# Identifying the Interaction between Genes and Gene Products Based on Frequently Seen Verbs in Medline Abstracts

**Takeshi Sekimizu** [1]
sekimizu@is.s.u-tokyo.ac.jp

**Hyun S. Park** [1 3]
hsp20@is.s.u-tokyo.ac.jp

**Jun'ichi Tsujii** [1 2]
tsujii@is.s.u-tokyo.ac.jp

[1] Department of Information Science, University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8654, Japan
[2] Department of Language Engineering, UMIST, PO Box 88, Manchester M60 1QD, United Kingdom
[3] Department of Computer Science, Sungshin Women's University, 249-1 Dongsun-dong, Sungbuk-gu, Seoul, Korea

**Abstract**

We have selected the most frequently seen verbs from raw texts made up of 1-million-words of Medline abstracts, and we were able to identify (or *bracket*) noun phrases contained in the corpus, with a precision rate of 90%. Then, based on the noun-phrase-bracketted corpus, we tried to find the subject and object terms for some frequently seen verbs in the domain. The precision rate of finding the right subject and object for each verb was about 73%. This task was only made possible because we were able to linguistically analyze (or *parse*) a large quantity of a raw corpus. Our approach will be useful for classifying genes and gene products and for identifying the interaction between them. It is the first step of our effort in building a genome-related thesaurus and hierarchies in a fully automatic way.

## 1 Introduction

Considering that there are a lack of standards for storing on-line text, the techniques in *Natural Language Processing* (NLP) are likely to become more and more important; it is necessary to extract useful information from the raw text and to store it in a computer-readable database format.

Much useful information would be extracted from the Medline abstracts, once we are able to analyze (or *parse*) the text and obtain useful linguistic knowledge such as *morphological*, *syntactic*, *semantic* and *discourse* information of the text[1] [1]. Thus, combining NLP techniques with genome informatics extends beyond the traditional realms of either technology to a variety of emerging applications.

However, handling real-time texts in Medline abstracts which contain many unknown words and human errors is not an easy task. Indeed, it is true that traditional NLP research has been hampered by the limitations of NLP systems which work only on a very small number of restricted or annotated texts. Still, recent advances in NLP technologies raise new challenges and opportunities for tackling genome-related on-line text for information extraction task [4].

In our approach, we first retrieved around one million word corpus from Medline abstracts (see Section 2). Then, we adopted a shallow parsing technology in place of a traditional full parsing to

[1] In computational linguistics, morphological knowledge concerns how words are constructed from more basic meaning units called morphemes, where a morpheme is the primitive unit of meaning in a language. Syntactic knowledge concerns how words can be put together to form correct sentences and determines what structural role each word plays in the sentence. Semantic knowledge concerns what words mean and how these meanings combine in sentences to form sentence meanings. Discourse knowledge concerns how the immediately preceding sentences affect the interpretation of the next sentence. A good introductory book about *natural language processing* in general may be James Allen's "Natural Language Understanding"

analyze the corpus, using a system called EngCG [9] from Lingsoft. After shallow parsing has been done, each word in the corpus was assigned *morphological*, *syntactic*, and *boundary* tags (see Section 3 for the meaning of these tags). The next step was to bracket noun phrases in the corpus in a fully automatic way, i.e., given a sequence of words in a sentence, we were able to set boundaries only around the noun phrases to distinguish them from other linguistic structures. This task, sometimes called the problem of '`noun phrase bracketting`' in NLP, has been successfully performed with 90% *precision*, which is relatively better than we had originally expected given a short period of time (see Section 4).

Once noun phrases were bracketted in the corpus, many useful applications could have been experimented. Among them, we tried to find subject and object terms for frequently seen verbs, listing the information as sentence-like assertions and saving them in the database. Some of these verbs include: *activate*, *bind*, *interact*, *regulate*, *encode*, *signal* and *function*. Considering that many of the subject and object terms are actual names of genes and gene products, we thought that these assertions would give vital information in identifying the interaction between genes in the future.

## 2 The Characteristics of Our Domain

Initially, we used 'protein binding' as our key word in retrieving abstracts from Medline. But considering that the term 'protein binding' is simply too abstract, we decided to use more specific key words such as "leucine zipper", "zinc finger" and "helix loop helix motif", which are parts of nuclear proteins. The corpus of ours, collected in this way from Medline contains 898441 words ( or 34343 sentences ).

Then, we examined several most frequently seen verbs from our corpus: *activate*, *bind*, *interact*, *regulate*, *encode*, *function* and etc. Depending on the verbs, the distribution of the conjugational forms was slightly different. For example, there were 2382 occurrences of the words, related to the stem word, *activate*. Among them, the most frequently seen conjugational forms are "*activation*"(1230 cases) which accounts for some 50% of the whole occurrences of the words derived from the word *activate*. Others include: *activated* (187 cases as a verb; 81 cases as an adjective), *activate* (279 cases) *activates* (183 cases), *activator* (298 cases), *inactivate* (18 cases), '*...-activated*' (71 cases), and '*...-activating*' (34 cases).

## 3 Shallow-Parsing the Medline Abstracts

In NLP, the techniques for analyzing a sentence and determining its structure are called *parsing techniques*. (i.e., 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.) It is extremely important to parse the corpus first, to get the maximum morphological and syntactic information for each sentence, and probably only after that, it is possible to achieve sophisticated tasks such as identifying the relationship between proteins and gene products by a fully automatic way as we will be showing here. However, it is well known that parsing unrestricted texts like the ones in the genome domain is extremely difficult. Thus, we use *partial* and *shallow parsing* techniques in place of traditional full parsing; *shallow* and *partial parsing* techniques aim to recover syntactic information efficiently and reliably from unrestricted text, by sacrificing completeness and depth of analysis. We used EngCG to parse the one million word corpus, which took about 35 minutes on a Pentium Processor.

**EngCG, A Shallow Parser:** EngCG has been originally developed by Voutilainen, Tapanainen, Koskenniemi, and Karlsson [6, 10]. EngCG reduces parsing as a problem of tagging, not only for

```
"<*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 1: The Result of Shallow Parsing by EngCG.

morphology, but also for syntax, and boundary of the phrases. To show the meaning of shallow parsing, we show the shallow-parsed result of the following sentence in Fig. 1:

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

Fig. 1 shows basically three kinds of tags. Here, the feature names which start with @ are called syntactic tags, and the rest are called either morphological or boundary tags. Some morphological tags can be seen such as A (adjective), N (noun), DET (determiner), and etc. Other tags include SG (singular), PL (plural), NOM (nominative), ABS (absolute form), CMP (comparative) and so on. Especially, notice 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 and boundary tags. Notice that some tags include an angle bracket, "<" or ">". The angle bracket indicates the direction where the head of the word is to be found. In Fig. 1, @+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. 1: <?>, meaning a morphological reading assigned by the guessing component, and <**CLB>, meaning a clausal boundary tag.

**Unknown Words:** Considering that 52053 words, or 5.7% out of the million words were unknown, it is quite important to consider how to deal with these. The strong point about EngCG is its ability to guess some morphological or even syntactic tags of unknown words, which is especially useful for tackling genome domain texts. Notice that the unknown words IL-4 and p21ras in Fig. 1 could not have been recognized as they were not registered in the computer dictionary (i.e., tagged as <?>). But still the two words could have been correctly guessed as a subject (@SUBJ) and object (@OBJ), respectively. This is possible because the parser looks at local constraints only, ignoring global constraints and recursive aspects of natural language. (For example, the parser is guessing that 'p21ras' is an object just by taking it into consideration that *the word is directly preceded by a transitive verb.*)

Some of the examples of unknown words are listed as below:

```
"viridis" <?> N NOM SG/PL @<P
"heterodimerizing" <?> PCP1 @<P-FMAINV
"immunoblotting" <?> PCP1 @<P-FMAINV
"immunoprecipitated" <?> <SVO> PCP2 @<NOM-FMAINV
"two-dimensional" <?> <Nominal> A ABS @AN>
"ligand-activated" <?> A ABS @AN>
"carboxy-terminal" <?> <Nominal> A ABS @AN>
"genomic" <?> <Nominal> A ABS @AN>
```

Among 52053 unknown words, 37796 words have been tagged as NN (73%), 5610 words as @SUBJ (11%), 4280 words as @OBJ (8%), 139 words as a verb (0.2%), and 10553 words as an adjective (20%)[2] .

**Ambiguity:** According to [9], EngCG, the constraint grammar parser of English, performs morphological and syntactic analysis of English based on *elimination rules*. The current version of EngCG parser uses 282 syntactic mapping rules, 492 syntactic constraints and 204 heuristic syntactic constraints. The current error rate, when parsing genome-related unrestricted running text, is approximately 2%, i.e., 2 words out of 100 get the wrong syntactic code. But the ambiguity rate is still fairly high, 16.4% in our sample, which means that 16 words out of 100 still have more than one morphological or syntactic alternative. For example, notice that the word "that" in Fig. 1 has not been fully disambiguated and that it has two readings, i.e.,

```
"<that>"
        "that" <NonMod> <**CLB> <Rel> PRON SG/PL  @SUBJ
        "that" PRON DEM SG  @SUBJ
```

meaning that the parser was not able to decide whether the word "that" in this case represent a relative pronoun or simply a pronoun.

## 4 Noun Term Recognizer

Given the syntactically tagged corpus from the method described in the previous section, the next step that we might want to do is to identify the noun phrases in the text [3] . This is traditionally called a problem of 'noun phrase bracketting'. Our method of recognizing noun phrases can be expressed approximately in the following regular expression:

```
[ D>N ? [ [ M>N + [ ( . + ) ] ? M>N * [ CC M>N + [ ( . + ) ] ? M>N * ] * ] *
            HEAD CD ? [ ( . + ) ] ? ]
      + [ N< D>N ? [ [ M>N + [ CC D>N ? M>N + [ ( . + ) ] ? M>N * ] * ]
                    * HEAD CD ? [ ( . + ) ] ? ] + ]  * | PRON ] !
```

where [ and ] are for grouping, + for one or more occurrences of args, * for zero or more occurrences of args, M>N for premodifiers, D/M>N for determiners and premodifiers, HEAD for nominal heads except pronouns, N< for prepositions starting a postmodifying prepositional phrase.

The relationship between our notations and the EngCG tags are approximately as follows:

```
M>N:  @NN>, @AN>, @QN>, @GN>, @AD-A>                                                      D>
HEAD: ( @SUBJ , @OBJ, @I-OBJ, @<P )  &&  ( NOM, PRON ,NUM ,PCP1 )
M>N:  ( @SUBJ , @OBJ, @I-OBJ, @<P )  &&  ( not ( NOM, PRON , NUM , PCP1 ))
```

[2] SG/PL (singular or plural) means that the parser was not able to decide the number of the word".

[3] The task of identifying noun phrase in the unrestricted text has been traditionally regarded as a very difficult task. Church was the first one to tackle unrestricted large text by Hidden Markov Model [2].

```
N<:   @<NOM-OF ,  (@<NOM' && PREP)
CC:   @CC &&  not("but")
CD:   CARD , "I", "II"
PRON: ``you'', ``he'', ``she'', ``it'', ``they'', ``we'',  ``them''...
```

Fig. 2 shows our sample output where the noun phrases have been identified by the above regular expressions plus some algorithmic constraints. Basically, we are only concerned about recognizing simplex noun phrases at this moment. However, notice that some of the prepositional attachment could have been determined and could have been combined with the previous noun phrase as can be seen below and in Fig. 2.

<NP> coexpression with mutant Env </NP> .

This has been possible as the EngCG tags carry some syntactic information such as `@<NN`, meaning that the noun is modifying the previous term.

Though the overall performance was much better than we had originally expected, there are still many cases where the noun phrase has been recognized incorrectly. Some of the failures were simply due to the fact that we are in an early stage of developing our system.

> nevertheless , <NP> all viruses </NP> produced from <NP> mixed transfection </NP> showed <NP> decreased infectivity </NP> compared with that of <NP> the wt virus </NP> when <NP> a multinuclear-activation beta-galactosidase induction assay </NP> was performed . <NP> the ability of wt Env </NP> to induce <NP> cytopathic effects </NP> was inhibited by <NP> **coexpression with mutant Env** </NP> . <NP> coexpression of mutants </NP> inhibited <NP> the ability of the wt protein </NP> to mediate <NP> virus-to-cell transmission </NP> , as demonstrated by <NP> an env trans-complementation assay with a defective HIV-1 proviral vector </NP> .

Figure 2: Noun Phrase Bracketting.

For example, consider the following example where '*the suppressor of Hairy wing (...*' has been wrongly recognized.

... <NP> *the suppressor of Hairy wing* **( su ( Hw )** </NP> **)** <NP> *protein* </NP> *inhibits promoter with respect to ...*

The above error occurred in our earlier draft version, simply because we did not consider a case of nested parentheses (i.e., **'( su ( Hw ) )'** ). This error had been fixed simply by modifying our algorithm slightly to consider upto 3 nested parentheses.

Still in many cases, the problems were more fundamental, arising from the EngCG parsing ambiguity. Much of the remaining ambiguity is of the prepositional attachment type or noun/verb ambiguity for some words. This particular type of ambiguity accounts for approximately 40% of all remaining ambiguity.

The most problematic case occurs when a sequence of the words matches the above regular expressions, and some words in the sequence retain an ambiguous reading. For example, the following sentence:

> It has been previously shown that <NP> a proline substitution </NP> for <NP> any of the conserved leucine or isoleucine residues </NP> located in <NP> the leucine zipper-like heptad repeat sequence of human immunodeficiency virus type 1 ( HIV-1 ) gp41 *renders* viruses </NP> noninfectious and envelope (Env) ...

has been wrongly bracketted, because *EngCG* was not able to decide the part-of-speech of the word '*renders'* between verb and noun readings as shown below, and our algorithm simply could not throw away the noun reading in this case.

```
"<renders>"
        "render" <SVOC/A> <SVO> V PRES SG3 VFIN @+FMAINV
        "render"  <DER:er> N NOM PL  @<P
```

This specific problem could have been remedied to a certain degree simply by applying several simple heuristics. For example, one of the heuristics says that if we have a very long noun phrase (more than 7 words), and a certain word in that phrase has two readings (usually noun/verb ambiguity), we simply throw away the noun reading.

Of course, many of the remaining ambiguities (especially PP attachment) are genuine and should be retained. Currently, we have studied many cases of various unresolved ambiguities and we believe that, in many cases, these ambiguities can be resolved by applying domain-specific heuristics for genome domain.

# 5 Identifying the Interaction between Genes

Once noun phrases can be identified from the text, the next target is to analyze the text to find out the interaction between genes, which are usually expressed by frequently seen verbs in our domain such as '*activate*', '*interact*', '*bind*' and etc. For example, we can identify a subject and object term of the verb '*activate*' for the following sentence;

> *... dCREB-A is a member of the bZIP family of transcription factors, shows specific binding to the ( CRE ), and can* **activate** *transcription in cell culture.*

Our system extract a simple assertion as follows from the above sentence, by finding the right subject and object term of the verb *activate*:

```
activate(@SUBJ: dCREB-A
         @OBJ:  transcription in cell culture
```

This is only possible if the noun phrase has been correctly bracketted, and the head word in each noun phrase has been given the correct syntactic tag such as `@SUBJ` or `@OBJ`. The algorithm we have used for finding a subject and object term for each verb is roughly as follows:

1. *Find a target word, which has been derived from the most frequently seen verbs in Medline.*
2. *In the case that it is a verb, judge the voice of the word (e.g., active or passive voice)*
3. *Find a subject or object noun phrase:*
   (a) *Look left for finding a nominal head tagged as a subject (cf. if the verb is in passive voice, it will be an object.).*
      - *Stop finding a subject when '.' or main verb is reached and return nothing.*
      - *When dealing with relative clauses, check the EngCG tag to see the characteristics of the relative pronouns, and search for the previous noun phrases.*

   (b) *Combine words tagged as premodifiers and postmodifiers(except determiner and post-modifying preposition) with nominal head: they will be combined only when EngCG clearly indicates that those modifiers are actually modifying a nominal head. If there are parenthesized words next to premodifier or postmodifier, regard them as a part of a noun phrase.*

   (c) *If the verb is in active voice, look right for finding an object word.*
      - *Stop finding an object when '.' or main verb is reached and return nothing.*

   (d) *If the verb is in passive voice, look right for identifying a subject.*
      i. *While a word is an adverb, look right.*
      ii. *If the word is 'by' , accept the following noun phrase as a subject term. Otherwise stop looking for a subject word.*

*iii. Look right for identifying head noun.*

*(e) Combine words tagged as premodifiers and postmodifiers with nominal head. Do as 3b*

The actual algorithm which has been implemented is slightly more complicated than the above one as we had to consider all the different cases for each verb, due to the different characteristics of it. For example when we deal with the verb '*activate*', we also consider the case of nominal predicates such as '*activation of*', where the following word is usually the target of something being activated. Also, negative expressions such as '*inactivation*' and '*does not activate*' give vital information for interaction between genes. Notice that the word "*activated*" can be both `adjective` and `past verb`, and this can only be disambiguated by observing the surrounding context. Without the reasonably accurate performance of EngCG for morphological tagging, this task could have been quite difficult.

Some of the failure to recognize the correct subject or object terms have arisen from parsing errors. For example, for the following sentence;

> *Taken together, these results indicate staurosporine specifically* **activates** *a JNK isoform, which may contribute to biological activities including neurite outgrowth.*

The subject of the verb 'activates' has been wrongly recognized as 'these results' instead of 'staurosporine'. This occurred because 'staurosporine' has been wrongly tagged as an object, and also because our algorithm sometimes skip certain part of the information (algorithmic error). Here is an another example due to the errors in our algorithm.

> *The cyclin A promoter in a reporter plasmid was* **activated** *by nearly 10-fold in quiescent rat 3Y1 cells by cotransfection with the expression of plasmids encoding ATF2 and Jun family members.*

In the above case, the object term of the verb *activate* is '*the cyclin A promoter in a reporter plasmid*' and the subject term is roughly *cotransfection.* (Remember that '*activated*' in this case is in passive mood). But our algorithm assigned '*quiescent rat 3Y1 cells*' as a subject term. This happens because, when the verb is in passive mood, our algorithm searches for the noun phrase just after the word 'by'. Unfortunately in this case, there are two phrases preceded by the word '*by*': '*by nearly 10-fold in quiescent rat 3Y1 cells*' and '*by cotransfection with the expression of plasmids* '. As our algorithm just looks for the noun phrase appearing closest to the first occurrence of the word '*by*', it skips the word *nearly*, *10-fild*, and *in*, and finally the algorithm selects '*quiescent rat 3Y1 cells*' as a subject terms.

Though there are some failures as shown above, we were able to extract large quantity of these assertions for various verbs with a reasonable precision rate. (For example, for the verb 'activate' alone, we extracted some 652 assertions). These assertions are all the more useful because most of the arguments of these verbs are names of protein or genes. Among the arguments of the verb 'activate' for example, we found out that about 65% of the terms were the names of gene or proteins. We used the system given in [3] from the Human Genome Center of the University of Tokyo, which recognizes material names in genome domain,

# 6 Overall Performance

**EngCG Tagging Performance:** We have tested our documents, not completely, but quite extensively. EngCG seldom discards an appropriate morphological reading: after morphological analysis, 99% of all words retain the appropriate analysis. But only 3% to 7% remain partly ambiguous, depending upon our samples (error rate 0.4%). After syntactic analysis, around 75% to 85% of words become syntactically unambiguous.

Table 1: # of Assertions for each verb.

| Verb | activate | interact | encode | regulate | prevent | contain | inhibit | bind |
|---|---|---|---|---|---|---|---|---|
| Asst. | 652 | 534 | 1070 | 642 | 142 | 1980 | 510 | 1659 |
| Prec. | 72.9 | 83.0 | 67.8 | 80.0 | 80.7 | 84.0 | 83.3 | 72.0 |

** Asst.: # of retrieved assertions
* Prec.: Precision of the retrieved assertions

Though the current NLP technologies did not reach to a point to parse a text with the performance comparable to human ability, considering that our purpose is to recognize particular target information, ignoring irrelevant information, the above performance was good enough to extract some useful information from the corpus.

**Noun Phrase Bracketting:** We calculated the precision of our noun phrase bracketter (see Section 4) based on the following formula:

$$\frac{(\#\ of\ correctly\ bracketted\ noun\ phrases)}{(\#of\ the\ whole\ noun\ phrases\ bracketted\ by\ our\ algorithm)} \quad (1)$$

Traditionally, the task of NP bracketting is restricted to only finding simplex noun phrases. But because of the nature of EngCG tag, we are doing slightly more than that, i.e., we try to include some noun phrases with prepositional phrases whenever possible. Thus, there had been some errors caused by the wrong attachment of prepositional phrase. Still the precision of our noun phrase bracketter reached upto 90%. The *recall* has not been examined extensively due to lack of time. However we think that the *recall* rate is at least above 92%, by observing several sample np-bracketted paragraphs and counting the *recall* rate for them.

**Assertions:** The *precision* of our assertions has been hand-counted by two people for various verbs. We randomly selected several hundred assertions for frequently seen verbs in Table 1, and checked whether a verb has been correctly associated with its subject and object. The *precision* of those assertions are shown in Table 1. Depending upon the verbs, the *precision* ranged from 67.8% to 83.3%. For example, for the verb '*activate*', about 652 assertions have been extracted. Though we were not able to check all these several thousands assertions, we assume that the *precision rate* reached approximately upto 73% from the sample assertions.

While we were examining each case of the failure, we had felt that many errors were simply trivial, because we were in an early stage of developing our system. Various cases of errors have been recorded to get better heuristics and algorithm in the future.

# 7 Future Direction: Recognizing Referential Links

As we are dealing with the unrestricted text written in natural language, the terms for each assertion obtained in the previous Section would naturally contain many kinds of anaphoric terms such as proper names, definites, indefinites, bare nominals, possessed nominals, and pronouns. We can get more useful information if only we can "*co-refer*" all these expressions to the actual protein or gene names. Thus, we decided to do a little more than just listing assertions.

For example, given the sentence;

> *... when cellular sterol levels are low, the SREBPs are released from the endoplasmic reticulum membrane, allowing them to translocate to the nucleus and activate SREBP target genes.*

our algorithm automatically extracts the following assertion;

```
activate(@SUBJ: them,
         @OBJ:  SREBP target genes)
```

However, rather than this, we wanted to get more specific information by specifying what '*them*' actually indicates in the context.

*Anaphora resolution* in NLP is a component technology of an overall discourse understanding system, and it is regarded as one of the most difficult problems. As the design architecture of our system is based on deep level processing by combining genome informatics and NLP techniques together, we decide to implement a preliminary reference resolution module for future use, but at this moment only for resolving impersonal pronouns. (A much more sophisticated reference resolution module will be developed in the future, dealing with definites, indefinites, and other anaphoric terms).

**Algorithm:** Our algorithm is similar to the ones described in [5, 7] except that we are not concerned about all the anaphoric terms. Our algorithm is as follows:

1. *Input: EngCG output. Entities with the following morphological and syntactic features are useful (among various features in EngCG and features in our application)*
   - *determiner type (e.g., DEF, INDEF, PRON)*
   - *grammatical or numerical number (e.g., SG or PL)*
   - *Head String (e.g., Head)*
   - *Modifier Strings*
   - *Sentence and Paragraph Positions*
2. *For each impersonal anaphoric entity in the sentence, Do (from left to right)*
   - *COLLECT antecedent entity within the same paragraph. We set the maximum number of antecedent terms to five.*
   - *FILTER with syntactic and semantic consistency between the anaphoric entity and the potential antecedent entity, based on number consistency, sortal consistency, and modifier consistency. As our concern is basically focused on protein and gene names, we give higher preference to protein and gene names. In identifying protein or gene names, we used the system developed by [3].*
3. *ORDER by dynamic syntactic preference. Ordering at this moment is only by recency and salience though there are many more sophisticated theoretical and practical works of others.*

# 8 Conclusion

We have shown a fully automatic way of extracting the relationships between the proteins and gene products expressed by some frequently seen verbs in the genome domain. Identifying the characteristics of the proteins or genes is the most important task, as proteins and genes play a central role in genome field.

Though the precision rate of our system did not reach a practical level yet, the readers should notice that our method of identifying the interaction between genes has not been applied to a small domain with several simple pattern matching rules; our method could have been applied to a quite large corpus, and this could have been possible as our method is fundamentally based on NLP techniques.

There have been some previous attempts by others to build tools for the automatic classification of sequences in functional classes using the detailed functional annotations provided by human experts or automatic systems [8]. We think that our research will also contribute to this specific application or constructing a medical thesaurus or hierarchies to classify protein names, according to their characteristics.

As we are in an early stage of developing a system, many modifications will follow to our current algorithm and design architecture as a whole. By adopting relatively deeper level analysis approach of the unrestricted raw text when compared to the previous approaches of others in genome informatics, we think we have shown (or alluded) some new possible application areas in genome informatics for us and for others, too.

## Acknowledgements

We would like to thank Nigel Collier for the comment. This research has been partially supported by the project of JSPS (JSPS-RFTF96P00502).

## References

[1] Allen, J., *Natural Language Understanding*, The Benjamin/Cummings Publishing Company, Inc., 2nd edition, 1995.

[2] Church, K., A stochastic parts program and noun phrase parser for unrestricted text, In *Second Conference on Applied Natural Language Processing*, 136–143, Assoc. Comput. Linguistics, Morristown, NJ, USA, 1988.

[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] Hishiki, T., Collier, N., Ohta, T., Ogata, N., Sekimizu, T., Steiner, R., Park, H.S., and Tsujii, J., Developing NLP tools for genome informatics: an information extraction perspective, In *Genome Informatics 1998*, Universal Academy Press, Inc., Tokyo, Japan, 1998.

[5] Kameyama, M., Recognizing referential links: An information extraction perspective, Technical report, AI Center, SRI International, 1997.

[6] Karlsson, F., Constraint grammar as a framework for parsing running text, In *the 13th International Conference on Computational Linguistics, H. Karlgren, editor*, 168–173, 1990.

[7] Lappin, S., Leass, H.J., An algorithm for pronominal anaphora resolution, *Computational Lingiustics*, 20(4):535–561, 1994.

[8] Tamames, J., Ouzounis, C., Casari, G., Sander, C., and Valencia, A., Euclid: automatic classification of proteins in functional classes by their database annotations, *Bioinformatics: Applications Note*, 14(6):542–543, 1998.

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

[10] Voutilainen, A. and Tapanainen, P., Ambiguity resolution in a reductionistic parser, In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, 394–403, 1993.