Formulating queries for assessing clinical trial eligibility [⋆]

D.W. Lonsdale [a,∗], C. Tustison [a], C.G. Parker [a], D.W. Embley [a]

[a]*Brigham Young University, Provo, UT, USA 84602*

Abstract

This paper introduces a system that processes clinical trials using a combination of natural language processing and database techniques. We process web-based clinical trial recruitment pages to extract semantic information reflecting eligibility criteria for potential participants. From this information we then formulate a query that can match criteria against medical data in patient records. The resulting system reflects a tight coupling of web-based information extraction, natural language processing, medical informatic approaches to clinical knowledge representation, and large-scale database technologies. We present an evaluation of the system and future directions for further system development.

Key words: clinical trials, predicate logic, clinical data, virtual medical record

1 Background and overview

As electronic texts become more available to researchers (and humans in general), an interesting dichotomy has emerged. On one hand, Web texts cater to users' abilities to read and analyze that information; Web publishers design the data's structure to be easy for humans to digest. Hence it must adhere to conventional syntactic and semantic constraints of the users' natural language. On the other hand, humans have very limited computational capacity for analyzing the vast amounts of electronic information now available. Information extraction research focuses on helping humans access and process large quantities of Web data. Often this work involves

[⋆] This work was partially funded under National Science Foundataion Information and Intelligent Systems grant IIS-0083127. See also www.deg.byu.edu.

[∗] Corresponding author.
Email addresses: `lonz@byu.edu` (D.W. Lonsdale), `tustison@gmail.com` (C. Tustison), `craigparker@cgpmd.com` (C.G. Parker), `embley@cs.byu.edu` (D.W. Embley).

devising new strategies and algorithms to convert electronic natural language text into various formats that feed subsequent automatic processing.

The task is complicated by several types of textual layout formats. Text is often classified into one of three categories: unstructured (or free), structured, and semi-structured [1]. Unstructured text is the most natural for humans to process, but treating the information automatically is nontrival. Structured text is stored in a very rigid format (e.g. a database or a table) and hence more readily processed automatically, but is often less natural for humans to work with. Semistructured text falls somewhere in between; some structure is imposed—often just enough to render it not quite grammatical—though not enough to help in automatically processing the contents. With the advent of various markup languages and other annotation conventions, Web text often includes other extratextual information that may or may not aid in extracting information. In this paper we discuss processing a repository of semistructured medical text.

Researchers design information extraction systems to perform various tasks, and these tasks require various levels of linguistic processing. Some systems are only concerned with parsing out the extracted information and therefore only require the use of a syntactic parser. Others need more in-depth processing and include a semantic component that can give some meaning to the extracted information. Yet other systems are dependent on real-world knowledge and require a pragmatic component to relate the data gathered from the system to outside information.

One area receiving recent attention is the medical domain. Much of the natural language processing (NLP) research done with medical literature has involved developing systems that extract different types of relationships from text. For example, NLP techniques have been used to extract interesting and novel relationships from Medline[1] abstracts. The Medline repository contains vast amounts of useful information about various disease- and health-related issues. Many researchers have succeeded in extracting various types of relationships found in this repository, including gene relations [2], protein relationships [3,4], acronym-meaning pairs [5], abbreviation definitions [6], and molecular binding relationships [7].

For its part, the field of medical informatics has produced large-scale resources, largely in database format, that specify the vast knowledge required for medical research and patient services. Highly specialized tools for representing clinical information and patient data have also been developed. Unfortunately, there has been only a modest amount of crossover between the NLP and medical informatics fields. The topic of information extraction is a salient one for demonstrating how applications can leverage the developments from both fields.

This paper describes our approach to identification, extraction, and query formulation of information regarding medical clinical trials. Figure 1 shows an overview

[1] See http://www.medlineplus.gov.

of the system. In Step 1, extraction and formula generation, we extract patient criteria from a web-based natural language description of qualifications for clinical trial participants, and create predicate logic expressions (PLE's) that reflect the semantic content of the text. In Step 2, code generation, the system processes parsed criteria and their PLE's. The system then attempts to map the criteria to concepts in an electronic medical record. For the criteria that map successfully, the system outputs appropriate logic for computing patient eligibility. In Step 3, eligibility assessment, the system evaluates the eligibility of a potential participant by executing the logic generated in Step 2 against that patient's electronic medical record. The system produces a generated report that can help a clinician make an informed decision about whether to further evaluate the patient for enrollment in the clinical trial.

In Section 2 we describe Step 1 of the system, which involves the NLP component. Section 3 describes the subsequent medical records database query component. We then discuss the system evaluation in Section 4. Finally, we sketch ways the system could be enhanced in the future to provide better results.

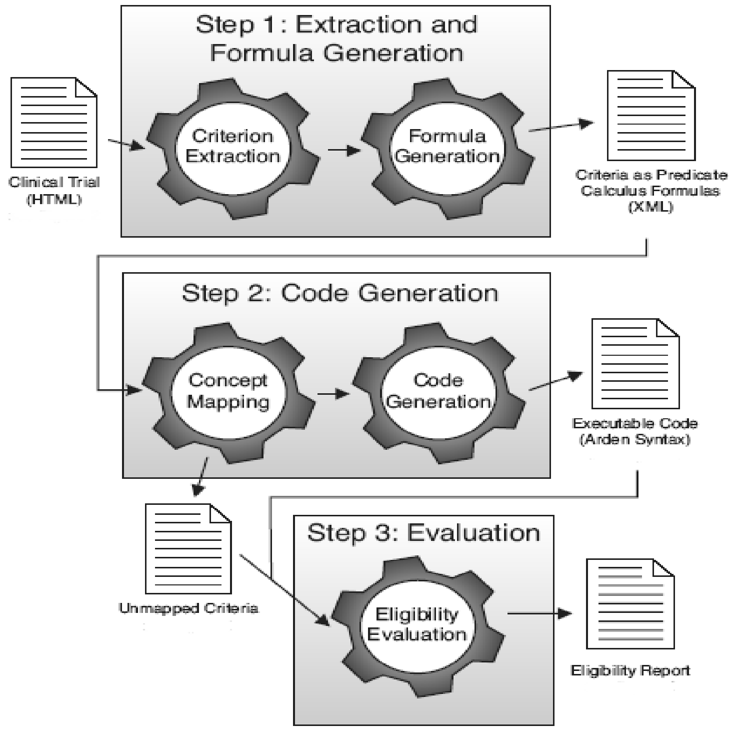


Fig. 1. Stages of processing in the system with data formats (input, intermediate, and output).

2 Extraction and formula generation

The domain that our system addresses is clinical trials, which medical professionals use as a tool to assess diagnostic and therapeutic agents and procedures. Such trials require voluntary human subjects to undergo the new treatments or receive experimental medications. With the increasing cost of bringing experimental new drugs to the public, there is a crucial need for improving and automating access to the information in clinical trials including the directed recruitment of experimental participants, which is otherwise costly and labor-intensive.

Greater enrollment of subjects leads to greater confidence in the experimental results, but identifying ideal participants is costly and time-consuming. Trials often have very specific criteria for age, gender, state of a given disease, number and types of co-existing diseases, and trial timeframe/location.

Eligible potential participants are identified in various ways. One way is for clinicians to recommend their own patients. This results in few recommendations, though, since the clinician is limited to current patients and must be aware of which trials are soliciting patients.

Another common method for identifying candidates is through advertisements distributed via television, radio, the internet, newspapers or magazines. This method reaches a large number of people, including those who are not seeing a clinician. Disadvantages include the high cost of advertising, the inability of the general public to understand and self-diagnose complex technical criteria, and the cost and time involved in having a clinician screen applicants as potential participants.

A third method for identifying candidates is a systematic review of medical records. Screeners with some clinical training can perform this work, though it is laborious and costly. Furthermore, the patient information may be out-of-date or incomplete, so in-person evaluations are usually necessary. Since patient medical data is increasingly available in electronic form, a variation on this third approach is becoming increasingly feasible. Automated processes can sift through the available data, identifying possible trial participants.

In this section we first discuss the web corpus we have targeted. Then we sketch the first stage of the system—how the pertinent text is processed by the NLP components of the system.

2.1 The corpus: clinical trials

From 1997 to 1999 the U.S. National Library of Medicine (NLM) and the National Institutes of Health (NIH) developed an online repository of clinical trials [8]. This repository currently contains about 25,000 trials which are sponsored by various governmental and private organizations[2]; the repository receives about 8,000,000 page views per month[3].

Providers develop web pages for the clinical trials website using a simple user interface[4] including a text box for the eligibility criteria. No format restrictions are currently enforced on the text, though some boilerplate material can be entered (e.g. patient ages and gender) via dropdown boxes.

Each trial in the online repository comprises a series of sections that contain specific information regarding the trial that is useful to providers and patients. Figure 2 shows a sample web page for an individual clinical trial and the hierarchy of different components it contains.

For this paper we extract information from one section of the web page: the Eligibility section. This section contains a listing of the requirements that a person must satisfy in order to participate in the trial. For example, nearly every eligibility section specifies the patient age and also the gender.

Each web page undergoes two levels of preprocessing: (i) locating, retrieving, and converting the Eligibility section to an XML format with each item embedded in `<criterion>` tags; and (ii) manipulating the natural language text of some criteria to enable further processing. Often eligibility criteria are expressed telegraphically, for example with elided subjects or as standalone noun phrases. Parsing works best on full sentences, but only a small percentage have eligibility criteria structured as complete sentences. For elided subjects, a dummy subject and verb (i.e. *A criterion equals...*) are prepended to the criterion.

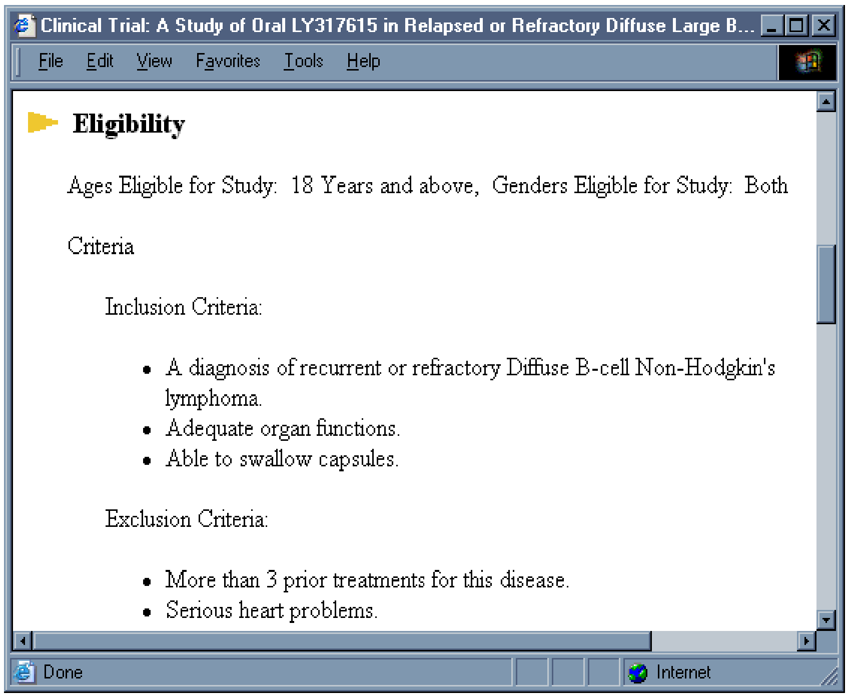
In other instances the first word in the criterion needs to be nominalized in order to produce a grammatical sentence. For example, the criterion *able to swallow capsules* is reformulated as *an ability to swallow capsules*, and then the dummy subject and verb are prepended.

Figure 2 shows an example clinical trials web page, its corresponding XML version, and the linguistically-annotated rendition of its eligibility criteria.

2.2 Deriving syntactic and semantic information

The next step in the process involves using a syntactic parser to process the natural language criteria and produce a corresponding syntactic representation. We use the

[2] See http://www.clinicaltrials.gov.
[3] See http://www.clinicaltrials.gov/ct/info/about.
[4] See http://prsinfo.clinicaltrials.gov/elig.html.


Clinical Trial: A Study of Oral LY317615 in Relapsed or Refractory Diffuse Large B...

Eligibility

Ages Eligible for Study: 18 Years and above, Genders Eligible for Study: Both

Criteria

Inclusion Criteria:

- A diagnosis of recurrent or refractory Diffuse B-cell Non-Hodgkin's lymphoma.
- Adequate organ functions.
- Able to swallow capsules.

Exclusion Criteria:

- More than 3 prior treatments for this disease.
- Serious heart problems.

(a) ***Clinical trial web page NCT00042666.***

```
<criteria trial="http://www.clinicaltrials.gov/ct/show/NCT00042666">
 <criterion>
  <text>Eligibility</text>
  <text val="1">Ages Eligible for Study: 18 Years and above,</text>
 </criterion>
 <criterion>
  <text>Eligibility</text>
  <text val="2">Genders Eligible for Study: Both</text>
 </criterion>
  ... (ADDITIONAL CRITERIA) ...
 <criterion>
  <text>Eligibility</text>
  <text>Criteria</text>
  <text>Exclusion Criteria:</text>
  <text val="6">More than 3 prior treatments for this disease.</text>
 </criterion>
 <criterion>
  <text>Eligibility</text>
  <text>Criteria</text>
  <text>Exclusion Criteria:</text>
  <text val="7">Serious heart problems.</text>
 </criterion>
</criteria>
```

(b) ***Criteria annotated with XML tags.***

1. *A criterion equals* `an age greater than 18 years.`
2. *A criterion equals* `both genders.`
3. *A criterion equals* `a diagnosis of recurrent or refractory Diffuse B-Cell Non-Hodgkin’s lymphoma.`
4. *A criterion equals* `adequate organ functions.`
5. *A criterion equals* `an ability to swallow capsules.`
6. *A criterion equals* `more than 3 prior treatments for this disease.`
7. *A criterion equals* `serious heart problems.`

(c) ***Criteria with linguistic elements added.***

Fig. 2. Portion of clinical trial NCT00042666 and preprocessed versions of eligibility criteria.

Table 1
Differences between dependency and link grammars

Dependency Grammar	Link Grammar
Notion of a root word	No notion of a root word
Links are not labeled	Links are labeled
Dependencies exist between heads & dependents	Links are undirected
Cycles are not allowed in the structure	Links may form cycles
Grammar rules are dependency rules	Grammar rules are lexical rules

link grammar (LG) parser [9]. We chose this tool because of its open-source availability, efficiency, robustness in the face of ungrammaticality and out-of-vocabulary words, and flexibility [5] .

Most traditional parsers are based on theoretical approaches to syntactic constituency and consequently produce phrase-structure trees that encapsulate these assumptions. For information extraction purposes, such parsers are often too time-consuming to execute, are too complex to manage, and produce output that is too detailed for efficient downstream use. Recently dependency parsers and statistical-based approaches have become more widely used to parse text for run-time applications. The LG parser is similar to a dependency-based parser, though subtle differences exist. The differences between dependency and link grammar parses are summarized in Table 1; a more complete discussion is found in [10].

The system reads in a .txt file containing each criterion (as extracted from the XML file described above) on a separate line in the file and parses each sentence individually. Because of structural ambiguities in English, a single input sentence might produce multiple parses; in this project, we only consider the highest-scored parse for subsequent processing. Figure 3 shows how a parse of *A criterion equals serious heart problems.* is represented syntactically by the LG parser. Different labeled links connect the words in the sentence in a way that expresses their dependencies. These links are the key to the next step, extracting the semantic meaning from the syntactic output. Three properties characterize a successful parse: planarity (i.e. links cannot cross), connectivity (i.e. links must indirectly connect all words together), and satisfaction (i.e. the link-word correspondences must follow the grammar's specifications). However, the LG parser will often output a partial parse even when a complete parse is impossible; this property is leveraged whenever necessary in subsequent processing.

Once syntactic parsing of a sentence has been completed, the sentence is analyzed by the syntax-to-semantics conversion engine. This is a component (that was previously developed for other applications) specifically designed to take the output

[5] See http://www.link.cs.cmu.edu/link.

Table 2
Sample predicate logic expressions output by LG-Soar

Original Source	**LG-Soar Output**
Adenocarcinoma of the pancreas	adenocarcinoma(x) & of(x,y) & pancreas(y)
Brain metastasis	brain_metastasis(x)
Femoral neck osteoporosis	femoral(x) & neck_osteoporosis(x)
Pregnancy	pregnancy(x)
High risk of VTE	VTE(x) & risk(y) & of(y,x) & high(y)
Controlled COPD	controlled(x) & COPD(x)
Genders eligible for Study: Female	female_gender(x)

from the LG parser and convert its content to PLE's (though other semantic formats are also supported by the system).

The engine is built on Soar[6], a rule-based symbolic intelligent agent architecture that uses a goal-directed, operator-based approach to problem solving [11,12]. The system involves hierarchical on-line machine learning to "chunk up" prior actions, creating new operators which can be invoked recognitionally when situations arise that are similar to those encountered in prior experience [13].

For the task at hand, sentences are fed one-by-one from the LG parser output to the Soar engine. For the highest-scoring parse, all words and links with their associated labels are created on the agent's input buffer. The system creates a discourse model and populates it with concepts representing each entity, property, and action (or state) derivable from the lexical and link content of the input. The relationships between these concepts are annotated, and the result is a data structure encoding the salient semantic features of the input sentence.

The engine then executes operators (specified via several dozen pertinent hand-crafted rules) to map components of the discourse semantic model to equivalent logical predicates and their associated arguments. Variables are generated for predicates to specify with appropriate arity which referents the predicates refer to. The system attempts to build nominal compounds, which are frequent in the domain, based on the link specifications. Table 2 shows some sample input fragments with corresponding PLE's.

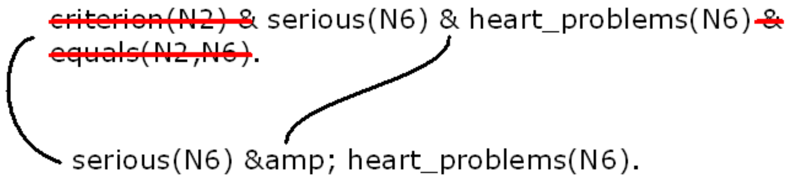
Further pursuing our criteria from Figure 2, the parsed sentence *A criterion equals serious heart problems.* would yield the PLE: "criterion(N2) & serious(N6) &

[6] Freely available at http://sitemaker.umich.edu/soar.

```
    +----------------------------Xp--------------------------+
    |                       +-------------Op------------+     |
    +-----Wd-----+          |        +---------A--------+     |
    |     +--Ds--+----Ss----+        |        +----AN---+     |
    |     |      |          |        |        |         |     |
LEFT-WALL a criterion.n equals.v serious.a heart.n problems.n .


       LEFT-WALL      Xp      <---Xp---->  Xp        .
 (m)   LEFT-WALL      Wd      <---Wd---->  Wd        criterion.n
 (m)   a              Ds      <---Ds---->  Ds        criterion.n
 (m)   criterion.n    Ss      <---Ss---->  Ss        equals.v
 (m)   equals.v       O       <---Op---->  Op        problems.n
 (m)   serious.a      A       <---A----->  A         problems.n
 (m)   heart.n        AN      <---AN---->  AN        problems.n
       .              RW      <---RW---->  RW        RIGHT-WALL
```

(a) ***Link grammar output for a criterion's sentential form.***



~~criterion(N2) &~~ serious(N6) & heart_problems(N6) ~~&~~
~~equals(N2,N6)~~.

serious(N6) &amp; heart_problems(N6).

(b) ***Predicate logic expressions before and after postprocessing.***

```
<criteria trial="http://www.clinicaltrials.gov/ct/show/NCT00042666">
 <criterion>
  <text>Eligibility</text>
  <text val="1">Ages Eligible for Study: 18 Years and above,</text>
  <pred val="1">age(N4) &amp; quantification(N5,greater_than)
                        &amp; measurement(N4,N5) &amp; units(N5,years)
                        &amp; magnitude(N5,18)</pred>
 </criterion>
 <criterion>
  <text>Eligibility</text>
  <text val="2">Genders Eligible for Study: Both</text>
  <pred val="2">both_genders(N4)</pred>
 </criterion>
... (ADDITIONAL CRITERIA) ...
 <criterion>
  <text>Eligibility</text>
  <text>Criteria</text>
  <text>Exclusion Criteria:</text>
  <text val="7">Serious heart problems.</text>
  <pred val="7">serious(N6) &amp; heart_problems(N6)</pred>
 </criterion>
</criteria>
```

(c) ***XML file with tagged predicate logic expressions added.***

Fig. 3. Final result of natural language processing stages.

heart_problems(N6) & equals(N2,N6)". Note that the dummy subject and verb, which were added for parsing purposes, are present in the PLE. For this reason, a postprocessing stage removes this extraneous information. Then the resulting PLE is placed in the abovementioned XML file.

Figure 3 illustrates the parse, its PLE, and the XML file after the NL processing stages have finished.

3 Query generation

Once the source web page has undergone the NL processing techniques described above, the resulting extracted information feeds a database query stage to match them with patient medical records. In this section we can only briefly mention the technologies germane to the task at hand; more details are available elsewhere [14].

3.1 The target

Medical information systems manage patient information for a wide variety of tasks including patient care, administration (e.g. billing), research, and regulatory reporting. Coded medical vocabularies have been developed in order to ensure consistency, computability, and sharability. Often they are conceptually based and have associated lexicons or vocabularies which are sometimes hierarchical in nature. For example, the SNOMED-CT [15] coded vocabulary has a code "254837009" that represents the concept "breast cancer".

Representing patient data usually requires more information than simple concepts. A data model called a detailed clinical model defines relationships between coded concepts or (other data values) and information of clinical interest. For example, a detailed clinical model might define a diagnosis in terms of a type and a subject/person, so that a statement "The patient has breast cancer." could be encoded with the diagnosis type from SNOMED-CT as described above, and the subject/person with the relevant patient ID number. Detailed clinical models thus combine coded concepts into meaninful expressions of a higher-order nature. We make extensive use of both coded concepts and detailed clinical models in the concept mapping process shown in Step 2 in Figure 1.

The target electronic medical record for this project is Intermountain Health Care's Clinical Data Repository (CDR)[7]. The CDR makes extensive use of coded vocabularies; it also defines detailed clinical models using Abstract Syntax Notation One (ASN.1) [16], an ISO standard for describing electronic messages [17], including binary and XML encodings for many different application areas ranging from telecommunications to genome databases.

All coded concepts in the CDR are drawn from IHC's Healthcare Data Dictionary (HDD) [18], a large coded vocabulary (over 800,000 concepts with over 4 million synonyms). The names of all the detailed clinical models used in the CDR and the

[7] See http://www.3m.com/us/healthcare/his/products/records/data_repository.jhtml. Intermountain Health Care (IHC) is a regional, nonprofit, integrated health system based in Salt Lake City, UT. The CDR is the result of a joint development effort between IHC and 3M Health Information Systems.

fields they contain are defined as concepts in the HDD.

The CDR comprises a database and its associated services. Besides providing a common access mechanism (for security, auditing, and error handling), the services crucially provide for handling of detailed clinical models as the basis for information access and retrieval. For example, an application can pass an instance of a detailed clinical model to the services, which will then return relevant instances of other detailed clinical models.

One of the outputs of Step 2 in Figure 1 is executable logic in Arden Syntax format [19], an ANSI standard for handling medical data. Arden Syntax is written in units called medical logic modules (MLMs). Each MLM contains the logic necessary for making one medical decision. One category of information in an MLM defines knowledge required for making clinical decisions; this category is what we use in this project. The most significant slots in this category are the data slot and the logic slot. The data slot contains mappings of symbols used in an MLM to data in the target electronic medical record. The logic slot, as its name implies, contains the logic that operates on the data.

Finally, since electronic medical records vary widely in content and structure across applications, it has been useful to use an abstraction called the virtual medical record (VMR) [20]. This assures that any number of healthcare organizations can write, maintain, and share clinical decision logic no matter what the structure of their own repositories. For eligibility criteria we use a small subset of VMR attributes called observations.

3.2 Concept mapping

The process outlined in Step 2 of Figure 1 takes the XML file described above as input. It attempts to map each criterion to concepts and data structures in the target electronic medical record. For each successful mapped criterion we generate executable code for determining if any patients meet the criterion.

Since IHC's CDR stores clinical data as instances of clinical models with coded concepts, and since all coded concepts are in the HDD, the mapping task involves matching words and phrases from the eligibility criteria to concepts in the HDD that represent either names or values in detailed clinical models.

The concept mapping portion of the system thus iterates through each criterion, attempting to map it to coded concepts from the HDD used in the CDR's detailed clinical models. The system uses multiple matching strategies executed sequentially, and once a match is found, subsequent matches are not sought. Seven decision points formulate the matching strategy; we sketch each below.

(1) Execute special case handling. We use string comparisons and regular expression matching for processing predictable boilerplate material (e.g. age and gender). (2) Match the raw text of a criterion to concepts in the database, in case subsequent processing does not succeed. Note that these two steps do not require PLE's, and thus are executed for every criterion. The remaining steps, however, are executed only for criteria that are successfully parsed into predicate calculus formulas.

(3) Match predicate names to the HDD. For example, the criterion "heart disease" yields the formula: heart(x) & disease(x). In this stage the mapper retrieves the best coded concept from the HDD that includes both predicate names. (4) Match the predicate with a measurement. Measurements are extracted as predicates; they include magnitudes, units, and other information. Here the criterion "LDL-C 130-190 mg/dL" is successfully matched to a query that searches LDL-C measurements (a valid HDD concept) in medical records and returns those within the acceptable range.

If the full matches above are not possible, partial matching is then tried. (5) Match name-value pairs. The predicate names are processed to find possible name-value pair relationships. For example, the criterion "diagnosis of appendicitis" does not map to a single concept in the HDD, but it does map to concepts in the CDR. Furthermore, the HDD recognizes "diagnosis" as a valid name for a clinical observation, and "appendicitis" as a valid value. We thus combine them to form a name-value pair. (6) Match a conjunction/disjunction. Often criteria are conjoined, and in such cases we process all elements. For example, the elements of the criterion "Hyperthyroidism or hypothyroidism" are mapped separately and then related with the relevant operation (conjunction or disjunction). (7) Partial match. The best possible match with all available predicate names is attempted, preferring nouns over other parts of speech. Thus, for example, a criterion "active neoplasms" would not match on the predicate "active" but would on the other one, "neoplasm". This heuristic is generally useful, though not always correct. For example, in the concept "renal disease," the adjective "renal" is more useful than the noun "disease".

3.3 Code generation

The second stage of Step 2 is code generation, where we generate executable code from the output of the concept mapping process. The code that we generate for this project is an Arden Syntax MLM (Medical Logic Module) that specifies VMR queries for data access[8]. The process has two steps.

The first step takes place in tandem with the mapping process described above. Each database mapping for a criterion spawns a related VMR query. Abstracting

[8] Generating code in a different language would only require an appropriate reimplementation of the generator interface.

away from the details, this process can be summarized as a rather straightforward conversion from and to nested attribute/value structures.

The second and subsequent step in generating code involves creating the Arden Syntax MLM. For each criterion that does not have a mapping to the target electronic medical record, we generate a comment stating that this criterion could not be mapped, but we do not generate any executable code. For the criteria that do have mappings, we generate an Arden Syntax “read” statement. The VMR queries generate a non-empty return value when the criterion is satisfied. After iterating through the criteria, we generate code that writes out the results.

Even though the vast majority of slots in an MLM are required by the specification, only a handful are useful for machine execution; most of the remaining slots are intended for human perusal. Therefore, for this project we populate only the small number of slots that are useful for automated processing, primarily in the knowledge category slots for type, data, and logic. The only valid value for the type slot is “data-driven”, so we populate it appropriately.

To generate the data slot, we iterate through the eligibility criteria. For each criterion that does not have a mapping to the target electronic medical record, we generate a comment stating that this criterion could not be mapped, but we do not generate any executable code. For the criteria that do have mappings, we generate an Arden Syntax “read” statement. The VMR queries generate a non-empty return value when the criterion is satisfied. After iterating through the criteria, we generate code that writes out the results.

The third code generation step, assessing the applicability of an encoded criterion, involves the straightforward querying of electronic patient records. A report summarizes for the clinician which criteria parsed and matched the stated values. Figure 4 shows an Arden Syntax VMR query and a sample eligibility report.

Our system can be used in a number of different ways. It could drive a process that searches through a large collection of patient records, looking for candidates for a given trial. A threshold could be set for the percentage of criteria necessary for suggesting a given patient for further consideration. Alternatively, a threshold could specify the number of patients to suggest for the trial in question.

Another use of the MLM would be to incorporate it in a process that evaluates patient records after a patient schedules an appointment and before the visit, so that the clinician can suggest possible trial participation during the visit.

```
Criterion1 := READ {
  <VMRQuery class="Observation">
      <value op="equals">
           <cd code="1450395" displayName="heart disease"/>
      </value>
  </VMRQuery>
```

(a) ***A sample Arden Syntax read statement containing a VMR query.***

Eligibility Report	
Header	
Title of Trial	A Study of Oral LY317615 in Relapsed or Refractory Diffuse & Large B-Cell Non-Hodgkin's Lymphoma
Patient Name	J. Doe
Medical Record #	1234567
Eligibility Summary	
Criteria met	6
Mapped Criteria for which eligibility could not be determined	7
Criteria not mapped	5
Total criteria	18
Criterion Detail	
Criterion 1	
...	
Criterion 3	
Criterion	LDL-C 130-190 mg/dL
Mapped	Yes
Status	Patient meets this criterion
...	
Criterion 11	
Criterion	Heart disease
Mapped	Yes
Status	Unable to determine if patient meets this criterion

(b) ***Portion of sample eligibility report.***

Fig. 4. Results for query generation and assessment stages.

4 Evaluation results

In order to evaluate the natural language PLE output of the LG-Soar system, we performed a preliminary standalone evaluation of that stage of processing [21]. We followed the logical form identification evaluation method developed by organizers of the Senseval '03 competition [9] . This was the first standardized logical form identification evaluation task, and the organizers developed a gold standard and an open-source evaluation tool for assessing PLE identification for a selection of texts

[9] See www.senseval.org.

Table 3
Preliminary LG-Soar PLE extraction results

	Precision	Recall	F-Measure
Argument	70%	61%	65%
Predicate	65%	63%	64%

[10] . The results are returned in terms of precision and recall for both predicates and for their arguments.

In order to test this part of the system, we randomly selected twelve different trials from the clinical trials website. From these trials, 102 criteria were extracted and, after exact duplicates were removed, a total of 77 criteria remained. These 77 criteria were run through the system, and 34 were successfully parsed. For these criteria, we thus achieved the results shown in Table 3. Though the results fall short of predicate extraction from newspaper texts (around 90%), it compares favorably with other kinds of information extracted in the medical domain (see [3]).

Encouraged by these results, we recently carried out an end-to-end system performance evaluation involving both the natural language and code generation components. From www.clinicaltrials.gov we randomly chose one hundred unseen clinical trials and ran them through Steps 1 and 2 in Figure 1. Afterwards we manually inspected each report, comparing them to the generated queries, and characterizing their success or failure. We tallied these results numerically, and a summary appears in Figure 5.

The 85 parsable trials varied in size and complexity, having from 3 to 71 criteria per trial. They also varied widely in subject matter, covering conditions from cancer to infertility to gambling. Two main factors contributed to the failure of 15 trials: some had unexpected special characters (e.g. the HTML character “&#252;” representing the umlat u character), and others had sentences of such complexity that the parser failed.

These 85 trials yielded 1,545 eligibility criteria; logical forms were successfully created for 473 of these criteria. All but 49 of these yielded queries, and another 96 queries could be generated without logical forms, so a total of 520 queries were formulated. Of these, 140 completely and exactly represented their original eligibility criteria. Another 113 of the queries were not entirely correct or complete but still yielded useful information for clinician decision-making. Four queries were technically well-formed based on the logical form though did not reflect the intent of the original criteria. In total, 257 queries were either completely correct, usefully correct, or technically correct. The remaining 263 queries were neither correct nor useful in determining eligibility.

[10] See http://www.cs.iusb.edu/vasile/logic/indexLF.html.

Trials evaluated	100
Trials successfully completing Steps 1 & 2	85
Criteria extracted	1545
Criteria parsed into logical forms	473
Criteria parsed but not mapped into queries	49
Queries generated	520
Completely correct queries	140
Other useful queries	113
Technically correct queries	4
Incorrect queries	263

Fig. 5. Results from end-to-end system evaluation.

Processing stage	Heuristic/Assumption	Challenges/Issues
Retrieve criteria	Standard tokenization	Ad-hoc abbreviations
Convert to XML	Scripting (Perl/Python)	—
Make full sentence	Elision predictable	Some ill-formed input
Parse sentence	LG parser + scoring	Some sentences fail to parse
Extract predicates	Soar linguistic agent	Negation, modals, quantifiers
Postprocess predicates	Scripting	—
Map concepts to HDD	String matching	HDD coverage, match cost
Create query structures	Recursive descent	Execution time/complexity
Create query statements	Arden Syntax	Defining relevant subset
Query patient records	Existing software	Patient data completeness
Generate final report	Scripting	End-user usefulness

Fig. 6. Summary of processing stages with relevant assumptions and issues.

Figure 6 summarizes the system's processing stages along with their associated heuristics, assumptions, challenges, and issues.

5 Discussion and future work

Our experimental system demonstrates that some degree of automatic evaluation of eligibility criteria is feasible. The system currently generates useful queries for about half of the number of criteria that produce formulas. We are encouraged by these preliminary results, and anticipate that planned improvements like those discussed below will substantially increase system accuracy and performance.

One issue has been the consistent authoring of parsable natural language statements by data providers. Tighter editorial controls could help solve this problem. A solution less intrusive to the users would be to develop collection of medical knowledge in the form of potentially reusable ontologies and axioms that could be used to assist in bridging the gap.

Currently dummy subjects are added to the usually terse criteria in order to create full grammatical sentences; sometimes, though, there are already complete sentences, and this preprocessing strategy renders them ungrammatical. More intelligent preprocessing could help in determining when dummy subjects are in fact required.

So far we have done little to customize the LG parser for our purposes, and we foresee improving it in at least three ways: (i) extending the range of acceptable grammatical structures; (ii) refining the parse scoring algorithm to return the most plausible parse; and (iii) integrating it with a large-scale medical lexicon as others have done [22]. Currently the semantics engine only handles a limited number of syntactic structures—far less than those provided by the LG parser—and we have not as yet experimented with the semantic engine's inherent machine learning capabilities either.

In several cases the system correctly mapped the name portion of a pair, but incorrectly mapped the value portion, rendering the query incorrect. For example, consider the criterion *blood products or immunoglobulins within 6 months prior to entering the study*. The system found a mapping to an appropriate concept, "blood products used"; it also found a mapping to the valid concept "months". However, the latter is not a permissible value for the former, so processing failed. If appropriate constraint checking could mediate name-value pairings, the system would be able to more gracefully reformulate such instances.

The synonyms supplied by the HDD produced frequent successes, but occasional ambiguity proved problematic. The system mapped the abbreviation "PCP" to the drug "phencyclidine", whereas the trial intended "pneumocystic carinii pneumonia". It also mapped "PG" to "phosphatidyl glycerol" whereas the trial used it in an ad-hoc fashion for "pathological gambling".

Often unsuccessful queries reflected an absence of relevant concepts from the HDD. This is not unexpected, given the domain's focus on experimental medications. We could use additional sources of clinical concepts such as the National Library of Medicine's Unified Medical Language System [23] or a database of experimental drugs. New concepts, though, would not be helpful unless patient records contain such concepts, which is unlikely.

Several queries provided partial information that was useful, but could not fully assess eligibility. For example, the system mapped the criterion "uterine papillary serous carcinoma", to the concept "papillary carcinoma". Matching "papillary carcinoma" in a patient's record does not necessarily satisfy the criterion, but it could suggest further action by a clinician.

With some criteria a match will never be possible. EMR's typically do not store patient information that would reflect such criteria as "plans to become pregnant during the study" or "male partners of women who are pregnant".

Criteria we missed could be evaluated based on data in the EMR, by adding further inferencing with external knowledge. For example, "meets psychiatric diagnostic criteria for depression" requires the system to know what these diagnostic criteria are before this criterion can be evaluated.

Another possibility for improving the system includes mapping criteria to more VMR classes than just the observation class. This would facilitate more accurate queries against information such as procedures, demographics, and medications.

References

[1] M. Magnani, D. Montesi, A unified approach to structured, semistructured and unstructured data, Tech. Rep. UBLCS-2004-9, University of Bologna (2004).

[2] M. Stephens, M. Palakal, S. Mukhopadhyay, R. Raje, , J. Mostafa, Detecting gene relations from Medline abstracts, in: Proceedings of the Pacific Symposium on Biocomputing, 2001, pp. 483–496.

[3] J. Pustejovsky, J. Castao, J. Zhang, M. Kotecki, B. Cochran, Robust relational parsing over biomedical literature: Extracting inhibit relations, in: Proceedings of the 7th Pacific Symposium on Biocomputing, 2002, pp. 362–373.

[4] R. Bunescu, R. Mooney, A. Ramani, E. Marcotte, Integrating co-occurrence statistics with information extraction for robist retrieval of protein interactions from Medline, in: Proceedings of the HLT-NAACL Workshop on Linking Natural Language Processing and Biology: Towards deeper biological literature analysis (BioNLP-2006), 2006, pp. 49–56.

[5] J. Pustejovsky, J. Castano, B. Cochran, M. Kotecki, M. Morrell, Automatic extraction of acronym-meaning pairs from Medline databases, in: Proceedings of Medinfo, 2001, pp. 371–375.

[6] A. Schwartz, M. Hearst, A simple algorithm for identifying abbreviation definitions in biomedical text, in: Pacific Symposium on Biocomputing, 2003, pp. 451–462.

[7] T. Rindflesch, J. Rajan, L. Hunter, Extracting molecular binding relationships from biomedical text, in: Proceedings of ANLP-NAACL, Morgan Kaufmann, San Francisco, CA, USA, 2000, pp. 188–195.

[8] A. T. McCray, Better access to information about clinical trials, Annals of Internal Medicine 133 (8) (2000) 609–614.

[9] D. Sleator, D. Temperley, Parsing English with a link grammar, Tech. Rep. CMU-CS-91-196, Carnegie Mellon University (October 1991).

[10] G. Schneider, A linguistic comparison constituency, dependency, and link grammar, Master's thesis, University of Zurich (1998).

[11] J. E. Laird, A. Newell, P. S. Rosenbloom, Soar: An architecture for general intelligence, Artificial Intelligence (33) (1987) 1–64.

[12] A. Newell, Unified Theories of Cognition, Harvard University Press, 1994.

[13] J. E. Laird, C. B. Congdon, The Soar user's manual: Version 8.6, Tech. rep., EECS Department, University of Michigan (2006).

[14] C. Parker, Generating medical logic modules for clinical trial eligibility, Master's thesis, Brigham Young University (2005).

[15] K. A. Spackman, K. E. Campbell, Compositional concept representation using snomed: Towards further convergence of clinical terminologies, in: Proceedings of the American Medical Informatics Association Annual Symposium, 1998, pp. 740–744.

[16] S. M. Huff, R. A. Rocha, H. R. Solbrig, M. W. Barnes, S. P. Schrank, M. Smith, Linking a medical vocabulary to a clinical data model using Abstract Syntax Notation 1, Methods of Information in Medicine 37 (4-5) (1998) 440–452.

[17] I. S. I. 8824-1, Information technology—Abstract Syntax Notation One (ASN.1): Specification of basic notation, Tech. rep., iTU-T Recommendation X.680 (2002).

[18] R. A. Rocha, S. M. Huff, P. J. Haug, H. R. Warner, Designing a controlled medical vocabulary server: the VOSER project, Computers and Biomedical Research 27 (6) (1994) 472–507.

[19] G. Hripcsak, P. D. Clayton, T. A. Pryor, P. Haug, O. B. Wigertz, J. v. d. Lei, The Arden Syntax for Medical Logic Modules, in: R. A. Miller (Ed.), Proceedings of the Fourteenth Annual Symposium on Computer Applications in Medical Care, IEEE Computer Society Press, Washington D. C., 1990, pp. 200–204.

[20] C. G. Parker, R. A. Rocha, J. R. Campbell, S. W. Tu, S. M. Huff, Detailed clinical models for sharable, executable guidelines, Medinfo 11 (Pt 1) (2004) 145–148.

[21] C. A. Tustison, Logical form identification for medical clinical trials, Master's thesis, Brigham Young University (2004).

[22] P. Szolovits, Adding a medical lexicon to an English parser, in: Proceedings of the American Medical Informatics Association Annual Symposium, 2003, pp. 639–643.

[23] C. Lindberg, The Unified Medial Language System (UMLS) of the National Library of Medicine, Journal of the American Medical Reccord Association 61 (5) (1990) 40–42.