National Research Council Canada | Conseil national de recherches Canada

Canada

Reprinted from

Réimpression du

Computational Intelligence

Intelligence informatique

# What is a heuristic?

Marc H. J. Romanycia and Francis Jeffry Pelletier

Volume 1 • Number 2 • 1985

Pages 47–58

# What is a heuristic?

MARC H. J. ROMANYCIA
*Information Services, Engineering and Planning, Gulf Canada, Calgary, Alta., Canada T2P 2H7*
AND
FRANCIS JEFFRY PELLETIER
*Departments of Philosophy, Computing Science, University of Alberta, Edmonton, Alta., Canada T6G 2E5*

Received February 5, 1985
Revision accepted March 29, 1985

From the mid-1950's to the present the notion of a heuristic has played a crucial role in the AI researchers' descriptions of their work. What has not been generally noticed is that different researchers have often applied the term to rather different aspects of their programs. Things that would be called a heuristic by one researcher would not be so called by others. This is because many heuristics embody a variety of different features, and the various researchers have emphasized different ones of these features as being essential to being a heuristic. This paper steps back from any particular research program and investigates the question of what things, historically, have been thought to be central to the notion of a heuristic and which ones conflict with others. After analyzing the previous definitions and examining current usage of the term, a synthesizing definition is provided. The hope is that with this broader account of 'heuristic' in hand, researchers can benefit more fully from the insights of others, even if those insights are couched in a somewhat alien vocabulary.

*Key words:* heuristic, rule of thumb, algorithm, problem solving, artificial intelligence, cognitive science, philosophical implications of AI, history of AI.

---

Depuis le milieu des années cinquante jusqu'à nos jours, la notion d'heuristique a joué un rôle crucial dans les descriptions que faisaient les chercheurs en IA de leurs travaux. Ce qui n'a généralement pas été relevé, c'est que les différents chercheurs ont souvent appliqué ce terme à des aspects assez différents de leurs programmes. Ce qu'un chercheur particulier appellerait une heuristique sera nommé différemment par d'autres. Ceci, parce que beaucoup d'heuristiques incorporent une variété d'aspects différents, et les divers chercheurs n'ont pas mis l'accent sur les mêmes aspects comme étant essentiels à la formulation d'une heuristique. Cet article se tient à l'écart de tout programme particulier de recherche et examine la question de savoir quels éléments, historiquement, ont été considérés comme centraux dans la notion d'heuristique et lesquels sont en conflit. Après avoir analysé les définitions antérieures et examiné les usages courants du terme, nous proposons une définition synthétique. Notre espoir est que, disposant d'un compte-rendu plus complet sur la notion d'heuristique, les chercheurs pourront bénéficier plus pleinement des approches de leurs collègues, même si celles-ci sont formulées dans un vocabulaire quelque peu différent.

*Mots clés:* heuristique, règle ad hoc, algorithme, résolution de problème, intelligence artificielle, science cognitive, implications philosophiques de l'IA, histoire de l'IA.

[Traduit par la revue]

Comput. Intell. **1**, 47–58 (1985)

## Introduction

That the concept of a heuristic has been, and continues to be, central in AI is too well known to require documentation. Less well known, perhaps, is the fact that this central concept has always had a number of distinct "dimensions of meaning" associated with it, and throughout the history of its use in AI different theorists have emphasized different ones of these "dimensions," so that what was once thought to be a clear instance of a heuristic would later be seen as only a marginal instance. In this paper we canvas the history of this concept in AI with an eye to teasing out these "dimensions of meaning." We present four dimensions: uncertainty of outcome, basis in incomplete knowledge, improvement of performance, and guidance of decision making. A thorough investigation is then made of each dimension to see exactly where the concept of heuristic fits along each dimension. Finally, with the entire analysis behind us, we conclude by providing our own definition of 'heuristic', one which we believe accurately summarizes what the majority of AI theorists mean by the term.

Why is a solid definition needed, it might be asked. Haven't we been getting along fine without one? It is true that very few of the research efforts that employ heuristics actually offer any detailed analysis of the concept. Individual heuristics are discovered, tested, and modified in conjunction with a particular task or subtask, but the concept of a heuristic itself is rarely reflected upon. As a rule, definition by example is the primary method of introducing the concept to a newcomer. Even such noteworthy works as Lenat's (1982, 1983*a*,*b*) are not a careful exposition of the relevant concepts, but are rather a variegated mixture of hypothetical key ideas and speculations presented as an account of his (and his colleagues') latest reflections on the subject. This is no criticism: obviously such work is of the utmost value when addressing issues at the forefront of scientific research. But we think that an equally valuable task is to try to untangle the web of distinct pronouncements made about the concept *without specific reference to any ongoing research project*, both so that future researchers can find a basis for commonality in comparing their work with the apparently dissimilar work of others, and also so that newcomers to the field will be better able to comparatively judge the success of projects which employ (what their authors call) heuristics, and will also be better able to judge the extent to which any such success is genuinely due to the heuristics, as opposed to any other techniques.

## History

*heuriskein* (ancient Greek) and *heuristicus* (Latin): "to find out, discover."

*Heuretic*: The branch of Logic which treats of the art of discovery or invention. 1838 Sir W. Hamilton *Logic* App. (1866) II. 230 That which treats of these conditions of knowledge which lie in the nature, not of the thought itself, but of that which we think about

Gelernter emphasizes that the necessity of avoiding algorithmic, exhaustive search is the rationale for introducing heuristics into a problem situation. Gelernter is also one of the first to point out that heuristics work in effect by eliminating options from an impractically large set of possibilities:

> A heuristic is, in a very real sense, a filter that is interposed between the solution generator and the solution evaluator. [Feigenbaum and Feldman 1963, p. 137]

This remark is noteworthy as an example of something that is common in AI: a researcher's program or theory of problem-solving influencing his conception of heuristic. Polya and Newell *et al.* spoke of a mathematician groping for a solution, but here we have posited a formal "solution generator" and "solution evaluator." These have actual counterparts in Gelernter's computer program, but we doubt if there are any such identifiable procedural components in a mathematician's thought processes.

In Tonge's (1960) discussion of his heuristic program for minimizing the number of workers needed on an assembly line, the nonguaranteed element plays a lesser role in the definition of heuristic and the filtering element is not present. He emphasizes efficiency and effort reduction in achieving a *satisfactory* solution. His definition also shows the tendency to abstract the meaning of *heuristic* away from "process" and towards any arbitrary "device." Often the "device" is a portion of his program with an identifiable function. He also speaks of heuristics as providing "shortcuts," and as employing "simplifications," in contrast with several of the algorithmic methods that theoretically guarantee solutions. His official definition is:

> ... by heuristics we mean ... principles or devices that contribute, on the average, to reduction of search in problem-solving activity. The admonitions "draw a diagram" in geometry, "reduce everything to sines and cosines" in proving trigonometric identities, or "always take a check – it may be a mate" in chess, are all familiar heuristics.
>
> Heuristic problem-solving procedures are procedures organized around such effort-saving devices. A heuristic program is the mechanization on a digital computer of some heuristic procedure. [Feigenbaum and Feldman 1963, p. 172]

Minsky (1961*a*) was one of the first to use *heuristic* in the context of "search" through a large "problem space." Speaking of chess, which Shannon had estimated to have $10^{120}$ paths through its game tree, he says (Feigenbaum and Feldman 1963, p. 408) "we need to find techniques through which the results of *incomplete analysis* can be used to make the search more efficient." His official definition, like Tonge's, emphasizes efficiency rather than an oppposition to algorithms:

> The adjective "heuristic," as used here and widely in the literature, means related to improving problem-solving performance; as a noun it is also used in regard to any method or trick used to improve the efficiency of a problem-solving system. A "heuristic program" to be considered successful, must work well on a variety of problems, and may often be excused if it fails on some. We often find it worthwhile to introduce a heuristic method which happens to cause occasional failures, if there is an over-all improvement in performance. But imperfect methods are not necessarily heuristic nor vice versa. Hence, "heuristic" should not be regarded as opposite to "foolproof"; this has caused some confusion in the literature. [Feigenbaum and Feldman 1963, p. 408]

Here Minsky is saying that a foolproof algorithm could be called a heuristic, provided it shows an improvement in efficiency over some other method. He is also emphasizing, like Polya, that a heuristic must be applicable to more than just a restricted set of problems. An effort-saving method that worked on only one problem would be more properly called a specific tool rather than a heuristic method.

Slagle's (1963) description of his program to solve integration problems in mathematics uses *heuristic* primarily to stand for any of a class of rules that transform a problem into one or more subproblems. Examples of such rules would be "try integration by parts" and "try a trigonometric substitution." He distinguishes algorithms from heuristic transformations, the latter being defined as follows:

> A transformation of a goal is called heuristic when, even though it is applicable and plausible, there is a significant risk that it is not the appropriate next step. [Feigenbaum and Feldman 1963, p. 197]

This particular usage, however, disagrees with his formal definition where the heuristic actually makes the decision as opposed to being a passive rule chosen by the executive:

> Although many authors have given many definitions, in this discussion a heuristic method (or simply a heuristic) is a method which helps in discovering a problem's solution by making plausible but fallible guesses as to what is the best thing to do next. [Feigenbaum and Feldman 1963, p. 192]

We will return to discuss this kind of confusion later.

Finally we come to the definition of Feigenbaum and Feldman (1963), the editors of *Computers and Thought:*

> A heuristic (heuristic rule, heuristic method) is a rule of thumb, strategy, trick, simplification, or any other kind of device which drastically limits search for solutions in large problem spaces. Heuristics do not guarantee optimal solutions; in fact, they do not guarantee any solution at all; all that can be said for a useful heuristic is that it offers solutions which are good enough most of the time. [Feigenbaum and Feldman 1963, p. 6]

This definition combines many of the features present in the other definitions we have discussed. It contains the elements of lack of guarantee, of arbitrary device, of effort reduction, of eliminating options, and of satisfactory solution. Following their definition Feigenbaum and Feldman also bring up a new element, that of domain dependence. Some heuristics are very special purpose and domain specific, like chess heuristics, whereas others, like "means-ends analysis" and "planning," apply to a much broader class of problem domains.

This brings us to the end of what we might call "the early AI period." As we see it, in this period researchers were groping with the concept of a heuristic and felt compelled to provide their readers with definitions of the term. After this period, researchers no longer felt that it was such a novel concept that it required any special explanation or justification, except perhaps when talking to lay audiences. One can already see, just from the examples cited, how the concept of heuristic was transformed since its original introduction to the AI community via Polya. Polya used 'heuristic' primarily in the context of logic or psychology of discovery. His heuristic methods were to apply helpful reasoning processes like asking certain questions, drawing diagrams, guessing, looking at the problem from a different perspective, etc. Somehow these methods direct the mind towards seeing a solution. 'Discovery' is used here very

But there are some novel interpretations emerging, which appear to be "second generation ideas" on what heuristics really are. Unfortunately, these are never very clearly defined and explained. For example, Hofstadter (1979) has a view of heuristic as "compressed experience":

> Of course, rules for the formulation of chess plans will necessarily involve heuristics which are, in some sense, "flattened" versions of looking ahead. That is, the equivalent of many games' experience of looking ahead is "squeezed" into another form which ostensibly doesn't involve looking ahead. In some sense this is a game of words. But if the "flattened" knowledge gives answers more efficiently than the actual look-ahead—even if it occasionally misleads—then something has been gained. [p. 604]

Another example of a cursorily presented novel interpretation comes from Albus (1981):

> Procedures for deciding which search strategies and which evaluation functions to apply in which situations are called heuristics. Heuristics are essentially a set of rules that reside one hierarchical level above the move selection and evaluation functions of the search procedure. A heuristic is a strategy for selecting rules, i.e., a higher level rule for selecting lower level rules. [p. 284; see also pp. 222, 223]

And finally, Lenat (1982) appears to have a view of heuristics similar to Hofstadter's:

> Heuristics are compiled hindsight: they are nuggets of wisdom which, if only we'd had them sooner, would have led us to our present state much faster. This means that some of the blind alleys we pursued would have been avoided, and some of the powerful discoveries would have been made sooner. [p. 223]

It is our belief that definitions like these last three are not sufficiently popular in the general AI community to warrant being included as part of a comprehensive definition of 'heuristic'. This situation may of course change with time.

So far we have just reviewed some of the assorted definitions of heuristic that have appeared over the past 40 years. We have seen that different researchers have emphasized different properties as being relevant to whether a heuristic is being employed, and we have seen a shift in emphasis in the concept. We would like now to distinguish more carefully the different (but interrelated) "dimensions of meaning" that the concept embodies. We can distinguish four dimensions along which various researchers have judged whether a process is a heuristic: uncertainty of outcome, basis in incomplete knowledge, improvement of performance, and guidance of decision making.

### The role of uncertainty: heuristics vs. algorithms

We have seen how, in many of the definitions, 'heuristic' has been opposed to terms like 'algorithmic', 'guaranteed', and 'complete'. We will argue in this section that the central idea underpinning these definitions is that heuristics exist in a context of *subjective uncertainty as to the success of their application*. We will explain in what respects those, like Minsky, who think that heuristics are perfectly compatible with algorithms are correct, even though there is a genuine conflict between these two notions. In illustration, we shall give the sense in which even the brute force British Museum Algorithm is a heuristic.

Central to this key property of uncertainty and, as we saw, present at the very earliest adoptions of the concept of heuristic by AI is the notion of *algorithm*. 'Algorithm' has many meanings, although it is doubtful that the ambiguity has caused any of the disagreements over definition. If we define algorithm as merely "a set of [formally defined and uniquely interpreted] rules which tell us, moment to moment, precisely how to behave" (Minsky 1968, p. 106), then any procedure for making decisions is algorithmic, and hence all heuristics implemented on computer, or otherwise strictly formulated, are algorithmic. We use "procedure-algorithm" to mean this type of algorithm. However, when 'heuristic' has been considered opposed to 'algorithm', 'algorithm' has always had a much stronger sense which includes an element of guarantee about finding a solution. Korfhage (1976, p. 48), following normal usage, characterizes an algorithm as follows:

1. Application of the algorithm to a particular input set or problem description results in a finite sequence of actions.
2. The sequence of actions has a unique initial action.
3. Each action in the sequence has a unique successor.
4. The sequence terminates with either a solution to the problem, or a statement that the problem is insoluble.

If the last restriction is too strong we may define 'semi-algorithm' as follows: "a method that will halt in a finite number of steps if the problem posed has a solution, but will not necessarily halt if there is no solution." For some problems there is always a solution, e.g., adding two integers, and so this distinction does not apply. We call such algorithms "simple-algorithms."

'Heuristic' has often been opposed to some such notion of algorithm, although not universally by all authors. Newell *et al.* (Feigenbaum and Feldman 1963, p. 114) opposed it to semi-algorithms, while Tonge (Feigenbaum and Feldman 1963, p. 172) and Slagle (1963, p. 194) opposed it to simple-algorithms. Feigenbaum and Feldman (1963, p. 6) implied a contrast with simple-algorithms and also seemed to say that there is no issue here. We have seen that Minsky, Raphael, and Sampson denied any opposition with algorithms and that both Nilsson (1980, p. 72) and Jackson (1974, p. 95) denied that heuristics need sacrifice a guarantee of finding a solution (although neither said anything about starting out with a guarantee, or what happens if one should find that the putative heuristic *does* guarantee finding a solution). Boden (1977, pp. 347, 348) argued on the one hand that there is no opposition with simple-algorithm or with semi-algorithm, but on the other hand there is a contrast insofar as heuristic programs postpone decision making, whereas algorithms require all decisions be precisely specified beforehand.

Given this mix of conflicting claims one could simply do as Barr and Feigenbaum (1981, pp. 28, 29) do, namely, state that *heuristic* is an ambiguous term and that to keep things clear one will be using such and such a definition. This response is inadequate, however, because it ignores several reasons for believing that there *is* one correct definition. These are the single origin of the term in the AI literature, i.e., the work of Polya; the fact that AI authors have placed so much theoretical weight on this specific term; and the fact that AI authors have not given their definitions with the air of "for convenience I use the term ..." but with the impression that they have captured what is really important and common to this branch of research, namely, the use of rules that save effort, or provide satisfactory solutions, or lack a guarantee, or what have you. Therefore, we believe a proper analysis of heuristic must result in one definition, and this one definition must show adequate appreciation for all the ideas which have been linked to it, and it must be able to explain any incompatibility among such ideas.

improve problem-solving performance. Since so many real world problems are of this form, it is no wonder heuristics have become so popular and are so worth studying.

Lenat (1982, p. 222) has remarked similarly on the domain of heuristic applicability:

> At an earlier stage [of knowing a domain], there may have been too little known to express very many heuristics; much later, the environment may be well enough understood to be algorithmized; in between, heuristic search is a useful paradigm. Predicting eclipses has passed into this final stage of algorithmization; medical diagnosis is in the middle stage where heuristics are useful; building programs to search for new representations of knowledge is still pre-heuristic.

Thus we have a spectrum of confidence levels in decision making. At one extreme are efficient algorithms and other decision processes which we believe are optimal (whether or not they guarantee a solution), and at the other extreme we have the most inefficient algorithms and other unprofitable processes in which we place little confidence. *Heuristics fall in between: they are plausible without being certain.* The placement of a particular process along this spectrum is, however, relative to our perception of the extremes. For example, Newell *et al.* (Feigenbaum and Feldman 1963, p. 116) originally spoke of their British Museum algorithm as producing such "simple and cheap" expressions that it could not be heuristic, whereas they (Newell and Simon 1972, pp. 120, 121) later call it heuristic because its generator is only apparently "blind-trial-and-error", since by generating only theorems it is so much more selective than one that generates *all* well-formed formulas.

As a defining ingredient in heuristics, partial insight offers more than just confidence. Insight is the core of a heuristic's intelligence, its reason for being. A particular heuristic is represented by its particular insight; without a genuine grasp of some aspect of the problem a device must perforce contribute nothing to problem solving. It could only masquerade as a heuristic until its luck wore out. It is with this dimension of meaning of *heuristic* in mind that Polya, Gelernter, Slagle, and Jackson offered their definitions. Here are two simple examples of the sense in which heuristics might represent partial insight into a problem domain.

In symbolic logic we know that $\neg\neg A$ is equivalent to $A$. This is a piece of knowledge about how logic formulas relate to one another. We also know that theorem provers bog down with more and more complex formulas, and that a big part of theorem proving is matching for similar patterns in other formulas. We can employ all these insights to construct a heuristic that simplifies pattern matching: "under such and such circumstances eliminate excess negations." Other heuristics could make use of other equivalences (e.g., those expressed by DeMorgan's rules) to recommend the conversion of all formulas to some type of normal form.

In chess, the piece of knowledge that at one point in play one's bishop can in two moves go to more possible squares than one's rook, might allow one to generate the temporary heuristic "use the bishop on this turn."

As can be seen from these simple examples, the possibilities for generating heuristics are endless. One discovers something about the problem and constructs a device to make use of this insight. The rule will thereby be plausible, and if one does not know enough about the problem to tell if the device is optimal then it can also be a heuristic.

When we analyze insight we see that it comes in a variety of forms. There is a simple insight that can be expressed in simple terms. The example from symbolic logic, that $\neg\neg A$ is equivalent to $A$, is a simple insight since we can describe it simply. Then there are insights that are not easily expressible, but are nonetheless present. For example, Samuel's (Feigenbaum and Feldman 1963, pp. 71–105) checker-playing program employed a polynomial evaluation function that included features like "center control," "mobility," "number of forceable exchanges," etc. This 16-element polynomial represents an insight into checkers, but how would one express it simply? For one thing the insight is highly dependent on Samuel's particular program and his test samples. One might argue that hence it is really only an insight into how to play good checkers with this particular program. We think, however, that the insight is more universal; it tells us, among other things, that as a general rule kings in the center are more powerful than we might have expected.

These two examples also show us that some insights are known prior to their heuristics while others are discovered by examining heuristics. Hence these two forms of knowledge, the aspects of the problem (factual knowledge) and how to make use of these aspects (procedural knowledge), can exist quite independently.

Some AI researchers have reflected on the abstract nature of heuristic insight. Boden (1977, p. 351), Minsky (Feigenbaum and Feldman 1963, p. 409f), and Newell *et al.* (Feigenbaum and Feldman 1963, p. 122) speak of moving from the start state to the goal state and avoiding many fruitless paths by *sensing* whether one is getting warmer or colder. A kind of negative feedback keeps one on the right track. At each point where alternatives are presented a decision is made. Only some of these decisions need to be fruitful to keep one from going too far astray. Evaluation functions fit this description well.

Another set of reflections comes from Boden (1977, pp. 341–344), Minsky (1968, p. 425ff), Pearl (1984, pp. 113–118), and Polya (1945, pp. 37–46, 180), who speak of the power of *analogies* and *models*. Analogies may be as complicated as or even more complicated than the original problem. If sufficient parallelism between the two cases exists then they allow us to transfer both the insights and the heuristics based on these, rather than be forced to rediscover these same insights and heuristics. Models are a form of analogy. They are simplified representations which allow us to focus on, or make more salient, some of the more relevant aspects of a problem. They offer a more compact representation of a problem's essentials and are thus a form of partial insight. Their simplicity may also make it easier to discover new insights. And those discovered are likely to concern the more essential aspects of the problem.

### Heuristics as performance improvers

Heuristics are often seen as improving performance, as some of our definitions above have illustrated. But how do they do it? In this section we will show what may seem obvious, but should not be, that heuristics are used to help improve the performance of a problem-solving system. In this regard they are like tools introduced to fix or enhance a system. The notion of performance improvement under consideration is that of increased efficiency, that is, receiving more benefit out for effort put in. We believe, but shall not thoroughly discuss here, that the various permutations of decreasing effort and increasing benefit explain many of the forms in which heuristics occur.

First of all, it should be clear that we would not be using heuristics in problem solving, in discovering solutions, guiding

> If concept $G$ is now very interesting, and $G$ was created as a generalization of some earlier concept $C$, give extra consideration to generalizing $G$, and to generalizing $C$ in other ways. [Lenat and Harris 1978, p. 43]

He designed his system to facilitate the addition of new rules and he hoped to add more in time (cf. Lenat 1982, 1983*a*,*b*). Again each new rule is seen as potentially improving the discovery abilities of the program. Lenat also experimented with AM; he appears to have added rules in a try-and-test fashion as various ideas for enhancing AM's performance occurred to him (Lenat 1982, pp. 205–207).

We have just seen how performance improvement is a popular activity in AI and how heuristics are associated with this activity by patching impractical theories or by being incrementally added to a general problem-solving schema. These are ways of expressing the basic benefit-greater-than-cost intuition and show that a number of properties that are ascribed to heuristics can be derived from it. For instance, Gelernter's (Feigenbaum and Feldman 1963, p. 137) idea of heuristics as "sufficiently nonporous *filters*" and the popular notion of "selective *pruning* of decision/game trees" both focus on our desire to eliminate from consideration more useless items than valuable ones. We also have Tonge's (1960) "shortcuts," "simplifications," and "adequate solutions." These are attempts to keep the costs (of having to perform detailed analyses) down but the benefits (quality of solutions) sufficiently high. Abstractions or generalizations of decision devices, insofar as they reduce the number of detailed devices that need to be memorized and also reduce the need to consider each one every time a decision is required (but do not grossly mishandle too many of the exceptional cases), are also candidates for being heuristic. Or, more generally, any area where we can trade off resource utilization for a slight loss of number of solvable problems, or of quality of solutions, is an area open to performance improvement by heuristic methods. Conversely, if by whatever means we can marginally increase resource utilization (time, memory, tool, etc.) costs, but recoup a dramatic increase in solvable problems, or a significant increase in quality of some solutions, then this too can qualify as heuristic performance improvement. Expert systems are good sources for finding such effort-increasing heuristics since we typically add rules to them, which implies occupying more space and spending more time considering extra rules. A consequence of the existence of this last class of heuristics is that all those definitions of heuristic that use the phrases "effort reduction" or "search reduction" are misleading. "Performance improvement" is the more accurate phrase since it covers all the cases of relative cost–benefit improvement.

## Heuristics as decision guiders

Heuristics have been variously presented in the form of proverbs, maxims, hints, suggestions, advice, principles, rules of thumb, criteria, production rules, programs, procedures, methods, strategies, simplifications, option "filters," goal transformers, and no doubt there are others. (See the History section, given earlier, for an example of each.) What is common to all these forms? In this last section on properties we hope to show that heuristics always try to help the problem solver by *guiding his decisions* during the course of moving from initial to solution state. Since this is not really a contentious point with anyone, we will not belabour it. Nonetheless, because it is a key property it deserves a clear statement. In the end we will discover a few new things about decision guidance; in particular we hope to clear up the issue of whether heuristics can be passive options presented to an executive decision maker or whether they must be the higher-order decision rules guiding the search for a solution.

To show that decision guiding is the primary function of heuristics, we first show that the element of choice is always present when heuristics are discussed, and that heuristics as a group do not consistently influence any other element of a problem solver or his situation. For example, they are not devices that consistently influence memory, clarity of vision, creativity, thoroughness, or any other feature of problem solving. To phrase it differently, we claim that the use of 'heuristic' always presumes the existence of a decision mechanism and that the heuristic's effect is to lead this mechanism down one path as opposed to another. The influence may be direct, i.e., the heuristic actually decides where to go. For example, evaluation functions are direct. Or the influence may be indirect, i.e., the heuristic simply changes some aspect of the problem situation. For example, "eliminate complex theorems from the subproblem list." There is no sharp line dividing these two types of influence.

By way of illustration, we bring forth a representative sample of usages and definitions to support the claim about the universality of decision guidance. We can start with Polya, whose usage we recall was rather different from what is prevalent in AI today. For Polya, any behavioral method considered useful while problem solving could be a heuristic method. This includes asking oneself certain key questions, drawing a diagram, or trying to rephrase a problem. Since Polya did not use the paradigm of search when describing mathematical problem solving, these behavioral methods need not affect any decision making. They could influence some unconscious processes which suddenly inspire the solver to see a solution. Nevertheless, Polya only speaks of his methods as being chosen by a solver. The student should try this, think of that, ask himself this question, etc.

In the early AI period, the paradigm of heuristic use is one of guiding search through a problem space. This applies to every author covered above. Like Polya, Newell *et al.* officially leave open the possibility of heuristics being arbitrary useful processes applied during problem solving. Yet in fact they solely use them to influence the order of development of the solution path along the subproblem tree. The value of the heuristics is explained by their effect on movement through the subproblem tree, and all their heuristics are clearly decision guiding. The four primary methods in the Logic Theorist directly choose some of the paths to be followed, while the "similarity test" acts as a filter, screening some theorems prior to matching and hence indirectly guiding the course of search.

Gelernter employed a similar tree search paradigm and uses his heuristics to filter out less promising decision options. Tonge used heuristics to simplify wherever possible the entire pattern of activity used to balance assembly lines. Slagle uses the Logic Theorist framework where heuristics both decide what problem transformations to apply next, as well as transform the problems themselves. Minsky introduces heuristics in a context of search where they guide the solver gradually to a solution. (He gives "hill climbing" as a typical example.) Feigenbaum and Feldman mention state-space search reduction in their definition and give assorted rules of thumb as examples. For these the solver is portrayed as trying one thing rather than another and is thereby led down a different problem-solving path.

prioritizing projects on the "agenda" of things to do next, without individually choosing what exactly is done next. Indeed, Lenat's executive is very simple and runs without any heuristics at all. Likewise, all declarative (as opposed to procedural) expressions of heuristic knowledge about a domain, such as MYCIN's "if evidence *E* then assert *A* with confidence factor CF," are heuristics that do not choose what to do next. In the case of MYCIN, a modified depth-first algorithm makes these choices (Barr and Feigenbaum 1982, pp. 187–191).

## Conclusion

We set out to define *heuristic* against a historical backdrop of conflicting definitions. What emerged from our survey of definitions was that *heuristic* could refer to any device used in problem solving, be it a program, a data structure, a proverb, a strategy, or a piece of knowledge. But not just any such device. There had to be an element of "rule of thumbishness" about the device; it had to be useful but need not guarantee success. This lack of guarantee, however, applies to the entire, *real* practical picture of supplying a solution. A heuristic device *can* guarantee supplying a solution, but if it is also provably the optimal device for arriving at a solution, then it is not a heuristic. As for its utility, this is derived from the heuristic's having captured some fact, some insight, about the problem domain. All in all, therefore, heuristics fit on a spectrum of devices between those that are random and uninspired and those that are applied automatically because they never fail to please, or if they do fail then we resign ourselves to this because we have a proof that there can be no better device.

Although these two properties should be sufficient to eliminate the majority of nonheuristic devices, most AIers use *heuristic* more restrictively still. They reserve the term for just those devices they have added to their experimental system in hopes of improving its performance. Although we suspect they would relinquish this property upon a little reflection, this restricted usage is nonetheless prevalent. For instance, it is to be found in the majority of definitions given by AIers themselves. Therefore we must admit this property if we are to give *an AIer's definition* of 'heuristic'. As for what "performance improvement" means, we found that, contrary to many authors, it did not mean search or effort reduction, that this was only half of the equation, the other half being the possibility of improvement in solution quality in exchange for a modest increase in search effort.

With the addition of performance improvement we have all the properties needed to restrict the set of problem-solving devices to those that AIers call heuristic. Technically this would suffice as a definition. Yet when we examine all the remarks made about heuristics in the literature we find that there is a popular theme not covered by fallibility, plausibility, and performance improvement, namely, the function of heuristics in problem solving. They work by guiding search, suggesting behavior, making decisions, or transforming the problem so that different courses of action are open. These properties are reflected in the choice of words used to make heuristics concrete: rules, advice, procedures, filters, etc. We suggested that the search guidance characterization is so popular because of the popularity in AI of the state-space framework for describing problems. Of course, the state-space framework is so popular in AI because, as scientists, AIers can benefit by analyzing a problem's essentials into paths, option nodes, states, etc. Heuristics in this framework naturally affect the decisions as to which paths to follow.

Having described all this we concluded the discussion of decision guidance by establishing that heuristics could be involved in direct active decision making, or merely passively as options to execute, and that therefore some authors were incorrect in thinking that all heuristics chose what course problem solving would follow next.

Concisely put, *a heuristic in AI is any device, be it a program, rule, piece of knowledge, etc., which one is not entirely confident will be useful in providing a* ***practical*** *solution, but which one has reason to believe will be useful, and which is added to a problem-solving system in expectation that on average the performance will improve.*


ALBUS, J. S. 1981. Brains, behavior, and robotics. Byte Publications Inc., Peterborough, NH.

BARR, A., AND FEIGENBAUM, E. A. (*Editors*). 1981. The handbook of artificial intelligence. Vol. 1. William Kaufmann, Inc., Los Altos, CA.

——— 1982. The handbook of artificial intelligence. Vol. 2. William Kaufmann, Inc., Los Altos, CA.

BLEDSOE, W. W. 1971. Splitting and reduction heuristics in automatic theorem proving. Artificial Intelligence, **2**, pp. 55–77.

BODEN, M. A. 1977. Artificial intelligence and natural man. Basic Books, Inc., New York.

CHANG, C.-L., and LEE, R. C. 1973. Symbolic logic and mechanical theorem proving. Academic Press, Inc., New York.

COHEN, P. R., and FEIGENBAUM, E. A. (*Editors*). 1982. The handbook of artificial intelligence. Vol. 3. William Kaufmann, Inc., Los Altos, CA.

ERNST, G. W., and NEWELL, A. 1969. GPS: A case study in generality and problem solving. Academic Press, New York.

FEIGENBAUM, E. A., and FELDMAN, J. (*Editors*). 1963. Computers and thought. McGraw-Hill Inc., New York.

FINDLER, N. V. 1976. Heuristics. *In* Encyclopedia of computer science. *Edited by* A. Ralston. Van Nostrand Reinhold & Co., New York, pp. 606, 607.

GELERNTER, H. 1959. Realization of a geometry-theorem proving machine. *In* Proceedings of an International Conference on Information Processing, Unesco House, Paris, pp. 273–282. (Reprinted in Feigenbaum and Feldman (1963), pp. 134–152.)

GRONER, R., GRONER, M., and BISCHOOF, W. F. (*Editors*). 1983. Methods of heuristics. Lawrence Erlbaum, Hillsdale, NJ.

HOFSTADTER, D. R. 1979. Gödel, Escher, Bach. Basic Books, New York.

HUNT, E. B. 1975. Artificial intelligence. Academic Press, Inc., New York.

JACKSON, P. C., JR. 1974. Introduction to artificial intelligence. Petrocelli Books, New York.

KORFHAGE, R. R. 1976. Algorithm. *In* Encyclopedia of computer science. *Edited by* A. Ralston. Van Nostrand Reinhold & Co., New York, pp. 47–50.

LENAT, D. 1977. The ubiquity of discovery. Artificial Intelligence, **9**, pp. 257–287.

——— 1979. On automated scientific theory formation: A case study using the AM program. *In* Machine intelligence. Vol. 9. *Edited by* J. E. Hayes, Donald Michie, and L. I. Mikulich. Ellis Horwood Ltd., West Sussex, England, pp. 251–283.

——— 1982. The nature of heuristics. Artificial Intelligence, **19**, pp. 189–249.

——— 1983*a*. Theory formation by heuristic search. The nature of heuristics II: Background and examples. Artificial Intelligence, **21**, pp. 31–59.

——— 1983*b*. EURISKO: A program that learns new heuristics and domain concepts. The nature of heuristics III: Program design and results. Artificial Intelligence, **21**, pp. 61–98.

LENAT, D., and HARRIS, G. 1978. Designing a rule system that searches for scientific discoveries. *In* Pattern directed inference