Appendix A. Annotation Guidelines

The corpus used in the development of MutationFinder[a] is annotated using the Knowtator plugin for the Protege Ontology Editor and Knowledge Acquisition System. (See Figure 1.) A new "mutation ontology" has been developed for this purpose, the components of which are summarized as follows:

- mutation event
  - deletion (subclass)
    - deleted element (slot for *biological sequence element, biological subsequence*, or *biological sequence position*)
  - insertion (subclass)
    - inserted element (slot for *biological sequence* or *biological sequence element)*
    - insertion start (slot for *biological sequence position*)
  - substitution (subclass)
    - wild type element (slot for *biological sequence element*, *biological sequence position*, or *biological subsequence*)
    - mutant element (slot for *biological sequence element* or *biological sequence*)
- biological sequence
  - polypeptide sequence (subclass)
  - sequence (slot for string)
- biological sequence position
- biological sequence element
  - amino acid (subclass)
    - Alanine, Ala, A
    - Glycine, Gly, G
    - etc.
  - position in sequence (slot for *biological sequence position*)
- biological subsequence
  - polypeptide subsequence (subclass)

[a]At present, MutationFinder leverages only a subset of the annotation data in the annotated corpus. (The MutationFinder distribution includes data files that have been generated from the Knowtator projects used in this annotation task, but have been preprocessed to simplify their contents.) As of this writing, the data sets provided with MutationFinder include only ordered lists of substitution events. Insertion and deletion events are not currently supported by the tool, and all information relating to how specific spans of text are associated with each substitution is absent. While the MutationFinder system itself is capable of conveying basic positional information about where substitution-event mentions begin and end in piece of text, the system's output cannot currently be used to automate the full annotation process described here.

- start element (slot for *biological sequence element* or a *biological sequence position*)
- end element (slot for *biological sequence element* or a *biological sequence position*)

A generic biological sequence may be thought of as a sequence of *sequence elements*. Mutation events are simply changes to these sequence elements. The ontology as presented here is capable of capturing only amino-acid sequences that appear in nature and the ways in which these sequences have been modified (mutated) experimentally. However, the ontology class structure outlined above is designed to accommodate generic biological sequences. In the future, nucleotide subclasses may easily be added to allow modeling of mutations at the pre-translational level. This focus in the current ontology merely reflects the initial focus of the MutationFinder project on proteins.

Thus, your task at present is to identify text elements that describe a mutation in a protein sequence. Any references to nucleic-acid mutations or sequences or sequence elements should *not* be captured during this annotation task.

To make this task less tedious, much of the text you review will have been preprocessed to contain text elements that have already been annotated automatically. It is important for you to verify each of these automatic annotations in addition to identifying mutation content that was missed by the automated process.

A.1. *Elements of the Ontology*

We now describe the various components of the mutation ontology in more detail. In the forthcoming discussion, we will refer several times to the following text sample:

> In this study, the structural characterization of the nitrogenase Fe protein with Asp at position 39 substituted by Pro provides an explanation. . . .

A.1.1. *Biological sequence position*

This class is used to capture the sequence position in the text. In our text sample above, for example, the string *39* should be annotated as a *sequence position*. This class has no slots (attributes). Use this class to annotate the position in the sequence only. The text spanned by this class should not include any information about *what* resides at this position. For example, confronted with the

> The experiment has shown Asp-213 of the L subunit (Asp-213) to be important. . . .

text string, you should create two *biological sequence position* annotations, each spanning one of the strings *213*. There should not be an annotation spanning *L* and *213* or *Asp-213*.

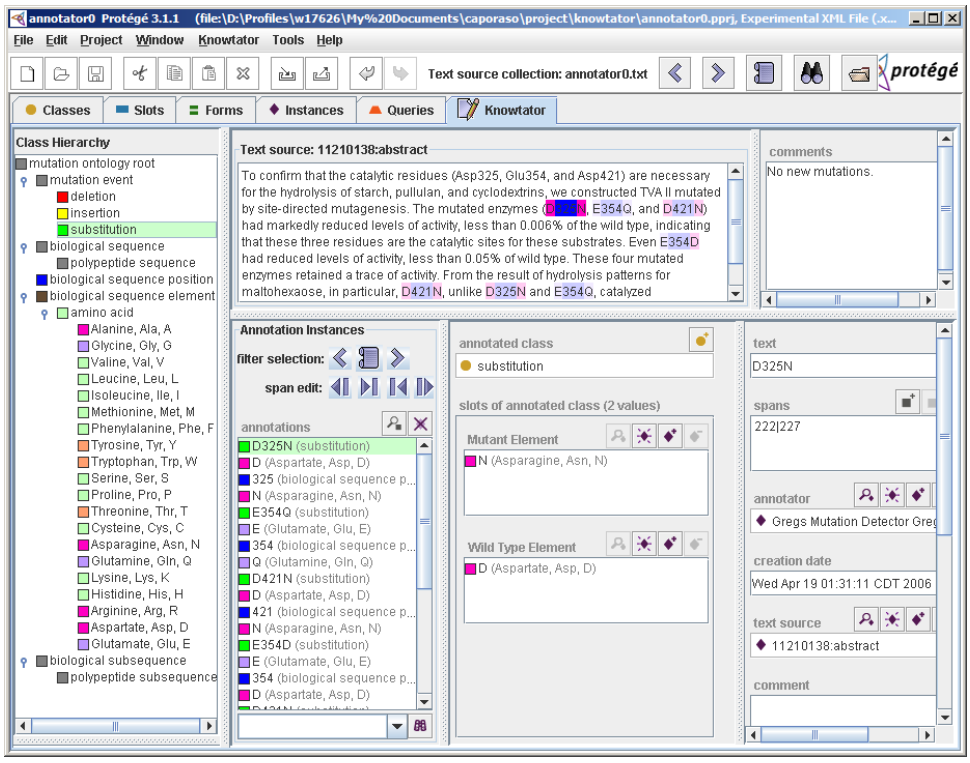

Fig. 1. Working with the mutation ontology in Knowtator/Protege.

A.1.2. *Biological sequence element*

This class comprises the base elements of a biological sequence. Currently, all 20 amino acids are represented as subclasses. This class has a single slot—*position in sequence*—which is constrained to be a *biological sequence position*. The *position in sequence* slot is an optional field that should only be filled if the position information is present.

You must make a distinction between sequence elements where we know the position and sequence elements that have no position. For example, in our initial text sample above, the annotation for *Asp* should have its *position in sequence* slot filled with the *biological sequence position* annotation for *39*, since the Asp residue is located at position 39 in the original sequence. The annotation for *Pro*, however, should have an empty *position in sequence* slot because it is not associated with a position in any sequence that occurs in nature. (*Pro* is only associated with position 39 by way of its participation in a mutation event.)

A.1.3. *Biological sequence*

This class is used to represent a biological sequence that is explicitly described in the text by a string, e.g. *ARYTGP.* This class is primarily used to fill the *inserted element* slot of an *insertion* event. However, it could also be used as the *mutant element* in a *substitution* event. Currently, *polypeptide sequence* is the only subclass of this class. There is a single slot for the *biological sequence* class called *sequence*, which will accept any string.

A.1.4. *Biological subsequence*

This class is used to represent a part of a sequence. The difference between a *biological sequence* and a *biological subsequence* is important. A *biological sequence* is a complete, free-standing sequence, and as such it has no associated "position." On the other hand, a *biological subsequence* only exists within the context of a complete *biological sequence*. The subsequence is defined by the *biological sequence positions* that define its range within a particular *biological sequence* (before any mutation event occurs). The *biological subsequence* class therefore has two slots, *start element* and *end element*. Both slots are constrained to be either a *biological sequence elements* or a *biological sequence positions*. Whenever possible, the more descriptive *biological sequence element* is used as the slot filler. However, if only the sequence position is used (as is often the case with deletion events), the *biological sequence position* is used to fill the slots.

For example, consider the text string *Delta109-114*, referring to a *deletion* event. For this text, you first create a *biological subsequence* denoting the span *109-114* within the original sequence. To do this, annotations of the *biological sequence positions* 109 and 114 are made, and then an annotation spanning *109-114* is created as a *biological subsequence* (or *polypeptide subsequence* if it is known that this is a protein sequence). The slot fillers for the new *biological* or *polypeptide subsequence* are the *sequence position* for *109* as the *start element* and the *sequence position* for *114* as the *end element.*

A.1.5. *Mutation event*

The subclasses of this class are used to annotate references to mutation events in the text. Three subclasses are included in the ontology: *substitution*, *deletion*, and *insertion*.

Substitution. This *mutation event* subclass is by far the most commonly referenced mutation in the biomedical literature. Typically, substitutions involve the replacement of one *biological sequence element* with another. The *substitution* class has two slots, *wild type element* and *mutant element*. The *wild type element* slot is constrained to be an instance of a *biological sequence element*, *biological sequence position*, or *biological subsequence*. The *mutant element* slot is constrained to be an instance of *biological sequence element* or *biological sequence*. When annotating

a *substitution* event, either the trigger word or the entire standardized representation should be tagged as the *substitution*. For example, in our primary text sample above, the word *substituted* would be tagged as the trigger word for the *substitution* event. Some examples of standardized representations, which should be tagged in their entirety as *substitutions*, include the following:

- `Met 213----Ala`
- `Met213Ala`
- `M213A`
- `Met213-->Ala`

Deletion. This *mutation event* subclass is used to represent deletion mutations. These mutations are typically represented by *deletion* or *delta* or *del* followed by a range of numbers, e.g. *13-17*. This class has a single slot, *deleted element*, which is constrained to be either an instance of a *biological sequence element*, *biological subsequence*, or *biological sequence position*.

Insertion. This *mutation event* subclass is used to represent insertions into the original sequence. Insertions seem to be the rarest of the three *mutation event* subclasses in the literature. This class has two slots, *inserted element* and *insertion start*. The *inserted element* slot must be an instance of a *biological sequence* or *biological sequence element*. The *insertion start* slot is constrained to be an instance of a *biological sequence position*.

A.2. *General Principles of Annotation*

- Always assign the most specific class that you can. For example, given the text *Asn*, it would be correct to annotate this as a *biological sequence element*, but it makes more sense to annotate it as the more specific *Asparagine, Asn, N*.
- When you come across a mutation event in the text, annotate the "trigger" word as the specific *mutation event* that it represents. By "trigger" word, we mean the word (or words) that caused you to think that a mutation event is being described. If you encounter a standardized mutation reference (e.g. *wNm*), annotate the entire token as the "trigger" word.
- Single residues are annotated as the specific amino acid they reference.
- The position of a mutation event is annotated as a *biological sequence position*. The text for this annotation is usually an integer, but it could also be something like *C-terminal*.
- In a *substitution* event, assign a *biological sequence position* only to the *wild type element*. The *mutant element* is coming from parts unknown and has no meaningful sequence position.
- If you see the word *or* separating two amino acids, this probably calls for the annotation of two distinct mutation events. Consider the following:

  The replacement of Trp 109 by Phe or Ala. . . .

This would result in the annotation of two *substitution* events.

- If your are unsure about how to annotate a given section of text, make note of this in the **Comment** area in the interface. This will make it easy to find the item later when you meet with other annotators to build consensus. We believe that these annotation guidelines are comprehensive, but it is likely that this document will continue to evolve as more annotators perform this task.

A.3. *Examples*

Finally, we present a few examples that describe the details of the annotation procedure in practice.

Simple substitution event. Consider the following text:

> The protein with Asp at position 39 substituted by Asn in. . . .

Here, the word *substituted* is annotated as a *substitution* event, the *Asp* term as an *Aspartate, Asp, D* instance of *amino acid*, the *Asn* term as an *Asparagine, Asn, N* instance, and the *39* as a *biological sequence position*. When this is in place, the *biological sequence position* element (for *39*) is used to fill the *position in sequence* slot of the *Asp* annotation (because *Asp* is the wild type residue in this example). Then the slots for the *substitution* frame are filled, entering the *Asp* element in the *wild type element* slot and the *Asn* element in the *mutant element* slot.

Splitting tokens. Sometimes it will be necessary to split tokens manually:

> . . . .with Asp129 substituted by Glu provides. . . .

Here, Knowtator will select *Asp129* as a single token by default. You must use the arrows located under **Annotation Instances** to change the span after you have selected it so that you have the substring *Asp* refer to *Aspartate, Asp, D* and *129* refer to *biological sequence position*.