Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2009), *23*, 331–338. Printed in the USA.
Copyright © 2009 Cambridge University Press 0890-0604/09 $25.00
doi:10.1017/S0890060409990035

Problem solving methods and knowledge systems: A personal journey to perceptual images as knowledge

B. CHANDRASEKARAN
Laboratory for Artificial Intelligence Research, Department of Computer Science and Engineering, The Ohio State University, Columbus, Ohio, USA

(Received December 1, 2008; Accepted January 15, 2009)

Abstract

I was among those who proposed problem solving methods (PSMs) in the late 1970s and early 1980s as a knowledge-level description of strategies useful in building knowledge-based systems. This paper summarizes the evolution of my ideas in the last two decades. I start with a review of the original ideas. From an artificial intelligence (AI) point of view, it is not PSMs as such, which are essentially high-level design strategies for computation, that are interesting, but PSMs associated with tasks that have a relation to AI and cognition. They are also interesting with respect to cognitive architecture proposals such as Soar and ACT-R: PSMs are observed regularities in the use of knowledge that an exclusive focus on the architecture level might miss, the latter providing no vocabulary to talk about these regularities. PSMs in the original conception are closely connected to a specific view of knowledge: symbolic expressions represented in a repository and retrieved as needed. I join critics of this view, and maintain with them that most often knowledge is not retrieved from a base as much as constructed as needed. This criticism, however, raises the question of what is in memory that is not knowledge as traditionally conceived in AI, but can support the *construction* of knowledge in predicate–symbolic form. My recent proposal about cognition and multimodality offers a possible answer. In this view, much of memory consists of perceptual and kinesthetic images, which can be recalled during deliberation and from which internal perception can generate linguistic–symbolic knowledge. For example, from a mental image of a configuration of objects, numerous sentences can be constructed describing spatial relations between the objects. My work on diagrammatic reasoning is an implemented example of how this might work. These internal perceptions on imagistic representations are a new kind of PSM.

Keywords: Generic Tasks; Knowledge Systems; Problem Solving Methods

1. INTRODUCTION

1.1. Knowledge-based systems (KBS), tasks, and methods

The artificial intelligence (AI) community's *raison d'être* is to provide an understanding of intelligence in the computational paradigm and to use this understanding to create a general intelligent machine. At the time of the emergence of KBS (also called "expert systems" at that time) a view had arisen that intelligent behavior was the product of an interaction between knowledge represented in an agent and various inference and problem solving techniques that the agent possessed. By "knowledge" was meant something specific: encodings in a linguistic–symbolic[1] representation.

The field of KBS built on this view of the relation between intelligence and knowledge representation, but proposed that intelligent behavior, at least behavior of experts in various domains, arose from the possession of vast amounts of domain-specific knowledge, and not by the experts making use of complex inferences and problem solving strategies. The

Reprint requests to: B. Chandrasekaran, Laboratory for Artificial Intelligence Research, Department of Computer Science and Engineering, The Ohio State University, Columbus, OH 43210, USA. E-mail: chandra@cse.ohio-state.edu

[1] In AI, the term symbolic is used to refer to a specific type, one that is composed of symbolic expressions describing properties of and relations between individuals in the domain of interest, similar to sentences in natural language. The processes that manipulate them use only the syntactic properties of the expressions. This is a rather narrow construal of symbolic representation systems: the range of symbolic representations is much wider, as discussed in Goel (1995), and exemplified by the idea of perceptual symbol systems (Barsalou, 1999; Chandrasekaran, 2006). However, in this paper we use symbolic to refer to predicate symbolic representations.

field was launched by a small number of demonstration systems that simulated professional problem solving in selected domains and tasks: diagnosis and treatment in a medical area (Mycin; Shortliffe, 1976), abducing molecular structures from relevant data (Dendral; Buchanan & Feigenbaum, 1978), and a kind of engineering design (R1; McDermott, 1982).

For people within AI like me, whose interest was and remained the understanding of intelligence, the emerging field was attractive because its apparent success in modeling complex if narrowly focused problem solving suggested that it could provide an arena in which deeper issues about the nature of cognition could be investigated. In contrast, the field attracted wide attention outside of AI because it seemed to promise a technology for building high-performance systems to automate expert problem solving. Researchers in various professional areas, and entrepreneurs outside, saw opportunities to build valuable applications. This journal owes its founding to the excitement surrounding the new technology, as one of the many, covering different professional areas (medicine, engineering, and financial decision making, to name a few) that wanted to explore the promise of KBS in their respective domains.

The idea of problem solving methods (PSMs) arose when some of us saw structure in the knowledge used in professional problem solving. The structure we saw was that the knowledge explicitly or implicitly contained strategies, even though the dogma of the incipient field, with its credo "Knowledge[2] is power," was dismissive toward methods and strategies. In our laboratory, Gomez and I (Gomez & Chandrasekaran, 1981) were working on medical diagnosis and noted that underlying the diagnostician's behavior was *classification* in a disease hierarchy. Brown and I (Brown & Chandrasekaran, 1986) noted that underlying our designer–collaborator's behavior was instantiating a skeletal plan and refining it. Mittal et al. (1984) noted that before classification the diagnostician went through a set of activities whose effect was to transform patient data into a form that could help in classification. We also noted that determining if a disease class applied to the case at hand called for a kind of concept matching that had its own interesting hierarchical structure. Josephson et al. (1987) pointed out that diagnosis was "best explanation" reasoning, which was a form of abductive inference, and which often made use of classification to generate good diagnostic hypotheses.

I attempted to unify the above empirically obtained insights into a framework. In my 1986 work (Chandrasekaran, 1986) I called the activities *classification, concept matching, abductive inference, skeletal planning*, and so forth, *tasks* without realizing the ambiguity in the term: task could be the goal or means or both. In fact, in our initial conception, they could be seen as goals ("To Classify") and as methods ("Establish and Refine"). I also noted that these tasks were independent of the domain: whether the diagnosis was done in some medical area or in mechanical systems, the underlying problem solving behavior seemed to involve the same tasks, abductive best explanation, hierarchical classification, and so forth, so I called them *generic tasks* (GTs). My coworkers and I showed how the GT/PSM analysis helped in knowledge acquisition (Bylander & Chandrasekaran, 1987; Chandrasekaran, 1989) and explanation of problem solving (Chandrasekaran et al., 1989; Tanner et al., 1993).

About the same time we were getting started and with intuitions similar to ours, Clancey (1985) noted that, contrary to the claim that knowledge in the form of domain-specific rules was doing all the work, the diagnostic component of Mycin actually followed a strategy he called "heuristic classification," which had three parts: *data abstraction, heuristic match*, and *refinement*. Some of the rules in Mycin were not domain rules, but encoding of this strategy. Clancey's motivation for analyzing Mycin arose from his attempts to build a tutorial system based on Mycin. Tutoring requires explanation, and an explanation of how Mycin solves problems required, well, an understanding and explication of its strategy.

There was soon substantial work in this mold of tasks and methods in many other research centers in the United States, Europe, and Japan. Without intending any disrespect, I will forgo a history of this effort here for two reasons: my earlier papers, for example, Chandrasekaran and Johnson (1993), review them and two, my goal here is a rapid recapitulation of some of the basic ideas so I can get to my current thinking on them.

In regard to the evolution of our thinking on GTs, with our initial ideas, it was tempting to think of a problem solving architecture that consisted of a collection of task-specific problem solving modules, each built from a PSM shell with domain-specific knowledge and calling on other modules[3] for the subtasks outside of its competence (Gomez & Chandrasekaran, 1981; Chandrasekaran, 1986). For instance, we built a diagnostic system that had the following modules: Data Retrieve/Abstract (DA), Establish–Refine Hypothesis (ER), Hypothesis Match (HM), and Abductive Assembly of Hypotheses (AA). DA would abstract patient-specific data into descriptions that HM would use to help the Establish subgoal of ER decide the degree of confidence in a hypothesis and what it can explain, AA would collect all the hypotheses and their explanations and assemble a subset as the best explanation composite hypothesis.

Our initial work and the later GT Toolkit were based on this approach to the architecture. However, as we applied this architecture to real-world diagnostic problems of some complexity, we found that the interaction between the problem solvers needed to be much finer grained than between PSMs as such. The fine-grained interaction between individual subtasks required that they be represented, invoked, and composed more flexibly. To accomplish this we turned to the Soar architecture (Laird et al., 1987). All the subtasks

[2] Again, this is a rather narrow view of knowledge: that it consists of domain-specific rules linking situation to action. That strategies are also knowledge was somehow not noticed by those who did not feel warmly to the introduction of knowledge use strategies.

[3] These modules, at first called "specialists," were functional units: they carried out an activity, hence the origin of the term "tasks" in our original GT work.

of the relevant PSMs were represented individually and invoked as appropriate to the problem solving state (Johnson et al., 1993). For example, the most effective sequence in a particular case might be that the presence of a datum D triggers consideration by HM of hypothesis H1, which causes AA to ask HM to consider a confounding hypothesis H2. However, in another case, the appropriate sequence might be that AA builds the best partial composite hypothesis with ER having explored only a part of the hierarchy, and asks ER for further hypothesis space exploration only if the explanation is not sufficiently satisfactory.

Particular compositions of subtasks, the PSMs, now served as *analysis frameworks*, rather than fully hard-wired collections of steps. This might strike one as a bit of retrogression if one viewed Soar as simply a rule-based architecture similar to the backward-chaining architecture in which Mycin was built. However, there are three important differences. First, under appropriate conditions, the learning mechanism of Soar chunks, or "compiles" in our terminology, PSMs as packages of subtasks. Second, these chunks do not stay as abstract representations of strategies. Domain-specific rules are compiled in which PSMs are implicit.[4] In other words, the agent behaves in a way that corresponds to the strategy without the strategy being explicitly set up and followed. Third, when necessary and appropriate, PSM shells can be built in this framework to obtain the advantages of PSMs. The composed versions of PSMs are best viewed as emergent regularities of problem solving behavior, instead of as a theoretically primitive building block.

Returning to compilation of expertise, even when an expert diagnostician's problem solving behavior, analyzed from the outside, appears to be an instance of say, classification, she may be unaware of it, let alone be explicitly applying the strategy. In fact, our interlocutors in medicine would often tell us their method was Bayesian, something they had been taught as the rational way to do diagnosis. Not only was there no evidence of that in their behavior, but also their behavior fit neatly with the classification/abductive inference explanation. They would go from a symptom to "That is high," to "Looks like a Liver problem," to looking for evidence for say Cirrhosis. It was as if they had a rule, "If Liver disease confirmed, consider Cirrhosis," rather than a rule, "When a hypothesis is established, consider its children in the hierarchy." A protocol like this actually tends to confirm the original KBS dogma—it is all domain knowledge—while clearly the behavior matches the method.

The first explanation for the seeming paradox is that of compilation by the agent, as modeled by the Soar implementation mentioned earlier. While solving an earlier case, if the agent had decided on liver, then explicitly looked for its subtypes and then went on to consider cirrhosis, he would have learned the rule, "If Liver confirmed, consider Cirrhosis." A second explanation is that knowledge might be acquired directly in a compiled form; for example, the diagnostician might have been trained to consider cirrhosis if liver disease is confirmed. Community knowledge evolves into a form that is useful, without the acquirer/user of this knowledge necessarily being aware of the reason for the efficacy. Another example of this community knowledge with a useful underlying form is the existence of disease hierarchies that a medical student might directly acquire. Whether the agent is explicitly using the PSM, the underlying PSM, in this case classification, is important as a theoretical construct: the PSM provides an explanation for how and why the specific items of knowledge work for this problem, how to adapt when the classification hierarchy changes, and how to acquire knowledge in new domains.

[4] This has echoes of an area of research in psychology called *implicit learning* (for a review, see, e.g., Berry & Dienes, 1993).

2. FROM PSMs TO TASK STRUCTURES

PSMs are organized collections of generic subtasks intended for a type of problem solving goal. However, the subtasks of a PSM can be useful for other tasks, and conversely, a task might have more than one PSM, depending on availability of knowledge or other situation-specific differences. Data Abstraction and Heuristic Match can be useful for tasks are other than Classification. For example, during design, a specification (e.g., "1200 psi") might be abstracted to a qualitative value ("high pressure"). A set of such qualitatively abstracted specifications (e.g., "high pressure" and "light weight") might be heuristically matched to subdevice alternatives (e.g. "titanium valve #58" and "stainless steel valve #60," with matching strengths 0.8 and 0.6) that might realize them and the best one chosen (in the case at hand, "titanium valve #58"). Conversely, a classification task might use a different method than Heuristic Match for assessing a hypothesis: for example, a diagnostic hypothesis might be assessed for fitness with the observations by simulating the malfunction and seeing if the observations correspond to simulation results. This two-way multiplicity of connections, between a task and a variety of methods and between a method and a variety of tasks in which the method could play a role, suggested a broader conception of PSMs to us: the notion of a *task structure* (Chandrasekaran et al., 1992). The idea is that a generic problem solving goal can be decomposed into an AND–OR graph of generic subtasks, representing alternate methods, and each of the subtasks might be similarly decomposed, this process repeated to a level that might be considered primitive enough for the purpose. Chandrasekaran (1990) provides a task analysis for the design task, and Josephson and Josephson (1994) and Chandrasekaran et al. (1992) provide analyses for diagnosis.

3. PSMs AS KNOWLEDGE LEVEL REPRESENTATIONS OF STRATEGY TRANSCENDING KBS

The task structure lays out a flexible strategy for achieving the goals of the task, but there is nothing about the strategy that requires that it be implemented in a certain way.

In particular, there is no reason to prefer implementation in the framework of KBS technology, that is, a software technology in which atomic pieces of rulelike knowledge in a domain are encoded in a knowledge base, and a domain- and task-independent mechanism accesses the elements of the knowledge base to solve problems in the domain.

For example, diagnostic systems following an abductive inferencing task structure could be implemented by appropriately combining the component problem solvers, the abductive inferencer, the classifier, and so forth, each implemented in any suitable framework. In fact, many diagnostic systems in industrial applications are built as quite conventional algorithms not requiring representations of abstract knowledge. The algorithm might not even follow the procedural arc of a PSM, even when it informed the construction of the algorithm; for example, it might not have a modular structure, where the modules correspond to the subtask–method combinations in the PSM. In the diagnostic example, a general top-level PSM for the task is AA, which requires that the final diagnostic solution explain all the observations, and that it is a better explanation than competing alternatives. The corresponding PSM explicitly sets up a subtask of comparing alternative explanations. In specific diagnostic applications, the evidence available for establishing or rejecting various diagnostic hypotheses from sensor data may be strong enough that the conclusion is definitive.[5] In such domains, the algorithm in the deployed application might not explicitly generate and compare alternate hypotheses. Nevertheless, such an algorithm can be deemed correct only after the algorithm constructor has explicitly considered the comparison subtask and justified to herself that the nature of the evidential strength in the specific application ensures that the subtask is implicitly satisfied.

Thus, the strategic content of a PSM can be useful in building systems in any framework. The GT approach's separation of the GT level from implementation level systems such as rule-based systems fitted in nicely with the knowledge level–symbol level distinction that Newell (1982) was formulating, approximately contemporaneously. Newell introduced the Problem Space Computation Level between the Symbol Level of the architecture implementation and the Knowledge Level. PSMs can be viewed as content theories of the Problem Space Computation Level.

4. ARE PSMs INTERESTING?

If PSMs are a representation for computational strategies in general with no features that distinguish them as ideas for AI or cognition,[6] as we just argued, do they matter for AI? There are two reasons why they are important. First, certain specific PSMs say interesting things about cognition; second, the usefulness of these specific things in accounting for the power of cognition provides a caution about attempts to explain cognition solely at some architecture level.

KBSs were originally developed to perform certain tasks that were of practical interest and that seemed to require large amounts of expert knowledge, tasks such as diagnosis, design, and planning. PSMs were then developed in the service of these tasks. Diagnosis may sound like a specific application in engineering and medicine, and design, similarly, an activity in engineering, with little to say about AI in general; after all, we do not regard finite element analysis, another problem area in engineering to which much computational attention has been paid, as a task of interest to AI. However, diagnosis and design are instances of more general tasks that are quite fundamental to the operation of agents: goal-seeking systems interacting with the physical world. At a high level of analysis, any agent of this sort has to perform three activities, in whatever mixture of implicit and explicit that works. The three tasks are: making a model of the relevant aspects of the external world where the goals need to be achieved; planning actions to take in the world so as to achieve the goal; and predicting the performance of the plan, most commonly by simulating the plan in some manner, to verify that it can accomplish the goals. These tasks need not be done explicitly, but the requirements of the tasks have to be met in some way. The tasks interact: prediction requires the use of the model of the world, model making requires prediction of the behavior of the model to verify the model itself, and of course planning requires prediction.

Making a model of the world (a picture of what is going on) is an abductive inference task, as my colleague Josephson has argued (Josephson & Josephson, 1984). The input from the senses are data to be explained, and the best model is one that provides the best explanation of the perceptions. Diagnosis is an instance of model making; in this case, a model of the malfunctioning entity that best explains the symptoms.[7] In fact, part of the model-making task for an agent will indeed include diagnosis-like activities: what is broken in the world to explain certain observations so that the model can be updated. Of course, biological agents come to the task with a good deal of knowledge about the world precompiled in their perception, action, and cognitive systems, so the complexity of model making, especially when it needs to be done in real time, is not prohibitive. Hence, the task structure of diagnosis, a contribution of PSM research, is very useful for constructing general agents, a core concern of AI.

Consider classification that plays such a prominent role in many analyses of diagnostic activity. Why is classification so ubiquitous in human knowledge and behavior? Consider why diagnosis is done: the need is to map from a situation description to a proposed action. The computational complexity of doing the mapping, let alone learning the mapping, can be extremely high, if the agent is going to use a table of mappings. The number of situation descriptions can be huge: for example, temperature 100.1°, temperature 100.2°, and so on, just

[5] This kind of a relation between an observation and a diagnostic hypothesis is termed *pathognomonic* in medicine.

[6] Particular implementations of PSMs may have AI interest, for example, as *specialists* (Gomez & Chandrasekaran, 1981), a premodern version of AI agents. However, as argued earlier, the concept of PSMs transcends specific implementations.

[7] Even when diagnosis is implemented by simple classification, assigning a class label, say, Liver Disease, to a set of symptoms, is building a model of an aspect of the relevant reality, "the patient's Liver is malfunctioning."

for one variable, and similarly for action descriptions. The computational strategy of first mapping from situations to an equivalence class of situations, and then mapping from the situation class to an action class and possibly refining it, would result in a substantial reduction in the computational complexity of learning and run-time performance, if appropriate knowledge is available. It so happens that the causal structure of the world is cooperative: systems, natural or artificial, are more often than not compositional and hierarchical, what Simon (1947) called loosely coupled systems of subsystems. These structure hierarchies provide a basis for malfunction hierarchies.

Similarly, planning for actions can take advantage of the fact that things can be composed hierarchically. The computational complexity of design and planning can similarly be reduced if they can take advantage of this structure. In fact, choosing a design template, instantiating it for some of the specifications, and then refining the abstract elements into subsystem designs, is a ubiquitous strategy in design and planning (Brown & Chandrasekaran, 1986). Similarly, simulation for prediction makes use of the hierarchies of structure and behavior (Iwasaki & Chandrasekaran, 1992). Human knowledge and strategies evolve to reflect these advantages of classification, plan refinement, and behavior projection. Psychology seems recently to be rediscovering these ideas from AI and KBS; the term "macrocognition" (Schraagen et al., 2008) has been proposed to describe some of the regularities in cognitive behavior, and the proposed types of macrocognitive activities are quite similar to the various GTs and the PSMs that go with them. For example, "sense making," a frequently given example of macrocognition, corresponds to the task of building a model that best explains the observations, an abductive inference task.

To summarize the preceding discussion, it is not the idea of PSMs or something that AI uniquely brings to representing them that is interesting from an AI point of view. It is the specific PSMs, classification, hierarchical instantiation and composition, simulation making use of hierarchies of structure, abductive explanation building, and so forth, that were and are interesting because they say something about the structure of knowledge and the implicit or explicit strategies that underlie cognition. This leads us to the second point about why PSMs are interesting for AI.

Let us look at the historic context in which the idea of PSMs arose and was seen as interesting. The original proposal of PSMs was a counter to the claim that there was nothing theoretically interesting about how intelligent behavior arises beyond a rather minimal architecture: a knowledge base of rules (or frames or sentences, in variant architectures), a working memory to hold goals and problem specifications, a set of basic matching and rewrite rules, and a general control strategy. From then on, it was all allegedly due to expertise and domain-specific knowledge, which are not especially interesting theoretically for AI or cognition: for instance, why would the medical knowledge of a diagnostician be of interest to theorists of cognition? As my historical account recapitulated, many of us demurred with this stance, and contended that the knowledge base contained implicitly or explicitly general strategies that indeed were relevant for understanding intelligent behavior. The PSM movement was a shot across the bow of the then dominant claim to reduce explanation of intelligence to the architecture level alone. Some kind of content mattered theoretically, the movement said, content having to do with a variety of knowledge-use strategies shared by the human community as a whole, often implicitly[8] organizing our knowledge, and at times explicitly available as strategies. I think this message continues to be relevant for understanding cognition.

This is also an occasion to bury the worry often expressed, not least by me, about how many tasks and PSMs there are. The constituents of PSMs are operations on information structures, and the ontology of these operations is not a closed set. There is no periodic table of methods containing the atoms of such operations. However, the top-level regularities in knowledge use: model making, action planning and prediction, as well as the computational and organizational strategies such as classification and hierarchical planning, are basic and widely useful, and that is why these tasks and PSMs are worth the attention.

5. THE STATE OF KBS APPLICATIONS

Two decades after the excitement about KBS as a new software technology framework to build applications, as a particularly attractive alternative to traditional software technologies, it is fair to say that the technology has not had the explosive growth that was hoped, although interesting research continues to be conducted. One kind of application seems to have been moderately successful, and the other seems to have not been so successful.

The first kind, applications of the successful kind, often may not even appear to be built in the knowledge system framework, but their development owes much to PSMs developed in the KBS work. We mentioned earlier about diagnostic and design applications in industry that are built as conventional software, but which nevertheless owe much to PSM research. One reason why these applications are built in conventional frameworks is the need to scope their applicability. In industry, there are significant incentives associated with deploying systems with provable, or at least reliable, performance. Thus, unless the system builder is in a position to precisely characterize the range of problems solvable and has a clear sense of closure about the relation between encoded knowledge and the range of problems it can support, users of any specific KBS intended for some task might have to live in fear that it would exhibit unpredictable, brittle performance. It is true that no such guarantees are sought or given in the case of human experts, but we are far from being able to claim sufficiently complete encoding of what an expert knows. It is therefore hard to have a similar faith in systems

[8] See our earlier discussion of how learning can produce implicit knowledge.

even when they are notionally based on human expertise. As a result, there is a strong incentive to extract a traditional algorithmic solution from a prototype KBS solution, and use these algorithms in deployed applications. The performance and the scope of applicability of such algorithms are often easier to characterize. As mentioned earlier, PSMs have a useful role to play in the design of such algorithms. For example, R1 demonstrated the use of the KBS systems technology in a class of design problems. Later analysis revealed the implicit PSM as a form of constraint satisfaction. Since then, numerous applications have been built and deployed for that class of design problems, but by using constraint–satisfaction algorithms instead of a general KBS technology. Similarly, diagnostic applications are built and deployed in the traditional algorithmic framework, and these applications often make use of PSMs such as classification or abductive inference PSMs as design templates.

The second type of application might seek its advantages in what is supposed to be the selling point of KBS in the first place: large knowledge bases that embody the knowledge of experts. If the advantages are sufficiently great and the deployment context is sufficiently forgiving, systems with not entirely predictable performance would still be useful. However, soon it was realized that there was no clear separation between expertise and more general, so-called common sense knowledge. There was a developing consensus that for KBSs not to exhibit "brittle" behavior, in other words, behavior that is catastrophically inappropriate, the knowledge bases needed to have commonsense knowledge that is shared by humans. This, in turn, launched research efforts to encode commonsense knowledge into a shareable repository, the most famous of which was CYC (Lenat & Guha, 1989). The CYC knowledge base is now a commercial enterprise, and an objective evaluation of its successes and failures is hard to come by. My sense is that CYC's domain-specific ontologies and knowledge bases built using them have found much more use than the so-called common sense base. A conservative judgment is that the vision of practical systems that tap into large bases of commonsense and professional knowledge to solve complex problems that currently require experts is quite far from realization. I next turn to why such knowledge has been difficult to capture.

6. THE FUTURE, OR MODESTLY, A FUTURE

I agree with Clancey's critique (1991) of the view, which is accepted almost universally in symbolic AI but without deep examination, that human memory is a repository of information, which is encoded in a functional equivalent of predicate-symbolic expressions, from which information or knowledge items are *retrieved*. We, of course, have much factual knowledge, such as "New Delhi is the capital of India," that we more or less retrieve.

However, not *all* knowledge that seems to appear in our deliberation is retrieved: much of it is constructed in response to the goals of the moment. In fact, as he has further argued (Clancey, 1989), knowledge bases are domain models; the knowledge base of Mycin is not a core dump of some expert's relevant knowledge as much as it is some expert's construction in rule form of the relevant parts of his expertise. We as agents are generally unaware of the process of construction, so the retrieval story, inspired by the ubiquitous computer memory metaphor, seems unproblematic on first acquaintance.

Clearly, something is in our memory, and that something generates information useful to pursue our goals. The problem is the nature of what is in memory. Let us consider two examples to highlight the issues involved.

When I was growing up in India, a prototypical example we children were told of behavior violating common sense was that of a person sitting at the edge of a branch on a tree and sawing the branch on the side between him and the tree. Even a child instantly recognizes this as comically foolish. If asked why, the child might say something like, "He's going to fall down when he's finished sawing!" What exactly was retrieved from the child's memory? It is highly unlikely that it was a piece of knowledge of the form, "If a person sits on a tree branch, saws it on the side closer to tree, he'll fall," nor even of the form, "If an object is sticking out at a height, . . ." One could imagine abstract axioms about gravity from which the inference could be made, but the amount of computing involved to generate the answer would be substantial.

Let us look at another example. Imagine a person was at a party one night and someone asks the next day, "Was Stephanie standing closer to Bill than to Stu?" If the partygoer had no reason to expect that he would be quizzed the next day on this, it is quite unlikely that he would have noticed the relative distances between Stephanie, Bill, and Stu, and thus it is unlikely he could reply by retrieving the appropriate linguistic symbolic unit from his memory. For one thing, slightly different forms of the question, for example, "Was Stephanie close enough to Bill that she could hear what he said, but too far from Stu to hear what he said?" require different sentences to be stored in memory. Storing all possible answers in memory in linguistic symbolic form would it be highly inefficient in storage, time consuming at the time of the experiencing the event, and the partygoer would not be able to anticipate all of the questions. It follows that the relevant part of the partygoer's memory had to be of the sort from which different answers could be generated, depending on the question.

Much of our memory, including parts corresponding to so-called commonsense knowledge, is body and perception based, and multimodal, as many have argued (Johnson-Laird, 1983; Chandrasekaran, 2006). What appears to be linguistically represented knowledge is generated from these multimodal memory fragments by means of a process that is functionally like a form of internal perception. In the partygoer example, what he has in his mind is a visual, almost diagrammatic, representation of the locations of Stephanie, Bill, and Stu. This basic representation plus internal perception,[9]

[9] No need to worry about homunculi. Internal perception does not call for an internal screen and eye, but activation of the same neural processes involved in perceiving categories and conceiving relations among them.

being able to "see" spatial relations in the mental image, is sufficient to support a wide variety of linguistic descriptions: "knowledge," as people in AI term it. In the case of the child laughing at the image of the man on the branch sawing the branch, the memory is visual as well as kinesthetic.

One way of looking at what I have been doing lately is as an entirely new variety of PSM. The notion of knowledge adopted in AI, that it is something expressed as a collection of predicate–symbolic expressions describing properties of and relationships between individuals in a domain of discourse, is an entirely too narrow conception of knowledge for knowledge *representation* in cognitive agents. If I say Jill knows a lot about Beethoven's music, I do not intend that she has in her head a lot of symbolically represented propositions about Beethoven's music. Among other things, I would expect her to recognize Beethoven's music, perhaps play some of it, and so on, none of which is reducible to a set of propositions in one's head. Knowing New York City includes being able to visualize its layout and avenues and streets. Of course, this knowledge might be *used* to generate linguistically expressed versions such as "Subway station A is closer to Bloomingdales than station B," to support decisions about which subway line to take. However, what is in memory is in an altogether different form than what is taken to be knowledge representation in AI.

Therefore, knowledge, broadly conceived, is all the representations in all modalities that an agent might have about his world. In particular, much of so-called commonsense knowledge, the acquisition of which precedes the development of the linguistic mode in children, is in the form of multimodal representations, including that of the body in relation to the environment. This view of internal representations as perceptual and imagistic finds support from recent work in neuroscience (Damasio, 1994) and psychology (Barsalou, 1999).

With this broader conception of knowledge, the new sets of PSMs I am interested in understanding are the sets of internal perception and action routines (Chandrasekaran, 2006) available for operations on representations in each of the modalities, and of course, the representations themselves. The representations have to be compositional, but are not compositions of Turing symbols, but rather perceptual symbols, which is quite a different notion. My collaborators and I (Chandrasekaran et al., 2004) have defined and implemented such a representation for diagrams, a limited version of representations in the visual modality. The diagrammatic representation we propose retains the spatial information about the diagrams and its elements. The diagram as a whole is not represented as an image, an array of pixels. Instead, in our representation, the diagram is a spatial composition of diagrammatic objects, each of which is a point, curve or region, and each of whose spatiality is fully available. A repertoire of perceptions and diagram modification/creation actions is also available; these may be thought of as new type of PSM. In the partygoer example, his relevant working memory representations are such diagrams: the partygoer applies perceptions of relative distances to his memory fragments from the party to answer the questions.

Diagrammatic representations are a subset of visual representations, and we chose that domain for two reasons. First, they are important in practice: what would engineering problem solving do without diagrams? Second, they provide an arena in which larger questions about perceptual representations as part of cognition, memory, and internal perceptions can be studied.

This is the story of a certain continuity and a certain shift in my research over the last two decades. The continuity is that the subject matter has remained the same: the nature of cognition, especially the representations and the processes associated with it. The shift is a fairly radical change in the conception of the cognitive state and knowledge: abandonment of what I now think is too narrow a conception of the language of thought and adoption of a view that gives pride of place to our perceptual and kinesthetic experiences.

ACKNOWLEDGMENTS

This work was supported by participation in the Advanced Decision Architectures Collaborative Technology Alliance sponsored by the US Army Research Laboratory under Cooperative Agreement DAAD19-01-2-0009. I am deeply indebted to my collaborators in the enterprise reviewed here. I thank David C. Brown for inviting me to write this article and putting up with me during a bad case of writer's block. Bill Clancey provided a characteristically generous review that nevertheless managed to set me straight on many points and suggested a large number of improvements. The anonymous reviewers were very forthcoming with useful comments that I took to heart, and I thank them sincerely. The determined efforts of all these people notwithstanding, the author is solely responsible for the contents.

REFERENCES

Barsalou, L.W. (1999). Perceptual symbol systems. *Behavioral and Brain Sciences 22*, 577–609.

Berry, D.C., & Dienes, Z. (1993). *Implicit Learning: Theoretical and Empirical Issues*. Hove: Erlbaum.

Brown, D.C., & Chandrasekaran, B. (1986). Knowledge and control for a mechanical design expert system. *IEEE Computer 19(7)*, 92–100.

Buchanan, B.G., & Feigenbaum, E.A. (1978). DENDRAL and Meta-DENDRAL: their applications dimension. *Artificial Intelligence 11*, 5–24.

Bylander, T., & Chandrasekaran, B. (1987). Generic tasks for knowledge-based reasoning: the "right" level of abstraction for knowledge acquisition. *International Journal of Man–Machine Studies 26*, 231–224.

Chandrasekaran, B. (1986). Generic tasks in knowledge-based reasoning: high-level building blocks for expert systems design. *IEEE Expert 1(3)*, 23–30.

Chandrasekaran, B. (1989). Task–structures, knowledge acquisition and learning. *Machine Learning 4*, 339–345.

Chandrasekaran, B. (1990). Design problem solving: a task analysis. *AI Magazine 11(4)*, 59–71.

Chandrasekaran, B. (2006). Multimodal cognitive architecture: making perception more central to intelligent behavior. *Proc. Natl. Conf. Artificial Intelligence*, pp. 1508–1512. Menlo Park, CA: American Association for Artificial Intelligence.

Chandrasekaran, B., & Johnson, T.R. (1993). Generic tasks and task structures: history, critique and new directions. In *Second Generation Expert Systems* (David, J.M., Krivine, J.P., & Simmons, R., Eds.), pp. 239–280. Berlin: Springer–Verlag.

Chandrasekaran, B., Johnson, T.R., & Smith, J.W. (1992). Task structure analysis for knowledge modeling. *Communications of the ACM 33(9)*, 124–136.

Chandrasekaran, B., Kurup, U., Banerjee, B., Josephson, J.R., & Winkler, R. (2004). An architecture for problem solving with diagrams. In *Diagrammatic Representation and Inference* (Blackwell, A., Marriott, K., & Shomojima, A., Eds.). LNAI, Vol. 2980, pp. 151–165. Berlin: Springer–Verlag.

Chandrasekaran, B., Tanner, M.C., & Josephson, J.R. (1989). Explaining control strategies in problem solving. *IEEE Expert 4(1)*, 9–24.

Clancey, W.J. (1985). Heuristic classification. *Artificial Intelligence 27(3)*, 289–350.

Clancey, W.J. (1989). Viewing knowledge bases as qualitative models. *IEEE Expert 4(2)*, 9–23.

Clancey, W.J. (1991). Situated cognition: stepping out of representational flatland. *AI Communications 4(2/3)*, 109–112.

Damasio, A.R. (1994). *Descartes' Error: Emotion, Reason, and the Human Brain*. New York: Putnam.

Goel, V. (1995). *Sketches of Thought*. Cambridge, MA: MIT Press.

Gomez, F., & Chandrasekaran, B. (1981). Knowledge organization and distribution for medical diagnosis. *IEEE Transactions on Systems, Man, and Cybernetics 11(1)*, 34–42.

Iwasaki, Y., & Chandrasekaran, B. (1992). Design verification through function and behavior-oriented representations: bridging the gap between function and behavior. *Artificial Intelligence in Design '92* (Gero, J.S., Ed.), pp. 597–616. New York: Kluwer Academic.

Johnson, T.R., Smith, J.W., & Chandrasekaran, B. (1993). Task-specific architectures for flexible systems. In *The Soar Papers: Research on Integrated Intelligence* (Rosenbloom, P.S., Laird, J.E., & Newell, A., Eds.), pp. 1004–1026. Cambridge, MA: MIT Press.

Johnson-Laird, P. (1983). *Mental Models*. Cambridge, MA: Harvard University Press.

Josephson, J.R., Chandrasekaran, B., Smith, J.W., & Tanner, M.C. (1987). A mechanism for forming composite explanatory hypotheses. *IEEE Transactions on Systems, Man and Cybernetics 17(3)*, 445–454.

Josephson, J.R., & Josephson, S.G. (1994). *Abductive Inference: Computation, Philosophy, Technology*. New York: Cambridge University Press.

Laird, J., Newell, A., & Rosenbloom, P. (1987). SOAR: an architecture for general intelligence. *Artificial Intelligence 33(1)*, 1–64.

Lenat, D., & Guha, R.V. (1989). *Building Large Knowledge Based Systems: Representation and Inference in the CYC Project*. Reading, MA: Addison–Wesley.

McDermott, J.P. (1982). R1: A rule-based configurer of computer systems. *Artificial Intelligence 19(1)*, 39–88.

Mittal, S., Chandrasekaran, B., & Sticklen, J. (1984). Patrec: a knowledge-directed database for a diagnostic expert system. *IEEE Computer 17(9)*, 51–58.

Newell, A. (1982). The knowledge level. *Artificial Intelligence 18(1)*, 87–127.

Schraagen, J.M., Militello, L., Ormerod, T., & Lipshitz, R., Eds. (2008). *Naturalistic Decision Making and Macrocognition*. Abingdon: Ashgate.

Shortliffe, E.H. (1976). *Computer-Based Medical Consultations: MYCIN*. New York: Elsevier.

Simon, S. (1947). *Administrative Behavior*. New York: MacMillan.

Tanner, M.C., Keuneke, A.M., & Chandrasekaran, B. (1993). Explanation using task structure and domain functional models. In *Second Generation Expert Systems* (David, J.M., Krivine, J.P., & Simmons, R., Eds.), pp. 599–626. Berlin: Springer–Verlag.

B. Chandrasekaran is a Professor Emeritus of computer science and engineering and Director of the Laboratory of Artificial Intelligence Research at The Ohio State University. He and David C. Brown authored *Design Problem Solving* (Morgan Kaufmann), and he is Co-Editor of *Diagrammatic Reasoning: Cognitive and Computational Perspectives* (MIT Press). Dr. Chandrasekaran was Editor-in-Chief of *IEEE Expert/Intelligent Systems* from 1990 to 1994. He is a Fellow of the Institute of Electrical and Electronic Engineers, Association for Computing Machinery, and American Association for Artificial Intelligence. His major research activities are in diagrammatic reasoning, knowledge systems, decision support architectures, and cognitive architectures.