Discovery Tools for Science Apps

Speed and smarts propel new tools for scientific applications.

Raúl E. Valdés-Pérez

One good characterization of science is that it is the systematic study of some phenomena. Under this liberal view, discovery in science is not fundamentally different from discovery in, say, business. As argued by [1], science's reputation for finding reliable knowledge owes more to the perseverance of its practitioners and to its organizational checks and balances than to the use of any special method. Thus, insights gained from developing discovery tools

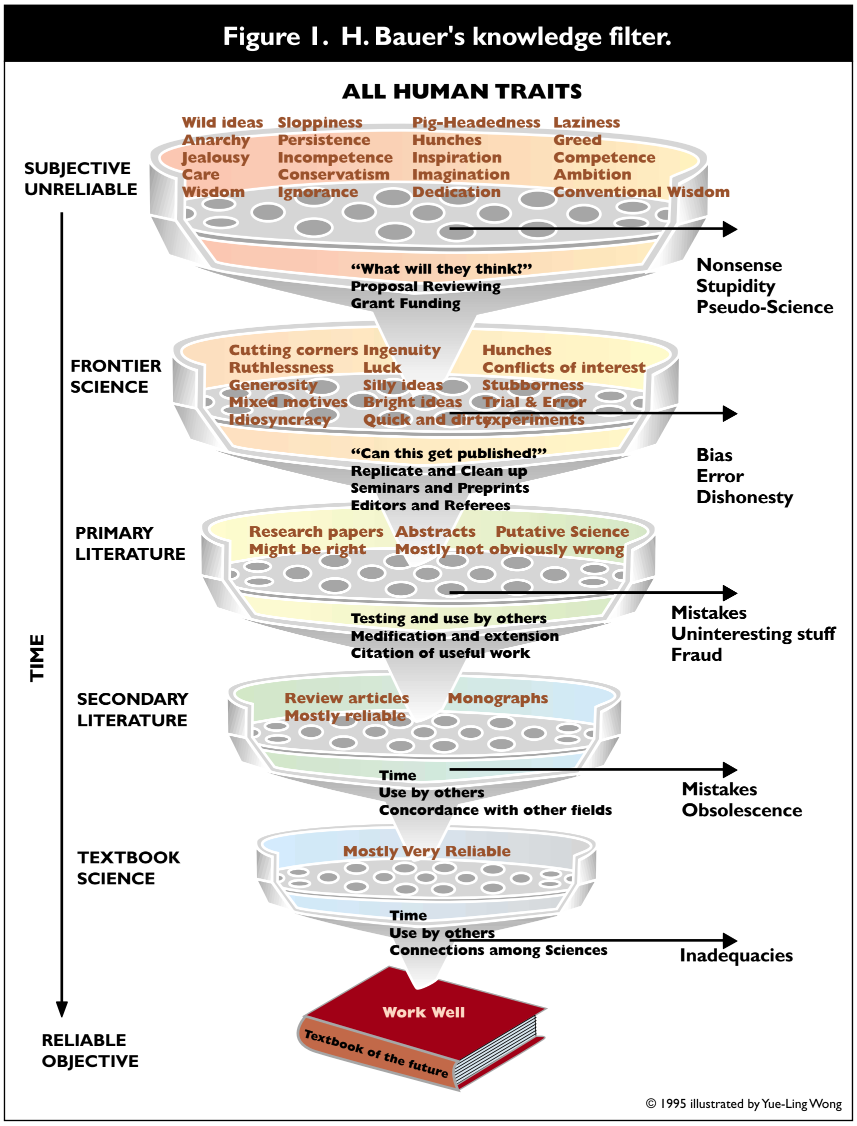
ANGELA WILCZYNSKI/WIZZLE.SIMPLENET.COM/FRACTALS/FRACTALINTRO.HTM

in science, which has such refined criteria of success, can guide knowledge discovery as a whole. This article aims to present some basic concepts about knowledge discovery, convey the state of the art by reporting on success stories, and suggest lessons that are relevant to application areas besides science.

To understand what computers can do in science, it helps to review some basic concepts from AI/cognitive science and the sociology and philosophy of science.

Heuristic search in combinatorial spaces. Since discovery-oriented workers (discoverers, for short) undergo a long apprenticeship to become experts in their field, one might conclude that a discoverer's reasoning process resembles expert reasoning based on recognition, for example, the chess grandmaster who chooses a strong move after a glance at the board, or the physician who quickly selects a likely diagnosis based on the first few symptoms. If discovery is like such expert reasoning, then pattern-recognition approaches such as neural nets or discrimination trees (that match a current problem against previous experiences) would be the methods of choice for computerizing most discovery tasks. However, discovery by definition happens at the frontiers of knowledge where nobody is an expert, so the better analogy is to chess beginners or to physicians-in-training, whose reasoning is based partly on trial-and-error, or heuristic search, as it is known in AI.

Figure 1. H. Bauer's knowledge filter.



© 1995 illustrated by Yue-Ling Wong

The basic idea of heuristic search is that solving many reasoning tasks can well be viewed as a search within a large combinatorial space. We say the space is combinatorial because at each choice point there are many choices, and these effects can accrue to create notoriously large spaces. In many practical tasks, people cannot search the entire space in their heads, so we must focus on some subspaces in preference to others. The knowledge that permits such focusing is called "heuristics." Heuristics can be absolutely reliable or they can be rules-of-thumb; the key point is they direct one's scarce resources toward more promising avenues. In chess, a beginner's heuristic is "consider moves that check the opposing king first."

Data-driven and knowledge-driven approaches. A program or approach is knowledge-driven if it uses relatively general knowledge (including knowledge of how to search combinatorial spaces) as a main source of power. A data-driven program instead relies on specific measurements, statistics, or examples. Most approaches are some combination of both.

A same task might be approached in either way: consider a chess program that bases its next move either on matching against a large database of examples of previous positions and good moves, or on general heuristics about strategy and tactics combined with a search over possible successive moves. The input to a data-driven program can be megabytes of data, whereas the input to a knowledge-driven program could be a mere few lines. Depending on the specifics of the problem, either might take longer to run.

Enhancing human discovery. Science is quite successful in technical achievements like putting a man on the moon, curing disease, creating reliable miniature circuits, and so on. This raises the question of what knowledge discovery can hope to achieve. Is the goal to make expertise more widely available, as was promised by the expert systems movement of the 1980s? In reality, the goal is to augment the capabilities of even the best experts. The accompanying figure (reproduced from www.chem.vt.edu/ethics/hbauer/hbauer-fig2.GIF) shows why: reliable textbook science accounts for the technical achievements, but cutting-edge, frontier science—the stuff of primary journal articles—is notoriously unreliable for the listed reasons [1] and because of the well-known bounded rationality of human beings, which limits our ability to reliably infer correct models, notice subtle patterns, foresee the implications of assumptions, and so on. Scientists with computers can do better frontier science.

The goals of scientific discovery. The goal of discovery is to find knowledge that is novel, interesting, plausible, and understandable [6]. Thus, a discovery program that too often leads to familiar, dull, wrong, or obscure knowledge won't be used. These four dimensions are separable: for example, the number of blades of grass viewable from my office window may be a novel, plausible, and understandable fact, but it fails to be interesting. It helps to analyze how a specific program addresses each of the dimensions, since this exercise can help pinpoint the reasons for user dissatisfaction and identify scope for improvement.

Discovery Programs

Three examples of successful systems—taken from medicine, mathematics, and chemistry—are described here.

Arrowsmith. Literature-based discovery refers to using documents—a special case of data—as a source of power. The Arrowsmith program developed at the University of Chicago (kiwi.uchicago.edu) makes conjectures about possible treatments or causes of medical diseases using the Medline collection of titles and abstracts from the medical literature. Given a target disease or other physiological state C, the program searches for two associations BC and AB where A is typically a dietary factor, drug, or other possible intervention, which suggests that A may cause or alleviate C through the intermediary B. For example, the user may pose a C which is migraine, and the program may come up with A=magnesium (a light metal which is essential to the human diet) and B=spreading depression.

After subsequent human examination of the literature, which reports that "magnesium can inhibit spreading depression in the cortex, and spreading depression may be implicated in migraine attacks," there is the plausible suggestion that magnesium could be a treatment for migraine.

Altogether Swanson has reported eight examples of successful matching of complementary but disjoint literatures, four of these in collaboration with Neil R. Smalheiser, a neurobiologist. The best confirmed example to date is the connection between magnesium deficiency and migraine headaches [5]. Subsequent to that publication, more than 12 laboratories have independently reported direct clinical or laboratory tests that provided supportive evidence. More recently, Arrowsmith was used to illuminate an already noticed and reported association between estrogen supplementation and Alzheimer's disease by pointing out a possible indirect mechanism of such an association.

The method's conjectures tend to be novel because a citation analysis verifies that no or few Medline articles cite both subliteratures responsible for the associations AB and BC. There are heuristics, similar to stoplists in language processing, that filter out overly broad (hence, uninformative) words like "hormone" or "pressure." The conjectures are plausible because they exploit the frequent transitivity of relations like causality. Finally, the conjectures are understandable because they are short statements like A may be a treatment for C, which suggests, for example, obvious clinical tests.

Graffiti. The Graffiti program developed at the University of Houston makes mathematical conjectures in such domains as graph theory and geometry (see math.uh.edu/~siemion).

Graffiti has motivated many graph theoreticians, including its designer, to try to refute or prove the generated conjectures which are broadcast on an email list. Many of the program's conjectures have been proven (by mathematicians) and published as regular mathematical contributions. Recent applications of Graffiti to chemistry have exploited the fact that molecules can be represented as graphs.

The program keeps a database of previous conjectures so that when the program is run it does not repeat itself and instead will tend to produce novel conjectures. The program's *echo heuristic* tends to preserve only interesting conjectures by postponing consideration of a new conjecture if it seems to be implied by a previous conjecture that has not been refuted, which is therefore more interesting because it is more general. Every conjecture is tested against a file of qualitatively different graphs and thus becomes plausible if no counterexamples are found. Finally, the conjectures are understandable because they are conventional statements of the form: *a short sum of graph properties is ≤ another short sum of graph properties,* which tend to be easier to prove than more complicated formulations.

Mechem. The Mechem program developed at Carnegie-Mellon University (in recent collaboration with A.V. Zeigarnik in Russia) finds explanatory hypotheses in chemistry (www.cs.cmu.edu/~sci-disc describes this and other discovery projects). That is, given the chemicals that start a reaction and which are formed by it, as well as prior background knowledge expressed to the program as constraints, the task is to find all the simplest plausible hypotheses (reaction mechanisms) that can explain how the products are formed.

The program's mechanisms tend to contain novelty because the pieces (elementary reactions and chemical substances) that make up a hypothesis are not drawn from any stored catalogue of common reactions; rather, they are generated from basic principles using algorithms minimally slanted toward particular solutions. The mechanisms are often interesting because they are the simplest, that is, the program reports mechanisms that contain fewest intermediate substances and steps. The mechanisms are understandable because the space being searched is taken directly from chemistry. Finally, the output is plausible because the user articulates any objections, via a graphical interface that allows for well over 100 kinds of constraints, and runs the program again with augmented input. This interaction repeats until no further problems remain, at which point all remaining hypotheses are deemed plausible.

Other programs have enhanced discovery processes in science, such as [4, 6]. Some use combinatorial search as their basic approach, whereas others use more specialized methods that can exploit mathematical properties of the subject matter, such as strings in genomics.

The aim here is not to provide a survey, but to state key concepts and illustrate them with a few programs that have led to published findings.

Lessons

A general procedure for (partially) automating many discovery tasks can be based on the following questions. What is a specific example of a discovery when the task is done humanly (if it *is* done humanly)? This question asks for specific models, patterns, conjectures, and so forth, which then feed the next question. From what larger conceivable set is this specific example drawn? In other words, what is the space within which solutions are, or should be, sought? Are some discoveries more presentable or preferable than others? One might consider that some discoveries are too complex to merit priority. A preference for simplicity or conciseness (shorter reaction mechanisms), or more concise mathematical conjectures, often makes sense.

What is the starting point (for example, data) for the task? Data-driven tasks are easier to automate. Sometimes the task is not initiated by data but by a problem. Mechem and Graffiti both have this flavor. Is background knowledge necessary for competent performance? If so, then it must be accommodated somehow, for example, by involving the user in an interactive collaboration with the program. In Mechem, building on existing background knowledge is absolutely critical for a competent program.

How can one design an algorithm that starts with the available data, generates solutions starting from the more presentable, and respects the available background knowledge? This is a key step that requires some knowledge of algorithm design, of the task from the user's viewpoint, and of knowledge engineering. Of course, it must all be turned into software.

Patterns of use/computer collaboration. Programs must be adopted by users, and this presents its own set of issues. In my experience, some general ways to improve the user's chances of finding novel, interesting, plausible, and understandable knowledge [6] are:

- Search a combinatorial space comprehensively. Since people cannot do this without computers, novelty will often turn up, often as alternatives to solutions that people mistakenly believe to be unique or best.
- Report the simplest (for example, most concise) solutions first. Simplicity correlates highly with interestingness.
- Select, or design from scratch, a tool that searches a space whose elements are highly understandable to the users.
- If the task is knowledge-driven, enable users to

input their relevant knowledge interactively and require the program to respect that input, by not reporting solutions that conflict with it. If the task is data-driven, then use abundant data; if the data are scarce, then use permutation tests to improve your confidence in the plausibility of the results.

Conclusion

Increases in computer speed are continual, and the positive relation between speed and discovery smarts is easy to see in terms of heuristic combinatorial search. By thoughtfully cultivating interdisciplinary collaborations, computer scientists can begin building, and research scientists begin building on, computer programs that are able collaborators in scientific discovery as well as other fields in which the pursuit of new, reliable knowledge is taken seriously. **C**

References

1. Bauer, H. *Scientific Literacy and the Myth of the Scientific Method.* University of Illinois Press, Urbana, Ill., 1994.
2. Bruk, L., Gorodskii, S., Zeigarnik, A., Valdés-Pérez, R., and Temkin, O. Oxidative carbonylation of phenylacetylene catalyzed by Pd(II) and Cu(I): Experimental tests of 41 computer-generated mechanistic hypotheses. *J. Molec. Catal. A: Chem. 130*, 1–2 (1998), 29–40.
3. Chung, F. The average distance and the independence number. *J. Graph Theory 12,* 2 (1988), 229–235.
4. Langley, P. The computer-aided discovery of scientific knowledge. In *Proceedings of the 1st International Conference on Discovery Science* (1998). Springer-Verlag, New York.
5. Swanson, D. Migraine and magnesium: Eleven neglected connections. *Perspect. Biol. Med. 31,* 4 (1988), 526–557.
6. Valdés-Pérez, R. Principles of human-computer collaboration for knowledge discovery in science. *Artif. Intel. 107,* 2 (1999), 335–346.

Raúl E. Valdés-Pérez (valdes@cs.cmu.edu) is a senior researcher in the Computer Science Department of Carnegie Mellon University in Pittsburgh.

This work has been supported by NSF grants #IRI-9421656, #DBI-9727699, and IRI-9819340 and by the Center for Light Microscope Imaging and Biotechnology at Carnegie-Mellon University in Pittsburgh.