A Cognitive API and Its Application to AGI Intelligence Assessment

Ben Goertzel[1] and Gino Yu[2]

[1] OpenCog Foundation
[2] School of Design, Hong Kong Poly U

Abstract. An Application Programming Interface for human-level AGI systems is proposed, aimed at bridging the gap between proto-AGI R&D systems and practical AI application development. The API contains simply formalized queries corresponding to the various key aspects of human-like intelligence, organized so as to be independent of the algorithms used under the hood for query resolution and associated supporting cognitive processes. A novel, qualitative (and in principle quantifiable) measure of software general intelligence is proposed (the APIQ), measuring the degree to which a system succeeds at fulfilling the various API functions using a compact set of representations and algorithms.

1 Introduction

Currently AGI is a relatively distinct pursuit from the engineering of practical AI applications. Indeed, the two pursuits can even sometimes seem opposed to each other. If one posits an "AGI vs. Narrow AI" dichotomy [1], focus on specific practical applications may be seen as one of the primary factors driving the majority of the AI field's attention toward Narrow AI (other major factors including the greater ease of doing theoretical analysis and empirical testing on narrower systems).

On the other hand, it's also the case that a number of proto-AGI systems have been customized, in whole or in part, to serve as parts of various practical applications. But this has been done on an ad hoc basis in each case, with considerable specialized effort required. I believe it is possible to connect the AGI and application development worlds more systematically, so that AGI R&D and AI application development can proceed more synergistically, each more richly benefiting the other.

One way to manifest this potential, I suggest, is the development of a "Cognitive API" for a proto-AGI software system, enabling application developers to access the system in specific ways that tend to be useful for application software, according to an interface that requires no knowledge of the underlying algorithms of the proto-AGI system, and doesn't change as the particulars of these algorithms change. Adoption of such an API would lead to more users for early-stage AGI systems, hence to more incentive for various organizations to fund or sponsor work on AGI systems, and thus could accelerate AGI development as well as improving application quality.

B. Goertzel et al. (Eds.): AGI 2014, LNAI 8598, pp. 242–245, 2014.

In this paper I will outline one approach to defining a Cognitive API of this nature. While the API indicated here would work effectively with the OpenCog framework that the author is currently involved with [5] [6], in its overall outline it is not OpenCog-specific in any way, and could be used just as well together with many alternative AGI systems.

The pursuit of a Cognitive API also provides a novel perspective on one of the vexing issues at the heart of the AGI field – the difficulty of measuring the level of general intelligence displayed by a given system. A novel intelligence measure, the APIQ, is proposed and defined. Roughly speaking the APIQ measures the degree to which a system fulfills a broad subset of the API functions using a concise set of representations and algorithms. Full formalization of the APIQ appears difficult, but it does provide a novel, and highly pragmatic, approach to conceptualizing the notion of general intelligence in the context of real-world applied software systems.

1.1 Needs of Application Developers vs. AGI Researchers

A key point underlying the current suggestion is that the needs of AGI researchers, versus application developers aimed at using early-stage AGI software in their applications, are substantially different.

For a developer whose focus is the creation or improvement of AI algorithms, the appropriate interface for an AI system is one that is extremely general and flexible, providing the maximum possible latitude in experimenting with new ideas and increasing the intelligence, scope and/or efficiency of existing code. For example: While rough around the edges in various places, OpenCog's current Scheme shell is a reasonable first approximation of an interface of this sort.

On the other hand, for a developer whose focus is the creation of AI-based application systems, a different sort of interface is appropriate an Application Programming Interface or API that supplies a limited but powerful set of application functionalities, in a manner providing: a) Simplicity of access to the AI functionalities commonly required by application developers; b) for each functionality, either a reliable level of intelligence, or a reliable confidence level associated with each output (indicating the systems assessment of its own confidence in the intelligence of its output in a given instance); c) robustness as a software system, when carrying out the specific application functionalities directly accessible by the API.

2 Representation

It seems inescapable that a Cognitive API will need to be associated with a flexible knowledge representation language. For instance, if a cognitive API were to be realized using OpenCog, then the representation language would be the language of the Atomspace, OpenCog's weighted, labeled hypergraph knowledge representation, e.g. as realized via the Scheme representation now commonly used to load Atoms into the Atomspace.

In general, one wants a KR (Knowledge Representation) language that is highly general, and is reasonably easily both human and machine readable. The bulk of the API indicated here, which comprises a set of queries to be made of a cognitive system, assumes the existence of a KR that can straightforwardly be used to provide descriptions of the following entities, for use in input and/or output of queries: actions, agent, bodies, categories, communication media, communications, constraints, datasets, expressions, events, maps, movements, object, patterns and situations.

Another relevant factor is standardization of concepts referenced in queries. For a cognitive system to have a better chance of understanding a users queries, it will be easier if the user poses his knowledge in terms of standard ontologies wherever relevant and feasible, e.g. WordNet [3] and ConceptNet [10].

3 Queries

The wiki page `http://wiki.opencog.org/w/A_Cognitive_API` comprises critical supplementary information to this paper, and presents a set of queries that we suggest a cognitive API should support. The set of queries is designed to cover all the key aspects of human-like intelligence, as identified in the AI Magazine paper [1] summarizing the conclusions of the 2009 AGI Roadmap Workshop. For each core category of human-intelligence functionality as identified there, a handful of essential questions is identified. And, each of these essential questions may be cast as a formal API call. While a lot more work would need to go into honing a formal API along these lines, first-draft "API call" versions of the questions are given here, for the clarity that this sort of concreteness brings.

For concreteness we give here three arbitrary examples from the long, structured list on the above wiki page ($*$ denotes an optional argument):

- *Expression-List GetTemporalPatterns(Situation-List S)*: what temporal patterns exist in a certain set of situations?
- *DirectAttention(Entity X, *Time T)*: instructs a cognitive system to focus its attention on a certain entity for a certain time period
- *GetAssociatedEmotions(Entity X, *Situation S, *Time T): finds the emotions associated with a certain entity*

4 An Application-Oriented Measure of General Intelligence

The notion of a Cognitive API also has implications in the area of AI intelligence assessment. Quantification of general intelligence levels poses a significant challenge for the AGI field, to which various approaches have been suggested [8] [9] [7] [4] [2] [1]. The Cognitive API proposed here suggests a different approach, an "APIQ" that measures the degree to which a software system can carry out a broad variety of humanly useful, intelligent-seeming functions using a small class of representations and mechanisms.

Suppose that, for each of the queries in the API: a) one has identified a measure of the quality of performance of a given software system at responding to that query, scaled between 0 and 1; b) one has a meaningful qualitative (or formal) way to identify the different representations and algorithms in a given software system; c) for each of these representations and algorithms, and each API query, one can quantify the degree to which the representation/algorithm plays a role in the response to the query, with some degree between 0 and 1. Then one can estimate the *total API quality* of a system as the sum of the performance quality the system displays on each API query; and the *query API generality* of a system as the entropy of the set of performance quality values displayed by the system on the API queries. One can also estimate, for each representation or algorithm, the *total contribution*, defined as the sum over all API queries of: the system's performance quality on the query, multiplied by the degree to which the representation/algorithm plays a role in the query; and the *component API generality*, defined as the entropy of the set of total contribution values, across all the representations/algorithms.

The **applied human-level general intelligence** of a system, then, can be defined roughly as (where a, b, c are nonnegative weight values summing to 3)

$$APIQ = \text{total API quality}^a * \text{API generality}^b * \text{component API generality}^c$$

References

1. Adams, S., Arel, I., Bach, J., Coop, R., Furlan, R., Goertzel, B., Hall, J.S., Samsonovich, A., Scheutz, M., Schlesinger, M., Shapiro, S.C., Sowa, J.: Mapping the landscape of human-level artificial general intelligence. Artificial Intelligence Magazine (2011)
2. Bringsjord, S., Schimanski, B.: What is artificial intelligence? psychometric ai as an answer. In: IJCAI, pp. 887–893 (2003)
3. Fellbaum, C.: WordNet: An Electronic Lexical Database. Addison-Wesley (1990)
4. Goertzel, B.: Toward a formal definition of real-world general intelligence. In: Proceedings of AGI (2010)
5. Goertzel, B., Pennachin, C., Geisweiller, N.: Engineering General Intelligence, Part 1: A Path to Advanced AGI via Embodied Learning and Cognitive Synergy. Atlantis Thinking Machines. Springer (2013)
6. Goertzel, B., Pennachin, C., Geisweiller, N.: Engineering General Intelligence, Part 2: The CogPrime Architecture for Integrative, Embodied AGI. Atlantis Thinking Machines. Springer (2013)
7. Hernandez-Orallo, J., Dowe, D.L.: Measuring universal intelligence: Towards an anytime intelligence test. Artificial Intelligence Journal 174(18), 1508–1539 (2010)
8. Legg, S., Hutter, M.: A definition of machine intelligence. Minds and Machines 17 (2007)
9. Legg, S., Veness, J.: An approximation of the universal intelligence measure. CoRR abs/1109.5951 (2011)
10. Speer, R., Havasi, C.: Conceptnet 5. Tiny Transactions of Computer Science (2012), `http://tinytocs.org/vol1/papers/tinytocs-v1-speer.pdf`