Developing NLP Tools for Genome Informatics: An Information Extraction Perspective*

Teruyoshi Hishiki [1] † hishiki@is.s.u-tokyo.ac.jp
Nigel Collier [1] nigel@is.s.u-tokyo.ac.jp
Chikashi Nobata [1] nova@is.s.u-tokyo.ac.jp
Tomoko Okazaki-Ohta [2] okap@ims.u-tokyo.ac.jp
Norihiro Ogata [1] † ogata@is.s.u-tokyo.ac.jp
Takeshi Sekimizu [1] sekimizu@is.s.u-tokyo.ac.jp
Roland Steiner [1] steiner@is.s.u-tokyo.ac.jp
Hyun S. Park [14] hsp20@is.s.u-tokyo.ac.jp
Jun'ichi Tsujii [13] tsujii@is.s.u-tokyo.ac.jp

[1] Department of Information Science, University of Tokyo 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-0033, Japan
[2] Human Genome Center, Institute of Medical Science, University of Tokyo, 4-6-1 Shirokanedai, Minato-ku, Tokyo 108-8639, Japan
[3] Department of Language Engineering, UMIST, PO Box 88, Manchester M60 1QD, United Kingdom
[4] Department of Computer Science, Sungshin Women's University, 249-1 Dongsun-dong, Sungbuk-gu, Seoul, Korea

Abstract

Huge quantities of on-line medical texts such as Medline are available, and we would hope to extract useful information from these resources, as much as possible, hopefully in an automatic way, with the aid of computer technologies. Especially, recent advances in Natural Language Processing (NLP) techniques raise new challenges and opportunities for tackling genome-related on-line text; combining NLP techniques with genome informatics extends beyond the traditional realms of either technology to a variety of emerging applications. In this paper, we explain some of our current efforts for developing various NLP-based tools for tackling genome-related on-line documents for information extraction task.

1 Introduction

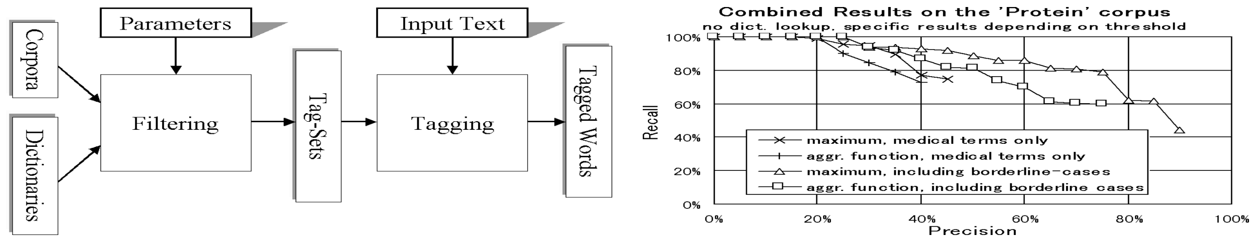
Nowadays, huge quantities of genome-related documents are available online. But the problem is that most of the documents are written in a human language rather than in a specific database format which computers would deal with more easily. Therefore, there is a natural and potentially important marriage between genome informatics and Natural Language Processing (NLP) technologies, and this synergy extends beyond the traditional realms of either technology to a variety of emerging applications.

Natural Language Processing (NLP) is a multidisciplinary subject which draws on a diverse range of areas including linguistics, computer science, psychology, logic, and philosophy. It is a field of study that has developed from an intersection between the general discipline of linguistics and the subset of computer science called "artificial intelligence". The goal of this research is to create computational models of language in enough detail so that computer programs can be written to perform various tasks involving natural language.

On the other hand, it is true that researchers in genome informatics, with rare exceptions, paid little attention to the most recent advances of NLP technologies. However, without understanding the

*This research has been partially supported by the project of JSPS (JSPS-RFTF96P00502) and Human Genome Program by Ministry of Education, Science and Culture.

† Partly supported by Research Fellowships of the Japan Society for the Promotion of Science for Young Scientists.



(a) Overview of the 4 Gram Model (b) Combined Results in Medline Corpus

Figure 1: 4 Gram Model.

state of the art in both genome informatics and NLP technologies, as well as the current and future capabilities and limitations of these two technologies, it is difficult to see what we can do or develop for the future system.

In the following sections, we describe various on-going approaches of ours by combining natural language processing techniques and genome informatics. Section 2 shows our current effort in developing a tool for retrieving genome-domain specific suffixes and morphological patterns, based on 4-gram model. Section 3 discusses the shallow parser EngCG, and the applications based on the tagged corpus generated from it. In section 4, we are developing automatic construction of ontological structures:*taxonomy*, *mereology* and *synonymy*, of the genome domain using semantic and discourse processing. Section 5 discusses how to manage text about genomes in a way that would be useful to extract biological information. Finally in section 6, we will explain the definition of *information extraction* and our final goal in the genome domain.

2 Automatic Identification of Genome-specific Character 4-grams

For various applications in NLP which are specific to medical applications it is necessary to reliably decide whether a given word is medical or not. The main problem in this is that the number of words in medical use is virtually unbounded and new terms are constantly being created. Since it is infeasible, if not impossible, to keep up-to-date dictionaries on all medical terms in existence, a more generic approach should be more advantageous.

There are currently various algorithms in existence with a very high recognition rate, e.g., to recognize protein names in [3], written explicitly for this specific purpose. The algorithm we will present here has a different focus instead: to automatically find genome-specific pre- & suffixes and other identifying morphemes. It is very generic and not bound to a unique set of words[1] and does not need user-defined rules, etc. Being this generic, this algorithm is meant as an additional tool to augment other methods, rather than being a means on its own.

Outline of the system: In our approach we are taking advantage of the fact that medical terms by and large are quite distinct from normal English words, when looking at their character patterns. Thus, we tested various possible training configurations and parameters, using character 4-grams[2] to find specific morphemes. Fig. 1(a) displays the schematic outline of the system.

Due to lack of space we cannot give further detail the system here. The interested reader is instead referred to [9], which explains the workings of the system in a much more detailed way.

[1] Indeed it is not even genome or medicine specific, but could be used on a much broader range of 'special' words.

[2] Mainly to keep memory requirements down. In the meantime we also have a preliminary system that uses character n-grams of arbitrary length, but we cannot give meaningful results for that yet. For a general treatment on character n-grams see [6].

```
"<*i*l-4>"       "*i*l-4" <*> <?> N NOM SG @SUBJ
"<was>"          "be" <SV> <SVC/N> <SVC/A> V PAST SG1,3 VFIN  @+FMAINV
"<the>"          "the" <Def> DET CENTRAL ART SG/PL @DN>
"<only>"         "only" A ABS  @AN>
"<cytokine>"     "cytokine" N NOM SG  @PCOMPL-S
"<that>"         "that" <NonMod> <**CLB> <Rel> PRON SG/PL  @SUBJ
"<binds>"        "bind" <SVO> <SV> <P/with> V PRES SG3 VFIN @+FMAINV
"<to>"           "to" PREP  @ADVL
"<a>"            "a" <Indef> DET CENTRAL ART SG @DN>
"<hemopoietin>" "hemopoietin" <?> N NOM SG @NN>
"<receptor>"     "receptor" N NOM SG  @<P
"<and>"          "and" CC @CC
"<that>"         "that" <NonMod> <**CLB> <Rel> PRON SG/PL  @SUBJ
                 "that" PRON DEM SG  @SUBJ
"<did>"          "do" <SVO> <SVOO> <SV> V PAST VFIN  @+FAUXV
"<not>"          "not" NEG-PART @NEG
"<activate>"     "activate" <SVO> <DER:ate> V INF  @-FMAINV
"<p21ras>"       "p21ras" <?> <NoBaseformNormalisation> N NOM SG/PL @OBJ
"<$.>"
```

Figure 2: The Parsing Result by EngCG.

Results: Following the goal for our algorithm to be a general tool, we also tested it in a relatively generic environment. Thus, we measured its performance in a Text-Retrieval environment on unrestricted text. The results given here were taken from experiments without any dictionary lookup[3] . Due to the large number of results obtained, only the results for the overall best performing parameter set is shown in Fig. 1(b).

3 Developing Tools Based on the Shallow Parsing Techniques

The previous section discussed our experimental tool for automatic recognition of domain-specific suffixes and prefixes. In NLP, this work can be categorized as the work related to morphological analysis, where morphology is the study of the internal structure of words. This section concerns how words can be put together to form correct sentences and determines what structural role each word plays in the sentence and what phrases are subparts of other phrases. The techniques for analyzing a sentence and determining its structure are called *parsing techniques*.

It is true that traditional parsing techniques have been hampered by the limitations of NLP systems which work only on a very small number of restricted texts. Still, recent advances in natural language processing (NLP) technologies raise new challenges and opportunities for tackling genome-related on-line text for information extraction task.

We have adopted a shallow parser called EngCG to tackle Medline abstracts, where the system has been originally developed by Voutilainen, Tapanainen, Koskenniemi, and Karlsson [10]. EngCG, the constraint grammar parser, performs morphological and syntactic analysis of English based on *elimination rules*. EngCG reduces parsing as a problem of tagging, not only for morphology, but also for syntax, and boundary of the phrases. To show what we mean by shallow parsing, we show the shallow parsing result of EngCG in Fig. 2 for the following sentence:

> *IL-4 was the only cytokine that binds to a hemopoietin receptor and that did not activate p21ras.*

Fig. 2 shows basically three kinds of tags. For morphological tags, some part-of-speech tags can be seen such as `A` (adjective), `N` (noun), `DET` (determiner), and so on. Other tags include `SG` (singular), `PL` (plural), `NOM` (nominative), `ABS` (absolute form), `CMP` (comparative) and so on. Especially, notice

[3] In other words: *all* words from the test corpora were considered unknown and to be checked by the algorithm

that some of the derivational forms are also represented: `<DER:ate>` (i.e., derived verb in -ate). Each syntactic tag is prefixed by "@" in contrast to morphological tags. Notice that some tags include an angle bracket, "<" or ">", which indicates the direction where the head of the word is to be found. In Fig. 2, `@+FAUXV` indicates finite auxiliary predicator, `@SUBJ`, a subject, `@OBJ`, an object, `@AN>`, a premodifying adjective, and so on. Some special tags can also be found in Fig. 2: `<?>`, a morphological reading assigned by the guessing component, and `<**CLB>`.

We have applied EngCG to parse a one million word corpus, which has been extracted from the Medline abstracts. The morphological disambiguator seldom discards an appropriate morphological reading: after morphological analysis, 99% of all words retain the appropriate analysis. But only 3% remain partly ambiguous. (error rate 0.4%) After syntactic analysis, 64% of the sentences become syntactically unambiguous, while only 2% of all sentences contain more than 10 readings. 3-6% aer unresolved as the expression of the grammar rules without direct referece to syntactic functional labels and clause boundaries becomes inhibitively difficult.

Based on the EngCG tagging information, we were able to identify (or bracket) noun phrases contained in the corpus, with the precision rate of upto 90%. Based on the noun-phrase-bracketted corpus, we tried to find the subject and object terms for frequently seen verbs in Medline corpus. The precision rate of finding the right subject and object were about 72.9%. Considering that the percentage of the protein and gene names appearing as the subject or object terms for these verbs were about 65%, the current task will provide valuable information about the characteristics of each protein and gene. Further technical details of this work can be found in [7].

4 Towards an Automatic Construction of a Type-theoretic Taxonomy, Mereology and Synonymy

In this part of our project, we will try to extract information on taxonomy, mereology and synonymy in the genome domain by robust natural language processing based on type theory and discourse theory. This stage is based on morphologically and syntactically analyzed texts which do not need to be exact and fully processed. In this stage, we extract semantic information which includes discourse-based information, like anaphora.

A taxonomy of a domain is identified as the subtype structure of the domain. A mereology and synonymy is constructed from type information, predicates which imply part-of relations or identity relations. The structure is constructed from such extracted information and managed by a non-monotonic inference system of the structure, which can make us update or revise the structure easily.

Our processing algorithm for a medical texts is as follows:

$T_k = (l_i)_{i \in I(k)}$:text, $S = \emptyset$:stacks of discourse information

foralll_i in T_k

    $(w_j)_{j \in l(i)} = \mathbf{engcg}(l_i)$;
    forallw_j in $(w_j)_{j \in I(i)}$
        forallR_m in rule set R
            if $C_{R_m}(w_j)$ **then** $S_j = R_m(w_j, S)$ **else** $S_j = S$;
            push(S, S_j);

Add S **to** deductive data base Δ;

where C_{R_m} is a condition of a processing rule R_m. For **engcg**, see the previous section. Δ consists of the axioms of type-inference rules, definition of equivalence, axioms of 'part-of' relation, user's bias, and background knowledge of the domain. Each rule R_m has a word and stacks of discourse information as its inputs. R consists of discourse processing rules such as anaphora resolution rules, and rules defined based on the following general linguistic observations and a type theoretical design:

- Noun phrases with numeric expressions often denote typing information in the domain, e.g., [two$_{DET}$ A-T families]$_{NP}$ $\mapsto$ `A-T_family:type`; [two$_{DET}$ type A viruses]$_{NP}$ $\mapsto$ `type_A_virus:type`;
- Indefinite appositives often denote typing information in the domain: e.g., 'FMDV, a picornavirus'$\mapsto$`FMDV:picornavirus`, `picornavirus:type`;
- Exemplification constructions denote typing information in the domain: e.g., 'three different strains of FMDV, O1BFS, A10(61) and A22 Iraq'$\mapsto$ `O1BFS:strain(FMDV)`, `A10(61):strain(FMDV)`, `A22:strain(FMDV)`;
- Parenthesized expressions denote information on synonymy in the domain: e.g., 'foot-and-mouth disease virus (FMDV)'$\mapsto$ `FMDV`$\equiv$`foot-and-mouth_disease_virus`;
- Exemplification copula constructions denote typing information: e.g., *be one of* , denotes typing information;
- Some verbs denote the synonymy and typing information of the domain: e.g., *termed*, *designated*;
- Some verbs denote the mereology of the domain: e.g., *consist of*, *be ... part of*;
- etc.

The robustness for unknown words is achieved by the specification of the above rule. They are specified only on the grammatical construction and functional words, and domain-independent words. So, they need no domain-specific lexicon.

A taxonomy is basically constructed by the type inference based on the definition of subtype relation:

$$T_1 \sqsubseteq T_2 \Leftrightarrow \forall x : T_1 \rightarrow x : T_2.$$

However, since we adopt non-monotonic subtype system, our taxonomy can interact with user's bias or background knowledge. A mereology is basically constructed by the meaning postulates of verbs such as:

$$R(x, y) \rightarrow part\_of(x, y).$$

A synonymy is basically constructed from the definition of equivalence relation. Therefore, a query to the resultative data base constructs a taxonomy, mereology and synonymy written in the domain-specific texts.

5 Towards a Unified Structure of Genome Documents in Standard Generalized Markup Language (SGML)

Besides the gene sequences, there are various kinds of information e.g. materials and methods in biological research papers. It has been pointed out that contents of biological research papers are important for the enhancement of knowledge bases, and NLP is one of the key approaches [5]. We think that some measures are necessary to correlate specific linguistic structures in phrases or sentences to biological information they convey is necessary. The purpose of this section is to provide a framework of the document that can bear a variety of additional information generated by both NLP and human judgment. In this section, we also show how the text with this framework can be used to extract information related to genome domain.

In order to add information to text, parts of the text are usually bracketed (marked up) by tags representing a certain type of information. It should be done systematically so that consistent marking up is assured. We chose SGML for that purpose.

5.1 What is SGML?

SGML[1] is a methodology to mark up text systematically. An SGML document consists of elements that have been predefined.

A part of the document written in SGML (SGML instance) might look as follows:

```
<experiment><sentence><procedure>We cotransfected <cell>
<phrase linguistic-category = "np">TYS cells</phrase></cell>
 with <gene><phrase linguistic-category = "np">the sense
</phrase></gene> or <gene><phrase linguistic-category = "np">
antisense</phrase> <phrase linguistic category = "np">
expression vector</phrase></gene> and <virus>
<word linguistic category = "np">pSV2neo</word></virus>
</procedure> and <result>obtained more than 200
<phrase linguistic-category = "np">G418-resistant colonies</phrase>
 ...
</result>.</sentence>
<result><sentence>Approximately 80% of
<phrase linguistic-category = "np">representative <cell>
G418-resistant clones</cell></phrase> expressed
..... </sentence></result></experiment>
```

The above example is about an experiment consisting of two sentences. In the former half of the first sentence, the procedure of the experiment is described. In the latter half of the first sentence and in the second sentence, the results of the procedure are described. Linguistic features are added by tagging (marking up) the phrases and giving the "linguistic-category" attribute to it. Biological features are added by marking up the phrases with tags such as <protein> or <gene>.

5.2 What is Document Type Definition (DTD)?

The grammar to define a SGML document is called Document Type Definition (DTD). A DTD mainly consists of three types of declaration: element declaration, attribute definition list declaration, and entity declaration. These declarations have their syntax. Examples are as follows:

```
Element declaration: <!ELEMENT abstract - - (background
| purpose | experiment | conclusion | perspective )+>
```

means that the abstract element is a collection of an element that is either background, purpose, experiment, conclusion, or perspective.

```
Attribute definition list declaration: <!ATTLIST phrase
 linguistic-category (np | vp | adjp | advp | pp) #IMPLIED>
```

means that the phrase element has the attribute of "linguistic-categeory" whose possible values are either "np", "vp", "adjp", "advp", or "pp".

```
Entity declaration: <!ENTITY % genome-term "protein
| protein-motif|gene|gene-motif|cell|cell-fraction
|species|method|experiment-condition">
```

means that "genome-term" stands for either of "protein", "protein-motif", "gene", "gene-motif", "cell", "cell-fraction", "species", "method", or "experiment-condition".

The next sections describe how we developed the DTD for documents in the genome domain, especially abstracts of scientific papers.

5.3 Planning DTD

The features that the elements of documents would have were listed using some abstracts of scientific papers in the genome domain. Five types of features were identified as follows:
Logical constituents of a scientific paper: background, purpose, experiment, conclusion, etc.
Terms in the genome domain: gene, gene motif (substructure of a gene), protein, protein motif, cell, species (mouse, rat, human, etc.), etc.
Strings with linguistic categories: sentence, noun phrase, verb phrase, noun, verb, adjective,

preposition, etc.
The following two features were intended for Information Extraction (IE) tasks, and the terms "object" and "event" generally follow [2]. We are going to explain more about IE tasks later.
Components of an object: An object is defined as an aggregate of information to represent an entity such as protein or gene. For example, a set of information such as the name of the protein, characteristic motifs, and the gene encoding the protein are used to describe a protein. Strings representing this information have the feature "object".
Details of an event: An event is defined as an aggregate of information to represent any change in one or more objects. For example, the co-precipitation of proteins is an event involving different types of protein and producing a complex protein. In this example, the proteins are objects related to this event, "co-precipitation", by the "original protein" role, and the product complex protein is an object related to this event by the "product complex" role.

Then, detailed values for these types were discussed by some of the authors of this paper with biomedical background.

5.4 Designing and Verifying DTD

The features listed in the planning phase were mapped to the elements or attributes of DTD. For the convenience of the management of additional information, the first three features were represented by different elements. Tentatively, each logical constituent of a scientific paper and each genome domain term was allocated its element, and strings with linguistic categories were defined to be the attributes of either a word or a phrase consisting of more than one word.
Because an object may consist of elements scattered in a document, we defined an object element as an element that is the key of a object and holds the ID and the name of the object. An event element was defined to be consecutive phrases or sentences containing necessary objects.
The relationship between elements such as object-attribute relationship and event-object relationship was defined as the attributes of the elements that are the slots of an object or an event. Because the specification of SGML does not allow an attribute to be used more than once for an instance of an element, a notation to pack the multiple relationships in a string was defined.

We checked the consistency of the DTD and some instances using SGMLS , a SGML parser (see `http://www.iclark.com/sp/` for further details). In the checking process, we extended the original DTD to meet the needs in marking up. The resultant DTD is presented in Fig. 3(a).

5.5 Using Tagged Data

An example of the SGML instance is presented in Fig. 3(b). We are going to build Information Extraction (IE) environment using such data. Our works include the development of a Tool to support human marking up, semi-automatic pattern acquisition tool for IE, and IE tools. They are discussed in the next section.

6 Automatic Pattern Refinement for Verb Classification

The desirable result of Information Extraction in the genome domain is formally defined in the section 5. In this section, we will explain the definition of Information Extraction and our tool for the task.

Information Extraction: Information Extraction(IE) is a task to extract only the interesting part of a sentence or a document. The Message Understanding Conferences (MUCs [2]), a series of competitions joined by developers of information extraction systems, have defined several categories of tasks in IE. In MUC-6, there are four tasks[4].

```
<!DOCTYPE abstract [

<!-- Groups of elements with similar properties -->
<!ENTITY % content "background | purpose | experiment | conclusion |
perspective">
<!ENTITY % simple-content "background | purpose | conclusion |
perspective">
<!ENTITY % complex-content "experiment">
<!ENTITY % experiment-component "procedure | result | previous-result">
<!ENTITY % content-component "%experiment-component;">
<!ENTITY % sentence-component "phrase | word | #PCDATA">
<!ENTITY % genome-term "protein | protein-motif | gene | gene-motif
| gene-location | cell | cell-fraction | species | method |
experiment-condition">
<!ENTITY % genome-phrase "molecular-function">
<!ENTITY % object-type "object">
<!ENTITY % event-type "event |%content; |%content-component;">

<!-- Attributes shared by elements -->
<!ENTITY % linguistic-category "linguistic-category (np | vp | adjp | advp
|pp) #IMPLIED">
<!ENTITY % coref "id NUMBER #IMPLIED ref CDATA #IMPLIED">
<!ENTITY % object-key-property "object-id CDATA #IMPLIED object-name CDATA
#IMPLIED">
<!ENTITY % event-property "event-id CDATA #IMPLIED event-type CDATA
#IMPLIED">
<!ENTITY % related-object "relation-to-object CDATA #IMPLIED">
<!-- relation-with-object :=
related-object-id:attribute+related-object-id:attribute;...-->
<!ENTITY % related-event "relation-to-event CDATA #IMPLIED">
<!-- relation-to-event := related-event-id:role+related-event-id:role;...
-->

<!-- Most minute units of document elements -->
<!ENTITY % genome-text-component "%sentence-component; | %genome-term;">
<!ENTITY % genome-sentence-component "%sentence-component; | %genome-term;
| %genome-phrase;
| object | event">

<!-- Blocks for building domain-specific terms and expressions -->
<!ELEMENT (%genome-term;) - - (%genome-text-component;)+>
<!ELEMENT (%genome-phrase;) - - (%genome-text-component;)+>

<!-- A key element of an object -->
<!ELEMENT object - - (%genome-text-component; | object)+>

<!-- A larger block containing sentence(s) that represents an event -->
<!ELEMENT event - - (sentence | event | %content; | %content-component;)+>

<!-- Organization of sentences and phrases -->
<!ELEMENT sentence - - (%genome-sentence-component; | %content-component; |
%content;)+>
<!ELEMENT phrase - - (%genome-text-component;)+>
<!ELEMENT word - - (#PCDATA)>

<!-- Contents of abstract in genome domain -->
<!ELEMENT abstract - - (%content; | event)+>
<!ELEMENT (%simple-content;) - - (sentence | %genome-sentence-component;)+>
<!ELEMENT (%complex-content;) - - (%experiment-component; |
%genome-sentence-component;
| sentence)+>
<!ELEMENT (%content-component;) - - (sentence |
%genome-sentence-component;)+>

<!-- Linguistic attributes of words and phrases -->
<!ATTLIST phrase %linguistic-category;>
<!ATTLIST word %linguistic-category;>

<!-- Attributes of genome terms-->
<!ATTLIST (%genome-term;) %coref; %related-object; %related-event;>

<!-- Attributes of an object key -->
<!ATTLIST (%object-type;) %object-key-property; %related-object;
%related-event;>
<!-- Attributes of an event -->
<!ATTLIST (%event-type;) %event-property; %related-event;>
]>
```

(a) DTD of Abstracts for Papers in Genome Domain

```
<!DOCTYPE abstract system "genome_new.dtd" []>
<abstract><background event-id = "e1" event-type = "background">
<sentence><gene id = "1" ref = "4+5+6">A family of human genes</gene>
 encoding <protein id = "2"><protein-motif id = "3">basic-leucine zipper
(bZIP)</protein-motif> transcription factors</protein>, <object object-id
= "o1" object-name = "gene"><protein id = "4" relation-to-object =
"o1:eocoded-protein">p45-NF-E2</protein></object>,<protein id = "5">Nrf1
</protein> and <protein id = "6">Nrf2</protein>, have been isolated
independently.</sentence></background>
<experiment event-id = "e2"><result event-id = "e3"><sentence>Whereas
<protein id = "7" ref = "4+5+6">the encoded proteins of <gene id = "8">
the three genes</gene></protein> share <protein-motif id = "9">highly
conserved regions</protein-motif> distinct from <protein id = "10" ref
= "12+13">other <protein-motif id = "11">bZIP</protein-motif> families
</protein> such as <protein id = "12">Jun</protein> or <protein id = "13">
Fos</protein>, <protein-motif id = "14">remaining regions</protein-motif>
 diverged considerably from <protein-motif id = "15">each other
</protein-motif>.</sentence></result></experiment>
<experiment event-id = "e4"><result event-id = "e5"><sentence>
<method id = "16"><method id = "17">Chromosomal localization</method> by
<method id = "18">fluorescence in situ hybridization</method></method>
 demonstrates that <gene id = "19" ref = "8">these genes</gene> are
non-syntenic.</sentence></result></experiment>
<experiment event-id = "e6"><result event-id = "e7"><sentence>
<protein id = "20" ref = "4">p45-NF-E2</protein> mapped to <gene-location
id
 = "27" relation-to-object = "o1:chromosome-location">chromosome
12q13.1-13.3
</gene-location>, whereas <protein id = "21" ref = "5">Nrf1</protein> and
<protein id = "22" ref = "6">2</protein> mapped to <gene-location id =
"28">
17q21.3</gene-location> and <gene-location id = "29">2q31</gene-location>,
respectively.</sentence></result></experiment>
<conclusion event-id = "e8"><sentence>However, <gene id = "23" ref =
"8">these
three genes</gene> <phrase linguistic-category = "vp">were probably derived
from </phrase><gene id = "24">a single ancestor</gene> by chromosomal
duplication as <gene id = "25">other genes</gene> that also map in
<gene-motif
id = "26">these regions</gene-motif> are related to one another.</sentence>
</conclusion>
</abstract>
```

(b) An Instance of Tagged Text

Figure 3: Document Type Definition.

Named Entity (NE) Task is to bracket proper names and important expressions by SGML tags. In MUC-6, the names of people or organizations were the target of this task. *Coreference (CO) Task* is to bracket strings coreferring noun phrases by SGML tags. *Template Element (TE) Task* is to extract basic information related to organization and person entities, such as the organization name, a category for the organization, and the location of the organization. Finally, *Scenario Template (ST) Task* is to extract a specific event such as changes in corporate managers using the results of the above three tasks, and relate the event information to the organizations and people involved in the event.

We will show some instances of the IE task in the genome domain. In the genome domain, an NE task would be to extract names of genes, proteins, and cells, etc. In the SGML instance shown in Fig. 3(b), "`p45-NF-E2`", "`Nrf1`", and "`Nrf2`" would be the protein names. No specific name is given to the genes encoding these proteins. CO task would be to extract the coreferential relationship among noun phrases, e.g., "`basic-leucine zipper (bZIP) transcription factors`", "`the encoded proteins`", "`these genes`", etc. indicating the proteins or genes extracted in the NE task. A TE task would be to create the descriptions of entities such as genes or proteins. In the SGML instance shown in Fig. 3(b), the gene encoding the protein "`p45-NF-E2`" also encodes the motif "`bZIP`", and the location of the gene is "`12q13.1-13.3`", therefore their entities would be described as follows:

GENE:	
Gene ID in the document:	`<GENE-1>`
Gene name:	none
Encoding Protein ID:	`<PROTEIN-1>`
Location of gene:	`12q13.1-13.3`

PROTEIN:	
protein ID in the document:	`<PROTEIN-1>`
protein name:	`p45-NF-E2`
protein motif:	`ZIP`

An ST task would be to extract events or relationships among the above genes and proteins, and create their descriptions. It is quite difficult to define the final format of the events that should be extracted, so we will set the definition as our system progresses.

Tool for Information Extraction: Now we are creating a domain-independent Information Extraction tool. It is part of our future work to apply the tool and verify whether it is useful to a genome domain. Our purpose in the genome domain is to perform semantic classification of verbs automatically. This data will be the knowledge base for the ST task in the genome domain.

For this purpose, we will use an existing automatic analyzer of proper names, i.e. a Named Entity tagger[8], as a part of our system.

Named Entity Tagger: The Named Entity tagger is a supervised learning system using a decision tree technique, but the performance is pretty good with only about 100 training articles, which means the analyzer is quite adaptable to a new domain. In fact, it works well in the domain of an executive succession and also a vehicle accident. Obviously, the genome domain has its own peculiar jargon, and the documents are quite different form newspaper articles, therefore we need to create the initial knowledge base for the analyzer in addition to training corpus. However, the manual task will be reduced if the analyzer works in the genome domain to the same extent as in the previously tested domain.

Overview of the System: We will explain an overview of our system. We can already obtain a set of specific verbs that has a significant co-occurrence with the names of proteins. First, our system collects sentences including these verbs from the Genome documents, and then applies the above-mentioned Named Entity tagger. We expect there will be some similarities among these sentences with Named Entity information. Our system clusters the sentences based on a certain similarity measure, and finally makes patterns for the verbs. Each pattern should be the abstract description that expresses one usage of the verbs.

It is necessary to firstly decide on an appropriate similarity measure and threshold on clustered sentences. We will then make several experiments.

7 Conclusion

Our group has many researchers in various backgrounds, including medicine, computer science, linguistics and genetics, and the architecture of the future system is being carefully designed with the cooperation of each group. We have discussed some of our current efforts in developing various NLP tools. Though each of our tools will be able to extract the helpful information from input texts on its own, all these tools will be integrated into a single information extraction system in the near future. The task to extract the events among proteins and genes, which is our final goal, will require all levels of information, lexical, syntactic and semantic, from the given documents.

We should stress that we are still in the process of developing our system. Some of the works already showed promising results, while other works are still in a preliminary stage as we have just started our project. However, we think that much progress will be made in the future.

References

[1] Bradley, N., *The Concise <SGML> Companion*, New York, Addison Wesley Longman, 1997.

[2] DARPA, editor. *Proceedings of Sixth Message Understanding Conference (MUC-6)*, Morgan Kaufmann Publishers, Inc., 1995.

[3] Fukuda, K., Tamura, A., Tsunoda, T., and Takagi, T., Toward information extraction: Identifying protein names from biological papers, In *Proc. of the Pacific Symposium on Biocomputing '98 (PSB'98)*, 707–718, 1998.

[4] Grishman, R. and Sundheim, B., Message Understanding Conference-6: A Brief History. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING 96)*, 466–471, Copenhagen, August, 1996.

[5] Hafner, C.D., Baclawski, K., Futrelle, R.P., Fridman, N., and Sampath, S., Creating a knowledge base of biological research papers, *ISMB*, 147–155, 1994.

[6] Riseman, E.M. and Hanson, A.R., A contextual postprocessing system for error correction using binary n-grams, *IEEE Transactions on Computers*, C-23(5):480–493, 1974.

[7] Sekimizu, T., Park, H.S., and Tsujii, J., Identifying the interaction between genes and gene products based on frequently seen verbs in Medline abtracts, *Genome Informatics 1998*, Universal Academy Press, Inc., Tokyo, Japan, 1998.

[8] Sekine, S., Grishman, R., and Shinnou, H., A Decision Tree Method for Finding and Classifying Names in Japanese Texts, In *Proceedings of the Sixth Workshop on Very Large Corpora*, 171–178, Montreal, Canada, August, 1998.

[9] Steiner, R. and Tsujii, J., Character 4-grams as a tool for semantic tagging, *IPSJ SIG Notes*, 157–164, 1998.

[10] Voutilainen, A., Designing a (finite-state) parsing grammar, In *Finite-State Language Processing*, E. Roche and Y. Schabes (editors), A Bradford Book, The MIT Press, 1996.