An approach to the artificial intelligence problem

J. McCarthy

(I underline terms with which the reader may have difficulty. Consult the vocabulary if you aren't sure we agree on meaning.)

1. Basic objective: To write a calculator <u>program</u> which can solve <u>intellectual</u> <u>problems</u> as well as or better than a human being.

Method: Choose a suitable* class of problems.

2. Devise a specific routine for solving them, program it, and try it. Observe the results of the program's efforts and use them to improve the program. (If seeing the program in action tells us nothing we could not have guessed before running it, running it is a waste of machine time. If we know nothing which we didn't know before writing it, then writing it was wasteful. This seems to rule out the programming of known decision procedures.)

3. After a number (0 to 5) of cycles of improving and observing the activity of the program, the program becomes <u>intelligent</u> enough so that we can turn it to improving itself. Presumably, because of the greater diligence of an electronic calculator, this will be long before it is as intelligent as we are.

4. Since we intend eventually to turn the program to improving itself, it will be helpful to choose the original program so that what we are able to teach it will be helpful in making the transition to improving itself. It does not seem to be feasible, however, to make this the initial task; it lacks definiteness and may even be paradoxical.

However, what is the intellectual nature of the problem of writing a program or devising a procedure to do x? It seems to be two-fold. First there is the problem of collecting facts. These facts are derived from observation and deduction in an order not usually sequential in the direction of the ultimate procedure.

\- - - -

*The criterion for suitability will be taken up later.

Next, given enough facts to determine the procedure, an actual sequential order of steps or, more generally, a strategy is devised.

These two stages of course may blend, but it seems to me that as much separation as possible should be made.

5. I tend to shy away from methods which rely on statistics and expect order to emerge from a statistical chaos by counting the successes and failures of procedures, which differ from each other mainly in the setting of parameters. Solving a problem of even moderate difficulty involves a large number of decisions, the criteria for success are too far removed from an elementary act, and the methods of generating complex actions which cannot be selected component-wise seem too problematical. At present they do not even seem to have been described unambiguously.

Suitable problems.

1. Program writing.
   A. Given enough facts which specifically describe how to calculate μ given λ, the writing of a program seems to be fairly routine even if optimizing it is a design problem.
   B. Collecting facts which describe how to get from λ to μ and which are relevant to getting from α to ω is an example of an <u>ill defined problem</u>.
   C. Collecting facts relevant to getting from α to ω, which are not descriptions of how to get from λ to μ, is perhaps an even more poorly defined problem.

However, we should choose problems for our initial program so that the methods used will be relevant to this latter complex of problems.

The following have been suggested:

1. Games. Procedures are uppermost here, but it has been suggested that the presence of conflict is uncharacteristic of the problems in which we are really interested. Moreover, there may not be enough chance to derive general principles. Still, it may be the best place to practise setting up selection hierarchies.

2. Puzzles. The first objection to games is removed, but there seems to be a dearth of good puzzles not demanding a knowledge of English.
3. Theorem proving. Perhaps the problems here are too well defined.
4. Formation of visual or aural invariants.
   a. Results obtained and methods used probably will not help with program improvement.
   b. Simulating analogue equipment or building it is a difficult and irrelevant task.
5. Program writing. This may be a good bet if a good subject can be found.
6. Finding examples having given properties. This would help study the semantic side of axiomatic theories and in mathematics is the skill which is perhaps most closely associated with common sense.
7. Design problems. After a while there is just toc much enumeration.

* * * * * *

Program -- I suggest we use program as the agent in artificial intelligence, i.e. in sentences like "The program does this" rather than "The machine does this."

Intelligence -- We stick to the naive use of the word, avoid splitting hairs and borderline cases.

Problem -- A well defined problem is given when a procedure is specified for verifying whether a sequence of words is an answer. For less well defined problems there may only be a procedure for deciding whether one answer is better than another. A still less well defined problem is that of producing an intermediate result which will be of use in solving another problem.