Overview of NextKB

Kenneth D. Forbus, Qualitative Reasoning Group, Northwestern University
1/7/19

NextKB is an open-license knowledge base built by our group to support research into building knowledge-rich systems that support both human-like learning (via analogy), qualitative reasoning, visual and spatial reasoning, and natural language understanding. It builds upon material from OpenCyc, FrameNet, and VerbNet, in addition to what our group has developed. This document provides an overview of what is in the files that constitute a NextKB distribution.

File Format

We call the files which specify the knowledge in the KB *flat files*, because they are text files, by contrast with the highly cross-linked structures that are typically found in a knowledge base implementation. The extension is always `.krf`, for knowledge representation file. The syntax is Lisp-style syntax, although these expressions are interpreted by a reasoning system, either at KB build time or during reasoning. The basic ontology is that of OpenCyc, which is introduced in a note that is also on the NextKB web page. There are only two kinds of exceptions:

- `(in-microtheory <mt name>)` indicates that all of the facts in the file after this directive are considered to be in the microtheory `<mt name>`. The optional argument `:include-globals?`, when `nil`, prevents the named microtheory from inheriting from `BaseKB` or `UniversalVocabularyKB`, which it otherwise would by default (see OpenCyc introduction for details).
- A handful of forms like `(defSuggestion …)`, which are unpacked by the FIRE reasoning system into a number of assertions. These forms make manual knowledge entry much more convenient, but they never appear in the KB themselves. Please see the FIRE manual, also on the NextKB web page, for more information.

These files are what we use in building KBs for our FIRE reasoning engine. We also distribute a FIRE KB built from them, so that you can use our existing freely available tools (CogSketch, Case Mapper) to browse and experiment with NextKB:

- CogSketch is a sketch understanding system that is both a model of human high-level visual and spatial representation and reasoning, and a platform for sketch-based educational software. CogSketch has been using NextKB for several releases now, so if you install CogSketch, you will get have a copy of NextKB you can experiment with. Please see the CogSketch manual for details. CogSketch requires Windows to run.
- Case Mapper provides a cognitive-scientist friendly way to experiment with analogical matching and retrieval, as well as a socket-based API for providing analogical services to other programs. Case Mapper is updated less frequently, so we advise downloading the version of NextKB from the NextKB web page and opening it from Case Mapper rather than using the built-in version. There are Case Mapper binaries that run under Windows, Linux, and Macs.

Given that the files contain just over 1.3 million facts, you will find using the browser built into one of these tools a lot easier than file-level operations.

Aside from the special forms noted above, we have kept the representation format quite neutral, so that others should be able to import this content into their own knowledge bases and reasoning systems. This will not always be easy: The OpenCyc ontology is far more expressible than today's Semantic Web formats and tools support, for example. But as the field seeks to perform more human-like reasoning, we believe such expressivity will prove to be crucial.

Contents of the flat files

There are just over 900 files in the NextKB flat file distribution, so we provide a quick guide to help you quickly focus on what you are interested in. The directory structure reflects how we organize files in our group.

OpenCyc ontology

The version of the OpenCyc ontology that we use is in `fire\flat-files\nextkb`. It constitutes a curated subset of material from OpenCyc 0.7-4.0, focusing on the material that we found the most useful. All Cycorp-derived material is in this directory, one microtheory per file.

Representations for natural language understanding

This material is concentrated in `ea\v8\kbfiles`, and is further broken down as follows:

- `lexicon\nulex4` contains files defining version 4 of the Nulex lexicon, which was built by extracting data from a public-domain Webster's dictionary and by-hand additions.
- `semtranses` contains files that provide semantic translations, using a transmogrified version of data from FrameNet to link to the OpenCyc ontology.
- `grammar` contains files that define our parser's grammar. The parser is derived from James Allen's TRAINS parser.
- `qp-support` describes frame-based representations of QP theory concepts, which are used with narrative functions in extracting qualitative models from natural language text.
- `propernames.krf` contains proper names

Much less of our work has gone into generation, see `nlgen.krf` and `verbalize.krf` under `fire\flat-files\`.

Representations for sketch understanding, vision, and spatial reasoning

These include representations used by CogSketch and other QRG systems. They break down into

- `cogsketch` contains the basic representations used by CogSketch, including the representations for the Sketch Worksheets tutor. Other representational components of CogSketch are localized to subdirectories:
    - `design-coach` describes the ontological extensions and teleological reasoning and coaching strategies for the CogSketch Design Coach, which gives students feedback on their explanations of mechanical systems they are designing.
    - `NuSketch` provides the basic visual representations used by CogSketch, including qualitative visual reasoning and the spatial routines processor.
    - `QM` provides the ontological extensions and rules for qualitative mechanics reasoning from sketches, including forces, friction, gravity, springs, cords, and surfaces. It is used by the Design Coach to generate predictions from a student's sketched design.
    - `skea` provides additional CogSketch internal representations.

- `shape-library` contains more shape representations used by CogSketch

Representations for analogy

The analogy ontology is defined in fire\flat-files\analogy-ontology.krf. The inferential power of these predicates is derived from FIRE outsourced predicates, i.e. procedural attachments, that call on highly optimized Lisp code.

Representations for the FIRE reasoning engine

The rest of the files in `fire\flat-files\*.krf` provide support for the FIRE reasoning engine. Some miscellaneous points of interest:

- `solve\solve-saint.krf` is an implementation of Slagel's SAINT system for symbolic integration
- `rerep\` contains some rudimentary re-representation facilities.
- `NWU\`background-knowledge contains various KB extensions
- `NWU\geoquery` contains a rendering of the classic geoquery dataset, translated into the OpenCYc ontology
- `NWU\geoscience` contains descriptions of geological processes
- `NWU\kiosk` contains the ontology and data that drives the Companion-based Kiosk installed in the new Computer Science Department's space in Seely Mudd on the Evanston campus.

Representations for QP theory

The basic QP ontology is described in `fire\flat-files\qp-ontology.krf`. The FIRE-based implementation of QP theory is supported by the representations in `fire\flat-files\QQR`. Much of the work in this implementation is done by outsourced predicates for efficiency.

Representations for HTN Planning

FIRE includes an HTN planner, and the files in `fire\flat-files\planning` provide test cases for it.

Representations for Freeciv

Freeciv is an open-source version of Civilization 2 which we have been using for experiments with Companions. The directory `freecivai\flat-files` contains ontological extensions used by the simulator interface to reify what is happening in the domain world, basic information about the game, planning strategies, and learned knowledge that has been accumulated over multiple experiments.

Representations for the Companion cognitive architecture

The Companion cognitive architecture is knowledge-intensive, and so while there is a great deal of associated code, much of it is defined in companions\v1\flat-files and the various subdirectories. The .krf files in the main directory contain the most general capabilities. Subdirectories concern specific aspects of the architecture or experiments, i.e.

- `commonsense` concerns doing commonsense reasoning via analogical chaining.
- `cross-modal-interaction` supports language/sketching interactions, including a model of forced-choice tasks
- `executive` is the knowledge component of the Companion Executive agent, responsible for managing the system's operations.

- `interaction-manager` is the Companion agent that handles user interaction, running natural language and accessing sketch agents as needed.
- `mechanics` contains the representations used by our experiments in within-domain and cross-domain transfer learning in solving physics problems
- `multimodal-instruction` contains the representations used in teaching simple games via language and sketching
- `perceptual-agent` is a set of higher-level Companion capabilities for agents that perceive and act in some world, e.g. Freeciv.
- `sketch-agent` and `spatial-agent` contain the knowledge involved in connecting CogSketch into the Companion architecture.
- `tech-kb` is a technology knowledge base developed by Tom Hinrichs
- `test-plans` contains representations used in regression testing