HLT-NAACL 2004 Workshop: Biolink 2004,
Linking Biological Literature, Ontologies and Databases, pp. 53-60.
Association for Computational Linguistics.

A Large Scale Terminology Resource for Biomedical Text Processing

**Henk Harkema, Robert Gaizauskas, Mark Hepple, Angus Roberts,
Ian Roberts, Neil Davis, Yikun Guo**
Department of Computer Science, University of Sheffield, UK
biomed@dcs.shef.ac.uk

Abstract

In this paper we discuss the design, implementation, and use of Termino, a large scale terminological resource for text processing. Dealing with terminology is a difficult but unavoidable task for language processing applications, such as Information Extraction in technical domains. Complex, heterogeneous information must be stored about large numbers of terms. At the same time term recognition must be performed in realistic times. Termino attempts to reconcile this tension by maintaining a flexible, extensible relational database for storing terminological information and compiling finite state machines from this database to do term look-up. While Termino has been developed for biomedical applications, its general design allows it to be used for term processing in any domain.

1 Introduction

It has been widely recognized that the biomedical literature is now so large, and growing so quickly, that it is becoming increasingly difficult for researchers to access the published results that are relevant to their research. Consequently, any technology that can facilitate this access should help to increase research productivity. This has led to an increased interest in the application of natural language processing techniques for the automatic capture of biomedical content from journal abstracts, complete papers, and other textual documents (Gaizauskas et al., 2003; Hahn et al., 2002; Pustejovsky et al., 2002; Rindflesch et al., 2000).

An essential processing step in these applications is the identification and semantic classification of technical terms in text, since these terms often point to entities about which information should be extracted. Proper semantic classification of terms also helps in resolving anaphora and extracting relations whose arguments are restricted semantically.

1.1 Challenge

Any technical domain generates very large numbers of terms – single or multiword expressions that have some specialised use or meaning in that domain. For example, the UMLS Metathesaurus (Humphreys et al., 1998), which provides a semantic classification of terms from a wide range of vocabularies in the clinical and biomedical domain, currently contains well over 2 million distinct English terms.

For a variety of reasons, recognizing these terms in text is not a trivial task. First of all, terms are often long multi-token sequences, e.g. *3-methyladenine-DNA glycosylase I*. Moreover, since terms are referred to repeatedly in discourses there is a benefit in their being short and unambiguous, so they are frequently abbreviated and acronymized, e.g. *CvL* for *chromobacterium viscosum lipase*. However, abbreviations may not always occur together with their full forms in a text, the method of abbreviation is not predictable in all cases, and many three letter abbreviations are highly overloaded. Terms are also subject to a high degree of orthographic variation as a result of the representation of non-Latin characters, e.g. *a-helix* vs. *alpha-helix*, capitalization, e.g. *DNA* vs. *dna*, hyphenation, e.g. *anti-histamine* vs. *antihistamine*, and British and American spelling variants, e.g. *tumour* vs. *tumor*. Furthermore, biomedical science is a dynamic field: new terms are constantly being introduced while old ones fall into disuse. Finally, certain classes of biomedical terms exhibit metonomy, e.g. when a protein is referred to by the gene that expresses it.

To begin to address these issues in term recognition, we are building a large-scale resource for storing and recognizing technical terminology, called Termino. This resource must store complex, heterogeneous information about large numbers of terms. At the same time term recognition must be performed in realistic times. Termino attempts to reconcile this tension by maintaining a flexible, extensible relational database for storing terminological information and compiling finite state machines from this database to do term look-up.

1.2 Context

Termino is being developed in the context of two ongoing projects: CLEF, for Clinical E-Science Framework (Rector et al., 2003) and myGrid (Goble et al., 2003). Both these projects involve an Information Extraction component. Information Extraction is the activity of identifying pre-defined classes of entities and relationships in natural language texts and storing this information in a structured format enabling rapid and effective access to the information, e.g. Gaizauskas and Wilks (1998), Grishman (1997).

The goal of the CLEF project is to extract information from patient records regarding the treatment of cancer. The treatment of cancer patients may extend over several years and the resulting clinical record may include many documents, such as clinic letters, case notes, lab reports, discharge summaries, etc. These documents are generally full of medical terms naming entities such as body parts, drugs, problems (i.e. symptoms and diseases), investigations and interventions. Some of these terms are particular to the hospital from which the document originates. We aim to identify these classes of entities, as well as relationships between such entities, e.g. that an investigation has indicated a particular problem, which, in turn, has been treated with a particular intervention. The information extracted from the patient records is potentially of value for immediate patient care, but can also be used to support longitudinal and epidemiological medical studies, and to assist policy makers and health care managers in regard to planning and clinical governance.

The myGrid project aims to present research biologists with a unified workbench through which component bioinformatic services can be accessed using a workflow model. These services may be remotely located from the user and will be exploited via grid or web-service channels. A text extraction service will form one of these services and will facilitate access to information in the scientific literature. This text service comprises an off-line and an on-line component. The off-line component involves pre-processing a large biological sciences corpus, in this case the contents of Medline, in order to identify various biological entities such as genes, enzymes, and proteins, and relationships between them such as structural and locative relations. These entities and relationships are referred to in Medline abstracts by a very large number of technical terms and expressions, which contributes to the complexity of processing these texts. The on-line component supports access to the extracted information, as well as to the raw texts, via a SOAP interface to an SQL database.

Despite the different objectives for text extraction within the CLEF and myGrid projects, many of the technical challenges they face are the same, such as the need for extensive capabilities to recognize and classify biomedical entities as described using complex technical terminology in text. As a consequence we are constructing a general framework for the extraction of information from biomedical text: AMBIT, a system for acquiring medical and biological information from text. An overview of the AMBIT logical architecture is shown in figure 1.

The AMBIT system contains several engines, of which Termino is one. The Information Extraction Engine pulls selected information out of natural language text and pushes this information into a set of pre-defined templates. These are structured objects which consists of one or more slots for holding the extracted entities and relations. The Query Engine allows users to access information through traditional free text search and search based on the structured information produced by the Information Extraction Engine, so that queries may refer to specific entities and classes of entities, and specific kinds of relations that are recognised to hold between them. The Text Indexing Engine is used to index text and extracted, structured information for the purposes of information retrieval. The AMBIT system contains two further components: an interface layer, which provides a web or grid channel to allow user and program access to the system; and a database which holds free text and structured information that can be searched through the Query Engine.

Termino interacts with the Query Engine and the Text Indexing Engine to provide terminological support for query formulation and text indexation. It also provides knowledge for the Information Extraction Engine to use in identifying and classifying biomedical entities in text. The Terminology Engine can furthermore be called by users and remote programs to access information from the various lexical resources that are integrated in the terminological database.

2 Related Work

Since identification and classification of technical terms in biomedical text is an essential step in information extraction and other natural language processing tasks, most natural language processing systems contain a terminological resource of some sort. Some systems make use of existing terminological resources, notably the UMLS Metathesaurus, e.g. Rindflesch et al. (2000), Pustejovski et al. (2002); other systems rely on resources that have been specifically built for the application, e.g. Humphreys et al. (2000), Thomas et al. (2000).

The UMLS Metathesaurus provides a semantic classification of terms drawn from a wide range of vocabularies in the clinical and biomedical domain (Humphreys et al., 1998). It does so by grouping strings from the source vo-

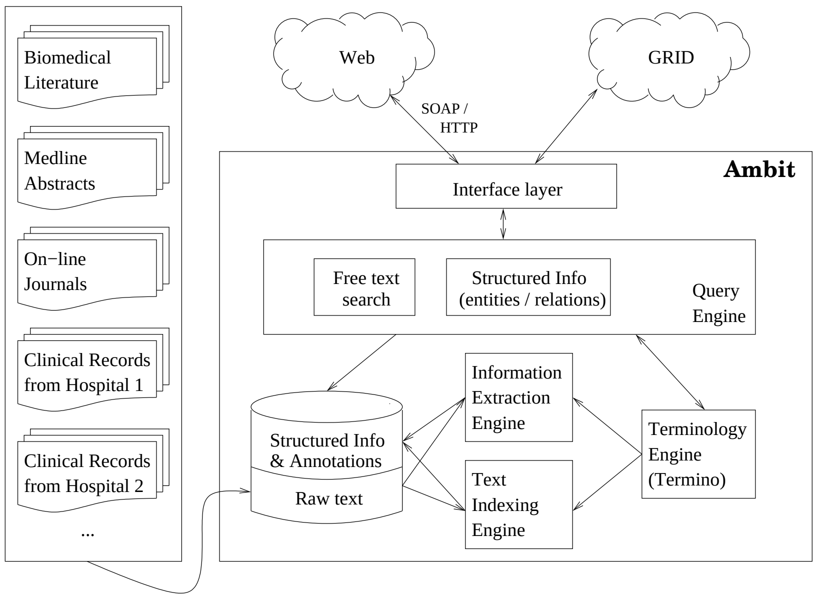
Biomedical Literature
Medline Abstracts
On−line Journals
Clinical Records from Hospital 1
Clinical Records from Hospital 2
...
Web
GRID
SOAP / HTTP
Ambit
Interface layer
Free text search
Structured Info (entities / relations)
Query Engine
Structured Info & Annotations
Raw text
Information Extraction Engine
Text Indexing Engine
Terminology Engine (Termino)

Figure 1: AMBIT Architecture

cabularies that are judged to have the same meaning into concepts, and mapping these concepts onto nodes or semantic types in a semantic network. Although the UMLS Metathesaurus is used in a number of biomedical natural language processing applications, we have decided not to adopt the UMLS Metathesaurus as the primary terminology resource in AMBIT for a variety of reasons.

One of the reasons for this decision is that the Metathesaurus is a closed system: strings are classified in terms of the concepts and the semantic types that are present in the Metathesaurus and the semantic network, whereas we would like to be able to link our terms into multiple ontologies, including in-house ontologies that do not figure in any of the Metathesaurus' source vocabularies and hence are not available through the Metathesaurus. Moreover, we would also like to be able to have access to additional terminological information that is not present in the Metathesaurus, such as, for example, the annotations in the Gene Ontology (The Gene Ontology Consortium, 2001) assigned to a given human protein term. While the terms making up the the tripartite Gene Ontology are present in the UMLS Metathesaurus, assignments of these terms to gene products are not recorded in the Metathesaurus. Furthermore, as new terms appear constantly in the biomedical field we would like to be able to instantly add these to our terminological resource and not have to wait until they have been included in the UMLS Metathesaurus. Additionally, some medical terms appearing in patient notes are hospital-specific and are unlikely to be included in the Metathesaurus at all.

With regard to systems that do not use the UMLS Metathesaurus, but rather depend on terminological resources that have been specifically built for an application, we note that these terminological resources tend to be limited in the following two respects. First, the structure of these resources is often fixed and in some cases amounts to simple gazetteer lists. Secondly, because of their fixed structure, these resources are usually populated with content from just a few sources, leaving out many other potentially interesting sources of terminological information.

Instead, we intend for Termino to be an extensible resource that can hold diverse kinds of terminological information. The information in Termino is either imported from existing, outside knowledge sources, e.g. the Enzyme Nomenclature (http://www.chem.qmw.ac.uk/iubmb/enzyme/), the Structural Classification of Proteins database (Murzin et al., 1995), and the UMLS Metathesaurus, or it is induced from on-line raw text resources, e.g. Medline abstracts. Termino thus provides uniform access to terminological information aggregated across many sources. Using Termino removes the need for multiple, source-specific terminological components within text processing systems that employ multiple terminological resources.

3 Architecture

Termino consists of two components: a database holding terminological information and a compiler for generating term recognizers from the contents of the database. These two components will be discussed in the following two sections.

STRINGS

string	*str_id*
...	...
neurofibromin	str728
abdomen	str056
mammectomy	str176
mastectomy	str183
...	...

TERMOID STRINGS

trm_id	*str_id*
...	...
trm023	str056
trm656	str056
trm924	str728
trm369	str728
trm278	str176
trm627	str183
...	...

PART OF SPEECH

trm_id	*pos*
...	...
trm023	N
...	...

SYNONYMY

syn_id	*trm_id*	*scl_id*
...	...	...
syn866	trm278	syn006
syn435	trm627	syn006
...	...	...

GO ANNOTATIONS

trm_id	*annotation*	*version*
...	...	...
trm924	GO:0004857	9/2003
trm369	GO:0008285	9/2003
...	...	...

UMLS

trm_id	*cui*	*lui*	*sui*	*version*
...	...	...	...	...
trm278	C0024881	L0024669	S0059711	2003AC
trm656	C0000726	L0000726	S0414154	2003AC
...	...	...	...	...

Figure 2: Structure of the terminological database

3.1 Terminological Database

The terminological database is designed to meet three requirements. First of all, it must be capable of storing large numbers of terms. As we have seen, the UMLS Metathesaurus contains over 2 million distinct terms. However, as UMLS is just one of many resources whose terms may need to be stored, many millions of terms may need to be stored in total. Secondly, Termino's database must also be flexible enough to hold a variety of information about terms, including information of a morpho-syntactic nature, such as part of speech and morphological class; information of a semantic nature, such as quasi-logical form and links to concepts in ontologies; and provenance information, such as the sources of the information in the database. The database will also contain links to connect synonyms and morphological and orthographic variants to one another and to connect abbreviations and acronyms to their full forms. Finally, the database must be organized in such a way that it allows for fast and efficient recognition of terms in text.

As mentioned above, the information in Termino's database is either imported from existing, outside knowledge sources or induced from text corpora. Since these sources are heterogeneous in both information content and format, Termino's database is "extensional": it stores strings and information about strings. Higher-order concepts such as "term" emerge as the result of interconnections between strings and information in the database. The database is organized as a set of relational tables, each storing one of the types of information mentioned above. In this way, new information can easily be included in the database without any global changes to the structure of the database.

Terminological information about any given string is usually gathered from multiple sources. As information about a string accumulates in the database, we must make sure that co-dependencies between various pieces of information about the string are preserved. This consideration leads to the fundamental element of the terminological database, a *termoid*. A termoid consists of a string together with associated information of various kinds about the string. Information in one termoid holds conjunctively for the termoid's string, while multiple termoids for the same string express disjunctive alternatives.

For instance, taking an example from UMLS, we may learn from one source that the string *cold* as an adjective refers to a temperature, whereas another source may tell us that *cold* as a noun refers to a disease. This information is stored in the database as two termoids: abstractly, '*cold*, adjective, temperature' and '*cold*, noun, disease'. A single termoid '*cold*, adjective, noun, temperature, disease' would not capture the co-dependency between the part of speech and the "meaning" of *cold*.[1] This example illustrates that a string can be in more than one termoid.

[1]Note that the UMLS Metathesaurus has no mechanism for storing this co-dependency between grammatical and semantic information.

Each termoid, however, has one and only one string.

Figure 2 provides a detailed example of part of the structure of the terminological database. In the table STRINGS every unique string is assigned a string identifier (*str_id*). In the table TERMOID STRINGS each string identifier is associated with one or more termoid identifiers (*trm_id*). These termoid identifiers then serve as keys into the tables holding terminological information. Thus, in this particular example, the database includes the information that in the Gene Ontology the string *neurofibromin* has been assigned the terms with identifiers GO:0004857 and GO:0008285. Furthermore, in the UMLS Metathesaurus version 2003AC, the string *mammectomy* has been assigned the concept-unique identifier C0024881 (CUI), the lemma-unique identifier L0024669 (LUI), and the string-unique identifier S0059711 (SUI).

Connections between termoids such as those arising from synonymy and orthographic variation are recorded in another set of tables. For example, the table SYNONYMY in figure 2 indicates that termoids 278 and 627 are synonymous, since they have the same synonymy class identifier (*scl_id*).[2] The synonymy identifier (*syn_id*) identifies the assignment of a termoid to a particular synonymy class. This identifier is used to record the source on which the assignment is based. This can be a reference to a knowledge source from which synonymy information has been imported into Termino, or a reference to both an algorithm by which and a corpus from which synonyms have been extracted. Similarly there are tables containing provenance information for strings, indexed by *str_id*, and termoids, indexed by *trm_id*. These tables are not shown in he example.

With regard to the first requirement for the design of the terminological database mentioned at the beginning of this section – scalability –, an implementation of Termino in MySQL has been loaded with 427,000 termoids for 363,000 strings (see section 4 for more details). In it the largest table, STRINGS, measures just 16MB, which is nowhere near the default limit of 4GB that MySQL imposes on the size of tables. Hence, storing a large number of terms in Termino is not a problem size-wise. The second requirement, flexibility of the database, is met by distributing terminological information over a set of relatively small tables and linking the contents of these tables to strings via termoid identifiers. In this way we avoid the strictures of any one fixed representational scheme, thus making it possible for the database to hold information from disparate sources. The third requirement on the design of the database, efficient recognition of terms, will be addressed in the next section.

[2]The function of synonymy class identifiers in Termino is similar to the function of CUIs in the UMLS Metathesaurus. However, since we are not bound to a classification into UMLS CUIs, we can assert synonymy between terms coming from arbitrary sources.

3.2 Term Recognition

To ensure fast term recognition with Termino's vast terminological database, the system comes equipped with a compiler for generating finite state machines from the strings in the terminological database discussed in the previous section. Direct look-up of strings in the database is not an option, because it is unknown in advance at which positions in a text terms will start and end. In order to be complete, one would have to look up *all* sequences of words or tokens in the text, which is very inefficient.

Compilation of a finite state recognizer proceeds in the following way. First, each string in the database is broken into tokens, where a token is either a contiguous sequence of alpha-numeric characters or a punctuation symbol. Next, starting from a single initial state, a path through the machine is constructed, using the tokens of the string to label transitions. For example, for the string *Graves' disease* the machine will include a path with transitions on *Graves*, *'*, and *disease*. New states are only created when necessary. The state reached on the final token of a string will be labeled final and is associated with the identifiers of the termoids for that string.

To recognize terms in text, the text is tokenized and the finite state machine is run over the text, starting from the initial state at each token in the text. For each sequence of tokens leading to a final state, the termoid identifiers associated with that state are returned. These identifiers are then used to access the terminological database and retrieve the information contained in the termoids. Where appropriate the machine will produce multiple termoid identifiers for strings. It will also recognize overlapping and embedded strings.

Figure 3 shows a small terminological database and a finite state recognizer derived from it. Running this recognizer over the phrase *... thyroid dysfunction, such as Graves' disease ...* produces four annotations: *thyroid* is assigned the termoid identifiers trm1 and trm2; *thyroid dysfunction*, trm3; and *Graves' disease*, trm4.

It should be emphasised at this point that term recognition as performed by Termino is in fact term look-up and not the end point of term processing. Term look-up might return multiple possible terms for a given string, or for overlapping strings, and subsequent processes may apply to filter these alternatives down to the single option that seems most likely to be correct in the given context. Furthermore, more flexible processes of term recognition might apply over the results of look-up. For example, a term *grammar* might be provided for a given domain, allowing longer terms to be built from shorter terms that have been identified by term look-up.

The compiler can be parameterized to produce finite state machines that match exact strings only, or that ab-

STRINGS

string	*str_id*
thyroid	str12
thyroid disfunction	str15
Graves' disease	str25

TERMOID STRINGS

trm_id	*str_id*
trm1	str12
trm2	str12
trm3	str15
trm4	str25

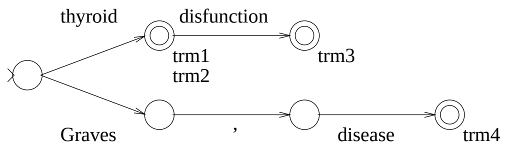

Figure 3: Sample terminological database and finite state term recognizer

stract away from morphological and orthographical variation. At the moment, morphological information about strings is supplied by a component outside Termino. In our current term recognition system, this component applies to a text before the recognition process and associates all verbs and nouns with their base form. Similarly, the morphological component applies to the strings in the terminological database before the compilation process.

The set-up in which term recognizers are compiled from the contents of the terminological database turns Termino into a general terminological resource which is not restricted to any single domain or application. The database can be loaded with terms from multiple domains and compilation can be restricted to particular subsets of strings by selecting termoids from the database based on their source, for example. In this way one can produce term recognizers that are tailored towards specific domains or specific applications within domains.

4 Implementation & Performance

A first version of Termino has been implemented. It uses a database implemented in MySQL and currently contains over 427,000 termoids for around 363,000 strings. Content has been imported from various sources by means of source-specific scripts for extracting relevant information from sources and a general script for loading this extracted information into Termino. More specifically, to support information extraction from patient records, we have included in Termino strings from the UMLS Metathesaurus falling under the following semantic types: pharmacologic substances, anatomical structures, therapeutic procedure, diagnostic procedure, and several others. We have also loaded a list of human proteins and their assignments to the Gene Ontology as produced by the European Bioinformatics Institute (http://www.ebi.ac.uk/GOA/) into Termino. Furthermore, we have included several gazetteer lists containing terms in the fields of molecular biology and pharmacology that were assembled for previous information extraction projects in our NLP group. A web services (SOAP) API to the database is under development. We plan to make the resource available to researchers as a web service or in downloadable form.[3]

The compiler to construct finite state recognizers from the database is fully implemented, tested, and integrated into AMBIT. The compiled recognizer for the 363,000 strings of Termino has 1.2 million states and an on-disk size of around 80MB. Loading the matcher from disk into memory requires about 70 seconds (on an UltraSparc 900MHz), but once loaded recognition is a very fast process. We have been able to annotate a corpus of 114,200 documents, drawn from electronic patient records from the Royal Marsden NHS Trust in London and each approximately 1kB of text, in approximately 44 hours – an average rate of 1.4 seconds per document, or 42 documents per minute. On average, about 30 terms falling under the UMLS 'clinical' semantic types mentioned above were recognized in each document. We are currently annotating a bench-mark corpus in order to obtain precision and recall figures. We are also planning to compile recognizers for differently sized subsets of the terminological database and measure their recognition speed over a given collection of texts. This will provide some indication as to the scalability of the system.

Since Termino currently contains many terms imported from the UMLS Metathesaurus, it is interesting to compare its term recognition performance against the performance of MetaMap. MetaMap is a program available from at the National Library of Medicine – the developers of UMLS – specifically designed to discover UMLS Metathesaurus concepts referred to in text (Aronson, 2001). An impressionistic comparison of the performance of Termino and MetaMap on the CLEF patient records shows that the results differ in two ways. First, MetaMap recognizes more terms than Termino. This is simply because MetaMap draws on a comprehensive version of UMLS, whereas Termino just contains a selected subset of the strings in the Metathesaurus. Secondly, MetaMap is able to recognize variants of terms, e.g. it will map the verb *to treat* and its inflectional forms onto the term *treatment*, whereas Termino currently does not do this. To recognize term variants MetaMap relies on UMLS's SPECIALIST lexicon, which provides

[3]Users may have to sign license agreements with third parties in order to be able to use restricted resources that have been integrated into Termino.

syntactic, morphological, and orthographic information for many of the terms occurring in the Metathesaurus. While the performance of both systems differs in favor of MetaMap, it is important to note that the source of these differences is unrelated to the actual design of Termino's terminological database or Termino's use of finite state machines to do term recognition. Rather, the divergence in performance follows from a difference in breadth of content of both systems at the moment. With regard to practical matters, the comparison showed that term recognition with Termino is much faster than with MetaMap. Also, compiling a finite state recognizer from the terminological database in Termino is a matter of minutes, whereas setting up MetaMap can take several hours. However, since MetaMap's processing is more involved than Termino's, e.g. MetaMap parses the input first, and hence requires more resources, these remarks should be backed up with a more rigorous comparison between Termino and MetaMap, which is currently underway.

The advantage of term recognition with Termino over MetaMap and UMLS or any other recognizer with a single source, is that it provides immediate entry points into a variety of outside ontologies and other knowledge sources, making the information in these sources available to processing steps subsequent to term recognition. For example, for a gene or protein name recognized in a text, Termino will return the database identifiers of this term in the HUGO Nomenclature database (Wain et al., 2002) and the OMIM database (Online Mendelian Inheritance in Man, OMIM (TM), 2000). These identifiers give access to the information stored in these databases about the gene or protein, including alternative names, gene map locus, related disorders, and references to relevant papers.

5 Conclusions & Future Work

Dealing with terminology is an essential step in natural language processing in technical domains. In this paper we have described the design, implementation, and use of Termino, a large scale terminology resource for biomedical language processing.

Termino includes a relational database which is designed to store a large number of terms together with complex, heterogeneous information about these terms, such as morpho-syntactic information, links to concepts in ontologies, and other kinds of annotations. The database is also designed to be extensible: it is easy to include terms and information about terms found in outside biological databases and ontologies. Term look-up in text is done via finite state machines that are compiled from the contents of the database. This approach allows the database to be very rich without sacrificing speed at look-up time. These three features make Termino a flexible tool for inclusion in a biomedical text processing system.

As noted in section 3.2, Termino has not been designed to be used as a stand-alone term recognition system but rather as the first component, the lexical look-up component, in a multi-component term processing system. Since Termino may return multiple terms for a given string, or for overlapping strings, some post-filtering of these alternatives is necessary. Secondly, it is likely that better term recognition performance will be obtained by supplementing Termino look-up with a term parser which uses a grammar to give a term recognizer the generative capacity to recognize previously unseen terms. For example, many terms for chemical compounds conform to grammars that allow complex terms to be built out of simpler terms prefixed or suffixed with numerals separated from the simpler term with hyphens. It does not make sense to attempt to store in Termino all of these variants.

Termino provides a firm basis on which to build large-scale biomedical text processing applications. However, there are a number of directions where further work can be done. First, as noted in 3.2, morphological information is currently not held in Termino, but rather resides in an external morphological analyzer. We are working to extend the Termino data model to enable information about morphological variation to be stored in Termino, so that Termino serves as a single source of information for the terms it contains. Secondly, we are working to build term induction modules to allow Termino content to be automatically acquired from corpora, in addition to deriving it from manually created resources such as UMLS. Finally, while we have already incorporated Termino into the AMBIT system where it collaborates with a term parser to perform more complete term recognition, more work can be done to with respect to such an integration. For example, probabilities could be incorporated into Termino to assist with probabilistic parsing of terms; or, issues of trade-off between what should be in the term lexicon versus the term grammar could be further explored by looking to see which compound terms in the lexicon contain other terms as substrings and attempt to abstract away from these to grammar rules. For example, in the example *thyroid disfunction* above, both *thyroid* and *disfunction* are terms, the first of class 'body part', the second of class 'problem'. Their combination *thyroid disfunction* is a term of class 'problem', suggesting a rule of the form 'problem' $\rightarrow$ 'body part' 'problem'.

References

A.R. Aronson. 2001. Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program. In *Proceedings of the American Medical Informatics Association Symposium*, pages 17–21.

R. Gaizauskas and Y. Wilks. 1998. Information extraction: Beyond document retrieval. *Journal of Documentation*, 54(1):70–105.

R. Gaizauskas, G. Demetriou, P. Artymiuk, and P. Willett. 2003. Protein structures and information extraction from biological texts: The PASTA system. *Journal of Bioinformatics*, 19(1):135–143.

C.A. Goble, C.J. Wroe, R. Stevens, and the myGrid consortium. 2003. The myGrid project: Services, architecture and demonstrator. In S. Cox, editor, *Proceedings of UK e-Science All Hands Meeting 2003, Nottingham, UK*. http://www.nesc.ac.uk/events/ahm2003/AHMCD/.

R. Grishman. 1997. Information extraction: Techniques and challenges. In Maria Teresa Pazienza, editor, *Information Extraction*, pages 10–27. Springer Verlag.

U. Hahn, M. Romacker, and S. Schulz. 2002. Creating knowledge repositories from biomedical reports: the medSynDiKATe text mining system. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 338–349.

L. Humphreys, D.A.B. Lindberg, H.M. Schoolman, and G.O. Barnett. 1998. The Unified Medical Language System: An informatics research collaboration. *Journal of the American Medical Informatics Association*, 1(5):1–13.

K. Humphreys, G. Demetriou, and R. Gaizauskas. 2000. Two applications of information extraction to biological science journal articles: Enzyme interactions and protein structures. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 505–516.

A.G. Murzin, S.E. Brenner, T. Hubbard, and C. Chothia. 1995. SCOP: A structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, (247):536–540. (http://scop.mrc-lmb.cam.ac.uk/scop/).

Online Mendelian Inheritance in Man, OMIM (TM). 2000. McKusick-Nathans Institute for Genetic Medicine, Johns Hopkins University (Baltimore, MD) and National Center for Biotechnology Information, National Library of Medicine (Bethesda, MD). http://www.ncbi.nlm.nih.gov/omim/.

J. Pustejovsky, J. Castaño, R. Saurí, A. Rumshisky, J. Zhang, and W. Luo. 2002. Medstract: Creating large-scale information servers for biomedical libraries. In *Proceedings of the Workshop on Natural Language Processing in the Biomedical Domain, Association for Computational Linguistics 40th Anniversary Meeting (ACL-02)*, pages 85–92.

A. Rector, J. Rogers, A. Taweel, D. Ingram, D. Kalra, J. Milan, R. Gaizauskas, M. Hepple, D. Scott, and R. Power. 2003. Joining up health care with clinical and post-genomic research. In S. Cox, editor, *Proceedings of UK e-Science All Hands Meeting 2003, Nottingham, UK*. http://www.nesc.ac.uk/events/ahm2003/AHMCD/.

C.T. Rindflesch, J.V. Rajan, and L. Hunter. 2000. Extracting molecular binding relationships from biomedical text. In *Proceedings of the 6th Applied Natural Language Processing conference / North American chapter of the Association for Computational Linguistics annual meeting*, pages 188–915.

The Gene Ontology Consortium. 2001. Creating the gene ontology resource: design and implementation. *Genome Research*, 11(8):1425–1433.

J. Thomas, D. Milward, C. Ouzounis, and S. Pulman. 2000. Automatic extraction of protein interactions from scientific abstracts. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 538–549.

H.M. Wain, M. Lush, F. Ducluzeau, and S. Povey. 2002. Genew: The human nomenclature database. *Nucleic Acids Research*, 30(1):169–171. (http://www.gene.ucl.ac.uk/nomenclature/).