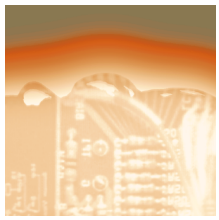
CYC: A Large-Scale Investment in Knowledge Infrastructure

Douglas B. Lenat



By codifying reams of commonsense knowledge, CYC automates the white space in documents to help standardize—and make more efficient—information retrieval, integration, and consistency checking.

Since 1984, a person-century of effort has gone into building CYC, a universal schema of roughly 10^5 general concepts spanning human reality. Most of the time has been spent codifying knowledge about these concepts; approximately 10^6 commonsense axioms have been handcrafted for and entered into CYC's knowledge base, and millions more have been inferred and cached by CYC. This article examines the fundamental assumptions of doing such a large-scale project, reviews the technical lessons learned by the developers, and surveys the range of applications that are or soon will be enabled by the technology.

One can think of CYC as an expert system with a domain that spans all everyday objects and actions. For example:

- You have to be awake to eat.
- You can usually see people's noses, but not their hearts.
- Given two professions, either one is a specialization of the other or else they are likely to be independent of one another.
- You cannot remember events that have not happened yet.
- If you cut a lump of peanut butter in half, each half is also a lump of peanut butter; but if you cut a table in half, neither half is a table.

Such assertions are unlikely to be published in textbooks, dictionaries, magazines, or encyclopedias, even those designed for children. These assertions, and a million more like them, embody knowledge the CYC authors safely assume is already known about the world. These assertions are so fundamental that stating them to another person, aloud or in print, would likely be confusing or insulting.

Such a commonsense substrate could serve as a standard ontology underlying the World-Wide Web and electronic commerce. As a universal schema, it could help standardize—and make more efficient—information

retrieval, integration, and consistency-checking. Prototype CYC-enabled applications of these operations are working; they and others are discussed in this article's Commercial Applications section.

To build CYC, we eschewed such free-lunch tactics as natural language understanding (NLU) and automatic machine learning (ML). We figured that progress in these two areas would generate demand for a system much like CYC is meant to be. For example, how can one determine the antecedent of "they" in "The police arrested the demonstrators because they feared violence" versus "The police arrested the demonstrators because they advocated violence"? How can one tell which meaning of "pen" is intended in "The box is in the pen" and "The pen is in the box"? How can one tell that, when translating the following sentence into Japanese, the first "water" is cold water and the second is hot water: "Mary poured the water into the teakettle; when it whistled, she poured the water into her teacup"? (Japanese does not provide a single word for liquid water.)

Moreover, statistics, colocation, and frequency do not resolve such questions. But the task goes from impossible to trivial if one already knows a few things about boxes and pens, police and demonstrators, and water and teakettles. The same sort of chicken-and-egg relationship characterizes CYC and ML because learning occurs at the fringe of what one already knows.

Therefore, in the early 1980s, when the rest of the world was so enthusiastic about NLU, ML, and AI in general, we were pessimistic [2]. We concluded the only way out of this codependency would be to prime the pump by manually crafting a million axioms covering an appreciable fraction of the required knowledge. That knowledge would serve as a critical mass, enabling further knowledge collection through NLU and ML, beginning in the mid-1990s. Mary Shepherd and I embarked on that task in 1984, knowing we had little chance of success, but seeing no alternative but to try.

We knew codifying common sense meant handling causality, time, space, substances, intention, contradiction, uncertainty, belief, emotions, planning, and so on. More precisely, we had to figure out how to represent the common cases of these phenomena, how to reason about them efficiently, and how to formulate sets of categories and attributes with which to carve up the world. Building something like CYC also forced us to develop methods and tools for browsing and editing immense knowledge bases, for keeping dozens of knowledge enterers from losing their focus, and for enabling would-be application builders to use CYC without having to spend months coming up to speed on its internal workings.

Due to the imagination and hard work of key individuals who joined us during this enterprise—Karen Pittman, R. Guha, Nick Siegel, Kathy Burns, Keith Goolsbey, Ken Murray, David Gadbois, and others—we are now moving toward the transition point where NLU and ML are supported. The rest of the world is disillusioned and pessimistic about symbolic AI, but ironically, as CYC reaches closure, our hopes for NLU and ML in the next 10 years are very high.

The following sections give some details about the technical problems that arose during CYC development and how they influenced the system's knowledge representation and reasoning. We then turn our attention to CYC applications.

Technical Lessons Learned

One thing to note about the five everyday assertions listed earlier is that they are true only as a default. In some contexts—such as during heart surgery—some of them are plain wrong. Each assertion should be considered true only in certain contexts, which are distinguished by the assumptions they make. For example, one context assumes all the people involved are more or less healthy, sane, non-babies, and sighted; that there is adequate light; and so on.

Different contexts make different assumptions and therefore involve different assertions that are true. For example, in the context of total darkness, there might be an assertion like: "You cannot see anything." This superficially contradicts the assertion about being able to see other people's noses, but there is no real contradiction. Why? One can import assertions from one context to another, but one must then collect and import assumptions not shared by the two contexts. Some context assumptions may be extreme—like a young child's model of the physical world or a fictional context in which vampires prowl the night. Some contexts may be ephemeral—like the context of a single point in a conversation when indexicals like "now" and pronouns like "it" and "he" have unambiguous meanings.

CYC puts each of its assertions into one or more explicit contexts. One can think of these as articulated plates in a suit of armor. Each of them is relatively small, solid, and flat and meets others at a small number of individually fashioned joints, but the whole suit of plate mail is strong and flexible.

Another issue is illustrated by the question: How likely is it that one can see another person's nose yet cannot see the other person's heart? It is tempting to make up numeric certainty factors (like 0.99982) for each assertion. But we do not really know these probabilities precisely, only that they are high. Builders of expert systems recognize this trap—encoding information into the fourth and fifth decimal places of numeric certainty factors to have one rule slightly override another or to have one assertion be slightly more likely than another. Since all rational

numbers are commensurable, this scheme becomes unmanageable when there are dozens of people making up numbers for hundreds of thousands of rules.

CYC therefore eschews numeric certainty factors, except in cases where statistics are known. Instead, each assertion is assumed true by default, and additional meta-level assertions might state that "Assertion A is less likely than Assertion B." Reasoning is done through argumentation, not by logic (propagating absolute True and False) nor by arithmetic (propagating and combining numeric certainty factors). Instead, pro and con arguments are marshalled and compared [1, 3].

CYC eschews certainty factors.

Instead, each assertion is assumed true by default.

Another point is that a standard sort of frame-and-slot language proved to be awkward in various contexts:

- For stating ternary and higher relations (e.g., "between" in an assertion like "Austin is between Dallas and San Antonio");
- For stating modals (e.g., "believes" and "wanted" in an assertion like "Israel believes the U.S. wanted Arafat to receive the Nobel Peace Prize");
- For stating quantifiers (e.g., "Every married person married someone");
- For stating assertions about other assertions (e.g., "Most of the rules about meningitis were entered by an expert in 1975," or "The assertion about seeing noses is more often violated than the one about seeing hearts"); and
- For explicit contextualizing (e.g., "While driving a car, eye contact is not socially required during conversations").

Such experiences caused us to move toward a more expressive language, namely first order predicate calculus with a series of second-order extensions [1, 8]. This move illustrates two important points about doing large-scale AI:

- One must not shrink from making changes, even fundamental changes, if the alternative is the sacrifice of the system's future stability and robustness.
- One should design the system, make changes, and be formal for pragmatic reasons rather than aesthetic reasons.

We entered approximately 10^6 general assertions into CYC's knowledge base, using a vocabulary with approximately 10^5 atomic terms, or basic concepts. The hundreds of long-lived contexts in the current system involve contexts most of us share and that are likely to remain useful in the system for years to come. The exact number of concepts and the exact number of assertions are not important.

Why is the precise number of assertions a red herring? CYC knows about a thousand different occupations. CYC includes an assertion about professions being more or less disconnected, plus a thousand rules of the form "Surgeons are doctors," and "Masons are builders." This is much more parsimonious than having another million rules of the form "Surgeons are rarely masons," and "Doctors are rarely plumbers." Our goal was neither to have as many axioms as possible nor as few axioms as possible, but to build CYC as rapidly as possible. This motivation caused us to lean toward the as-few-as-possible end of this spectrum, without worrying about occasional redundancy. This pragmatism is crucial to a successful large-scale AI effort.

Why is the precise number of concepts or terms a red herring? CYC has a set of functions that enable one to make assertions about non-atomic terms. For example, one can refer to LiquidForm(nitrogen) without having to create a new term like liquid-nitrogen. As with assertions, our goal is not to have as many terms as possible nor to have a provably minimal set of terms, but just enough, and no more.

There are dozens of general contexts in CYC today, including substances and events occurring in time, but almost all the day-to-day action is in creating and extending hundreds of much more specialized contexts. These new contexts deal with such diverse situations as a wedding, an office environment, a camping trip, a business meeting, driving a car, shopping in a supermarket, and the personnel department of a company. This organization is reminiscent of Schankian scripts; we imagine that if Schank had persevered with his 1970s paradigm, he would have constructed something like CYC long before we did.

Commercial Applications

This section reviews a much broader spectrum of applications—most not yet under way—than we previously reviewed in *Communications* [1].

The applications in the first group focus on information retrieval, with CYC helping in various ways. For example, consider stating a detailed user model, including assertions about the user's hobbies, job, family status, and values; areas of expertise, ignorance, interest, and disinterest; and personality. This is not a new idea, but user models are generally limited to almost trivial preference lists, filling in blanks in questionnaires. CYC can help because of its expressiveness—through both its syntax and its large universal schema of terms. It also enables the statement of rules about how these user-model state-

ments influence the choice of what to present and how to present it.

The issue of how to present it includes much more than modality and formatting, also covering the level of terseness, the level of sophistication, the adding or paring of particular background information, the sequencing of the information, the choice of what to highlight, and more. CYC should also enable more context-sensitive menuing and preemptive guessing of what the user is likely to want to see and do next.

This leads to the issue of what to present, including much more than just selecting particular articles or MOSAIC pages or messages. It incorporates the dynamic selection of specific sentence-sized pieces of information, depending on the user model. In an extreme case, one might eventually build a self-programming VCR the user would only occasionally have to reward and punish, but never have to set. Decision theory and other technologies would also be at work in such a device, but their strengths and weaknesses differ from those of CYC, and it seems likely the most powerful system will be some sort of hybrid.

CYC can be used as a substrate—a semantic backbone—for dynamically semiautomatically linking multiple heterogeneous external information sources, such as remote, third-party-maintained databases, spreadsheets, text news feeds, and more. CYC serves as a universal schema to enable such integration and as an automatic learner of the meaning of alien schemas, inducing what a particular field called INC means in a particular schema. "Semiautomatic" means it includes the possibility that the user will be asked to verify or disambiguate interpretations of a given relation, of a term in a cell, and of other parameters.

The important point is that users will be able to find information without having to be familiar with the precise way the information is stored, either through field names or by knowing which databases exist and can be tapped. For example, one could use CYC beneath a spreadsheet-to-database-like interface as follows: The user starts sketching in column headings and cell entries in an initially blank table. The system gradually fleshes out the table—automatically—as it figures out what the user means by each row and especially by each column. The system then goes off to various databases to find cell entries for the new table or, more interestingly, pieces of information from which it can infer cell entries for the new table. A prototype is in operation.

If the content of the information is understood, the information presented to the user can be more like a symbolic spreadsheet than like a static display. For example, certain words may be highlighted, and users can change the words and watch the consequences of the changes percolate through their current pageful of displayed information. What happens when the user changes a date, a person's name, or other information is that CYC goes off and infers the other changes to make. This function leads to another area of CYC application—dealing with the processing of structured data extracted from spreadsheets, databases, and other sources.

CYC technology can examine the retrieved data, recognizing inconsistencies, contradictions with specific data from other sources, and violations of common sense. For example, in one personnel table, employees might appear to be hired before they were born and might be listed as their own emergency contacts. Moreover, across tables, employees might appear to have multiple spouses at the same time, might appear not to age for five years, and might appear to be charging dinner in several different cities on the same night. A prototype of this application is in operation.

CYC should also increase the functional richness and power of word processors. For example, if a word is spelled incorrectly but the new spelling happens to be a valid word and grammatically correct, a word processor has difficulty flagging it as an error. But with CYC examining the sentences, some of these errors can be caught. For example, in the sentence "Huck thought it felt good to rest his bare toes on the warm rough wooden sock," we (and CYC) can guess that the user most likely just hit "s" instead of "d" and meant dock, not sock, despite the high statistical co-occurrence of sock and toes.

The same approach improves grammar checking—to allow a larger set of nongrammatical but comprehensible sentences. If CYC can understand what the user probably meant, so can another human.

A decade from now, CYC could help enable a whole new sort of checking—content-checking. For example, if an author promises, "Later, we will address . . ." but never delivers, the system could highlight the broken promise. If an author includes specific data in a document, presumably presented as up to date (not tagged as, say, 1992 data), the system could go out to online databases, spreadsheets, the World-Wide Web, and other sources, offering to update the article with more recent data if available and check earlier time-dated data if still available.

Another use of CYC in word processing is to offer to flesh out incomplete (even outlined) sentences and paragraphs and sections. This can be done today with legalese boilerplate, but tomorrow it may be done with common sorts of mail and email correspondence, and the day after (early next century) with much less constrained sorts of documents.

A nearer-term use of CYC would be to find and flesh

We eventually gave up completeness of inference *in return for expressiveness and efficiency.*

out incomplete bibliographic references. This ties back to the first application area—information retrieval—this time enabling creation of a semantic file system in which files are indexed not just by a few attributes and hierarchical links, but through inferences that match the file's caption to the user's current query or task.

Another major application area for CYC will be simulations. Users will demand ever greater fidelity of behavior of simulated intelligent agents, just as they have continually demanded ever greater fidelity of physical behavior of simulated tangible objects—in games, in training simulations, and in other programs. Eventually, this demand will flatten out near what a person knows and how a person or organization reacts.

In a role-playing game, the computer-run characters should not have trivially predictable daily routines or conversations limited to revealing clues to the user. Instead, they should have hobbies, jobs, social cliques, chores, memories, and lots of factual knowledge about their lives and their surroundings. They should also subtly change moods with every interchange they have with the user and with other simulated people and organizations. The knowledge we put into CYC is a necessary component of such a function; the knowledge we will add to CYC is close to a sufficient component.

Even if we focus on natural language input provided by the user, innumerable cases will be ambiguous unless the user has access to CYC's knowledge. For example, if, in an air traffic control training simulation, the user asks, "What is Flight 803's destination?" only context and common sense—or preconceived tinkering—will reveal whether the question involved today's particular airplane (just diverted to O'Hare) or the scheduled daily route for planes with this flight number. Context and common sense may be needed to solve the simpler problem of whether the answer should be the name of a runway, an airport, a city, or a country.

It is worth mentioning that CYC can help increase the accuracy of speech recognition and NLU understanding systems. For example, CYC could be applied after statistics and natural language processing have finished, performing a final sanity check on the supposedly understood spoken sentence. Can some adjectives or prepositional phrases make the utterance more sensible if they modify something other than what the parser claims they modify? Does some phonetically close word make the utterance more meaningful than the current best guess? Recall the earlier sock and dock example. Are some pronoun antecedents clear only because of common sense? Recall the earlier police and demonstrators example.

CYC can also be used to extend the functionality of email in the following way. Imagine smart routing of email based on user models, as well as on (partial) understanding of the content of the message. To speed it along, CYC could semiautomatically generate a caption for each message, and then have the user verify the caption (and edit, if necessary). The captions could also be used for retrieving relevant past messages.

CYC could also support such applications as direct marketing, smart corporate yellow pages, and machine translation [4]. In our 1994 article in *Communications*, we detailed the most well developed CYC applications, including semantic retrieval of captioned images [1].

Conclusions

By codifying commonsense knowledge, we have in effect been automating the white space in documents. We addressed this task reluctantly, however, after it was clear that AI technologies, specifically for NLU and ML, would probably never scale up. Only a large-scale paradigm-breaking effort could surmount the barriers.

Several technical problems were encountered and addressed, several applications are now under way, and many more are ripe for attack.

The system's technical history contains several hurdles we overcame. For example, all the assertions had to first be decontextualized as much as possible and dumped into a large common pool. Eventually, we gave CYC's representation language a construction for explicitly tagging each assertion with the context(s) in which it was true. Articulation axioms map between contexts, and nonshared assumptions are explicitly added to an assertion when it is imported from one context to another. Each context is a first-class CYC object, about which assertions can be made. Some of these assertions list the assumptions of the context; some list the contents of the context (assertions true in that context); and some relate one context to another.

Instead of trying to find a single, general solution for problems that have plagued AI researchers for four decades and philosophers for four millennia—including time, space, causality, modals, and substances—we sought to build a set of micro-theories that together cover the common cases of each problem. For example, several models of time and temporal reasoning are included. Each micro-theory inhabits its own context.

Originally, each assertion was given a numeric certainty factor. Eventually, we adopted a much more symbolic scheme. Now, most assertions are default-true, and there are explicit assertions like "X is more likely than Y." Each assertion is a first-class CYC object, about which other assertions can be made.

Originally, we tried to make the usual sort of tradeoff between expressiveness and efficiency through a conventional frame language. We eventually gave up completeness of inference in return for expressiveness and efficiency. The basic language is expressive

and formal, involving first-order predicate calculus plus ZF set theory, meta-level assertions, contexts, and modal operators. Also included are special-purpose representations, reasoning algorithms, and heuristics that identify situations in which such shortcuts can be used. We separated the epistemological level (what the system knows) from the heuristic level (how it efficiently reasons with and about what it knows) both conceptually and in code.

We pursued a large number of CYC applications, several demonstrable at least at the prototype stage today. Even for applications only in the planning stage, we can reel off reasons why users need commonsense knowledge not found in any dictionary, encyclopedia, newspaper, textbook, database, or other repository.

Is CYC necessary? How far would a user get with something simpler than CYC but that lacks everyday commonsense knowledge? Nobody knows; the question will be settled empirically. Our guess is most of these applications will eventually tap the synergy in a suite of sources (including neural nets and decision theory), one of which will be CYC. ◻

References

1. Guha, R. V. and Lenat, D. B. CYC: Enabling agents to work together. *Commun. ACM 37*, 7 (July 1994).
2. Lenat, D. B.and Brown, J. S. Why AM and Eurisko appear to work. J. *Artif. Intell.*, 23 (1984), 269–294.
3. Lenat, D. B. and Guha, R. V. *Building Large Knowledge Bases.* Addison-Wesley, Reading, Mass., 1990.
4. D. B. Lenat and R. V. Guha. Ideas for Applying Cyc. MCC Technical Report ACT-CYC-407-91. December, 1991.
5. Lenat, D.B., Guha, R. V., Pittman, K., Pratt, D., and Shepherd, M. CYC: Toward programs with common sense. Commun. ACM (Aug., 1990).
6. McCarthy, J. Programs with common sense. In Readings In Knowledge Representation, H. Levesque and R. Brachman, Eds. Morgan Kaufmann, Los Altos, Calif., 1986.
7. McCarthy, J. and Hayes, P.J. Some philosophical problems from the standpoint of AI. In *Readings In Nonmonotonic Reasoning*, M. Ginsberg, Ed. Morgan Kaufmann, Los Altos, Calif., 1987.
8. Pittman, K. and Lenat, D.B. Representing knowledge in CYC-9. MCC Tech. Rep. CYC-175-93P, 1993.
9. Quine, W.V. Natural kinds. In *Ontological Relativity and Other Essays.* Columbia University Press, New York, 1969.

About the Author

DOUGLAS B. LENAT is president of Cycorp and a consulting professor of computer science at Stanford University. He is working on getting machines to learn automatically and understand natural language and speech.

Author's Present Address: Cycorp, 3500 West Balcones Center Drive, Austin, TX 78759. Phone: 512-338-3436; fax: 512-338-3858.