Proceedings of the Twenty-Eighth International Florida Artificial Intelligence Research Society Conference

English Generation from a Knowledge Base and Overly General Classes

Vinay K. Chaudhri and **Nikhil Dinesh**
Artificial Intelligence Center
SRI International, Menlo Park, CA, 94025

Eva Banik
Computational Linguistics, Shakespeare Tower
Barbican, London, EC2Y 8DR, UK

Umangi Oza
Evalueserve Private Limited
Gurgaon - 122001, Haryana, India

Abstract

In this paper, we describe the range of sentences that we synthesize from a knowledge base (KB) using a natural language generation (NLG) system. We give an overview of our NLG system, and then focus on the specific challenge in handling overly general classes such as Tangible-Entity and Physical-Object. Such classes are unavoidable in the KB, and it is undesirable to show them in an output for end users. Our approach has been implemented in an intelligent textbook application that helps students learn better.

Introduction

To perform well at the Jeopardy! task, it was not necessary for the Watson system to synthesize novel sentences because 94.7% of the correct answers are the titles of the Wikipedia pages (Chu-Carroll and Fan 2011), and for the remaining 5.3% of the correct answers, the answer was limited to a word or a short phrase. In contrast, the focus of the current paper is on a question answering system that synthesizes novel sentences from a knowledge base (KB).

Our overall goal in creating this KB has been to support explanation, reasoning and dialog (Chaudhri, Dinesh, and Inclezan 2013). Previous work has reported the use of KB for reasoning (Chaudhri et al. 2014b; Chaudhri, Dinesh, and Heller 2013). The focus of the present paper is on the use of the KB in generating natural language sentences needed for answering questions. We describe example sentences produced by the system, give an overview of the computations that needed to be performed to integrate NLG into the overall system, and consider in detail the problem of presenting overly general classes. By an overly general class, we mean a domain-independent class (e.g., Tangible-Entity, Physical-Object) usually found in upper ontologies (Borgo and Masolo 2009; Spear 2006). Because the end users, who are students of biology, cannot be expected to have an understanding of such classes, it is not desirable to present overly general classes in the output. We analyze why such classes are present in a domain-specific KB, and present a preliminary approach for handling overly-general classes. The task of thoroughly evaluating our approach for onverly general classes remains a subject for future work.

This work was performed in the context of Project Halo which resulted in `KB_Bio_101` (Chaudhri, Wessel, and Heymans 2013; Chaudhri et al. 2013c). `KB_Bio_101` represents knowledge about biological processes and mechanisms from an introductory college-level biology textbook (Reece et al. 2011). `KB_Bio_101` is integral to a prototype of an intelligent textbook called *Inquire* that was designed to help students learn better (Chaudhri et al. 2013a). *Inquire* answers questions (Chaudhri et al. 2013c), gives explanations and engages in dialog through NLG (Banik, Kow, and Chaudhri 2013).

Several innovative NLG systems have been recently built. For example, QUILL is targeted at generating stories from the database facts in domains such as finance, sports, civics, etc (Birnbaum 2013). Our work differs from QUILL in that we are working from a first order logic KB as opposed to working from database facts. Using NLG for KBs in Web Ontology Language (OWL) is also of interest (Androutsopoulos, Lampouras, and Galanis 2013; Liang et al. 2013; Stevens et al. 2011). These prior systems present the taxonomic information in English, as does our system; our system can also present rich descriptions of events and entities. Cyc uses a template-based generation system that converts one axiom at a time into English (Lenat 1995). In contrast, our approach extracts information from multiple rules to synthesize sentences. Thus, the primary contributions of our work consist of (1) presenting an application that requires synthesizing English sentences with a much greater range and sophistication than any of the previous NLG systems, and (2) identifying an important challenge of overly general classes for the construction of robust NLG systems and presenting an initial approach for meeting that challenge.

We begin with some background and then give example sentences that the NLG system produces. Next, we consider

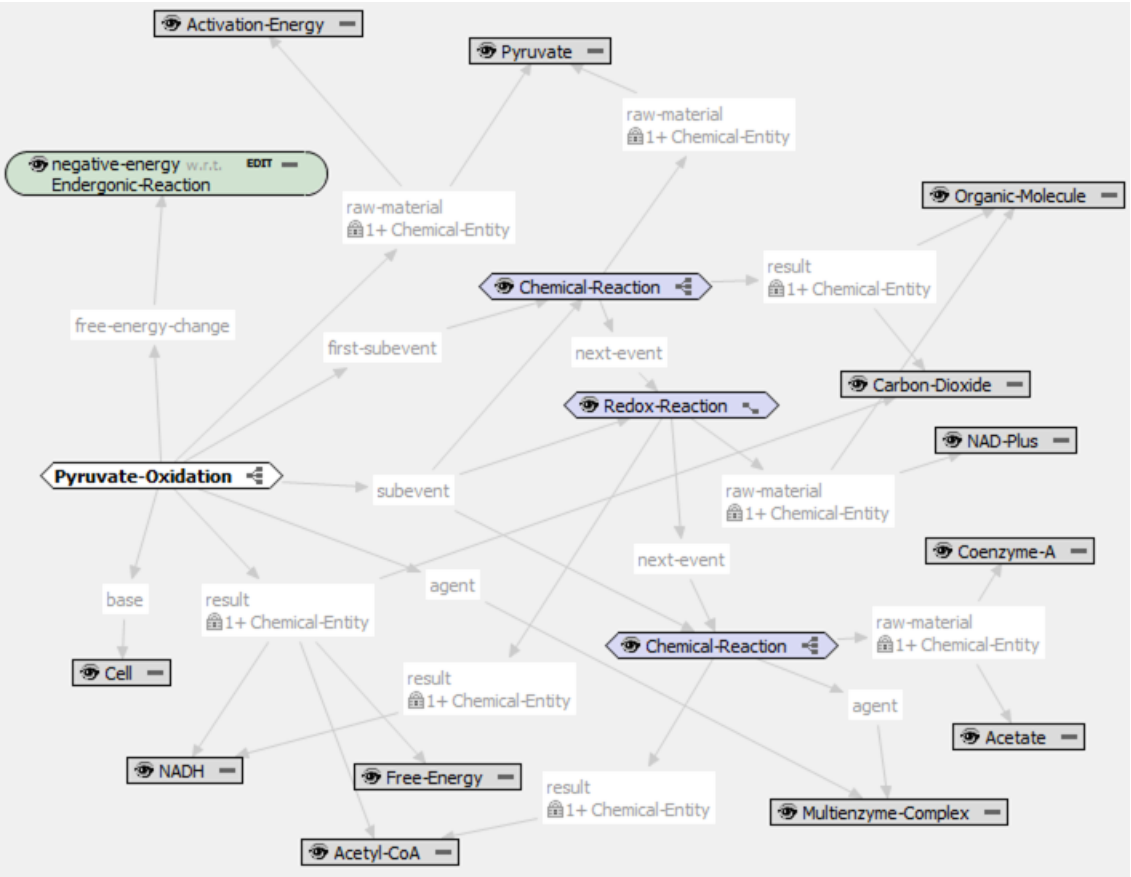

Figure 1: Concept Graph for Pyruvate Oxidation

the problem of overly general classes and our approach for addressing it. We conclude by identifying open challenges.

Background

We explain the relevant aspects of KB_Bio_101 and *Inquire* by examples. The graph in Figure 1 represents an existential rule (Baget et al. 2011) from KB_Bio_101 in which Pyruvate-Oxidation is universally quantified (shown in white) and every other node (shown in gray) is existentially quantified. Specifically, this graph represents that for every instance of Pyruvate-Oxidation, a Chemical-Reaction, a Redox-Reaction and another Chemical-Reaction exist; that these reactions are related to the instance of Pyruvate-Oxidation by the subevent relation; and further, that the first Chemical-Reaction is also related to the instance of Pyruvate-Oxidation by the first-subevent relation. Other relationships in this graph can be explained in an analogous manner. Here, the relationships such as first-subevent, subevent, agent, and base. are designed by the knowledge engineers (Chaudhri, Dinesh, and Inclezan 2013). This graph also represents cardinality constraints (e.g., there is a cardinality constraint that every Pyruvate-Oxidation has at least one Chemical-Entity as the value of the raw-material and result relationships).

The representation in KB_Bio_101 supports standard features such as class hierarchy, relation hierarchy, disjointness between classes, cardinality constraints and rules (Chaudhri et al. 2013b). The biologists encoded 11 chapters of a biology textbook (Reece et al. 2011) resulting in KB_Bio_101. *Inquire* supports question answering methods that can query KB_Bio_101(Chaudhri et al. 2013c; 2014b). For example, users can ask questions such as: "What are the steps of Pyruvate Oxidation? what is the difference between oxidation and reduction?"

Inquire is an integrated system that leverages question answering over KB_Bio_101 (Chaudhri et al. 2013a). Figure 2 presents a sample answer page that is a response to a concept description question in *Inquire*. In this answer, the sentence labeled as A is human authored, but sentences B-E are machine generated by the NLG system from the representation shown in Figure 1.

Sentence Classes Produced

We now consider different sentence classes produced by the current NLG system. The sentence classes considered here, obviously, do not cover all possible sentence classes that may be required. (A later section discusses the sentence classes that are not currently supported.)

S1. Super class statement: Sentences of this type state the super classes of a class. For example, the sentence marked

as B in Figure 2 is an example of a super class statement.

S2. Property value statement: Sentences of this type state a property value. In most cases, such sentences are short fragments. The sentence fragment marked C in Figure 2 is an example of a property value statement. In some cases, property value statements can lead to complete sentences (e.g., "The denseness of a bio membrane is inversely proportional to the fluidity of the bio membrane.").

S3. Event summary statement: Sentences of this type summarize an event and its participants. Three sentences that are in the section marked E are event summary statements. The first and the third sentences in that section are subtly different: although the first sentence begins by "Chemical reaction consumes ...", the second sentence has a parenthetic construction that begins by "Chemical reaction – a multi-enzyme ...". These two sentences each have a different form because for the second chemical reaction in Figure 1, the representation specifies an explicit agent relationship leading to a parenthetic construction. The event summary statements have two further special cases.

S3a. Event summary in the context of an entity: Sentences of this type summarize an event that occurs only in the context of a specific entity. For example, "In liver cell, alcohol, a barbiturate and a drug are detoxified by a liver cell using protein enzymes in a smooth endoplasmic reticulum." Here the detoxification event happens only in a liver cell.

S3b. Event summary in the context of another event: Sentences of this type summarize an event that occurs as a sub-step of another event. For example, "In secretion of insulin by pancreas cell, polypeptide and a water molecule are converted by a bound ribosome and a tRNA using an amino acid, a GTP and two monomers." Here, the conversion happens as a sub-step of the larger process of secretion of insulin by a pancreas cell.

S4. Energy statements: Sentences of this type state energy change during a reaction. The second sentence marked D in Figure 2 is an energy statement.

S5. Role example statement: Sentences of this type begin with an event and a specific role that an entity in that event plays. An example of such a sentence is: "Oxidation – an electron transfers from an acetyl CoA to oxaloacetate. Here, the acetyl CoA is an electron donor. "

S6. Spatial statement: Sentences of this type describe a spatial relationship between two entities. For example, "A glycoprotein is inside a phospholipid bilayer."

S7. Solute and solvent identification: Sentences of this type are applicable when presenting chemical solutions and they identify which entity is a solute vs a solvent. For example, when queried for the parts of a solution, the system returns the following parts: "a chemical that acts as a solvent and a chemical that acts as a solute."

For each of the sentence class shown above, the system always produces multiple alternative realizations that are ranked to select the most appropriate realization. Alternative realizations can emphasize different words in a sentence. For example, consider the following two different realizations E1 and E2: (E1) Cellulose synthase converts a chemical in a cell to cellulose; (E2) Chemicals in a cell are converted into cellulose by cellulose synthase. Here, E1 emphasizes cellulose synthase, and E2 emphasizes conversion into cellulose. Choice of realization is context-sensitive. For example, if the question asks for a concept summary of cellulose synthase, E1 will be preferred. If the question asks for how chemicals are converted to cellulose, E2 will be preferred.

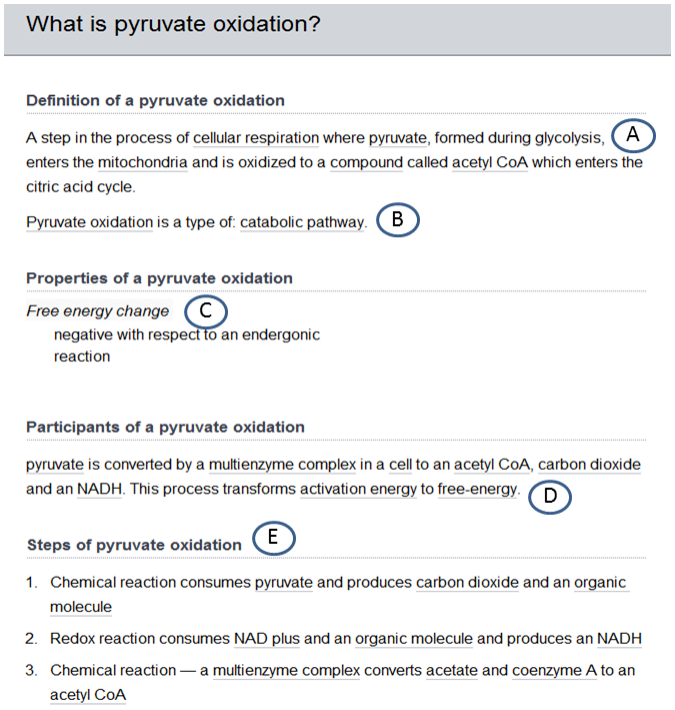

Figure 2: An Example Answer in *Inquire*

Let us consider the space of sentences in relation to the knowledge representation language, ontology and linguistic forms. A representation language includes taxonomic axioms (e.g., class-subclass relationships); cardinality constraints; disjointness statements; slot values; etc. Each of these axiom forms leads to a different input and corresponds to a different sentence type. In the examples above, S1 presents a taxonomic axiom, and S2 presents a slot value that is a property. Although we do not generate sentences for cardinality constraints on their own, some event descriptions do take the number constraints into account (e.g., *Energy investment phase of glycolysis in a cell using 2 ATP and glucose resulting in 2 ADPs and 2* glyceraldehyde-3-phosphate. *This sentence includes the cardinality constraints on* ATP, ADP, *and* glyceraldehyde-3-phosphate.) In our current system, we have chosen not to render cardinality constraints and disjointness statements as standalone sentences because they can as easily be shown using a tabular display.

Key distinctions in the ontology are Event, Entity, and Role. Each of these distinctions requires a different sentence type. In the examples above, sentences S3, S3a, and S3b correspond to events; S5 to roles; and S4, S6, and S7 to entities. Multiple sentence types for events and for entities illustrate that each ontological category needs to be handled differently, with several special cases covered.

Orthogonal to the space of representational axiom forms and ontological categories are the linguistic forms of sentences. For every sentence, our system generates both an active and a passive form. Sentence forms S4 and S5 always involve two sentences so that the second sentence adds further detail about the information introduced in the first sentence. Sentence S7 is an example of parenthetical construction in which an extra level of detail is added in the sentence.

The range of representational axioms, ontological categories, and linguistic phenomena considered here motivate and require a powerful NLG system of the kind we discuss here. It constitutes a substantial advance in the capabilities of the NLG systems built to date.

Overview of the NLG System

Synthesizing sentences from the KB involves three major computations: content selection, surface realization, and sentence finalization. These computations use three linguistic resources: concept word frames, a grammar, and a morphological lexicon. We first explain the linguistic resources, followed by a discussion of the computations involved in generating the sentences.

Concept word frames provide a mapping from an AURA concept to words and their syntactic requirements. For example, the concept Pyruvate-Oxidation is mapped to the verb *convert*, and its grammatical subject is indicated as agent, and grammatical object as raw-material. We capture the usage of preposition: for example, the *result* relation is associated with the preposition "to". The concept word frames are curated by the knowledge encoder biologists.

The generation grammar consists of a set of Tree Adjoining Grammar elementary trees (Joshi and Schabes 1997). It contains at least four trees for each verb: canonical, active, passive, and a complex noun phrase. Different combinations of input parameters can lead to additional trees. A tree can be thought of as a template with underspecified participant slot names and underspecified prepositions.

The morphological lexicon contains root forms for words and phrases and their inflected forms. For nouns, it contains singular, plural, the type of the noun (e.g., whether it is a mass noun, count noun, proper name, acronym), and its determiner. For verbs, it contains present tense form, past tense form, progressive form, nominalization, etc.

We now address the computations involved in synthesizing a sentence. The first step in the process is content selection. The goal of the content selection is to extract appropriate information from the KB that is needed for synthesizing an answer. For example, for producing the answer shown in Figure 2, the content selection module needs to extract the information required to produce all the sentences A-E. The content selection module is part of the query answering subsystem and produces an *answer bundle*, which is passed on to an answer presentation module. The answer presentation produces the page shown in Figure 2 by making appropriate calls to the NLG module.

When the NLG module is invoked, it processes the events to detect duplicate participants, aggregates triples that have an identical range, handles constraints, and deletes implied participants. Duplicate participants do not lead to fluent output, and it therefore picks one of the participant relations drawing on a heuristic order that captures their importance. The relation values that involve the same relation are aggregated so that they lead to a coordinated sentence (e.g., in sentence S3a, the sentence fragment "In liver cell, alcohol, a barbiturate and a drug are detoxified ...", is a result of range aggregation.). The constraint handling enables the system to produce sentence fragments such as: "Selective degradation by a protein enzyme in a cell using a water molecule resulting in at least 2 monomers, an organic molecule and a polypeptide. Here, the constraint that the selective degradation produces at least two monomers is immediately followed by two monomers. Implicit participants are factored out when the name of the verb already implies a participant (e.g., polymerization implies that the result is a polymer).

The task of surface realization, which is to create sentence templates consists of three main steps: (1) lexical selection in which elementary trees are selected from the grammar, (2) tree assembly in which the selected elementary trees are combined into derived trees, with the tree combinations that fail filtered out, and (3) ranking of the generated outputs according to optimality-theoretic ranking constraints.

The sentence finalization module carries out two main tasks: morphological realization and referring expression generation. The morphological realization module performs verb and noun inflection by looking up the morphological lexicon and, if no entry is found, by using morphological rules. The referring expression generation module fills the holes in the sentence templates from the previous step by generating noun phrases based on the discourse context.

A more detailed technical description of the NLG system is available elsewhere (Banik, Kow, and Chaudhri 2013).

Problem of Overly General Classes

We define an overly general class as any class name that is not in the vocabulary of the end user (i.e., is not introduced in the textbook). The biologist encoders of the KB curate a list of classes that are considered overly general (e.g., Tangible-Entity, Spatial-Entity). Many situations exist where such classes are present in the domain-specific rules of KB_Bio_101. When such concept names are used in *Inquire*, they lead to negative feedback from the users. Consider the following sentences E3 and E4: (E3) A bio membrane blocks a spatial entity; (E4) Energy is stored at a tangible entity. Sentence E3 is generated for a Block event in which the value of the object relationship has been specified as Spatial-Entity. Sentence E4 is produced for a Store event for which the value of the base relationships has been specified as Tangible-Entity.

Occurrence of an overly general class from an upper ontology in an answer leads to negative feedback because the students have never been introduced to such classes. Such classes cannot be eliminated from the upper ontology, because they are essential for cross-domain applicability. We see below that in some cases overly general classes exist in the KB because the knowledge in the textbook itself is abstract and too general. Under such circumstances, the overly

general classes need to be presented to the users, but a suitable NLG scheme must be designed for that presentation. In our analysis below we show that at least four reasons account for the existence of overly general classes that an NLG system needs to handle.

Reasons for the use of overly general concepts

In a recent version of KB_Bio_101, we analyzed all the descriptions of functions of entities, and found 3,148 occurrences. Because KB_Bio_101 contains 3,700 classes in total, this suggests that, on average, an overly general function will appear for 85% of the classes. We analyzed a sample of these occurrences and classified the reasons of their occurrence into the four broad categories discussed below. The categorization we present does not change the definition of an overly general class; it provides insight about why they occur, and justification for why they cannot be eliminated from the KB.

Knowledge gaps The first reason for using overly general classes is knowledge gaps (i.e., the textbook contains more specific knowledge than is encoded in the KB). On occasion, such gaps arise because more specific knowledge is present in a later chapter of the textbook that has not yet been encoded. Because we had encoded only 11 of the textbook's 55 chapters, we had no systematic way to eliminate such knowledge gaps.

Abstraction and Generalization The textbook uses abstraction and generalization for explaining things. For example, the textbook says that a membrane allows different things to pass through it. Only in a more specific context are we given information about which things it might allow to pass through. For example, for diffusion, the membrane allows solute molecules to pass through. In some situations, the textbook states knowledge only at an abstract level (e.g., a cell wall blocks some things, but the textbook does not offer information about which specific things are blocked). Thus, in such situations, the representation uses an overly general class such as Physical-Object to indicate that a cell wall blocks things. Use of general classes for such abstract situations is unavoidable.

Inherited values from upper ontology For many of the concepts in the upper ontology, we specify default values for some of the relationships. Some of these relationships are required whereas others are optional. The first reason for providing the default values is to clarify to the domain experts what they need to specify for each concept. For example, for the concept Move-Into, object and base are required relationships, whereas origin and destination are optional relationships. The second reason for providing default values is that rules exist that use these values (e.g., a rule exists in the KB that requires that the value of the base relationship be different from the value of the origin and destination relationships). When using this concept in the domain-specific KB, specializing all of the slot values is occasionally impossible, especially the optional slot values, but these inherited values remain in the KB.

General classes with domain specific values Many examples exist of the use of overly general classes that are specialized by adding more specific slot values, but do not have a domain-specific name. For example, consider a Detoxification for which the value of the object relationship is Tangible-Entity. In this case, the Tangible-Entity has a relation plays with a value of Toxin (i.e., the Tangible-Entity plays the role of a toxin). Thus, in this case, we have specialized Tangible-Entity by adding a value restriction on the plays relationship, but we have not created a domain-specific class name for it.

Handling overly general classes in NLG output

One could argue that the correct approach for handling overly general classes is to invest the knowledge engineering effort necessary to eliminate them from the KB. Our practical experience, however, suggests that no matter how many resources we invest, overly general classes will remain. Only some knowledge gaps can be eliminated. Because the use of abstractions is a fundamental aspect of textbook descriptions, abstract concepts must be handled in the knowledge representation and, hence, in the output of the NLG system. More generally, the NLG system needs to be robust because it is unreasonable to expect a perfect input all the time.

Our approach for handling overly general classes caused by KB gaps is to first use their appearance in the NLG output as a debugging tool. We implemented a tool that identifies such classes in the NLG output and show them as warnings to the knowledge encoder. The knowledge encoder can then pay special attention to such classes and appropriately specialize them if possible. In spite of supporting such a tool, we acknowledge that a large evolving KB will never be complete, and general classes that have not been specialized will always remain.

Our approach for classes that may have domain-specific values was to strengthen our content-selection algorithm. For an overly general class, we check its relation values, and if we find at least one relation value that is not overly general (i.e., domain-specific), we extract it as part of content selection. With the availability of a domain-specific relation value, the NLG system can generate a good sentence that is acceptable to the end users. For the example considered earlier, instead of generating a sentence such as Tangible-Entity *playing the role of a* Toxin *is detoxified*, the system can generate the sentence *Toxins are detoxified*.

To handle the overly general classes that cannot be further specialized, we select additional content from the KB and then summarize it as follows: whenever we have a statement that contains overly general classes, we first present it in English followed by several examples in which the overly general class has been further specialized. The overly general class is given a name that is friendly to the end users. We illustrate our approach by taking the example of the functions of bio membrane.

In Figure 3, we show a summary of the functions of a bio membrane. At the most abstract level, its functions are "A chemical is moved through a bio membrane and a chemical is blocked by a bio membrane." These general functions are further specialized in the subclasses of bio membrane.

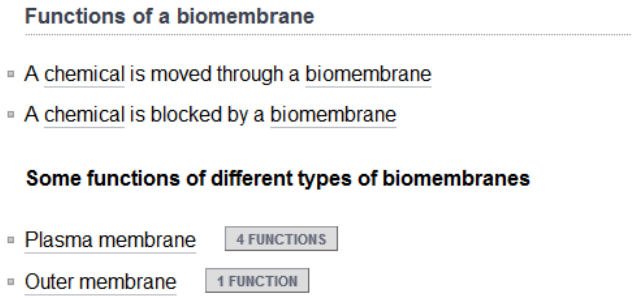

Figure 3: Summary of the Functions of Bio Membrane

Because numerous bio membrane specializations may exist, we organize bio membrane functions by each subclass that further specializes the function. The detailed functions can be viewed by clicking on the drill down button labeled as "4 functions". The result of expanding all the functions is shown in Figure 4. In Figure 4, we can see that one of the specialized functions of a Plasma-Membrane is to transport organic molecules. Here, transport is a specialization of Move-Through. Another function of Plasma-Membrane is to recognize other cells. This is a new function that is defined only for Plasma-Membrane, and is not a specialization of any of the inherited functions from bio membrane. In this case, such taxonomic organization of functions helps a user understand how special forms of bio membrane could have additional functions. Presenting the overly general statements along with their specialization affords much better insight into the knowledge in the KB.

Future Work

A thorough evaluation of the approach for handling overly general classes presented in the previous section is a problem open for future research. We present some data below on the functional evaluation of the use of NLG system in the overall application, and identify several functional improvements for its further evolution.

To measure the question answering performance of *Inquire*, we assembled a test suite of 1961 questions spread over the first 11 chapters of the textbook. Those questions contained instantiations of 25 different question templates including concept description questions and several other templates (Chaudhri et al. 2014b; Chaudhri, Dinesh, and Heller 2013). The teacher biologists, who were responsible for quality assurance, and were not the biologists who entered knowledge into the KB, graded 76% of the answers to be correct. Such a high rate of correctness would not have been possible without a well-functioning NLG module. The primary success in using NLG so far has been in presenting event descriptions in English that significantly enhanced the usability of the KB.

Even though our NLG module functions well, room for improvement exists. Several sentence forms required for answers are not currently handled. An example is a relationship question: "What is the relationship between a carbohydrate and a bio membrane?" The system displays an answer to this question graphically (Chaudhri et al. 2013a), but one that is not easily understood by the end users. The system also uses the KB to generate questions (Chaudhri et al. 2014a) that are currently being synthesized into English using question templates. Clearly that approach is not scalable and requires the NLG to be generalized. Finally, the system must generate paragraphs and chapters not just sentences.

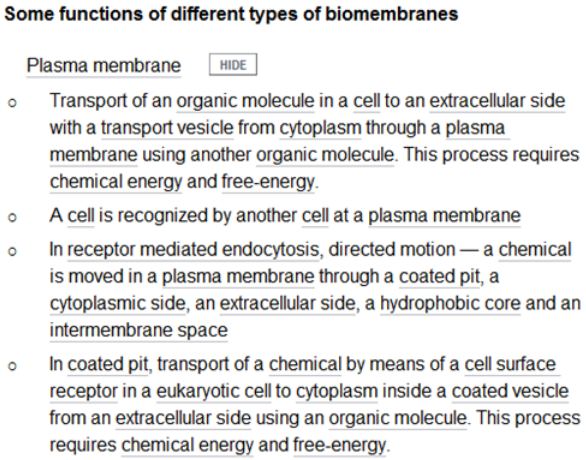

Figure 4: Details of the Functions of Biomembrane

As part of an NLG challenge called KBGen (Banik et al. 2012; Banik, Gardent, and E.Kow 2013), four systems competed on generating sentences from `KB_Bio_101`. When human experts rated the outputs on a scale of -50 to +50, the outputs received a score of 32.49 for fluency, and 37.5 for grammatical correctness. The results of KBGen have shown that even state-of-the-art NLG systems need to improve before they can produce human-like English from a KB and are already inspiring the development of novel techniques (Gyawali and Gardent 2014).

Summary

In this paper, we argued that an ability to synthesize novel English sentences is a critical capability for question answering systems that go beyond Watson. To generate sentences that have not been previously authored by a human, the system needs to work from a conceptual model of domain knowledge. Our work differs from prior work on generation systems for ontologies because of the variety of different sentence forms it produces and because the expressiveness of knowledge representation that it works from. We illustrated this by giving the range of sentences generated by the system in the context of an electronic textbook application. We presented an overview of the computations needed to integrate the NLG into the overall system, and we considered in detail the problem of presenting overly general classes. Mundane and specific as the problem may seem, it is an important requirement for a robust NLG system.

Our work brings new insights into the KB verbalization systems in two ways: First, KB verbalization systems must provide for appropriate content-selection schemes so that the

overly general classes can be displayed to end users in terms meaningful to them. Second, the surface realizations modules for KB verbalization must look beyond presenting taxonomic axioms because richness in linguistic construction is required when considering a broad spectrum of concepts.

A futuristic goal for intelligent textbooks is to first create a conceptual model of the textbook, and then generate the textbook content from that model. That new model for textbook authoring could facilitate automatic customization of textbooks for grade levels, across teaching standards in different states, and also in different languages. The current system capabilities are a stepping stone toward such a goal.

Acknowledgments

This work has been funded by Vulcan Inc. We thank the members of the AURA development team for their contributions to this work.

References

Androutsopoulos, I.; Lampouras, G.; and Galanis, D. 2013. Generating natural language descriptions from OWL ontologies: The NaturalOWL system. *Journal of Artificial Intelligence Research* 48:671–715.

Baget, J.-F.; Leclère, M.; Mugnier, M.-L.; and Salvat, E. 2011. On rules with existential variables: Walking the decidability line. *Artificial Intelligence* 175(9):1620–1654.

Banik, E.; Gardent, C.; Scott, D.; Dinesh, N.; and Liang, F. 2012. Text generation for knowledge bases as a new shared task. In *INLG 2012, The seventh International Natural Language Generation Conference*.

Banik, E.; Gardent, C.; and E.Kow. 2013. The KBGen challenge. In *14th European Workshop on Natural Language Generation (ENLG)*.

Banik, E.; Kow, E.; and Chaudhri, V. K. 2013. User-controlled, robust natural language generation from an evolving knowledge base. In *ENLG 2013 : 14th European Workshop on Natural Language Generation*.

Birnbaum, L. 2013. Telling stories on internet scale.

Borgo, S., and Masolo, C. 2009. Foundational choices in DOLCE. In Staab, S., and Studer, R., eds., *Handbook on Ontologies*. Springer, second edition.

Chaudhri, V. K.; Cheng, B.; Overholtzer, A.; Roschelle, J.; Spaulding, A.; Clark, P.; Greaves, M.; and Gunning, D. 2013a. Inquire Biology: A textbook that answers questions. *AI Magazine* 34(3):55–72.

Chaudhri, V. K.; Heymans, S.; Wessel, M.; and Tran, S. C. 2013b. Object-Oriented Knowledge Bases in Logic Programming. In *Technical Communication of International Conference in Logic Programming*.

Chaudhri, V. K.; Heymans, S.; Wessel, M.; and Tran, S. C. 2013c. Query answering in object-oriented knowledge bases in logic programming. In *Workshop on ASP and Other Computing Paradigms*.

Chaudhri, V. K.; Clark, P. E.; Overholtzer, A.; and Spaulding, A. 2014a. Question generation from a knowledge base. In *19th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*.

Chaudhri, V. K.; Heymans, S.; Overholtzer, A.; Spaulding, A.; and Wessel, M. 2014b. Large-scale analogical reasoning. In *AAAI-2014 Conference*.

Chaudhri, V. K.; Dinesh, N.; and Heller, C. 2013. Conceptual models of structure and function. In *Second Annual Conference on Advances in Cognitive Systems*.

Chaudhri, V. K.; Dinesh, N.; and Inclezan, D. 2013. Three lessons in creating a knowledge base to enable explanation, reasoning and dialog. In *Second Annual Conference on Advances in Cognitive Systems*.

Chaudhri, V. K.; Wessel, M. A.; and Heymans, S. 2013. kb_bio_101: A challenge for OWL reasoners. In *The OWL Reasoner Evaluation Workshop*.

Chu-Carroll, J., and Fan, J. 2011. Leveraging wikipedia characteristics for search and candidate generation in question answering. In *In AAAI-2011 Conference*.

Gyawali, B., and Gardent, C. 2014. Surface realisation from knowledge-base. In *52nd Annual Meeting of the Association of Computational Linguistics*.

Joshi, A. K., and Schabes, Y. 1997. Tree-adjoining grammars. In *Handbook of formal languages*. Springer. 69–123.

Lenat, D. B. 1995. CYC: A large scale investment in knowledge infrastructure. *Communications of the ACM* 38(11):33–38.

Liang, S. F.; Stevens, R.; Scott, D.; and Rector, A. 2013. Ontoverbal: A generic tool and practical application to SNOMED CT. *International Journal of Advanced Computer Science & Applications* 4(6).

Reece, J. B.; Urry, L. A.; Cain, M. L.; Wasserman, S. A.; Minorsky, P. V.; and Jackson, R. B. 2011. *Campbell Biology*. Boston: Benjamin Cummings imprint of Pearson.

Spear, A. D. 2006. Ontology for the twenty first century: An introduction with recommendations. http://www.ifomis.org/bfo/documents/manual.pdf.

Stevens, R.; Malone, J.; Williams, S.; Power, R.; and Third, A. 2011. Automating generation of textual class definitions from OWL to English. *Journal of Biomedical Semantics* 2(Suppl 2):S5.