Advances in Cognitive Systems 10 (2022) 1-6 Submitted 9/2022; published 11/2022

Addressing the Scaling Problem for Embodied Symbolic Reasoning

David Sekora DJSEKORA@UMD.EDU
Department of Computer Science, University of Maryland, College Park, MD 20740 USA

Abstract

Modern machine learning approaches are quite effective across many specific domains, but have several seemingly fundamental shortcomings. Traditional AI frameworks based on symbolic logics, knowledge bases, and explicit reasoning have desirable features that could complement these shortcomings in a hybrid system, but have lagged behind due to a fatal flaw: inefficient scaling. If all of the completeness and correctness properties we typically associate with symbolic logics are desired, then exponential time/space worst-case scenarios are inherently unavoidable. However, both coefficients and average-case behavior can be improved by imbuing the reasoning engine with appropriate notions of context. Furthermore, if certain properties are relaxed or deferred, as they must be for a practical embodied agent, then even greater improvements can be obtained. This paper identifies several primary sources of intractability in symbolic reasoning systems, and outlines approaches to ameliorate some of these sources. Additional detail is given to the incorporation of spatial and temporal context into data storage and retrieval. Taken together, these components form the basis for a new research direction to be undertaken by the author. Some references are made to Active Logic, but many of the ideas themselves are relatively agnostic to the choice of logic and reasoning system.

1. Introduction

Symbolic logics have a long tradition in the field of AI. But, they have always been marred by the issue of scalability - as the number of facts and inference rules increases, the number of possible inferences that can be made grows exponentially. While much work has been done on this problem (and in fact has had to be done for these systems to be practically useful at all), a satisfactory solution has not yet been reached for use in a real-time embodied agent context. In this setting, information is incomplete and constantly being updated, and reasoning must be done without the luxury of being able to consider all of the facts and make all possible inferences.

This paper is primarily concerned with the problem of improving symbolic logic-powered reasoning systems, so it may seem like the setting is irrelevant. However, I argue that the optimizaton of reasoner operation is actually quite entangled with the setting: an agent in a real or simulated world is situated within the time and spatial dimensions of that world, and those dimensions can and should inform how facts are stored and how reasoning is carried out.

1.1 Motivation: Why Symbolic Logics?

It may seem odd to be writing a paper about symbolic logic in 2022, with ML systems wowing the masses with ever higher performance peaks. But, the shortcomings of such systems are well-documented e.g. (Floridi & Chiriatti, 2020), with common criticisms being the lack of knowledge consistency, the inability to reliably perform explicit rule-based reasoning, and difficulty with zero- and few-shot learning. While much work is being done to try to mitigate these problems e.g. (Nye et al., 2021), it must be noted that these are areas in which traditional symbolic systems excel! Conversely, ML systems are strong at scaling and attention, which are traditional reasoner weaknesses, so it seems only natural to explore combining the two approaches. Whether this combination will play out as one being a tool for the other or as two equal and fully-fledged systems interfacing in complex ways is out of the scope of this paper, but either way the goal described here of improving the effectiveness of the symbolic component should be useful.

1.2 Related Work

This paper follows in the tradition of Active Logic e.g. (Elgot-Drapkin et al., 1999), (Anderson et al., 2008), in the sense that a distinction is being made between reasoning *about* the world vs reasoning *in* the world. The former is certainly required by the latter, in a technical sense, yet the former tends to imply a level of detachment, a sense of impassive and omniscient observation that doesn't take into account the limitations imposed by the latter.

Active Logic is a form of step-logic (Elgot-Drapkin et al., 1991), which means that reasoning is performed one step at a time in an ongoing fashion. Not all possible one-step inferences will necessarily be carried out within one time step. The index of the current time step is kept track of with a now predicate, and each fact is tagged with the time step at which it was acquired and the inference rule that produced it (if derived), which allows for such features as propagation of contradiction handling and introspection regarding time taken to reason (which allows for real-time reasoning as well as evaluation of cost). There is a rich body of work surrounding step-logics and Active Logic, much of which specifies consequences, additional features, and extensions to either the logics or the reasoners implementing them, but I will only focus on the above characteristics here because the themes explored in this paper are not tied to those other specifics.

1.3 Note on Terminology

The problem being addressed here is related to the notion of *inferential glut*, outlined in (Brody et al., 2021) as

> the problem that symbolic reasoners tend to produce vastly more deductive information than they can tractably process

and expanded upon in (Brody & Perlis, 2022), both of which focus on Active Logic and its associated reasoner ALMA. Those papers additionally define “pre-inferential glut", which constitutes computational overhead spent to [determine which inferences are valid] in contrast to [make valid inferences that are not useful]. I do not strictly adopt this terminology here because I am addressing a wider problem scope.

Other than “glut", this problem may be referred to in the literature as “swamping", “load", “bloat", “intractability", “information explosion"[1], and “expansion", among others. These terms will be used interchangeably in this paper.

One final piece of terminology: here, I use the term *embodied* to refer to an agent that is situated in space and time, with no distinction being made between real and simulated environments.

1.4 Goals of This Paper

This paper is part problem survey and part research statement. I will first examine scaling issues facing symbolic logics; some of these have been solved to varying degrees, but all will need to be addressed for a system that hopes to be a long-running embodied agent. Then, I will lay out the foundations of a novel direction of research regarding the spatial embedding of knowledge base content.

2. Sources of Intractability and Strategies for Reduction

Here is a breakdown of some common contributors to time and space issues for symbolic reasoning systems, along with thoughts on addressing some of them:

2.1 Superfluous Information

Superfluous information comprises information that is useful, but already adequately represented elsewhere in the knowledge base. This category can be broken down into duplicate information, near-duplicate information, semantically equivalent information, and easily derivable information:

2.1.1 Duplicate Information

Multiple copies of the same fact can arise when information is coming in from multiple sources. In a hypothetical system with the ability to sort through its entire knowledge base at every reasoning step, detection and elimination of duplicates is relatively straightforward. However, in a real-time system, only a subset of the knowledge base is being considered at any given time. If the duplicates never show up in the same context, then they would never be detected without a specialized subroutine made for the purpose. One could argue that this case doesn’t constitute much of a problem, since only one of the facts is contributing to the computational load at any given time during normal query and inference operations. However, this does create issues when the fact needs to be updated or removed, and also does contribute overhead for operations that must traverse the entire knowledge base. So, periodic sweeps for duplicates should be carried out.

2.1.2 Semantically Equivalent Information

This category is the same in spirit as the previous one, except that instead of multiple copies of the same fact we have several syntactically different facts that contain the same semantic content.

---

[1] Not to be confused with the “principle of explosion", which refers to the derivability of every fact from a contradiction in classical logics, and which we still do need to be careful to avoid.

Checking for this can be done in the same contexts as the previous, but the determination of what counts as “equivalent" is nontrivial. (It should be noted that humans encounter this problem quite frequently, and do not always make the appropriate connections to unify their knowledge across multiple domains.)

2.1.3 Near-Duplicate Information

Many facts are closely related to sets of other facts. For example, let’s say you observe that a party is occurring at 5PM. You also observe that the party is occurring at 5:01PM, and 5:02PM, and so on until 7PM. Rather than storing 121 facts of the form $OccurringAt(BigParty, n)$, instead it would be more efficient and intuitive to store a single fact including an interval:

$$OccuringAt(BigParty, interval(5PM, 7PM))$$

where the interval data would probably also include date information, but we are keeping things simple.

Intervals (and hyper-rectangles, more generally) work great for information that is consistent across the entire interval, and logics that incorporate intervals are quite well-developed (e.g. Gerevini & Schubert (1995)). However, what about information that changes over time? For example, let’s say you record the temperature throughout the day. If your system is granular enough, this value might change every hour or every minute, resulting in a deluge of facts that will bog down reasoning. Typical context-based methods to filter out irrelevant facts will be ineffective - these facts are so similar that if one is included in a search, the others likely will as well. But this similarity can actually work to our advantage: computers are traditionally great at storing lots of similar information, after all. So, I propose that a set of similar facts be condensed into a singular fact, which contains a pointer to a native data structure that holds the actual variable data history. This way, the fact is only considered for inference once, and the correct timepoint value can be extracted at inference time.

This approach works well for simple data types such as numbers and strings, but there is plenty of room for improvement by expanding the definition of “near-duplicate". There is also a conflict of granularity to consider here: if we have the fact that 4 people are at a party at 6:00PM, and there are 5 people at 6:30PM, should we just record that number, or should we actually have some separate facts regarding *who* was at the party at each time? If the latter, should we collate such facts by party-attendance, or by person-daily-activity?

2.1.4 Easily Derivable Information

This category can be considered a generalization of the previous one. Humans have access to many more facts than they explicitly store - for example, I don’t know offhand the answer to 47×73, but I know that I can easily derive it and that it is unlikely to come up, and therefore I know that it is not useful to me to store at this time (see Parikh (1987) for some examples along these lines). Certain classes of information are easier to apply this to than others (such as facts that are direct consequences of a single inference rule, or facts whose expressions and derivations are formulaic in nature and can be offloaded to specialized subroutines).

2.2 Unneeded, Outdated, and Irrelevant Information

Ideally, all of the information in our knowledge base would be relevant and up-to-date - right? Well, this is probably not actually the case. Sometimes, it's useful to have access to historical facts about misunderstandings, but we must be careful not to mix this information up with current knowledge, nor to let it take too many resources. As for “relevant", that by definition changes with context - what's relevant tomorrow may be the furthest thing from one's mind in the current moment.

2.2.1 Unneeded Information

We can't keep track of every piece of information we come across. Incoming and recently derived information should be filtered to determine whether it merits long-term storage, and information that hasn't been accessed much should be evaluated for archiving, compression, or removal.

2.2.2 Outdated Information

This is information that is incorrect, or information that has become incorrect. While there is a distinction between these, both mean that we no longer want to consider the information as correct. Of course, there is a bigger implication here that we need to re-evaluate other facts we have in case they have been “tainted", but even just from an efficiency standpoint it's clear that we should take every opportunity we can to remove information that we don't need.

2.2.3 Irrelevant Information

This category is trickier than the previous two, because we DO want this information in our knowledge base - we just don't want it to be part of the current inference or task. Assuming that the calculation of the relevance heuristic for a single fact takes constant or at most $O(log(n))$ time, where n is the number of facts, it may seem like the $O(nlog(n))$ cost for the simple algorithm of [assign a priority value to each fact and sort the facts by priority] is dwarfed by the cost of actually carrying out inference, which is higher-order polynomial due to the need to test all possible combinations of facts and inference rules. However, the computational load for inference scales with the number of *relevant* facts, not the number of *total* facts. Long-running embodied agents will presumably accrue large collections of facts over their lifespans, and therefore the ratio of total facts to relevant facts will need to be small. Therefore, the cost of [determining which facts are relevant] is itself quite relevant.

The main way to address this is with better means for storing and querying information, which will be primarily addressed in section 3. Briefly, if we have information about where the relevant information is stored or is likely to be stored, we may not need to look through the entire knowledge base - consider the efficiency of binary search vs linear search.

2.3 Poor Maintenance of Organization

The knowledge base for any reasoning agent needs to be able to be frequently updated to reflect newly gained information, and this holds especially true for an embodied agent. All data structures are not created equal with respect to updating: some rebalance themselves to maintain their useful properties, whereas others can grow more and more unbalanced if (often expensive) global refactoring isn't executed.

3. A Framework for Storing Time and Space Situated Knowledge

Now, we turn our attention to a framework that addresses some of the above problems.

Active Logic, as mentioned in the introduction, improves on traditional temporal logics by being situated *in* time rather than just reasoning *about* time. In a similar vein, the logics typically referred to as "spatial logics", such as those mentioned in (Aiello et al., 2007), reason *about* space, but are not situated *in* space. Active Logic achieves its situated-ness by linking facts in the knowledge base with timestep information; I propose extending this to also index each fact with information about the location in which it was acquired.

3.1 Why Spatial Indexing?

If we are going to use multi-dimensional metadata to index our facts, one might wonder: why choose something so mundane as space? After all, reasoners such as Cyc (Lenat et al., 1990) use elaborate microtheory hierarchies to establish domain, and there is plenty of literature on incorporating modular special purpose reasoners such as (Frank, 1999). If we expand our horizons to the ML world, we can clearly see the benefits of letting a system learn its own dimensional stratificaton such as in the famous word2vec (Mikolov et al., 2013).

Unfortunately, the Cyc approach requires copious amounts of hand-crafting, and word2vec and co. require huge corpuses of data, both of which can create bottlenecks when adapting to new domains. In comparison, spatial data is readily available to an embodied agent; at least one form of spatial data is relatively free of ambiguity and can be collected in an automated fashion without issue. Compare the query "is my central coordinate reference point within the rectangular prism defined by points (x_1, y_1, z_1) and (x_2, y_2, z_2)?" with "am I currently in a FoodPreparationTask scenario or a HouseholdChoreTask scenario?".

When it comes to interfacing with humans and the human-generated information ecosystem, spatial information is often more accessible, intuitive, and universal than a custom ontology or embeddings for your arbitrarily defined vector space, in both directions of information flow.

Finally, using spatial indexing doesn't preclude us from also incorporating other forms of context. I claim that space and time are natural and low-cost, not that they are sufficient for all context determination needs. Exact nearest-neighbor queries suffer from the curse of dimensionality (Weber et al., 1998), but if we restrict usage to deciding which facts are likely to be relevant then approximate nearest-neighbor methods should be sufficient.

3.1.1 Support By Existing Research

Spatial information is well-studied in the literature. Relational queries are made easier by well-studied topological relationships (e.g. Clementini et al. (1993)). There is a large body of existing work on spatial databases (Güting, 1994) to draw from, with a lot of focus in the last decade on performance at scale e.g. (Aji et al., 2013). Reasoning tasks can be difficult to parallelize, but space provides a natural structure on which partitioning can potentially be carried out, creating potential for partition-based theorem proving (MacCartney et al., 2003) (likely requiring the establishment of some general-purpose global knowledge).

3.1.2 Examples of the Usefulness of Spatial Locality

Consider the following example scenarios, with the illustrated concepts appended in parentheticals:

- You work as a chef in a restaurant. When you cook dinner in your kitchen at home, you are able to apply many of your professional skills, but you are rarely confused by the difference in tools and methods available to you. (Location determines context. Location can have multiple modes of context: you are in a kitchen, but it is a different kitchen than your work kitchen.)
- You live in an apartment complex. Your upstairs neighbor calls you and asks you to turn off the oven in her apartment. You have never been in her apartment, but you know that all of the apartments in your building have the same layout, so you have no trouble completing the task. (Hierarchical location, batch information transfer from one location to another.)
- You enjoy working from home, but find that you are typically more productive in the office. (Location eases access to information and skills.)

3.2 Comparison With Time

Space shares many characteristics with time, but there are some important differences that make “do the same thing we did with time, but with space instead" a nontrivial undertaking.

Time is sequential and unidirectional, whereas most spaces can be returned to repeatedly and arbitrarily. There is no direct spatial equivalent of “facts derived at time step t were derived from facts known prior to t", and spaces can evolve over time whereas time intervals are fixed. However, this can also be a blessing, as the fact that we can return to a location means that information we gain about that location is likely to be useful to us beyond the immediate future.

The discrete time steps outlined in our summary of Active Logic may seem a far cry from the continuous multi-dimensional regions we must now contend with. However, time is continuous as well, and we often need to interact with intervals rather than individual timepoints. Conversely, space can also be discretized, either with a grid or by mapping regions to entities (e.g. “I am in room 310 of the Computer Science building").

3.3 Gestures At Technical Details

This section will be kept light, as the main focus of this paper is on the overarching ideas guiding the research direction rather than the implementation details.

3.3.1 Data Structure Desiderata

More complex data structures and algorithms are necessary to support efficient query operations. R-trees and their variants seem a promising start, and again the spatial database community should prove fruitful; a cursory search reveals research on both domains with frequent updates e.g. (Silva et al., 2009) and history maintenance e.g. (Tao & Papadias, 2001), both of which are desiderata.

Ideally, the reasoner will be able to efficiently iterate through its knowledge base in order of decreasing relevance, stopping either when a goal condition is met (for goal-driven reasoning) or when the allotted time (either one time step, or a longer specified interval) for the reasoning task has expired. In a system indexed only by spatial coordinates, this can be achieved by beginning with a narrow query region around the reasoner's current position and gradually increasing the scope of the region; to increase efficiency further, the explored region can be excluded from future queries by splitting the consecutive queries into the (hyper-)rectangles surrounding the explored region.

3.3.2 Coordinate Considerations

For small-scale highly localized agents, a number of assumptions can be made to simplify the coordinates. An "absolute" coordinate system can be defined from a stable reference point, ground can be assumed flat, etc.. However, if an agent is expected to know or learn about locations far enough away that the Earth's curvature becomes relevant, or about any scientific topics where these assumptions break down, more sophisticated coordinate systems must be employed. In a system that reasons *about* space, this is less of a problem, but for our system space is something more fundamentally ingrained, and therefore it may be more sensitive to this type of paradigm shift.

For everyday tasks carried out by non-airborne agents, the two dimensions of horizontal movement are likely to be more important than the vertical dimension. The vertical dimension has a much smaller range of operation[2], and there is less variance within it.

4. Conclusion

This paper took some initial steps towards a research direction of daunting scale. Motivation and background were provided, problems to be addressed were described, and plans of attack for some problems were sketched. The research direction was positioned as enabled by existing research but not encapsulated by it, suggesting that it is both possible and novel.

Acknowledgements

Thanks to Don Perlis for laying the groundwork for many of these ideas, and Yiannis Aloimonos for pushing me to take these ideas into new directions.

---

[2] When specified in terms of distance from the ground; this ignores the utility of facts about elevation, but multiple coordinate systems can and should be employed in the same way that temporal systems are able to conceive of multiple types of time units.

References

Aiello, M., Pratt-Hartmann, I., Van Benthem, J., et al. (2007). *Handbook of spatial logics*, volume 4. Springer.

Aji, A., Wang, F., Vo, H., Lee, R., Liu, Q., Zhang, X., & Saltz, J. (2013). Hadoop gis: a high performance spatial data warehousing system over mapreduce. *Proceedings of the VLDB Endowment*, *6*, 1009–1020.

Anderson, M. L., Gomaa, W., Grant, J., & Perlis, D. (2008). Active logic semantics for a single agent in a static world. *Artificial Intelligence*, *172*, 1045–1063.

Brody, J. D., Austin, B., Khater, O., Maxey, C., Goldberg, M. D., Clausner, T., Josyula, D., & Perlis, D. (2021). Using neural networks to control glut in the active logic machine. *NeSy* (pp. 143–153).

Brody, J. D., & Perlis, D. (2022). Decision transformers for glut control in the active logic machine. *AAAI Spring Symposium: MAKE*.

Clementini, E., Felice, P. D., & Oosterom, P. v. (1993). A small set of formal topological relationships suitable for end-user interaction. *International Symposium on Spatial Databases* (pp. 277–295). Springer.

Elgot-Drapkin, J., Kraus, S., Miller, M., Nirkhe, M., & Perlis, D. (1999). *Active logics: A unified formal approach to episodic reasoning*. Technical report, University of Maryland, College Park, MD.

Elgot-Drapkin, J., Miller, M., & Perlis, D. (1991). Memory, reason and time: the step-logic approach. *Philosophy and AI: Essays at the Interface*, (pp. 79–103).

Floridi, L., & Chiriatti, M. (2020). Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, *30*, 681–694.

Frank, G. (1999). A general interface for interaction of special-purpose reasoners within a modular reasoning system. *Proceedings of the 1999 AAAI Fall Symposium on Question Answering Systems* (pp. 57–62).

Gerevini, A., & Schubert, L. (1995). Efficient algorithms for qualitative reasoning about time. *Artificial intelligence*, *74*, 207–248.

Güting, R. H. (1994). An introduction to spatial database systems. *the VLDB Journal*, *3*, 357–399.

Lenat, D. B., Guha, R. V., Pittman, K., Pratt, D., & Shepherd, M. (1990). Cyc: toward programs with common sense. *Communications of the ACM*, *33*, 30–49.

MacCartney, B., McIlraith, S., Amir, E., & Uribe, T. (2003). Practical partition-based theorem proving for large knowledge bases. *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*. Citeseer.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Nye, M., Tessler, M., Tenenbaum, J., & Lake, B. M. (2021). Improving coherence and consistency in neural sequence models with dual-system, neuro-symbolic reasoning. *Advances in Neural Information Processing Systems*, *34*, 25192–25204.

Parikh, R. (1987). Knowledge and the problem of logical omniscience. *ISMIS* (pp. 432–439).

Silva, Y. N., Xiong, X., & Aref, W. G. (2009). The rum-tree: supporting frequent updates in r-trees using memos. *The VLDB Journal*, *18*, 719–738.

Tao, Y., & Papadias, D. (2001). Efficient historical r-trees. *Proceedings Thirteenth International Conference on Scientific and Statistical Database Management. SSDBM 2001* (pp. 223–232). IEEE.

Weber, R., Schek, H.-J., & Blott, S. (1998). A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. *VLDB* (pp. 194–205).