Do Computers Need Common Sense?

Matthew L. Ginsberg
CIRL
1269 University of Oregon
Eugene, OR 97403-1269

Abstract

My aim in this paper is to make and defend three claims. First, it is incumbent on the knowledge representation and nonmonotonic communities to demonstrate that their ideas will eventually lead to improvements in the performance of implemented systems. Second, a reasonable working definition of "commonsense" reasoning is that it is the process of using polynomial techniques to convert a large instance of an NP-hard problem to a smaller instance on which search techniques can be applied effectively. And finally, it is a consequence of these first two claims that the most pressing problem facing the commonsense community is the identification of realistic problems and problem structures for which commonsense reductions are both necessary and effective.

1 INTRODUCTION

One might study formal aspects of knowledge representation for at least two reasons. In the first case, the elegance of the associated theories might itself be compelling; formal theories of reasoning might attract interest in the same way that number theory might enthrall a French jurist.

The other potential justification is more pragmatic. Knowledge representation in general, and nonmonotonic reasoning in particular, many of us have argued, will eventually be good for something. We remain in some disagreement with regard to what, and our critics (this author included) are becoming increasingly harsh over our failure to deliver the promised goods.

While the rest of artificial intelligence was similarly lacking, these criticisms could be deferred. Recently, however, the rest of AI has begun to pick up the pace: In areas as diverse as game playing, manufacturing scheduling, and natural language processing, results of practical importance are beginning to appear. In all of these cases, programs' successes have been a consequence not of insight and commonsense understanding, but of raw computational power. Commonsense is our human way of muddling through situations that are too complex for us to deal with exactly; the general impression from the above results is that machines may not need to do this.

At a fundamental level, however, this must be wrong: Interesting problems still appear to be NP-hard or worse, and the fact that useful approximating techniques have been found for vertical applications does not appear to mean that similar methods will work for broader-based projects. Machines may need common sense for the same reason we do: It is the only effective approach on problems of interesting size.

At the moment, however, the empirical evidence is against this. It is no longer appropriate for the knowledge representation reasoning community to simply assume that commonsense reasoning is a necessity and proceed with business as usual. Instead, we need to develop a coherent theory of application areas in which nonmonotonic and commonsense capabilities will lead to practical payoffs.

Let me begin, however, by discussing three specific focuses of AI research and potential applications: game-playing, scheduling, and planning. In Section 5, I will try to draw some common observations from these domains and suggest the impact these observations should have on future work.

2 GAME PLAYING

Machine game-playing programs are getting better. What's more, they are progressing not by imitating human methods for solving problems, but by exploiting the vast computational power that is available on current platforms.

Consider chess. A human player will consider some tens (or perhaps hundreds) of positions when selecting a move in a particular position. A machine, on the other hand, will consider millions. My paper on February 12, 1996 quoted the Associated Press report on the match between Kasparov and Deep Blue: "Deep Blue [is] an IBM computer that has a calculating capacity of 200 million moves a second."

There have been attempts to produce computer chess players that play the game in a more human-like way, such as Wilkins' PARADISE system [26]. Unfortunately, the performance of these systems is at best modest. Simon draws similar conclusions from this example, saying, "It could be that, because of the radical differences between electronic devices and brains, programs designed to be efficient [intelligent programs] would be totally different in architecture and process from systems designed to simulate human thinking [23, p. 121–122]."[1] Other games are similar; in checkers [21] and Othello [18], for example, machine performance (now exceeding human in both domains) is a consequence primarily of raw speed.

More recently, games of imperfect information have begun to have a similar feel. Koller and Pfeffer suggest that it will be possible to use novel pruning techniques to solve poker exactly, presenting preliminary results for an 8-card deck [16].

Bridge is another game of imperfect information that has thus far eluded competent computer play. But here, too, the power of exhaustive methods is beginning to be felt. Smith and Nau remark that, "Some programs ... do decision making by repeatedly generating random hypotheses for what hands the opponents might have, and doing a full game tree-search for each hypothesized hand – but this approach is feasible only late in the game, after most of the tricks have been played [24]." I have made a similar suggestion that Monte Carlo methods can be effectively combined with exhaustive search [11].

The time needed to search the full game tree in bridge has fallen dramatically, due in part to increasing speed in hardware and in part to the discovery of new pruning methods [12]. A recently implemented system plays full hands of bridge in tournament-acceptable times using the method described in the previous paragraph; the performance of this system is far superior to any of its predecessors [10].

In all of these cases, Simon's speculation has proven correct. The strongest machine players do indeed bear little resemblance to their human counterparts. The only games for which this observation is not currently true appear to be backgammon (Tesauro's *TD-Gammon* program is neural-net based and appears, at least on a surface level, to have a human-like architecture [25]) and Go, for which no competent computer program exists at all.

3 SCHEDULING

On a different note, consider the domain of job shop scheduling. In a problem of this class, there are a variety of tasks that need to be completed, each of which uses a specific type of machine and lasts for a given duration. Given that the number of machines of each type is limited and that some tasks must be finished before others can start, the goal is to produce a schedule that completes all of the tasks as quickly as possible.

There are many examples of problems of this type; typical is the construction of artifacts such as aircraft. Indeed, a large problem that is related to aircraft manufacture specifically has been made available over the web (http://www.NeoSoft.com/~benchmrx). This problem involves 575 tasks competing for 17 resources, with each task using multiple resources.

Work on solving problems of this type automatically has progressed through three distinct phases. In the first, expert systems technology was used to capture human expertise in finding effective solutions. Typical here is Fox's ISIS system [7].

In the second phase, more general search tools were used, but human intuition was still exploited to guide and focus the search. Typical here is Sadeh's work [27]; the search engines themselves exploit intuitive notions such as "bottlenecks" that are specific to job shop scheduling problems.

Current research simply applies the fastest search tools available, typically combining both systematic and local search methods in a fairly generic way. The best known solutions on the benchmark problems mentioned above, for example, combine the systematic technique of limited discrepancy search [14] and a local

[1] It is not clear, however, how sanguine Simon is about accepting this conclusion. He says earlier, "It would be of enormous interest today to know what knowledge, how organized, would be required for a chess program to play at grandmaster level without needing to search more (100 branches?) than a human grandmaster [23, p. 103–104]." Once again, the assumption is made that it is at least *possible* to get machines to behave like people.

optimization technique known as "schedule packing."[2]

In all cases, performance has improved as the search techniques have come to rely more on speed and less on human insight. Sadeh's methods outperform Fox's on academic problems involving 25 jobs using 5 machines; these problems themselves are no longer challenging for more modern methods [5]. Our earlier observations remain valid: Effective machine methods and effective human methods are virtually unrelated.

Other areas of AI are similar. There has been a great deal of recent progress in natural language processing, resulting in substantially greater accuracy in automated parsing of large corpora that are used as case studies by this community. Almost all of this progress has been a consequence of the community's recent focus on purely statistical techniques [3].

4 PLANNING

Finally, consider generative planning. This area is somewhat different from the preceding two in that little practical progress appears to have been made, but the lessons to be learned are unchanged.

In a typical generative planning problem, one is given a situation in which a handful of blocks has been scattered on a table and the goal is to build a tower of some kind. The problem is to construct a sequence of block movements that either achieves the goal, or does so in minimal time.

From a pragmatic point of view, these problems are best solved using simple state-space search. Start with the initial situation, and work forward. Simple pruning heuristics can be constructed that allow this method to quickly solve problems involving some 40 blocks.[3]

From an academic point of view, however, a completely different approach is used. Actions are selected that achieve each of the goals, and an attempt is then made to sequence the actions in a way that avoids possible conflicts in which a block is placed on top of another that is about to be moved, and so on. The interest in this kind of approach appears to be a consequence of the fact that backward reasoning is a much more "humanlike" planning mechanism than is state-space search.

As usual, there is an intuition that there is some potential computational payoff to the backward reasoning, action-based approach. Perhaps by selecting an action to achieve a goal in general as opposed to in a specific situation, the resulting planner will be more robust against changes in the domain or other difficulties. In actuality, however, these arguments have yet to yield effective implemented systems – a generative planner constructed in this fashion is typically incapable of dealing with more than seven or eight blocks. Penberthy and Weld's UCPOP system [20], in fact, cannot reliably handle more than four without the inclusion of domain-specific control information.[4]

Interestingly enough, it appears that a continuum is emerging in the planning community, with pure "generative" planners at one end and state-based planners on the other. This work began with Blum and Furst's GRAPHPLAN system and their observation [2] that the performance of generative planners could be improved if one committed to specific times at which the selected actions would be taken. Kautz and Selman took this idea further, extrapolating away the actions as well and producing an axiomatization that simply constructed potential states from their predecessors [15]. The axiomatization was then passed to one of a variety of propositional theorem provers, and the resulting system proved still more effective than GRAPHPLAN, solving problems involving 17 blocks. The axiomatization itself, however, is strikingly similar to a propositional encoding of the associated state-space search problem; as such, the Kautz/Selman planner is itself outperformed by dedicated state-space engines.

The range of tools available here raises the question of which of these implementations can actually be considered "planners." Massey's blocks-world planning program, incapable of solving any problem in any other domain, is clearly not a planner. UCPOP clearly is. Kautz and Selman's work, which contains no action descriptions and relies on the encoding being propositional instead of first-order, seems somehow intermediate with regard to this metric. The Kautz/Selman "planner" cannot be extended to either use a first-order axiomatization or to manipulate action descriptions without substantial computational cost.

In fact, the inference engine underlying the Kautz-Selman planner is WSAT [22], a fast tool for finding models of satisfiable propositional theories in which each clause consists of three randomly selected literals. WSAT's performance appears to degrade when it is applied to realistic problems with more structure [5, 17]. McAllester and I have explained this by suggesting that:

[2] Barry Fox and James Crawford, personal communication.

[3] Bart Massey, personal communication.

[4] Will Harvey, personal communication; also cited elsewhere [9].

> The most difficult random problems appear to be tangles of closely related individual variables while naturally occurring problems tend to be tangles of sequences of related variables. [13, p. 227]

WSAT and related algorithms appear to be strongest at solving problems involving "puzzle mode" reasoning (tangles of closed related variables) instead of "commonsense" reasoning (tangles of *sequences* of related variables). Problems in commonsense reasoning do indeed seem to involve lengthy partial analyses in which the branching factor is low. Once you make the difficult decision of deciding what to buy a friend for Christmas, there is typically a lengthy but far less difficult process involved in actually making the purchase.

The reason WSAT can be made to function effectively as a planner is because Kautz and Selman have managed to find a propositional encoding of the blocks world that highlights its puzzle mode nature; this encoding has extrapolated away all of the domain's commonsense features. The fact that they have removed the actions is no accident – the actions were, in some sense, the last vestige of commonsense reasoning in the blocks world. By using an axiomatization that avoids the inclusion of actions and refers to the state space alone, Kautz and Selman retain only the puzzle mode aspects of the problem.

Kautz and Selman appear to argue that their WSAT encoding is still a planner because it is possible, generally in linear time, to recover the plan from the state sequence they produce. In their words, "Each pair of adjacent states corresponds to the (easy) problem of finding a partially-ordered plan of length 1." But the arguments of the previous paragraph are exactly that what distinguishes commonsense from puzzle mode reasoning is little more than an additional polynomial complexity at each node. Kautz and Selman have essentially removed the commonsense aspect of the planning problem. In doing so, it is not clear to me that their results embody a more compelling message than the fact that planners working in puzzle mode (i.e., using state-space search) will outperform planners working with more commonsense representations. A dedicated blocks-world planner such as Massey's makes this point even more compellingly.

5 COMMONSENSE REASONING

What does all of this say about commonsense reasoning? The observations that we can draw from the preceding sections are completely uniform: Machines are far better than we are at puzzle mode reasoning, and we are far better at common sense. Furthermore, it often appears to be possible to encode problems involving common sense in a way that allows them to be solved using the machine's substantial puzzle-mode capabilities. Let me in fact take this observation one step further, and elevate it to the status of a definition:

Definition 5.1 Commonsense reasoning *is the technique whereby an inference problem is solved by reducing it (perhaps unsoundly) to a modified NP-hard problem on which exhaustive search is practical.*

My aim in making this definition is to say something sharp enough that instances of commonsense can be identified in a reasonably objective way, as opposed to by an appeal to an intuitive but otherwise unspecified standard. As with any attempt to define a previously unformalized notion, the justification for the definition should be that it does indeed manage to capture a wide range of examples of the phenomenon in question. Before moving forward, let me examine some instances of commonsense reasoning and see how they fit in this framework.

Consider, for example, the commonsense observation that the names of the individuals crossing the river are irrelevant in the missionaries and cannibals problem. Naming the missionaries Jimmy, Billy and Jerry (to pick three completely unmotivated choices) serves merely to increase the size of the search space without changing its structure in any interesting way. By recognizing that the names are not important to the solution of the problem, we reduce the problem to one that we can solve using exhaustive search.

The mutilated checkerboard problem is similar; the commonsense observation that one should consider only the colors of the covered squares corresponds, as in the previous example, to a reformulation for which exhaustive search becomes practical.

Other examples of commonsense reasoning are nonmonotonic. Consider the recognition that the best way to get to the grocery store is to drive, assuming that there is still gas in the car, that it's still parked where you left it, that the supermarket hasn't moved either, and so on. Without these assumptions, the problem of getting one's groceries would simply be insolubly difficult. As I have suggested elsewhere, the purpose of the nonmonotonic inference rules is to focus our attention on aspects of the problem that render it tractable [8].

Other examples of nonmonotonic reasoning are similar. In McCarthy's typology of uses of nonmonotonic reasoning, for example [19], he mentions the use of nonmonotonicities as "rules of conjecture." Of the other uses he proposes, perhaps the most awkward for our definition is the nonmonotonic nature of communication: If I tell you about a particular bird Tweety, and I know that Tweety can't fly, I am expected to inform you so explicitly. Absent such information, I can be taken to have sanctioned your likely conclusion that Tweety can fly after all.

Even here, though, there are computational overtones. If we are discussing building a cage for Tweety, the communication convention leads to a common focus on the problem of Tweety's possibly flying away. To the extent that communication is a mechanism for arriving at shared solutions to common problems, the communicative goals of nonmonotonicity are to guarantee a match in the computational (or perhaps other) assumptions made by multiple agents solving similar problems.

Finally, consider Kautz and Selman's "compiling away" of the commonsense elements of blocks-world planning. Our definition suggests that in this case, it is Kautz and Selman who are solving the commonsense aspects of the problem; their "planner" is solving the puzzle-mode kernel of the problem instead of the problem itself. This explains both the computational success of their method and the fact that they have not yet managed to automated the process by which a conventional description of a domain in terms of actions and fluents can be changed into one suitable for their planner. The fact that their planner is so incapable of dealing with the commonsense aspects of planning also explains the cognitive dissonance in calling it a "planner" at all.

Note, however, that Definition 5.1 views commonsense reasoning as fundamentally a heuristic activity, as opposed to an epistemological one. Although in keeping with my understanding of nonmonotonic reasoning [8] and with work such as Kautz and Selman's, this focus is very much at odds with the conventional view in the knowledge representation community. Nevertheless, I would argue that the definition works: Most if not all of what we refer to as common sense is covered, and little else is.

Given Definition 5.1, the work on scheduling and on game playing supports the view that automated commonsense reasoning is unnecessary because machines will simply deal with large NP-hard problems in their entirety. The commonsense portion of the reasoning exploits human insights regarding the nature of the search space. In scheduling, for example, progress in AI accelerated when we adopted the operations research community's idea that the branch points in the search space should not be the times at which various tasks are executed, but the selection of jobs that should be given priority when there is a resource conflict. In game playing, there is a long history of algorithmic developments, ranging from alpha-beta pruning to more recent work on singular extensions [1]. This last idea, for example, captures our commonsense intuition that forced moves should not contribute to the depth limit to which a machine searches. Alpha-beta pruning captures the fact that moves provably off the main line need not be considered.

In all of these cases, human insight is applied globally, transforming a problem in game playing, scheduling or planning into a modified form for which brute-force search is effective. The uniform success of this methodology supports the view that human and machine abilities simply differ. Perhaps we are meant to complement one another, with the humans reducing problems requiring common sense to problems that don't. Our job is to translate the problems into a form with which machines can cope effectively.

A more optimistic tack, however, and certainly the one best in keeping with AI's overall goals, is that machines will need common sense themselves. Such a need has been presupposed by the knowledge representation and reasoning community, and the time has come to justify that presupposition. The most compelling justification would involve the construction of algorithms that do just this, automatically modifying large and (apparently) intractable problems in ways that allowed them to be solved effectively.

This may be too hard, however. Only slightly less compelling would be the identification of problem types for which commonsense reductions were necessary. What structure can a problem have that will make an effective reduction possible while the problem itself becomes intractable for blind search methods? What evidence is there that such structure is present in naturally occurring problems? Is the structure related to our earlier quoted observation that "difficult random problems appear to be tangles of closely related individual variables while naturally occurring problems tend to be tangles of sequences of related variables?"

As stated, this is not an experimental question. In fact, much of the experimental methodology that has been proposed in AI recently is almost exactly off target here. Consider Cohen's "modeling, analysis and design" proposal for AI research in general, where the thrust of any particular piece of work is expected to be a theoretical or (more likely) experimental investigation into the performance of an implemented system as a parameter is varied continuously [4]. Global questions involving the purely theoretical identification of situations or problems for which classes of algorithms are well suited are outside the scope of this "MAD" methodology (the acronym is Cohen's).

Other experimental work has focused almost exclusively on randomly generated problems which are designed specifically so as *not* to have the kind of structure that might support reasoning mechanisms other than puzzle-mode. This work has been an invaluable source of insight with regard to puzzle-mode reasoning itself, but to believe that it can lead to similar insights when applied to problems of common sense seems to me to miss the mark completely. As such, I am distressed to see recent work such as that of Etherington and Crawford [6] proposing to investigate experimentally the viability of nonmonotonic techniques in a random setting. If our goal is to identify structures for which commonsense reductions are a necessity, randomly generated problems are exactly the *wrong* place in which to look.

At its heart, of course, the problem of finding structures that support or require some kind of commonsense reasoning is a theoretical question as opposed to an experimental one. It is unfortunate that it is an issue that has thus far received so little attention from the AI subcommunity that is both the best placed to solve it and that has the most to gain from doing so.

Acknowledgement

This work has been supported by ARPA/Rome Labs under contracts F30602-91-C-0036 and F30602-93-C-00031. I would like to thank David Etherington and the members of CIRL for discussing these ideas with me.

References

[1] T. Anantharaman, M. S. Campbell, and F. hsiung Hsu. Singular extensions: Adding selectivity to brute-force searching. *Artificial Intelligence*, 43:99–109, 1990.

[2] A. L. Blum and M. L. Furst. Fast planning through planning graph analysis. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 1636–1642, 1995.

[3] E. Charniak. *Statistical Language Learning*. MIT Press, Cambridge, MA, 1993.

[4] P. R. Cohen. A survey of the eighth national conference of artificial intelligence: Pulling together or pulling apart? *AI Magazine*, 12(1):16–41, 1991.

[5] J. M. Crawford and A. B. Baker. Experimental results on the application of satisfiability algorithms to scheduling problems. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, 1994.

[6] D. W. Etherington and J. M. Crawford. Toward efficient default reasoning. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 1996. Submitted.

[7] M. S. Fox. *Constraint-Directed Search: A Case Study of Job-Shop Scheduling*. Morgann Kaufmann, San Francisco, CA, 1987.

[8] M. L. Ginsberg. The computational value of nonmonotonic reasoning. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*, Boston, MA, 1991.

[9] M. L. Ginsberg. Approximate planning. *Artificial Intelligence*, 76:89–123, 1995.

[10] M. L. Ginsberg. GIB and the May, 1996 Bridge World. *The Bridge World*, 1996. Submitted.

[11] M. L. Ginsberg. How computers will play bridge. *The Bridge World*, 1996.

[12] M. L. Ginsberg. Partition search. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 1996.

[13] M. L. Ginsberg and D. A. McAllester. GSAT and dynamic backtracking. In *Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning*, Bonn, Germany, 1994.

[14] W. D. Harvey and M. L. Ginsberg. Limited discrepancy search. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 607–613, 1995.

[15] H. Kautz and B. Selman. Pushing the envelope: Planning, propositional logic, and stochastic search. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 1996.

[16] D. Koller and A. Pfeffer. Generating and solving imperfect information games. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 1185–1192, 1995.

[17] K. Konolige. Easy to be hard: Difficult problems for greedy algorithms. In *Proceedings of the Fourth International Conference on Principles of*

Knowledge Representation and Reasoning, Bonn, Germany, 1994.

[18] K.-F. Lee and S. Mahajan. The development of a world class othello program. *Artificial Intelligence*, 43:21–36, 1990.

[19] J. McCarthy. Applications of circumscription to formalizing common sense knowledge. *Artificial Intelligence*, 28:89–116, 1986.

[20] J. S. Penberthy and D. S. Weld. UCPOP: A sound, complete partial order planner for ADL. In *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, pages 103–113, Boston, MA, 1992.

[21] J. Schaeffer, N. Treloar, P. Lu, and R. Lake. Man versus machine for the world checkers championship. *AI Magazine*, 14(2):28–35, 1993.

[22] B. Selman, H. A. Kautz, and B. Cohen. Local search strategies for satisfiability testing. In *Proceedings 1993 DIMACS Workshop on Maximum Clique, Graph Coloring, and Satisfiability*, 1993.

[23] H. A. Simon. Artificial intelligence: An empirical science. *Artificial Intelligence*, 77:95–127, 1995.

[24] S. J. Smith and D. S. Nau. Strategic planning for imperfect-information games. In *Proceedings of the AAAI Spring Symposium on Games: Planning and Learning*, Stanford, California, 1993.

[25] G. Tesauro. Temporal difference learning and TD-Gammon. *Commun. ACM*, 38(3):58–68, 1995.

[26] D. E. Wilkins. Using patterns and plans in chess. *Artificial Intelligence*, 14:165–203, 1980.

[27] Y. Xiong, N. Sadeh, and K. Sycara. Intelligent backtracking techniques for job shop scheduling. In *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, Boston, MA, 1992.