Artificial Intelligence 59 (1993) 191–196
Elsevier

ARTINT 976

Notes on "Heuristic classification"

William J. Clancey

Institute for Research on Learning, 2550 Hanover Street, Palo Alto, CA 94304, USA

1. Introduction

The paper "Heuristic classification" [6] began as a memo written at Teknowledge, Inc. in late 1983, stemming from discussions with Steve Hardy and Denny Brown about how to teach knowledge engineering. We based our courses on a library of "sample knowledge systems" and looked for patterns that could be taught as design principles. Discussion raged about competing knowledge representations: rule versus frame languages, deep versus shallow systems, classification versus causal reasoning, first principles versus case-based modeling. Customers and students at Teknowledge pressed us to relate our products and terminology to our competitors' (what marketing people call "tool comparison"). Hardy and Brown wanted to relate our example systems to the "representation, inference, and control" framework, which they preferred for describing reasoning. I wanted to convince the developers of Teknowledge's S.1 and M.1 why a representation language should incorporate classification primitives.

Analyzing expert systems and articulating patterns was part of my continuing effort (with Jim Bennett) to encourage Teknowledge's programmers to develop task-specific tools [2]. We realized by 1982 that the Neomycin approach of abstracting the diagnostic inference procedure from the domain model produced a reusable shell that included not just an inference engine, but a task-specific representation language and reasoning procedure (the diagnostic metarules). The idea of reasoning patterns was therefore in the air and suggested commonalities that recurred across domains. Notably, Newell's "Knowledge-Level" AAAI Presidential Address at Stanford

Correspondence to: W.J. Clancey, Institute for Research on Learning, 2550 Hanover Street, Palo Alto, CA 94304, USA. E-mail: clancey@xerox.parc.

in August 1980 [18] impressed upon us the importance of levels of description, and the need to move our descriptions from the implementation (rules versus frames) to the conceptual level. I already had some experience in such cross-architecture comparisons (e.g., Section 7 of the "epistemology" paper [5] applied the structure–strategy–support framework to six knowledge-based systems ranging from AM to Hearsay).

2. A few clarifications

The strength of the "heuristic classification" article may be its generality, for it can be something to everyone. But there have been a few important misinterpretations:

(1) I strongly urged people not to view classification as an inherent property of problems. Classification is a *method* for constructing a situation-specific model; a given modeling *purpose*, such as diagnosis, might be accomplished in different ways, depending on the kind of model available. Problem types can be classified more usefully in terms of the purpose for constructing a model of some system in the world (i.e., the *tasks* of diagnosis, planning, control, repair, etc.).

(2) I distinguished between analytic and synthetic tasks, which I perhaps unfortunately labeled "interpret/analysis" and "construct/synthesis" (Figs. 5.1 and 5.2). A few readers confused these terms—which refer to analyzing an existing system in the world (e.g., predicting or explaining its behavior) or constructing a new system in the world (i.e., building it or repairing it)—with how the situation-specific model is *inferred*. The point of the article of course was to contrast inference by *selection* of models from a pre-enumerated classification with *construction* of models from structural and functional components related spatially, temporally, and causally. The typology of problem tasks refers to *why* the system is being modeled; the typology of inference methods refers to *how* the model is developed.

When writing the article, I was unsure how to contrast the process of constructing a line of reasoning in heuristic classification with the process of constructing a new model. Section 6.2.4 is one attempt, based on pre-enumerated versus new links between concepts. In "Model construction operators" (MCO) [11], I emphasize that construction of situation-specific model graphs is always occurring in expert systems; the distinction between Neomycin and Abel (for example) is what the nodes and links represent. For Neomycin, using heuristic classification, each node is a process description of the entire system being modeled (e.g., "there is a bacterial agent growing in the meninges of the central nervous system"). In Abel, which constructs a situation-specific model from primitives, the nodes represent physiological substances and processes *within* the system being modeled.

3. Redescribing by revisualizing

By the time I wrote MCO, I realized that many confusions about representations could be resolved if we see them as alternative perspectives on a single "virtual" formal system. In MCO, I relate node, path, subgraph, and graph views of inference [11, Section 3]. This theme plays throughout my work, as I realized that representations and reasoning processes that were commonly viewed as different could be related by a shift in visualization:

(1) Neomycin's causal-network-to-classification inference appeared in Casnet (I missed this at first because I drew horizontally what Weiss and Kulikowski drew vertically).
(2) Mycin's context tree constitutes a three-paneled blackboard (we missed this because we drew as a tree what others drew as layered boxes; we emphasized inheritance, they emphasized levels of description).
(3) Neomycin's differential (a list of diseases) can be better represented as an explanation-proof tree, which we call the *situation-specific model* (we missed this because we thought Abel was doing "deep" causal modeling while Neomycin was only doing "shallow" classification, that is, not modeling at all).

Many other examples appear in the figures of MCO (e.g., Figs. 17, 19, 27, 34, Table 5). Through this experience I developed the intuition that seemingly intractable debates about representations often stem from different visualizations or metaphors, and hence apparently incommensurable languages, not from inherent differences in the modeling methods of the programs being described. Bill Scherlis first impressed me with this possibility in 1979, when he argued that Mycin's rules could be expressed in predicate calculus, an idea that seemed sacrilegious at the time.

4. Revisualizing is reconceiving, not deriving, mapping, or compiling

Even when I realized that there were multiple perspectives for describing representations, I thought they must be derivable from each other, such as by a compilation process. For example, I had been asked by Keith Butler (at Boeing in 1984) to explain how common representational distinctions such as class/individual, type/subtype, and definition/schema relate to the heuristic classification framework. I found that these representational primitives play different roles: Definitions tend to be used for initial abstraction of data, schemas are used for heuristic association, and subtypes are used for both abstraction and refinement. Thus, we move from a concept or terminology-centered description of the knowledge base to a line-of-reasoning, data-to-solution perspective. When writing the article, I conceived of this analysis as "deriving" the horseshoe diagram from primitive relations (Section 4.4). But it is better to say that I shifted perspective from static concepts in isolation to how information about one concept is inferred from another (revealing an ordering of relations characterized as heuristic classification).

As I relate in my comments on the "epistemology" paper [12], attempting to generate patterns by reducing them to primitive representations is a powerful computational approach for constructing new and more generally useful process models. But it is also a scientific presumption about the nature of models and levels of description that is perhaps hampering our attempts to understand how people create and use representations (cf. Lave [15]). In particular, a common assumption is that there is always one "correct" description of a system and alternative representations must be logically inferable from each other (cf. Schön's [19] critique of analogical reasoning as structure mapping). This rests on the more general assumption that reality can be exhaustively modeled, with the idea that scientific laws (the "hidden truth of the matter") literally generate all the phenomena we observe. This parallels the prevalent belief of cognitive scientists that all human behavior is generated from a representational bedrock; in particular, tacit knowledge ("know how") must be compiled from representations. (Newell explicated this in his "brain as an orange" model, in which the "core" is knowledge compiled from a "rind" of production rules.) In effect, conflating knowledge, reality, and representations shaped the dilemmas of representational theory over the past few decades (Clancey [9,10]).

5. Lessons and impact

One idea in this article (Section 5) that I believe could be developed further is to integrate knowledge engineering with systems analysis. In MCO, I extend this argument to claim that AI programming should be conceived as a process-modeling technique, emphasizing qualitative or relational representations, as opposed to quantitative or numeric representations. Integrating these approaches to serve the needs of scientific and engineering modeling was at first obscured by the original emphasis that an expert system is necessarily related to how experts reason, by virtue of the knowledge "acquisition" process by which a human "transfers expertise" to the program.

For example, even though we weren't interested in strictly modeling human reasoning, we were biased against using numeric models in expert systems because we believed classification and rule-based inference to be a better model of human knowledge than equation manipulation.

Emboldened by the publication of the Winograd and Flores book [22], I first presented these ideas at the Oregon State Knowledge Compilation and Banff Knowledge Acquisition Workshops in October 1986. I argued that it is more fruitful and appropriate to characterize a knowledge base as a model of some system in the world coupled with a task-specific reasoning procedure [7], and to not equate a representation of knowledge (the knowledge base) with knowledge, a capacity to behave [10]. We should use whatever modeling techniques are useful for the problems at hand [8]. Furthermore, we should recognize that the qualitative modeling techniques of AI programming have a generality and value that extends beyond their initial development for representing human beliefs and reasoning (MCO). Framing AI research methods in this way helps us understand why numeric representations (for example, certainty factors) seem to violate the rules of the game; also this view is important for not dismissing the value of schema models as situated cognition calls assumptions about knowledge representation into question [10].

By focusing on graph manipulation operators in MCO, I aim to squarely place knowledge engineering in the realm of computer programming and operations research. Today we are less prone to confuse means (building on people's existing language and models) with goals (constructing models in order to facilitate scientific prediction and experimentation, as well as the design and maintenance of complex engineering and organizational systems). Significantly, the "information for authors" of the *Knowledge Acquisition* journal now says, "The emphasis is not on *artificial* intelligence, but on the extension of *natural* intelligence through knowledge-based systems."

The "heuristic classification" paper helped move arguments about representations from the level of programming constructs (e.g., rules versus frames) and conceptual networks (e.g., terminology classifications) to the level of *recurrent abstractions* in process modeling (e.g., kinds of taxonomies, how modeling tasks chain together). For example, the idea that causal inferences can feed into a classification, pioneered in Casnet and rediscovered in Neomycin, is now a commonplace modeling technique that can be taught explicitly to knowledge engineers and used to structure knowledge acquisition tools. Other researchers have gone beyond my promissory notes to deliver a second generation of process-modeling languages and tools (Alexander et al. [1]; Breuker and Wielinga [3]; Chandrasekaran [4]; Gruber [13]; Hayes-Roth et al. [14]; McDermott [16]; Musen [17]; Steels [20]; Stefik [21]). In many respects, the original hope behind my conversations with Steve Hardy and Denny Brown has been realized.

References

[1] J.H. Alexander, M.J. Freiling, S.J. Shulman, J.L. Staley, S. Rehfuss and M. Messick, Knowledge level engineering: ontological analysis, in: *Proceedings AAAI-86*, Philadelphia, PA (1986) 963–968.

[2] J.S. Bennett, ROGET: a knowledge-based consultant for acquiring the conceptual structure of an expert system, *J. Autom. Reasoning* **1** (1985) 49–74.

[3] J. Breuker and B. Wielinga, KADS: structured knowledge acquisition for expert systems, in: *Proceedings Second International Workshop on Expert Systems*, Avignon, France (1985).

[4] B. Chandrasekaran, Generic tasks in knowledge-based reasoning: high-level building blocks for expert system design, *IEEE Expert* **1** (3) (1986) 23–29.

[5] W.J. Clancey, Epistemology of a rule-based expert system—a framework for explanation, *Artif. Intell.* **20** (3) (1983) 215–251.

[6] W.J. Clancey, Heuristic classification, *Artif. Intell.* **27** (3) (1985) 289–350.

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

[8] W.J. Clancey, The knowledge level reinterpreted: modeling how systems interact, *Mach. Learn.* **4** (3–4) (1989) 287–293.

[9] W.J. Clancey, The frame of reference problem in the design of intelligent machines, in: K. VanLehn, ed., *Architectures for Intelligence: The Twenty-Second Carnegie Symposium on Cognition* (Lawrence Erlbaum, Hillsdale, NJ, 1991) 357–424.

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

[11] W.J. Clancey, Model construction operators, *Artif. Intell.* **53** (1) (1992) 1–115.

[12] W.J. Clancey, Comments on "Epistemology of a rule-based expert system", *Artif. Intell.* **59** (1993) (this volume).

[13] T. Gruber, Automated knowledge acquisition for strategic knowledge, *Mach. Learn.* **4** (3–4) (1989) 293–336.

[14] B. Hayes-Roth, M. Hewitt, M. Vaughn Johnson and A. Garvey, ACCORD: a framework for a class of design tasks, KSL Tech. Report 88-19, Computer Science Department, Stanford University, Stanford, CA (1988).

[15] J. Lave, *Cognition in Practice* (Cambridge University Press, Cambridge, England, 1988).

[16] J. McDermott, Preliminary steps toward a taxonomy of problem-solving methods, in: S. Marcus, ed., *Automating Knowledge Acquisition for Expert Systems* (Kluwer Academic Publishers, Boston, MA, 1988) 225–256.

[17] M.A. Musen, Automated support for building and extending expert models, *Mach. Learn.* **4** (3–4) (1989) 347–375.

[18] A. Newell, The knowledge level, *Artif. Intell.* **18** (1) (1982) 87–127.

[19] D.A. Schön, Generative metaphor: a perspective on problem-setting in social policy, in: A. Ortony, ed., *Metaphor and Thought* (Cambridge University Press, Cambridge, England, 1979) 254–283.

[20] L. Steels, Second generation expert systems, *Future Gen. Comput. Syst.* **1** (4) (1985) 213–221.

[21] M. Stefik, *Introduction to Knowledge Systems* (Morgan Kaufmann, San Mateo, CA, to appear).

[22] T. Winograd and F. Flores, *Understanding Computers and Cognition: A New Foundation for Design* (Ablex, Norwood, NJ, 1986).