Answers for the Written Exam of the Technical Area of my Qualifying Exam

Stefan Marti, February 14-15, 2002

Question 1

A problem with the idea of "common sense" is that it seems to cover so many subjects and situations. What is the scope of knowledge that we call “common sense” — how do you decide what's included and what's excluded? Is common sense an "AI-Complete" problem? That is, is there a way to get a computer that has a reasonable degree of common sense without having to solve the entire AI problem first?

Answer

This question consists of two parts:

- What is the scope of commonsense knowledge?
- Is the commonsense problem “AI-Complete”?

It seems that a truly unrestricted commonsense reasoning system may indeed be AI-complete. Nevertheless, the interesting question remains whether there is a commonsense reasoning system that is not perfect and still useful.

Scope of commonsense knowledge

I believe there is no clear distinction between commonsense knowledge and more specialized knowledge. It is rather a continuum along a (at least single) dimension.

First of all, let us ignore the actual transition between commonsense knowledge and more specialized knowledge, or the threshold, which may vary from individual to individual (e.g., my own father has a very wide commonsense knowledge base compared to me), from culture to culture, and certainly among different species.

Nevertheless, the answer to the question, “What is and is not commonsense knowledge?” is not obvious.

Pragmatic distinctions

Researchers who are interested in porting commonsense reasoning to machines have come up with definitions and examples for what commonsense knowledge might be. Let us see if they can clarify the dimension:

- McCarthy (1959) [8]: *A program has commonsense if it automatically deduces consequences from anything it is told or already knows.* This definition would exclude manually entered knowledge from the class of commonsense knowledge, putting focus on automatically acquired or learned knowledge.
- Lenat et al. (1990) [6]: *Commonsense knowledge is the knowledge that people intuitively use to understand the contents of an encyclopedia.* This would logically exclude all the knowledge that can be found in an encyclopedia, putting focus on the knowledge that is assumed from a reader.
- Guha et al. (1994) [5]: *Commonsense knowledge is what a high-school teacher assumes students have already when they enter the classroom. It consists of a vocabulary and knowledge about the*

vocabulary. Expected are notions about time, space, causality, human capabilities, limitations, goals, emotions, familiarity with art, literature, history, etc. Such a definition would exclude K-12 educational knowledge as well as all later acquired school and academic knowledge from being commonsense knowledge.

These examples show that the distinction between commonsense and non-commonsense knowledge depends highly on the approach to the commonsense problem, or specific focus of the researcher. However, it seems also that the kind of formalized knowledge that can be found in references (specifically encyclopedias, text books, etc.) does definitively not belong to the class of commonsense knowledge anymore.

Complexity of primitives

The continuum from commonsense knowledge to specialized knowledge can also be approached via the varying complexity of primitives (such as frames, production rules, predicates) that are used to describe the knowledge. The longer and the more complex the primitives are, the less likely they are part of an "honest" commonsense knowledge system (Lenat et al. 1990) [6].

Using too complex primitives has to be avoided when designing "true" commonsense reasoning systems. If one chooses a set of long, complex primitives that have a lot of knowledge compiled within them (e.g., the fictitious predicate LaysEggsInWaterThatIsLessThanFourInchesDeep), and writes rules that are also tailored to the program's domain (omitting premises that needn't be worried about in that particular task), then a system could be built that is useful for a special purpose, but may not be so in the long run (Lenat et al. 1990) [6]. (More about this so-called Representation Trap in the answer to question two.)

Expert systems vs. commonsense knowledge databases

Another approach to distinguish commonsense knowledge from non-commonsense knowledge is comparing classical expert systems with systems that are built specifically for commonsense reasoning.

Expert Systems have a narrow field of task and do symbol manipulation that needs to be interpreted by a user to make sense. The meaning of the terms lies in the eye of the beholder. Such a system gives the illusion that it works, but fails with unforeseen situations. Yet, expert systems help humans via "pushing tokens around," (Lenat et al. 1990) [6] but they are only meaningful to humans, not to machines.

Projects like Cyc are explicitly built to bridge the gap between specialized knowledge bases that do not interoperate easily because of incompatible assumptions and ontologies, different predicates (e.g., *Feverishness* and *BodyTemperature*). Cyc is supposed to assume the function of *semantic glue* between otherwise isolated expert systems (Lenat et al. 1990) [6].

AI completeness

Is there a way to get a computer that has a reasonable degree of common sense without having to solve the entire AI problem first?

I believe there are possibilities to make a system appear to have useful commonsense in restricted domains. However, as outlined above, finding the right balance between *usefulness* and *brittleness* is difficult. The narrower the knowledge domain and the more specialized the primitives, the more likely the system is useful in a restricted domain, especially when the output of the system is interpreted by a human, but also the more likely it will break when encountering situations which are outside its original domain.

However, there might be ways to make a commonsense reasoning system useful and still not too brittle.

User feedback

One way is to use the user's feedback to disambiguate or verify unclear situations or requests. If a moderately complex system is allowed to ask the user questions when it gets stuck, it is more likely to succeed and be useful even without being complete. For example, a user could teach the system small

pieces of commonsense knowledge with each interaction. Like that, commonsense representations can be built up slowly over time (Lieberman et al., 2000) [7].

Of course there is a trade-off for such a system: How often can the system ask the user, interrupting or even annoying her, in order to be more useful?

Combined approaches

It might be worth combining different approaches to cross-compensate for their respective weaknesses. For example, combining symbolic AI approaches (like Cyc; e.g. Lenat et al. 1990) [6] with statistical approaches (pattern recognition, data mining, etc.; e.g., Bacchus 1990) [1] and other techniques.

An example for two approaches that perhaps should be combined rather than used exclusively is shown in the ongoing debate within the AI community between those who go for neat, systematic, logical structures, which are intended to suit the demands of the computer as an information-processing device (e.g., symbolic AI, axiomatic theories, Cyc), and those who try to build knowledge representation systems which reflect the rich "messiness" of human experience and ideas. This controversy between the "neats" and the "scruffs" (first mentioned in Chapman, 1987) [3] has lead to an interesting divergence of styles of representing knowledge in computational systems. I believe trying to combine them is a better solution than just going for one or the other.

We still seem to be far from a complete commonsense reasoning system. The most likely scenario in which we may succeed is a combination of different methods. Mueller (1998) [9], for example, suggests bootstrapping AI through an iterative process: combining techniques at multiple levels by entering manually symbolic rules, by writing programs to discover rules automatically, and by augmenting automatically generated rules with manually entered ones.

Question 2

A difference between computer reasoning and human reasoning is that the more humans learn, the faster they tend to be at solving problems. On the other hand, the more facts and inference rules you give a computer, the more it tends to slow down by searching through and considering all the possibilities. How do we resolve this dilemma?

Answer

Short answer

- Restrict the search space via contexts. Basically, wrap the assertions in context and take only the ones in account that are relevant in a certain context
- Parallel computing architectures may help
- Generally good ontological engineering will help

Longer answer

Psychologically, acquiring new knowledge seems to happen on the verge of already known knowledge. The more somebody knows, the more easily she can absorb new knowledge. The more knowledge there is, the easier it is for a human to "attach" new knowledge to already known knowledge[1]. That is also the logical explanation why humans learn faster the more they know, and why their cognitive performance does not slow down with increased knowledge.

However, humans have a very unique "hardware" (brain) that seems to scale for reasoning performance very well. It is not clear if there is a theoretical limit of how much humans can know, but if there is a limit, we certainly seem to very far from that. (However, we are limited in how much we can *learn* in a given life span.)

But today's computers are obviously not built like human brains.

Parallel architecture

One of the significant differences between a brain and a contemporary computer is the underlying architecture: Massively parallel but slow processing vs. serial but very fast processing. The brain's parallel processing architecture seems to allow for fast responses even though the "clock speed" of neurons is rather low compared to today's computers. But a highly parallel architecture still does not explain how humans would *use* such a parallel infrastructure (e.g., Rumelhart et al., 1987) [10].

As mentioned earlier in my answer to question one, it is not clear whether just trying to imitate the "human infrastructure" would help us solve the problem. Computers are very different from brains, so some people may argue that we shouldn't even try to imitate the brain, but that's a whole different issue.

Context

The most important way to avoid the dilemma between increased knowledge base size and decreased inference performance seems to be related to *reducing the search tree*. In other words, putting assertions in *contexts* reduces the need to go through all possible assertions.

In Cyc, for example, "if you're trying to talk about the weather today, you don't want Cyc worrying about whether spiders have eight legs." (Lenat in Thompson, 2001) [12] As a solution, the knowledge enterers of Cyc have created contexts, clumpings of like-minded facts that help speed up inferencing. "If Cyc is told a fact about tour trips to Greece, for example, it begins with its existing knowledge about Europe, travel, trains, and the like. In this sense, Cyc's processing strategies are akin to human cognition; we can discuss any given topic only by ignoring 99.999 per cent of our knowledge" (ibid.)

Unfortunately, it is not that simple. The computer still has to find the 0.001 per cent of the assertions that are relevant. But that search is rather trivial compared to the exponentional problem when looking at the possible combinations of assertions and reasoning chains.

Semantic convergence

In a wider sense, this question asks how we can make sure that the phenomenon of *semantic convergence* occurs (Lenat et al., 1990) [6]. If a growing knowledge base is rather converging than diverging, we are probably on the right path to avoid the dilemma.

In the case of Cyc, Guha et al. (1994) [5] claim that Cyc's consensus reality seems to be converging, because of the following indications:

---

[1] More precisely: According to Vygotsky (1978) [9], at any given point in cognitive development there are certain problems that children are on the verge of being able to solve. While a child can solve some of these independently, others are outside the learner's capabilities and can only be solved under teacher guidance or in collaboration with a more advanced peer. At this point, the child is working in the *zone of proximal development*.

1. Basic topics like time, substances, etc. are stable over the last few years, meaning, they did not had to be revisited and redone over and over.
2. Topics use axioms and vocabulary of other topics. If this had not happened, Cyc would be just an agglomeration of many small knowledge bases.
3. It is not getting more and more difficult to enter knowledge.
4. Adding new features to Cyc was not necessary recently.

Cyc's knowledge enters have not only added a huge amount of assertions over the last decades, they have also consolidated them, which leads to less and less assertions, but better ones. This boils down to the problem that we just need the *right primitives* to keep the database small. In other words, we need to do good ontological engineering (Lenat et al., 1990) [6].

Beside having a good global ontology of human knowledge, one of the main things to watch out in ontological engineering is to avoid the *Representation Trap* (Lenat et al., 1990) [6]: Choosing a set of long, complex primitives (predicate names) that have a lot of knowledge compiled within them, and writing rules that are also tailored to the program's domain (omitting premises that needn't be worried about in that particular task). Reasoning seems to become trivial if we choose *task specific primitives*. However, this will not work in the long run, because we "cheat" by using long predicate names. Although the system seems to show commonsense, it cannot answer questions about the given situation. An example would be a script that describes how to drive a car. Although it is tempting to describe how to drive a specific car, we have to find the underlying element of "car driving," and then lift these assertions to a specific car later. Basically, we need to relate things by virtue of the relations between their constituents (Lenat et al., 1990) [6].

Question 3

How does designing for small and mobile devices affect the amount and kind of intelligence appropriate for a system? On one hand, a small device might be limited in the amount of memory or computing time it has. On the other hand, the inability to type and varied circumstances in which the device might be used might call for more intelligence or context sensitivity.

Answer

It is certainly true that mobile devices need more context sensitivity than any other stationary device because the people carrying them traverse many different kinds of social and environmental contexts, and the device and its interface ideally should adapt to these varying context situations.

However, I personally do not believe that the small form factor of such mobile devices will pose a problem, as I see many ways to avoid resource limitations for mobile devices via appropriate architecture. I will present a few ideas in the following sections.

"Commonsense chip" and "commonsense module"

My first idea to tackle this problem is to suggest a module (software) or component (hardware) that can be built into any device so that it "gets" human-level commonsense. It could be implemented as a network connection to a remote server that "gets" commonsense, or a stand-alone chip that has the condensed

content of such a server. Whatever the application or appliance is doing, there would be always this commonsense chip helping out, checking for actions and reactions that would not conform to common commonsense.

This notion of "putting commonsense into a chip that can be plugged into anything" is of course naïve (as naïve as an "emotion chip," another common theme in science fiction). Many researchers, especially those close to Cyc, have come to the conclusion that there does not seem to be a way around manually entering a big part of our "consensus reality" into a system. There seems to be no elegant shortcut, no secret logical or mathematical formula for commonsense, no unifying *Maxwell Equations of Thought*, but the need for a large database and some considerable amount of computing power.

It seems to be difficult to condense such a system into a single chip. One factor that might enable such a task eventually is to build a chip that is hardware-optimized for this commonsense reasoning task. That is not going to happen anytime soon, given that there is no consensus over how much consensus reality is actually needed, how to organize it efficiently, etc.

Context and Embodiment

Especially in the mobile communication setting, it is very important that commonsense reasoning happens in the appropriate *context*, since the user is per definition mobile, and most likely switching among different social and environmental context situations continuously. Sensing this context is a very difficult and tedious task. Therefore, the real problem is most likely not the limitedness of computing resources, but how to sense context information to make commonsense reasoning useful in the first place (e.g., Lieberman et al. 2000 [7], Dey 2001 [4]).

However, an isolated "commonsense chip" cannot be useful. In order to do relevant reasoning, it needs information from the outside world, both from the real world and cyberspace. It needs sensors for both physical and logical information, e.g., its location, the topography of its immediate surroundings, network connections to data which is only electronically available like databases, and of course a generic internet and WWW access. Such an "embodiment" (e.g., Shanahan et al., 1999) [11] in both the real and electronic universe seems to be specifically necessary for mobile devices in order to do useful reasoning.

Modularity

If the part of a device that is responsible for commonsense reasoning would have standardized interfaces, both physically and logically, it could be built to be modular. It could be implemented as a basic commonsense chip, enhanced with domain specific knowledge depending on what the application or appliance is supposed to do, or with the ability to connect dynamically to more specialized knowledge bases, depending on the current social and environmental context of the user of the device.

Such a modular approach would require fewer resources on the mobile device itself, but it would complicate the architecture and require a (wireless) network connection.

In a wider context, modularity would lead to *decentralized models of commonsense*, one of the ideas behind the semantic web, an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation (e.g., Berners-Lee 2001) [2].

Precompiled knowledge

There is another approach that could be helpful specifically for the setting of small devices. If the resources are limited, a system architecture could be useful that has *precompiled knowledge*—e.g., the most appropriate scripts for a given task such as human telecommunication, but still can connect to a remote commonsense server when its internal precompiled knowledge breaks. Such an architecture would be similar to how humans seem to work. Most of the time, we use compiled experiences for cognitive economy; stereotypes, schemes, and scripts guide our behavior and thinking. This allows us to respond somewhat mechanically to standard situations, using our previously acquired abstract black-box models. However, if problems occur, we are able to open the black box and reason on a lower and more detailed level, and so on until we have a handle on the problem. Being able to connect to a remote commonsense

server would allow the device to "open the black-box" that just broke and use more general knowledge to fall back on (Lenat et al., 1990) [6].

Given these options, I believe that there are no specific limitations for designing for small mobile devices *if* there is at least a network connection available. I believe one of the most useful applications for web enabled systems like Opencyc[2] and Openmind[3] should be to enable remote mobile devices with low computing resources to log into a more powerful server and retrieve relevant commonsense information.

References

[1] Bacchus, F. (1990). *Representing and Reasoning with Probabilistic Knowledge*. Cambridge, MA: MIT Press.

[2] Berners-Lee, T., Hendler, J. and Lassila, O. (2001). *The Semantic Web.* Scientific American Volume 284, Number 5, May 2001, pp. 34-43. http://www.scientificamerican.com/2001/0501issue/0501berners-lee.html

[3] Chapman, D. (1987). *Planning or Conjunctive Goals*. Artificial Intelligence 32:333-377.

[4] Dey, A.K. (2001). Understanding and Using Context. Personal and Ubiquitous Computing, Special issue on Situated Interaction and Ubiquitous Computing, 5(1). http://www.cc.gatech.edu/fce/ctk/pubs/PeTe5-1.pdf

[5] Guha, R.V. and Lenat, D.B. (1994). *Enabling agents to work together*. Communications of the ACM, 37(7):127-142. http://www.acm.org/pubs/citations/journals/cacm/1994-37-7/p126-guha/ http://www.acm.org/pubs/articles/journals/cacm/1994-37-7/p126-guha/p126-guha.pdf

[6] Lenat, D.B. and Guha, R.V. (1990). *Building Large Knowledge-Based Systems: Representations and Inference in the CYC Project*. Reading, MA: Addison Wesley. http://www.amazon.com/exec/obidos/ISBN%3D0201517523/002-3739191-0608065 http://www.cc.gatech.edu/~jimmyd/summaries/lenat1990-1.html

[7] Lieberman, H. and Selker, T. (2000). *Out of context: Computer systems that adapt to, and learn from, context.* IBM Systems Journal Vol. 39, Nos. 3 & 4, pp. 617-632. http://www.research.ibm.com/journal/sj/393/part1/lieberman.pdf

[8] McCarthy, J. (1959). *Programs with Common Sense*. In Mechanisation of Thought Processes, Proceedings of the Symposium of the National Physics Laboratory, London, U.K.: Her Majesty's Stationery Office, pp. 77-84. http://www-formal.stanford.edu/jmc/mcc59.pdf http://www-formal.stanford.edu/jmc/mcc59/mcc59.html

[9] Mueller, Erik T. (1998). *Natural language processing with ThoughtTreasure.* New York, NY: Signiform. http://www.signiform.com/tt/book/

[10] Rumelhart, D. E., McClelland, J. L., and PDP Research Group (Eds.). (1986). *Parallel distributed processing: Explorations in the microstructure of cognition* (Volumes 1 and 2). Cambridge, MA: MIT Press.

[11] Shanahan, M.P. (1999). What Sort of Computation Mediates Best between Perception and Action? In H.Levesque and F.Pirri (eds.) *Logical Foundations for Cognitive Agents: Contributions in Honor of Ray Reiter*, Berlin: Springer-Verlag, pp. 352-369.

[12] Thompson, C. (2001). *The Know-It-All Machine*. Lingua Franca, Volume 11, No. 6, September 2001. http://www.linguafranca.com/print/0109/cover.html

[13] Vygotsky, L. S. (1978). *Mind in Society. The Development of Higher Psychological Processes.* Cambridge, MA: Harvard University Press.

[2] http://www.opencyc.org/

[3] http://commonsense.media.mit.edu/