessing similar mediocre abilities, they were a good match for each other. This was fortunate for young Donald because, when not playing chess, he learned much from Turing about computation and intelligence. AlthoughTuring's investigation of machine intelligence was cut short by his tragic death, Donald continued his legacy. After an extraordinarily successful career in genetics, Donald founded the first AI group in Britain and made Edinburgh one of the top laboratories in the world, and, through a shared interest in chess with Ivan Bratko, established a connection with Slovenian AI.

I first met Donald when I was as a visiting assistant professor at the University of Illinois at Urbana-Champaign, working with Ryszard Michalski. Much of the team that Donald had assembled in Edinburgh had dispersed as a result of the Lighthill report. This was a misguided and damning report on machine intelligence research in the UK. Following the release of the report, Donald was given the choice of either teaching or finding his own means of funding himself. He chose the latter. Part of his strategy was to spend a semester each year at Illinois, at Michalski's invitation, because the university was trying to build up its research in AI at that time. The topic of a seminar that Donald gave in 1983 was "Artificial Intelligence: The first 2,400 years". He traced the history of ideas that lead to the current state of AI, dating back to Aristotle. Of course, Alan Turing played a prominent role in that story. His 1950 Mind paper [1] is rightly remembered as a landmark in the history AI and famously describes the imitation game. However, Donald always lamented that the final section of the paper was largely ignored even though, in his opinion, that was the most important part. In it, Turing suggested that to build a computer system capable of achieving the level of intelligence required to pass the imitation game, it would have to be educated, much like a human child.

> Instead of trying to produce a programme to simulate the adult mind, why not rather try to produce one which simulates the child's? If this were then subjected to an appropriate course of education one would obtain the adult brain. Presumably the child-brain is something like a notebook as one buys from the stationers. 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 be easily 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.

He went on to speculate about the kinds of learning mechanisms needed for the child machine's training. The style of learning was always incremental. That is, the machine acquires knowledge by being told or by its own exploration and this knowledge accumulates so that it can learn increasingly complex concepts and solve increasingly complex problems.

Early efforts in Machine Learning adopted this paradigm. For example, the Michie and Chambers [2] BOXES program learned to balance a pole and cart system by trial-and-error receiving punishments are rewards, much as Turing described, and like subsequent reinforcement learning systems. My own efforts, much later, with the Marvin program [3] were directed towards building a system that could accumulate learn and accumulate concepts expressed in a form of first order logic. More recent

systems that learn in this manner include the Robot Scientist [4] and the Xpero robot learning project [5]. However, most current machine learning research has adopted a somewhat different style. Turing could not have foreseen the internet and the effect it would have on AI. Access to huge amounts of data and computing resources suggest a kind of intelligence that may be built in a very different way than human intelligence.

Another significant historical figure did anticipate something like the internet. At around the same time of Turing's paper on computing machines [6], H.G. Wells speculated on what he called "the world brain" [7], a permanent world encyclopedia that would provide, "...a sort of mental clearing house for the mind, a depot where knowledge and ideas are received, sorted, summarised, digested, clarified and compared.... any student, in any part of the world, will be able to sit with his projector in his own study at his or her convenience to examine any book, any document, in an exact replica."

Wells thought that the technology for this repository of knowledge would be microfilm. He could not have known that the World Wide Web, Wikipedia and internet search engines could bring about his vision on a far larger scale than he imagined. Indeed, Google's company mission is "...to organize the world's information and make it universally accessible and useful"[1]. A core technology for such search engines is machine learning. However, it is of a form that is somewhat different form that described by Turing for his Child Machine. Unlike a human child, acquiring new knowledge incrementally, machine learning systems can access the enormous amounts of data available throughout the internet to produce, in some instances, superhuman performance. However, this is usually *all-at-once* and *single-concept-at-a-time* learning that must still be guided by humans.

The first programs capable of efficiently learning from moderately large numbers of examples began with Michalski's Aq [8] and Quinlan's ID3 [9], both of whom were influenced by Donald Michie. Quinlan was a PhD student studying with psychologist Earl Hunt when he attended a lecture by Michie at Stanford University. Donald challenged the students to devise a method of learning to determine a win in a chess end-game and Quinlan responded with the first version of his decision tree learner. The utility of these programs for finding patterns in large data sets encouraged new research into "batch" learning systems, eventually leading to the large-scale statistical learning methods commonly in use today. A characteristic of most systems that are used for mining patterns in "big data" is that they use fairly simple features and rely of masses of data to build robust classifiers. Structure in representation is generally created by the human data analyst. Currently, the construction of the world brain is a partnership between human and machine. But how far can this continue? Will humans be able to structure truly large knowledge sources as the amount and complexity of information increases, or will machines have to take over at least some of that job too? If that is the case, what mechanisms can be employed to do this?

Some search engines are already beginning to incorporate some *semantic integration*. For example, Google's Knowledge Graph[2] uses what is essentially a semantic net to supplement search information with the properties of objects and their relationships to other objects. However, most of the semantic information is derived from human built ontologies, and this poses a problem. Handcrafted ontologies reflect a knowledge engineers assumptions about structure in the data, which are not necessarily true, whereas a machine built ontology should be more faithful to the actual evidence for a particular semantic structure. Learning systems are capable if finding relations between entities [10] but the complexity is greater than learning propositional representations. This complexity can be reduced if concepts are learned incrementally. That is simple concepts are learned first and become background knowledge for further learning. Learning can also be assisted if the systems is capable of selected its own examples to test hypotheses. When these elements are incorporated, machine learning research once again resembles Turing's model of a child machine, but perhaps not entirely as he envisaged because it will also include the capabilities of the "world brain".

The study of learning in the style of the child machine, which accumulates knowledge active experimentation as been neglected, although there are some notable exceptions [11, 12, 13, 14]. There are some very significant open questions that are yet to be answered for this approach to work. We discuss these in the next section.

2 Cumulative active learning

Many of the problems associated with building a Child Machine have been addressed in the past. An unfortunate effect of the success of the "big data" approach to machine learning has been to distract researchers from investigating what I have termed cumulative active learning, which is essential for the Child Machine. The structure of a cumulative, or never-ending, learning system that acquires data by experimentation was nicely illustrated by Mitchell *et al.* [15] in 1983 (Figure 1) and his more recent work on never-ending language learning [13] follows a similar structure. An experiment or test of a hypothesis is created by a problem generator, the problem solver performs the experiment whose outcome is judged by a critic. The critic labels the results of the experiments as positive, if the problem solver succeeded in its task or negative if it failed. The learner then updates its hypothesis, based on the new training data and the process repeats as long as there are remaining learning tasks.

A requirement for cumulative learning is a mechanism

[1] http://www.google.com/about/company/

[2] http://www.google.com/insidesearch/features/search/knowledge.html

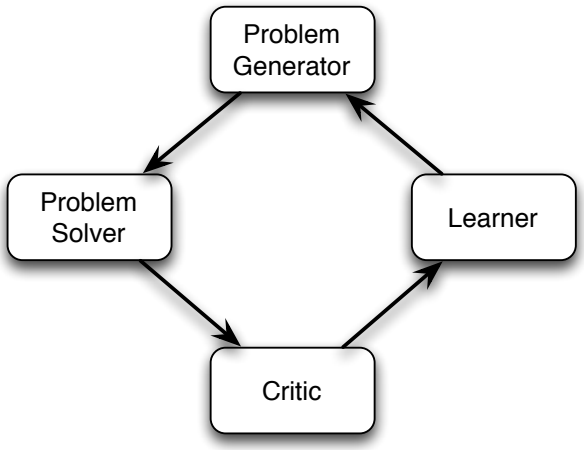

Figure 1: A theory is a network of concepts.

for storing and using learned concepts as *background knowledge*. Representing concepts in the form of expressions in first order logic makes this relatively easy. To my knowledge, Banerji [16] was the first propose such a mechanism and the first implementation of a learning system that could use learned concepts for further learning was by Cohen [17]. With Brian Cohen, I elaborated this idea to create a learning system in which the concepts were represented as logic programs and were, thus, executable [18]. This work evolved into Marvin [19] and other related systems [20, 21] developed similar capabilities. The ability to use of background knowledge has become one of the distinguish features Inductive Logic Programming [22]. However, even many ILP systems that use background knowledge to learn new concepts do not close the loop to allow those learned concepts to, themselves, become part of the background knowledge for future learning. Other systems, such NELL [13] currently only use background knowledge that is in the form of ground clauses [23].

3 Learning as debugging a logic program

Since, in practice, a learning agent can never be exposed to all instances of a concept, it is prone to making mistakes of two kinds:

- The system may observe an event for which there is no prior knowledge about how to respond.
- The system may observe an event for which there is prior knowledge. However, the known concepts are too general and the system may respond inappropriately.

When the system is learning only one concept at a time, repairing a theory is relatively easy. We just specialise an over-general theory or generalise a theory that is too specific. However, in a system that accumulates knowledge over time, learning many concepts, localising the error is not so easy. We can think of a theory as being a network of interdependent concepts, as in Figure 2, where the definition of concepts S and T rely on the definition of Q, which depends on P and E. Suppose we discover an error in the theory while attempting to use concept, T, but the real error is due to an incorrect definition of concept, E.

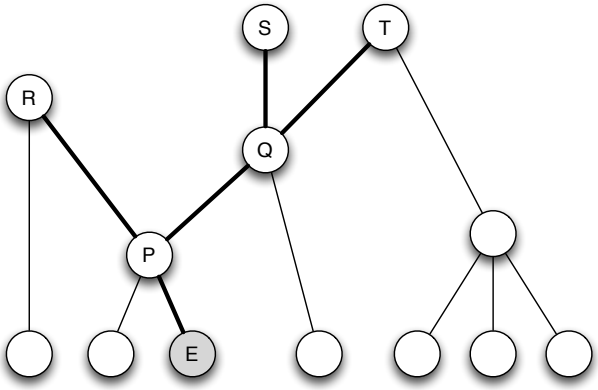

Figure 2: A concept hierarchy.

When we use a logical representation of concepts, a concept description is also an executable computer program, so one way of locating the problem is to trace through the execution of the program that lead to the unexpected result, testing each concept. That is, we debug the program, as Shapiro [24] did with MIS. To locate an error when an incorrect solution has been given (i.e. the theory contains an over-generalisation) Shapiro's debugging algorithm works backwards through the failed proof of a goal, searching for the procedure that caused the failure. In Figure 2, backtracing would begin with the last goal satisfied, that is, T. The debugger begins stepping back through the proof, i.e. down the dark path to node Q, then P if necessary, asking an oracle if the partial solution at each point is correct. If this is not true, then an erroneous clause has been found. Note that the algorithm assumes the existence of an infallible oracle. In a reactive environment, the learning program may do without an oracle since the program is able to perform experiments to test a concept. Thus, a failure suggests that the initial set of experiments that resulted in the formation of the concepts along the solution path was not extensive enough for at least one of the concepts.

A concept that is too specific may prevent the program from producing any solution at all. That is, the logic program that is supposed to satisfy the goal does not cover the initial conditions of the task. An attempt at debugging the theory can only be made when a correct solution has been seen, otherwise the learner has no indication that the task really is possible. A correct solution may be found, either by "mutating" the current theory in the hope that the goal can be satisfied by the mutant or the learner may observe

another agent in the world performing the task. Shapiro's debugging method for programs that fail to produce an answer is equivalent to the second alternative, that is, the oracle supplies the correct solution. The debugger again tries to work backwards seeking clauses in the program which could have produced the given solution. Once such a clause is found, its body provides further goals that should be satisfied in order to arrive at the solution. The debugger considers each of these intermediate goals to see if they can also be produced by other clauses. Any goal that cannot be achieved indicates where the program or theory is deficient.

Up to this point, we have described mostly past research that addresses many of the problems that are needed to build a Child Machine but further work is still required. In the next section we look at a problem that has received very little attention, mainly because we have not yet built AI systems that are sufficiently complex.

4 The Economics of learning and theory revision

Detecting and repairing an error in a single concept is one thing, but repairing an entire theory is another matter. Remember that in Figure 2, we envisaged a world model or domain theory as a hierarchy of concepts. Using a horn clause representation, the head of a clause corresponds to a parent node and the goals in the body correspond to the children. These goals match other clause heads and form links to the rest of the network. Also in Figure 2, we imagined that one concept, represented by the shaded node, E, was in error. When the concept is repaired, what effect will that have on the concepts which referred to the old concept? Since P, Q, R, S and T refer, directly or indirectly, to the erroneous node they must have been learned in the presence of the error. Are they, therefore also in error or will correcting E alone correct them all?

When faced with the problem of ensuring the consistency of its knowledge base, two strategies are available to the learning system.

1. After correcting E, the system may test each of the concepts that depend on E. However revising all of the concepts dependent on one that just been modified could involve a lot of work if the network of concepts is very extensive.

2. The system may wait to see if any further errors show up. In this case, each concept will be debugged as necessary. Although more economical this method requires a method for tolerating errors if the program has been assigned a task which it must continue to perform.

When a learning system is connected to the physical world, as in a robot, it cannot rely on the accuracy of measurements from vision systems, touch sensors, etc. Thus, a program may fail because its world model does not accurately reflect the real state of the world. This being the case, the learning system must not revise its domain theory prematurely since there may not, in fact, be any errors in its theory. Therefore, a prudent approach to error recovery is to delay revision of a domain theory until sufficient evidence has accumulated to suggest the appropriate changes.

Richards and Mooney [25] proposed a theory revision system, based on earlier work by Muggleton [26, 20]. Wrobel [27] also proposed a first order theory refinement system. The main idea behind these methods is to consider repairing a theory as a kind of compression. That is, the set of clauses that form a theory may be rewritten so that the same or greater coverage is achieved by a theory that is more compact in an information theoretic sense. An important operation for this is predicate invention [20, 14]. That is, the learner must have the ability to invent its own concepts when it detects patterns that can be reused in the description of other concepts.

When an experiment fails, we must not invalidate the concepts used in planning the experiment for, as mentioned earlier, the failure may be due to noise. Instead, we note the circumstances of the failure and augment the failed concept with a description of these circumstances. Several things could happen to the concept when this is done:

- The description of the concept is modified to the extent that it becomes correct. If an alternative, correct description already existed, then the alternative domain theories of which these concepts were components, converge.

- After several failures, there is no generalisation which covers the circumstances of failure. In this case, the failures may be either due attributed to noise or to some phenomenon not yet known to the system. In either case, nothing can be done.

A truth maintenance system (TMS) [28] may be used to maintain the network of concepts that form a domain theory. The TMS stores dependencies which, when errors are found will indicate where other potential weaknesses in the theory lie. The TMS may also allow a learning program to experiment with alternative domain theories by maintaining multiple worlds.

Whatever mechanisms are used, some important questions to investigate are: what is the cost of learning and when it it worthwhile adopting a new theory in favour of an existing one. That is, putting up with occasional errors in an existing theory my cost less than building a new theory.

5 Deep knowledge vs shallow knowledge

Before concluding, I want to return to Turing's paper on the imitation game and what is needed in a child machine to be

able to pass the test. Turing proposed the following criteria for success:

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

It should be noted that these conditions are not particularly onerous if the conversation is confined to superficial chat, but according to Donald Michie, Alan Turing never engaged in superficial chat, so the test should be understood to pose questions that require some thought. Turing, however, was not a typical conversationalist. Many human dialogues can be conducted with "canned" responses, for example, interactions at a supermarket or enquiries to a call centre, where, in fact, the operators follow quite strict scripts.

Leading up to the 50th anniversary of Turing's paper in 2000, Donald Michie suggested to me that we should try to build a conversational agent, to see how close AI was to Turing's prediction that the test could be passed within 50 years of the publication of the paper. We implemented several frameworks for building a conversational agent [29], borrowing partly from natural language processing theory, which represents deep understanding of an utterance, and also from chatterbots, which mostly rely on simple pattern matching rules, much like Eliza [30]. The idea of mixing deep and shallow knowledge is consistent with a theme of Donald's work, namely that many tasks that we think require conscious, deliberative thought are actually achieved by simpler, subcognitive processes. He was fond of a quote from A.N. Whitehead [31]:

> It is a profoundly erroneous truism... that we should cultivate the habit of thinking what we are doing. The precise opposite is the case. Civilisation advances by extending the number of important operations which we can perform without thinking about them.

This applies even to conversation. There are many circumstances in which we can, legitimately, talk without thinking. Even many aspects of an expert's decision making are subcognitive. An expert may be defined as someone who knows his or her job so well that it can be done with little thought, only requiring real deliberation when an unusual case is encountered. Learning subcognitive skills was one of Donald's great interests for many years. He coined the term *behavioural cloning* [32] to describe the process of building an operational model of an expert's skill by observing and recording traces of the behaviour and using them to create training examples for a learning program.

There are advantages, for the Child Machine, of having a mixture or representations, reasoning and learning methods. Shallow reasoning is usually fast and covers the most common cases encountered but may break when an unusual case is seen. Reasoning methods that require complex representations and inferences are computationally expensive to perform and to learn but they are more general and may work when shallow reasoning fails.

6 Conclusion

From its initial beginnings as an attempt to perform human-like learning, Machine Learning is now in danger of becoming just another branch of statistics. The majority of research done today only concerns itself with data analysis. The field has forgotten that there is more to learning than just looking for patterns in very large data sets. While this can be useful, machine learning will eventually hit the limitations of shallow representations and will have to face the problems that a human learner experiences in trying to make sense of the world over a lifetime of accumulated experience. So while we are currently preoccupied with building the world brain, eventually this effort will have to change to making the world brain into a child machine.

This essay reviewed some of the research that is needed to create a child machine and discussed open problems that are still unanswered. Indeed, some of the questions I posed are drawn from a 1991 paper [**?**] that set the program for much of our group's research since then, and will continue well into the future. In the words of Alan Turing, "We can only see a short distance ahead, but we can see plenty there that needs to be done" [6].

References

[1] Turing, A.: Computing machinery and intelligence. Mind **59**(236) (1950) 433–460

[2] Michie, D., Chambers, R.: Boxes: An experiment in adaptive control. In Dale, E., Michie, D., eds.: Machine Intelligence 2. Oliver and Boyd, Edinburgh (1968)

[3] Sammut, C.A.: Learning Concepts by Performing Experiments. Ph.d. thesis, Department of Computer Science, University of New South Wales (1981)

[4] King, R.D., Whelan, K.E., Jones, F.M., Reiser, P.G.K., Bryant, C H, Muggleton, S.H., Kell, D.B., Oliver, S.G.: Functional genomic hypothesis generation and experimentation by a robot scientist. Nature **427**(6971) (2004) 247–252

[5] Bratko, I.: Comparison of machine learning for autonomous robot discovery. In Koronacki, J., Ras, Z.W., Wierzchon, S.T., Kacprzyk, J., eds.: Advances in Machine Learning I. Volume 262 of Studies in Computational Intelligence. Springer (2010) 441–456

[6] Turing, A.: On computable numbers, with an application to the entscheidungsproblem. Proceedings of the London Mathematical Society **2**(42) (1936) 230–265

[7] Wells, H.: World brain: The idea of a permanent world encyclopaedia. In: Encyclopédie Française. (1937)

[8] Michalski, R.S.: Discovering classification rules using variable valued logic system vl1. In: Third International Joint Conference on Artificial Intelligence. (1973) 162–172

[9] Quinlan, J.R.: Discovering rules by induction from large collections of examples. In Michie, D., ed.: Expert Systems in the Micro-Electronic Age, Edinburgh (1979)

[10] Grobelnik, M., Mladenic, D.: Automated knowledge discovery in advanced knowledge management. Journal of knowledge management (9) (2005) 132–149

[11] Brown, S., Sammut, C.: A relational approach to tool-use learning in robots. In: International Conference on Inductive Logic Programming, Dubrovnik (September 2012)

[12] Bratko, I.: Discovery of abstract concepts by a robot. In Pfahringer, B., Holmes, G., Hoffmann, A.G., eds.: International Conference on Discovery Science. Volume 6332 of Springer Lecture Notes in Computer Science. (2010) 372–379

[13] Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, Jr, E.R., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010). (2010)

[14] Muggleton, S., Lin, D., Pahlavi, D., Tamaddoni-Nezhad, A.: Meta-interpretive learning: application to grammatical inference. Machine Learning (forthcoming)

[15] Mitchell, T.M., Utgoff, P.E., Banerji, R.B.: Learning by experimentation: Acquiring and refining problem-solving heuristics. In Michalski, R., Carbonell, J., Mitchell, T., eds.: Machine Learning: An Artificial Intelligence Approach. Tioga, Palo Alto (1983)

[16] Banerji, R.B.: A language for the description of concepts. General Systems **9** (1964) 135–141

[17] Cohen, B.L.: A Theory of Structural Concept Formation and Pattern Recognition. Ph.d. thesis, Department of Computer Science, University of New South Wales (1978)

[18] Sammut, C.A., Cohen, B.L.: A language for describing concepts as programs. In Tobias, J.M., ed.: Language Design and Programming Methodology. Volume 79 of Lecture Notes in Computer Science. Springer (1980) 111–116

[19] Sammut, C.A.: Concept learning by experiment. In: Seventh International Joint Conference on Artificial Intelligence, Vancouver (1981) 104–105

[20] Muggleton, S., Buntine, W.: Machine invention of first-order predicates by inverting resolution. In Michalski, R.S., Mitchell, T.M., Carbonell, J.G., eds.: Proceedings of the Fifth International Machine. Learning Conference. Morgan Kaufmann,, Ann Arbor, Michigan (1988) 339–352

[21] De Raedt, L., Bruynooghe, M.: An overview of the interactive concept-learner and theory revisor clint. In Muggleton, S., ed.: Inductive Logic Programming. Academic Press (1992) 163–191

[22] Muggleton, S.: Inductive logic programming. New Generation Computing **8** (1991) 295–318

[23] Quinlan, J.R.: Learning logical definitions from relations. Machine Learning **5** (1990) 239–266

[24] Shapiro, E.: An algorithm that infers theories from facts. In: Proceedings of the Seventh International Joint Conference on Artificial Intelligence, Vancouver, Morgan Kaufmann (1981) 446–451

[25] Richards, B., Mooney, R.: First-order theory revision. In: Proceedings of the Eighth International Machine Learning Workshop, Evanston, IL (1991) 447–451

[26] Muggleton, S.: Duce, an oracle based approach to constructive induction. In: Proceedings of the International Joint Conference on Artificial Intelligence, Milan (1987) 287–292

[27] Wrobel, S.: First order theory refinement. In: Advances in Inductive Logic programming. Volume 32. (1996) 14–33

[28] de Kleer, J.: An assumption-based tms. Artificial Intelligence **28**(1) (1986) 127–162

[29] Sammut, C.: Managing context in a conversational agent. Electronic Transactions on Artificial Intelligence **5**(B) (2001) 189–202 Full paper from Machine Intelligence 18.

[30] Weizenbaum, J.: Eliza - a computer program for the study of natural language communication between man and machine. Communications of the ACM **9**(1) (1966) 36–35

[31] Whitehead, A.: An Introduction to Mathematics. Henry Holt and Company, New York (1911)

[32] Michie, D., Bain, M., Hayes-Michie, J.: Cognitive models from subcognitive skills. In Grimble, M., McGhee, S., Mowforth, P., eds.: Knowledge-base Systems in Industrial Control. Peter Peregrinus (1990)