GeneTUC v2

A Biolinguistic Project, Next Generation

Rune Sætre

IDI, NTNU

June 17, 2002

Abstract

The purpose of GeneTUC v2 is to apply the "The Understanding Computer" (TUC) architecture to biolinguistic texts (about genes and protein interactions). Anders Andenæs started this work in the GeneTUC project two years ago, and he focused on collecting all known genes, proteins and substance names into the system's database, together with common "interaction"-verbs. In this follow-up project, the main goal has been to parse one abstract 100%. That means that GeneTUC v2 is a little more qualitative linguistic oriented than its predecessor, which used a large corpus and measured the success rate of parsed sentences (known as recall rate). The qualitative approach is used in order to discover particular weaknesses in the grammar, and to monitor if the semantic interpretation of any of the other sentences is affected by changes done in the grammar.

By the end of this project three of the original sentences had to be slightly modified, in order to be able to parse the entire abstract. These unsolved grammatical problems are linked to punctuation, the word "*indicating*", and a particular use of the word "*and*".

None of the grammatical changes that were done during the project affected the other sentences in a negative way. That means that all the changes improved the total quality of the grammar.

Acknowledgements

I would like to thank Tore Amble for all his support, and for inventing the term “Biolinguistics”. Astrid Lægreid has helped me understand all the complex names and terms in the biological texts. Per Kristian Lehre and Jörg Cassens have engaged in fruitful and inspiring discussions. And finally, Anders Andenæs have answered all my email questions about the work he did with the GeneTUC system last year.

I also want to thank Tore Bruland for volunteering to proofread this report, even though he had his own report to think about. And, as always, last (but not least!) my mother for thinking of me when I’m working late nights, encouraging me to keep doing my best, and making me good and healthy dinners when I need it the most.

Preface

When I started this project I didn't think it was going to be that hard. After all, Anders Andenæs had already done much of the foundation work, and I thought of this project as a comprehensive GeneTUC Tutorial (which is always a good idea to start with, when you're introduced to a new system).

It turned out that the TUC system was much more complex than I had imagined, and it took quite some time to understand how all the files, predicates and debugging commands are connected, and works together.

Thanks to Tore Amble, who spent many hours walking me through the complex process of debugging and adding code, I slowly started to see the greater picture. We decided early that I should focus on the semantics, while he should fix the grammar definitions as needed. As I started working more independently with the semantics, a new problem showed up: The semantics of the biolinguistic text we got (Appendix A, ICER-manus) were by no means obvious to my un-trained (not biologically educated) eye.

Even though I have taken quite a few "crash courses" in microbiology by now, I still had a hard time making the *correct* semantic network based on the text alone. This shows that one needs much more than just commonsense knowledge in order to create a biolinguistic expert system. Also, there is not yet full consensus in the biological community about what the exact *correct* knowledge is, but important work is being done (E.g. GO and others)!

This report is written in MS Word, using font *Palatino Linotype* 12pt.

Rune Sætre, June 17, 2002

Contents

1 Introduction

PubMed, a service of the (American) National Library of Medicine, provides access to over 11 million MEDLINE citations back to the mid-1960's and additional life science journals. PubMed includes links to many sites providing full text articles and other related resources [PubM]. This enormous resource of answers in plain text form is extremely valuable to biomedical researchers, but at the same time, they find it almost impossible to search for and find exactly the answers they need. It can best be described as the problem of finding a needle in a haystack.

1.1 Different approaches

The current approach to searching the MEDLINE Database (DB) is called Information Retrieval (IR). IR basically means to enter keywords into the search engine, and then a list of *all* articles containing the keywords are returned. In GeneTUC something called Information Extraction (IE) is used. IE, in this case, means that a full parser parses all the sentences in the DB, transforming each sentence into some logical formulae that represents the *meaning* of the original sentence. This transformation is called *text mining*. Then the researchers can state their questions using natural language sentences, and the system should be able to find the right answer to the question, or give a list of only the abstracts/articles that are semantically *relevant* to the question.

The purpose of this project is to find out if this kind of text mining (full parsing) works in the biomedical domain. The purpose is not to meet several end-user demands, or to create a commercial product (yet). It should also be shown that this method works better

than for example the statistical approach used in PubGene [Jen02] or syntactic pattern matching as in Entrez [PubM]. In PubGene a relation will connect two genes if there are *many* examples that they are *usually* mentioned in the same abstract. Entrez is just another example of IR, as described above.

Because of the computational complexity of the full parsing, some pre-processing might be needed before all the text can be parsed in a reasonable amount of time. One example of pre-processing that would be useful in an on-line system is to find a subset of articles that we want to do full parsing on, in a sorted order. In that case, less parsing would be needed before an answer is found. Such a system will be a hybrid between plain text searching and strict grammar parsing.

1.2 The task specification

1.2.1 Natural language processing of gene information

Current knowledge about genes and their interactions exist largely only as free text. Searching and cross-linking such information rely largely on existing indexes created either manually or by syntactic pattern matching. As a first step we want a tool that is able to correctly recognize occurrences of a ***gene*** *in free text, e.g. in an article abstract, and the* ***context*** *in which the gene is mentioned. The project will be in cooperation with Biomedical research at NTNU, and will partly be building on existing prototypes for text mining of biomedical texts (GeneTUC).*

1.2.2 Goals

The main goal is to successfully parse all the sentences in an entire article. The article (see Appendix A) was given to us, by Astrid Lægreid at the Medical Research Center (MTFS) at NTNU. It is a typical example of the type of article that the system will have to deal with. Parsing all the sentences is a difficult problem. In addition to all the "usual" problems with NLP, like synonyms, homonyms, ambiguities etc, biolinguistics are even harder, mainly because of the complex syntax in this kind of articles. Still TUC is relying on full parsing, so this is a necessary first goal.

The next goal is to try to make use of the semantics in the parsed sentences. GeneTUC should be able to answer questions like "what represses ATP-B", if this is stated in the input article. When we can make GeneTUC do this, then the next step is to evaluate if the system is useful to the biomedical society. That means that GeneTUC will have to solve actual problems, given to it by our contacts at MTFS or other biologists. The last step when making a commercial system is to make sure the system is better than other existing approaches. In other words GeneTUC should be compared to other systems that use for example, statistical analyses.

To sum it up, this is the path to follow:

1) Is our approach possible?
2) Is the system useful?
3) Is the system better than other systems?

The main goal of this project is to answer question 1).

1.3 Constraints

One side-goal is to keep the old bus-*grammar* to maintain backward and sideways compatibility with other TUC applications. There is no point in removing working parts of the system, as long as they don't create many extra problems and the system can benefit from improvements done by other TUC-programmers. The bus *semantics* database can be removed, since nobody ever needs to ask GeneTUC about busses or their schedules. Later one should think about modularizing the semantics database, so that at least parts of it could be reused from one project to another. That could save someone a lot of definition work in the future.

The semantic database in TUC is in English. That means that Norwegian sentences have to be translated to English, and then analyzed. This is not a problem in GeneTUC, since virtually every biologist in Norway (and the rest of the world) use English as his or her publishing language.

2 Background

2.1 Previous work

In order to work with GeneTUC one needs to know about both linguistics and human genetics. Chapters 3 and 4 in [And00b] provide an excellent introduction into these two fields, and chapter 5.1 in the same report gives some good historical notes on the GeneTUC system.

2.2 Other work

There are about 20 groups in the world trying to extract protein-to-protein interactions from abstracts using NLP (according to linguistic Prof. Franz Guenthner, University of Munich). The two most relevant works are "[Yak01]: Event extraction from biomedical papers using a full parser" and "[Par01]: Using Combinatory Categorial Grammar to Extract Biomedical Information". [Yak01] uses a full parser, just like GeneTUC does, and [Par01] uses a Categorial Grammar, also just like TUC. It is very encouraging to us that [Par01] mention full parsing as probable *future work*, in order to increase the precision-rate (at the cost of lower recall-rate) in certain precision sensitive applications.

2.3 Contacts

A lot of people have come up with ideas, help and inspiration the last year. Here a list of people that should be called upon as the project proceeds later.

2.3.1 NLP

Tore Amble has been the main NLP resource in this project. Another student, Tore Bruland, has also been working on TUC this year, and he has also been giving useful input.

In March the "Human Language Technology" Conference were held in San Diego [HLT02], and quite a few of the participants there were working with *Biolinguistics*. The main focus was on protein name discovery, and protein-interactions (verbs) mined from biomedical abstracts. The most interesting of the systems presented at this conference were by Kristofer Franzén and his fellows from the Swedish Institute of Computer Science (SICS). They are doing protein name discovery, and they are using Prolog, like us.

2.3.2 Biological

Astrid Lægreid at MTFS has been the main source of biomedical input. Jan Komorowski (IDI) and Finn Drabløs (SINTE Unimed) have been helpful and accessible through a class on "Data Mining and Knowledge Discovery" that was taught last semester. Komorowski is now starting a Bioinformatics center at the University in Uppsala, Sweden.

2.4 Sources

TUC's semantic network needs to be upgraded with more medical terms, and there are several places to look for these new words. One possible source is WordNet, which defines word contexts and semantically connected clusters of lexical words. Another source is Gene Ontology, which defines identified molecular functions, biological processes and cellular components in context with other functions, processes or components. Part of this project will be to incorporate words from these sources into TUC. Amble and Andenæs have already done much of this earlier. For example all adverbs and adjectives were imported into TUC from WordNet a few years ago. The verbs were not imported in this way, since it is believed that the group of relevant verbs is small enough to allow for manual insertion, which gives better precision than automatic definitions.

Some of the new words may be defined only in the abstract currently being parsed. These words must then be added manually to the TUC system. Our contacts at the biomedical center will help us get the semantics of these words right. Other terms may be general and well-established definitions that one might hope to find by searching Gene Ontology or some other online biology DB.

2.4.1 WordNet®

WordNet is an online lexical reference system whose design is inspired by current psycholinguistic theories of human lexical memory. English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept. Different relations link the synonym sets together, see [WNet].

It is possible to do online searching to find the synonym sets of specific words from the WordNet homepage. Just go to this Internet address: http://www.cogsci.princeton.edu/cgi-bin/webwn1.7.1

2.4.2 Gene Ontology™ (GO)

The goal of the Gene Ontology Consortium is to produce a dynamic controlled vocabulary that can be applied to all organisms even as knowledge of gene and protein roles in cells is accumulating and changing [Gene].

They are partly succeeding in doing this, and the GO have already been used in several projects by the DIS group at IDI, NTNU. Some critique have been made though, that the old data in the GO is never updated, so every time someone puts a "false" annotation in the database, it will be there forever, even if someone later discovers that the assumptions were not true.

The GO can be searched online by using one of many browsers, for example the QuickGO: http://www2.ebi.ac.uk/ego/QuickGO. In order to find out what one of the terms used in the abstract means, a search with the keyword "PKA" was done. The result told us that it means protein kinase (as expected), and in addition explained that it is cAMP-dependent. Other searches for acronyms were less fruitful. An example is CCK, which were not found. But searching for the full name "cholecystokinin" gave the expected information. This means that CCK is not defined as a synonym in the same way that PKA was, even though it should have been. This goes to show that the GO is by no means complete.

2.4.3 PubMed MEDLINE

The best source of abstracts is the MEDLINE Database, which is maintained by the American association "National Center for Biotechnology Information" (NCBI). NCBI was established in 1988 as a national (American) resource for molecular biology information, but their resources are now widely used throughout the world. NCBI creates public databases, conducts research in computational biology, develops software tools for analyzing genome data, and disseminates biomedical information - all for the better understanding of molecular processes affecting human health and disease [NCBI].

2.5 The abstract

The main goal of this project was to do a complete parsing of one given abstract. The abstract that was used will be referred to as the "ICER-manus", and it can be found in the Appendix A.

3 GeneTUC

GeneTUC is a large system. Only the grammar alone covers more than 30.000 lines of Prolog code. This section is meant to give newcomers a quick tutorial on the system: How to install, use and modify it.

3.1 Source code

The source code is stored on the "*~busstuc*" user on IDI's computers. Tore Amble is in charge of distributing the password to worthy programmers. The TUC part of the system is no longer to be freely distributed, because it has been sold to Team Trafikk, by the company LingIT.

3.1.1 Unzipping and unpacking the files

The file *"GENETUC2.tar.gz"* contains all the files needed, and it can be unzipped and extracted by typing:

1) *gunzip GENETUC2.tar.gz <enter>*
2) *tar –xvf GENETUC2.tar <enter>*

This will place all the extracted files in a folder "*GENETUC2*".

3.1.2 Compiling

To create a fresh executable version of the program the following commands must be executed (as *user* busstuc @ vier.idi.ntnu.no). Note the trailing periods in statements 3-5:

1) cd GENETUC2 <Enter>
2) sicstus <Enter>
3) [gtbase]. <Enter>
4) [genetuc]. <Enter>

5) save_program (genetuc). <Enter>

Next time, substituting lines 2-5 with just "genetuc <Enter>" will start this compiled and saved program.

3.2 Implementation

As stated earlier, GeneTUC is implemented in Prolog. The system has mainly two parts: The general-purpose grammar, and the domain specific semantic network.

3.2.1 Grammar

The grammar is meant to be the same for all applications of TUC, e.g. GeneTUC, BusTUC, LexTUC and so on. The main reason for doing this is that the grammar is not yet finished, and it is being updated quite often. As long as we manage to use the same grammar in all TUC projects, it's an easy task to "upgrade" the system whenever a new grammar is released/available. Sometimes one change in the grammar leads to unforeseen problems in other parts of the system, but that is something we have to deal with, until we get to the point where the grammar is complete and stable. Until then extra caution and testing is needed to make sure a change is not "a step backwards".

So far, it has been possible to keep the same grammar for all application, with a little help of "switches", or flags, that can be turned on or off. The most important one is called "*busflag*" and is turned off in all other applications than BusTUC. This causes the grammar rules that are very specific to bus questions to stay inactive in the other TUC applications.

3.2.2 Semantic network

The semantic network and the rest of the content in the files *semantic.pl* and *facts*.pl contains "all the knowledge" in the system, and can be thought of as the Knowledge System's (KS's) database. The KS's concepts and relations are defined in *semantic.pl*, while the leaf nodes of the semantic network are defined in *facts.pl*. Updating the semantic network is supposed to be a simple task, and Tore

Bruland has recently made a Graphical User Interface (GUI) that can generate the *semantic.pl* file automatically, to make life even simpler.

The main biological challenge in GeneTUC is that it's not always obvious how the different concepts relate to each other. Therefore it's important with good and frequent communication with knowledgeable biologists, in order to get the semantic network right. This communication should be done in some *neutral* language that is easily understood and agreed upon by both biologists and computer scientists. One example of such a language is semantic network diagrams.

3.3 Changes made to the system

TUC is slowly turning into a big multi-developer project, where different programmers work both in parallel, on the same files or on different files, and serially, picking up where someone else left off. Because of this, it's very important to keep track of all the changes that are made, since some of them might need to be propagated to other TUC applications. It's also important because every time a change is made it's a hazard to other previously working parts of the system, and in case some bug is discovered after a grammar change we want it to be easy to backtrack (rollback) the changes.

In Appendix B (Changes in the code) a list of all the recent changes can be seen. Note that every change is *stamped* with the initials of the programmer, and the date that the modification was done.

4 Biolinguistics

Biolinguistics is the most concise way to describe the GeneTUC project, so far. As the name suggests, it involves both NLP and biomedical texts. Below is a summary of the problems that were encountered because of the complex grammar/syntax of the medical language.

4.1 Greek letters

In order for GeneTUC to be able to "read" the ICER-manus, the document had to be converted from Word to Text format. The most obvious choice was to use the "save as txt"-function in Word, but then it was discovered that all the Greek letters were transformed into question marks.

In biolinguistics there are many Greek letters, mainly because they are used in protein naming, among other things. Fortunately there were no Greek letters in the abstract-part of the article, so we didn't have to deal with the problem in this project. In the future, though, it will be necessary to find a smart solution to this problem. One suggestion is as follows:

1) Save the text in some format that keeps the language codes for the Greek letters (e.g. RTF).
2) Use some kind of preprocessing language (e.g. Perl) to substitute α with a, βwith b, γ with g, and so on.
3) Save the preprocessed text as plain text, and there should be no more fake question marks present in the text.

4.2 Punctuation (in the middle of a word)

One particular problem that was encountered in the abstract was the use of a period in the middle of a word (e.g. the antagonist L740.093). This is a tough problem for TUC, because the only legal use of period is as an End-of-Sentence (EoS) symbol. The only way this could be solved was by rewriting the sentence, because it's important to keep period as a reserved symbol. Without having *period* as the EoS-marker, sentence segmentation would be an extremely complex task to solve.

Another problem with punctuation was encountered in the "Gq/G11"-construct. Usually TUC ignores all other punctuation than periods, but since "/" is used as a delimiter in the TQL-language, they make it into the inner loops in TUC's parsing algorithms, and cause havoc there. Again, rewriting the sentence without using "/" was the only simple solution. In this case the "/" was just substituted by a hyphen, which in turn is just ignored (and treated like a space) in the actual parsing algorithm.

4.3 Gene-to-protein correspondence

In many cases there is a one-to-one correspondence between gene and protein, but of course there are exceptions. Because of splice-variants, one gene can give rise to several similar proteins with potentially very different functions.

Often the same name is used both for the gene and the most common variant of the protein it describes. Then, only through analyzing the context of the name, can we find out if it refers to a gene or a protein.

5 Results

Some of the sentences in the original ICER manus had to be changed because of the punctuation rules in TUC among other things. Much effort were made to keep the abstract as unchanged as possible, but to meet the deadline, and to work around some of the inherent problems in TUC, a few of the "hard" problems like "sentence, *indicating* something" and "*L740.093*" were rewritten. These problems should be handled properly by the TUC grammar in a later version.

5.1 Changes in TUC

The most important result of this project is of course the modified version of the program GeneTUC. Only through steady refining can this system hope to finally one day be really useful to the biologists.

5.1.1 Grammar

A lot of changes were made in the grammar during this project. Tore Amble did all the programming of grammar definitions, after I had discovered the specific problems. Most of the grammar changes were done in the files *fernando.pl* and *gram_e.pl*. Look in Appendix B (Changes in the code) for a list of changes.

The grammar is better now than it was before the project. The reason is that a few standard, but hitherto, unused grammar constructs were "discovered" in the abstract, and then added to the TUC grammar. This was done in a way that did not "destroy" any of the constructs that were already working.

5.1.2 Semantics

Many changes were made in the semantic part of the system; see Appendix B (Changes in the code) for a complete list of them. In addition to the problems that were solved, a few other important problems have also been identified, but not yet fixed. Some of the problem sentences that were identified, but not yet fixed, are listed here:

1) Gastrin- and cck-response.
2) Sentence, indicating something
3) Gq/G11 –protein coupled receptor (G-protein is a signaling protein)

They are all caused by punctuation problems, or at least they would be easier to solve if TUC had some proper treatment of punctuation.

The problem with 1) is the special use of a *hyphen* ("-") between "gastrin" and "and". This means that the word that ends the last complex NP (e.g. -"response") is supposed to be added to the first NP, too, making it "gastrin-response".

TUC is not "trained" to handle this kind of grammatical constructs yet, and it's not trivial to implement it either. One has to be very careful not to "break" any other grammar rules, when one is trying to "fix" this specific problem.

Another problem with giving TUC to many choices about how to parse an NP high up in the parse tree is that it leads to a lot of re-parsing of several sentence fragments further down in the tree. This might potentially slow down the parsing a lot.

The problem with sentence type 2) is again based on the fact that TUC ignores all punctuation except periods. What should be extracted from this sentence is an "indicating" relation from the *main event* in the left sentence to the *main event/fact* in the right sentence. So far, TUC have no mechanism for doing this, but this should be possible to implement. It should be easy to identify and parse this construct in a text, even without using the *comma* (",") to help recognizing this special use of indicating. Since the use of the word "indicating" almost always indicates that a comma is present. We

can usually just assume that there is a *comma* to the left of the word *indicating*, meaning that it is the entire left sentence that is indicating something.

As always in NLP there are some exceptions. An obvious one here is the following sentence: "The man was *indicating* that something was wrong."

There are two problems with fragment 3). The first one is again connected to punctuation. The *slash* ("/") is not ignored like the other types of punctuation. Like mentioned in chapter 4.2, s*lash* is used as a separator in the TQL syntax. This means that the sentence has to be rewritten, since *slash* is a reserved symbol. In this project a *hyphen* was used, and it is just ignored when TUC parses the sentence. That in turn means that a *compword* (in the file *dict_e*.pl) must be defined, to cluster "Gq" together with "G11" as one word.

After the problem with the *slash* was solved, a new one emerged. "G-protein coupled receptors" are actually a well know class of proteins (they even have their own Internet domain, see [GPCR]). The correct semantic interpretation of 3) should have been

- gq_g11 isa gpcr

Or possibly

- sk(1) isa pcr
- adj/gq_g11/sk(1)/real

This is currently not possible to do in TUC, since there is no way to add a *name* ("Gq-G11") to this already complex NP ("protein coupled receptor"). As before, it is not impossible to implement this in the grammar, but it could cause a computational explosion if too many ways to parse a NP is added (high/early) in the parse tree. That means that this is also something that will be solved in a later version of TUC.

5.2 "Changes" in the programmer

A less obvious result of the GeneTUC project is the fact that I (the programmer) know a lot more about TUC now than I did before this project. This is important since the plan is to pursue a PhD degree in biolinguistics. So even if (or maybe just because) the project resulted in more questions than answer, this is a good starting point for further investigation and programming. The next step coming up is to find out how much more effort is needed to parse the second

abstract we got from Astrid Lægreid. See the next section for more details.

5.3 Translating and testing

After a successful parsing of the (modified) ICER-manus had been achieved, we got some more abstracts for testing purposes. The same authors wrote one of these abstracts, and all the abstracts were about the same biological phenomenon as the first one.

The other abstracts should be used as control texts, to see how useful our work with the first training text has been, in terms of improving the performance on *unseen* data. A very brief test was done at the end of the project, trying to parse 5 new abstracts without any preparation at all. Even though this is an extremely optimistic approach, it was somewhat disappointing that none of the sentences were parsed correctly. Almost all of the sentences failed because they introduced new words that were currently unknown to GeneTUC, but that is easy to fix by just adding one line per word into the *semantic* file.

To see if these unknown words were the real major problem, a second brief test was done with the *unknownflag* set to *true*. This means that the parser will not crash because of undefined words.

Even after setting the *unknownflag,* none of the sentences in the new abstracts would parse correctly. A little close inspection discovered that many of the sentences tended to fail largely because of the same reasons that made us rewrite the ICER-manus before: Problems with the use of punctuation ({}. /) and the difficult way of using the word "and". In addition there were a number of new grammatically constructs, that need to be identified and inserted into the general TUC grammar (as soon as possible).

Also, a problem similar to the "Greek letters" problem surfaced. This time it was because of *curly braces* ("{" and "}"). It turns out that these *braces* are translated into "æ" and "å" when the word-document is stored as plain text. This problem affected several of the new sentences, and the solution is the same as for the "Greek letters" case: Using some sort of text preprocessing and a proper conversion of the special characters should solve this problem.

5.4 Question answering

There has not been much focus on question answering in this project, but in its present state GeneTUC can still answer some very simple questions. To prove this a couple of examples are shown:

1) E: what blocks gastrin.

.....
[Which (A)::(gastrin isa substance, (block)/A/gastrin/B, event/real/B)]
.....
L740093.

2) E: what is cck.

.....
[which (cck):: cck isa gene]
.....
cck

Example 2) is a little flawed, since the correct output should of course be “gene”, and not “cck”, but this will be fixed in the next version.

6 Discussion

6.1 Semantics

The changes and add-ons to the *semantic* part of the system should have been easy to do, but this was not the case after all. The main reason for this is that the biologists themselves haven't done enough standardization work yet, especially when it comes to gene-ontologies and protein interaction naming.

This means that every time something is added to GeneTUC's semantic network, there's a chance that it will have to be changed later, because our "view of the world" is still changing. This is the same problem that were describe with the GO. Some "guesses" turn out to be erroneous, but nobody ever bothers to correct the entries.

In other cases you find that you get different answers about how to build the semantic network, depending on which sources you go to. A good example of this is the fact that in the old GeneTUC system, *process* is modeled as *ako activity*. When this was discussed with Astrid Lægreid, she spontaneously said that *activity ako process*, and not the other way around. Her statement was based on the fact that *biological process* is actually one of the general top three nodes in the Gene Ontology, while *activity* is describing smaller and simpler (sub) processes.

This is an old problem in NLP; you go from a language that is highly ambiguous, to the language of logic, with only one acceptable interpretation. In order to be able to do this, there must be some sort of consensus about what the actual meaning of the NL constructs are. This consensus is not yet established in the biolinguistic domain, but work is underway. Efforts like the Gene Ontology seek to identify and classify all processes, functions and locations of genes and proteins in the cells.

6.2 Syntax

The same thing can be said in many ways. TUC requires questions to be stated with the same form/syntax as the corresponding input fact was stated with, in order to be able to find the answer. This is an argument for using entire articles, and not just abstracts, since the same important facts will usually be stated many times, in different forms, throughout the article. This should make it easier to ask question, since there will usually be a couple of accepted ways to state the question.

6.3 Punctuation

It might be worth taking at least some punctuation into the TUC language. Especially *commas* would have the potential to solve a lot of ambiguity problems.

A good argument against starting to deal with punctuation, though, is that it will also be a new source of problems, not only solutions. The way people use punctuation is as ambiguous as the way they use the language in general. Still, it would probably be well worth the time looking into what positive solutions punctuation (or just comma) treatment would bring with it.

7 Conclusions

7.1 Possibility analysis

In this project we (almost without changing it) managed to parse one entire biolinguistic abstract using the TUC architecture. So it can be said that the TUC approach to biolinguistic is possible, but it is very time-consuming. If it turns out that the same amount of work is needed to parse the next (similar) abstract by Lægreid and her colleagues, one has to ask if this approach is efficient enough.

In this project the entire abstract had to be manually digested and understood by the programmer, in order to be able to make the right entries into the semantic network. If this much work is needed for every single future abstract (meaning that there are always new terms emerging), there is no way the programmers are going to keep up with the publishers.

7.2 Ontology tools

What is needed is a more automatic way of building the semantic network. One approach is to rely on the GO, WordNet and other ontologies. That way more people will be involved in keeping the semantic network updated, and TUC would only need to download fresh semantics every once in a while.

A tool that is supposed to support simple interchange of ontologies is the Resource Description Framework (RDF). The RDF specifications provide a lightweight ontology system to support the exchange of knowledge on the Web [RDF]. It is also a part of a bigger project, namely the "Semantic Web". As more people start using this standard, one can imagine a scenario where it would be possible to just "download" the appropriate semantic network for a given

domain from the Web. This would save the programmer a lot of time when doing the "common sense" coding for any new project.

7.3 Usefulness analysis

Franz Günther from Hamburg University said that around 20 groups in the world are doing this kind of protein interaction NLP work, so obviously someone believes that the systems are going to be useful. One of the groups is located at Korea Advanced Institute of Science and Technology, and they are working with Categorial Grammars, like the one TUC is using [Par01]. The main difference from TUC is that they use partial parsing.

First of all they filter the sentences by preprocessing, keeping only sentences that contain one of the ten or so predefined verbs, which denote *interesting* relations. Then they have a simple *Regular Grammar* that finds all possible *subject* and *object* Noun Phrases (NPs) to the left and right of the current verb (the Key Word, KW). During the search for NPs, all adverbs and adverbial phrases are skipped, except negative ones, like *never*, which are needed to get the correct semantic interpretation of the relation. Finally they give all candidate NPs to their Categorial Grammar parser, and lets it pick the longest *semantically meaningful* NPs that can be *subject* and *object* for the current KW.

With this approach, [Par01] gets a much better recall than GeneTUC (48% against 8%), but one thing that they will never be able to do with this approach is to capture modifiers like when authors say: "We *think* this indicates that…" or "*Previously* it was *assumed* that…" and so on. This can only be done using a full parser.

At the same time, [Yak01] prove that they can get 23% recall using full parsing and a simple preprocessor. If they accept ambiguous sentences, the recall goes to 47%. Their next project now is to design and implement a postprocessor. When this postprocessor is fed with all the results from the parser, including *partial* results, they expect to extract argument structures with a recall of 74%.

Another important result from [Yak01] is that full parsing can be done in a reasonable amount of time, even on just a single PC in the

top range. TUC solves the processing complexity problem using *cuts* that sometimes cause problems, especially with *garden path sentences*. It is reassuring to think that these problems can also be solved in depth without cuts, just using one of the powerful PCs on the market.

Based on all this, it seems that at least some of the biolinguistic research in the rest of the world is converging at full parsing and Categorial Grammars. So even if GeneTUC is still struggling with low recall numbers and some serious grammatical problems, it is to early to give up on this approach. It is probably about time we started looking at pre- and post- processors to help TUC take full advantage of the parts of the grammar that already well developed.

8 Future work

8.1 Pre-parsing

8.1.1 Use IR to find only specific relations (verbs)

Information Retrieval (IR) is just pure text searching (aka pattern matching). When I spoke to Franz Günther from Hamburg University he gave me a new idea to a way of pre-parsing the abstracts. His claim is that it's better to focus on one specific relation at a time, instead of trying to completely analyze every sentence in every abstract. The reason for this is that the language is ever expanding, and there are hopelessly many ways of (miss-) writing anything. Therefore it seems we have to build something that is robust enough to cover "everything", which is more or less impossible. This is basically the same argument that motivated [Par01] in their focus on certain verbs.

The situation is not so hopeless if we focus on only one specific relation at a time; then there is probably a finite amount of possible word configurations and misspellings to consider, and as soon as all of them are programmed, the system will be robust, considering just that specific relation. Another advantage of this is that the biologists will then immediately have some useful sub-program to start their exploring with. And this, in turn, will lead to useful feedback to the programmers.

8.1.2 Use Perl to solve the punctuation problems

Many of the punctuation problems can be solved with a good text-manipulation system like Perl. That way the punctuation can already be standardized by the time TUC starts chewing on the

sentences, and TUC would not fail because of ungrammatical use of *periods* or *slashes*. This preprocessing approach is possible since there is a finite amount of words with periods in them, or at least a finite amount of templates for these words (E.g. The *LetterNumber.Number* template for all terms like L740.093). Preprocessing is also a good thing to do, since it separates the strange biolinguistic punctuation problem from the general TUC grammar/parsing problems. Also worth noting is that Perl has much more powerful syntactic text manipulation functions than Prolog.

8.2 Problematic grammar constructs

A few of the difficult sentences that were abandoned in this project should definitely be made working in later versions. The most obvious example is the following template: "*sentence*, indicating *something*". This is a very standard and legal way, in terms of English grammar, to express that one complex event points to another one.

The reason that this sentence is not already working in TUC is that it's never been encountered before, in any of the simpler TUC applications. It is also a little trickier to solve than some of the other general problems that were identified in this project, so it was put on the "waiting list" (future work).

The biggest challenge in GeneTUC will probably be to build a grammar that is good enough to deal with all the complex language constructs that the biologists uses in their reports.

In this project it's been shown that it is possible to do full parsing on a relevant text abstract, even though it took quite an effort to do it. The next step is to find out if the effort to parse a similar abstract will be less than the work already done with the first one. This is what has happened to BusTUC, as the same types of questions are repeated over and over again in the long run.

8.3 Partial parsing

Other research communities seem to have great success with partial parsing of biolinguistic texts. The average biomedical sentence is very long, and very often it contains more than one fact. Common examples of this are the use of "and" to join two sentences

into one, or apposition like in this construct: "L740093, *a specific CCK-B receptor antagonist* blocks …". Even if the rest of this sentence cannot be parsed, some useful facts are already discovered. But the present version of TUC cannot handle this scenario. When the parser fails, nothing is returned.

A better solution would be to return as many facts/events as possible, together with a warning that this is just part of the facts represented by the sentence. The advantage of this is that it would almost double the recall rate, since most sentences represent more than one extractable *event*.

References

[And00] Anders Andenæs. GeneTUC - /j&'-ne-tük/. Technical report, Department of Computer and Information Science, Norwegian University of Science and Technology, 2000.

[And00b] Anders Andenæs. GeneTUC – An NLP System for Biomedical Texts. Master's thesis, Department of Computer and Information Science, Norwegian University of Science and Technology, 2000.

[Bra97] Jon S. Bratseth. Bustuc – a natural language bus traffic information system. Master's thesis, Norwegian University of Science and Technology, 1997.

[Gene] Gene Ontology Consortium. Official webpage: http://www.geneontology.org/

[GPCR] GPCRDB: Information system for G protein-coupled receptors (GPCRs). Official Home Page: http://www.gpcr.org/

[HLT02] Human Language Technology Conference, San Diego, March, 2002. Webpage: http://www.hlt2002.org/

[Jen01] Jenssen T-K, Lægreid A, Komorowski J, Hovig E, A literature network of human genes for high-throughput nalysis of gene expression, Nature Genetics, 28:21-8, May 2001.

[NCBI] National Center for Biotechnology Information. Webpage: http://www.ncbi.nlm.nih.gov/

[Par01] Jong C. Park. Using Combinatory Categorial Grammar to Extract Biomedical Information, Intelligent Systems in Biology, 1094-7167/01 IEEE Intelligent Systems, 2001.

[RDF] Resourse Descriptive Framework, by the World Wide Web Consortium, W3C. Official Webpage: http://www.w3.org/RDF/

[PubM] PubMed, a service of the National Library of Medicine, provides access to over 11 million MEDLINE citations. Webpage: www.ncbi.nlm.nih.gov/PubMed/

[WNet] The WordNet Project. Official webpage: http://www.cogsci.princeton.edu/~wn/

[Yak01] Yakushiji et al., "Event Extraction from Biomedical Papers Using a Full Parser," Proc. Pacific Symp. Biocomputing 2001, World Scientific Publishing, River Edge, N.J., 2001, pp. 408-419.

A ICER-manus

A.1 Original version

The CREM gene encodes both activators and repressors of cAMP-induced transcription. ICER (Inducible cAMP Early Repressor) isoforms are generated upon activation of an alternative, intronic promoter within the CREM gene. ICER is proposed to down-regulate both it's own expression and the expression of other genes that contain cAMP responsive elements (CREs) such as a number of growth factors. Thus, ICER has been postulated to play a role in proliferation and differentiation. Here we show that ICER gene expression is induced by gastrin, cholecystokinin (CCK) and epidermal growth factor (EGF) in AR42J cells. The time course of gastrin- and CCK-mediated ICER induction is rapid and transient, similar to forskolin- and PMA- induced ICER expression. The specific CCK-B receptor antagonist L740.093 blocks the gastrin- but not the CCK -response, indicating that both the CCK-B and the CCK-A receptor can mediate ICER gene activation. Noteworthy, CREB is constitutively phosphorylated at Ser 133 in AR42J cells, and ICER induction proceeds in the absence of increased CREB Ser 133 -P. Gastrin-mediated ICER induction was not reduced in the presence of the PKA inhibitor H-89, indicating a PKA independent mechanism. This is the first report on ICER inducibility via Gq/G11 –protein coupled receptors.

A.2 Modified version[1]

(1) The CREM gene encodes both activators and repressors of cAMP-induced transcription.
(2) ICER (Inducible cAMP Early Repressor) isoforms are generated upon activation of an

[1] The changes are marked with *italics* and a leading * before the sentence.

alternative, intronic promoter within the CREM gene.

(3) ICER is proposed to down-regulate both it's own expression and the expression of other genes that contain cAMP responsive elements (CREs) such as a number of growth factors.

(4) Thus, ICER has been postulated to play a role in proliferation and differentiation.

(5) Here we show that ICER gene expression is induced by gastrin, cholecystokinin (CCK) and epidermal growth factor (EGF) in AR42J cells.

(6) The time course of gastrin- and CCK-mediated ICER induction is rapid and transient, similar to forskolin- and PMA- induced ICER expression.

(7) * The specific CCK-B receptor antagonist *L740093* blocks the gastrin- but not the CCK – response*. This indicates* that both the CCK-B *receptor* and the CCK-A receptor can mediate ICER gene activation. ("gastrin-" should be *"gastrin response", semantic problem?)*

(8) Noteworthy, CREB is constitutively phosphorylated at Ser 133 in AR42J cells, and ICER induction proceeds in the absence of increased CREB Ser 133 -P.

(9) * Gastrin-mediated ICER induction was not reduced in the presence of the PKA inhibitor H-89*. This indicates* a PKA independent mechanism.

(10) * This is the first report on ICER inducibility via *Gq-G11* protein coupled receptors.

B Changes in the code

Here is "a map" to the most needed files in this GeneTUC project, and also a list of changes that were done in the different files.

B.1 Important files

Below is a brief explanation of the content in the most important files, regarding this project:

database/facts.pl
- Contains the leaf nodes in the semantic network, written on the form "constant *isa* concept."

database/semantics.pl
- Contains the semantic network concepts, written on the form "concept *ako* concept."
- Also contains the templates for how to combine concepts with *verbs, adverbs & adjectives*.

tuc/dict_e.pl
- Contains rules for how to read complex words that are made up by more than one word.

tuc/fernando.pl
- Changed by Tore Amble a few times.
- Deals with complex NPs like "the antagonist l740093 exists."
- Deals with sentences like "GenaX, GeneY and GeneZ exists."

tuc/gram_e.pl
- Contains all the grammar rules in *consensical* form.
- To many things were fixed in this file to include all of them in the list below.

B.2 database/facts.pl:

```
%% REVISED  RS-020614
serin isa amino_acid.   %% RS-020529
l740093  isa antagonist.          %% RS-020515
ar42j   isa cell.          %% RS-020529
```

```
pka isa kinase.          %% RS-020529, Protein Kinase A??
h_89 isa pka_inhibitor.          %% RS-020531
%% To be moved to protein.pl?!          RS-020523
crem  isa protein.       %% RS-020515
g11   isa protein.       %% RS-020529
gq    isa protein.       %% RS-020529
gq_g11 isa protein.      %% RS-020603
gq_g11_pcr isa gpcr.     %% RS-020603 G-Protein coupled receptor
%% To be moved to substance.pl?!          RS-020523
epidermal_growth_factor  isa substance. %% RS-020523
```

B.3 database/semantic.pl:

```
%% REVISED RS-020614
activator       ako protein.        %% RS-011018
cckbr_antagonist        ako antagonist.     %% RS-020527
%% expression           ako process.        %% RS-020513 %% iflg Astrid Lægreid
pka_inhibitorako inhibitor.         %% RS-020531
process         ako activity.       %% AAn-000404 %% RS-020510 opposite!, ref Lægreid
%% protein      ako agent.          %% AAn-000320 %% RS-020528 GIVEN by marker ?!
protein         ako substance.      %% RS-020528 Biologically correct, ref Lægreid
repressor       ako protein.        %% RS-011018, Eg. substance; men hva med attr
gpcrako receptor.       %%RS-020603
adj_templ(cck_mediated,activity).           %% RS-020527, løste et annet problem ;-O
adj_templ(forskolin_pma_induced,activity).          %% RS-020525
adj_templ(gastrin_cck_mediated,activity).           %% RS-020525
adj_templ(gastrin_mediated,activity).       %% RS-020515
adj_templ(increased,protein).       %% RS-020527
adj_templ(pka_independent,abstract).        %% RS-020603, eg.mechanism
%%adj_templ(pka,inhibitor).      %% RS-020531, Comp_word!
adj_templ(pma_induced,activity).            %% RS-020527
adj_templ(gq_g11_protein_coupled,protein).          %% RS-020603
%% Hva med generelt adjname_templ(Gene, agonist).         %%RS-020527
%% adjname_templ(cckbr,antagonist).     %% RS-020527,->compword!
adjname_templ(creb,protein).        %% RS-020525 What's wrong???
%%adjname_templ(gq_g11,receptor).        %% RS-020603
adjname_templ(icer,activity).       %% RS-020524 Includes activation
%%n_compl(by,activity,gene).        %% RS-020524 ikke n_compl (passiv!)
iv_templ(phosphorylate,protein).            %% RS-020525 (transitiv!, by agent)
iv_templ(proceed,activity).         %% RS-020525
tv_templ(induce,gene,activity). %% RS-020513
tv_templ(mediate,agent,thing). %% AAn-000914 %%RS-020524 (gene/prot etc)
tv_templ(normalise,agent,thing).            %% AAn-000921, RS-020527 modified
tv_templ(phosphorylate,agent,protein).      %% RS-020525
tv_templ(play,agent,role).          %% TA-011203, RS-020514 modified
n_compl(of,course,activity).        %% RS-020524
adv_templ(constitutively,phosphorylated).           %% RS-020528
v_compl(phosphorylate,agent,at,amino_acid).         %% RS-020525
```

B.4 tuc/dict_e.pl

```
synword(constitutively, constantly).       %% RS-020529 Riktig, Astrid?
synword(ser, serin).    %% RS-020529, amino_acid
%%noisew(constitutively).       %% TA-011025, rem RS-020528 kontinuerlig
synword(cholecystokinin, cck). %% RS-020515 (juks...?)
synword(similar, equal).          %% RS-020525
compword(cck,[a,receptor], cckar).          %% RS-020527
compword(cck,[b,receptor,antagonist], cckbr_antagonist).  %% RS-020527 Juks!
compword(cck,[b,receptor], cckbr).          %% RS-020525
compword(cck,[mediated], cck_mediated).          %% RS-020515 "garden path"
compword(creb,[ser,133,p], creb).          %% RS-020527 Riktig??
compword(epidermal,[growth,factor], epidermal_growth_factor).  %% RS-020522
compword(forskolin,[and,pma,induced], forskolin_pma_induced). %% RS-020525
compword(gastrin,[mediated], gastrin_mediated).          %% RS-020515 "garden path"
compword(gastrin,[and,cck,mediated], gastrin_cck_mediated). %% RS-020525 Juks!
compword(gq,[g11,protein,coupled,receptor], gq_g11_pcr).          %% RS-020603 Juks
compword(gq,[g11,protein,coupled,receptors], gq_g11_pcr).          %% RS-020603 Juks!
compword(h,[89], h_89).          %% RS-020529
compword(icer,[gene], icer).     %% RS-020514
%%compword(icer,[isoforms], isoform). %% RS-020521 rem
compword(pka,[independent], pka_independent).          %% RS-020603
compword(pka,[inhibitor], pka_inhibitor).          %% RS-020531
compword(pma,[induced], pma_induced).          %% RS-020527
compword(protein,[coupled], protein_coupled).   %% RS-020603
compword(ser,[133], ser).          %% RS-020525 Riktig?? Semantisk?
compword(Word,List,WholeWord) :-
      user:gene_cmpl(Word,List,WholeWord).     %% RS-020525, Slow?? Ok!
```

B.5 tuc/fernando.pl

```
%% REVISED TA-020527
constrain0(_:thing,_):-!. %% TA-020103 "det"  koster
%% NEW PREDICATE   %% TA-020419
bealign(X,Y,S, P, Q):- %% Explicitly  X is Noun // not Noun X
      \+ value(textflag,true),
      !,
      align(X,Y,S, P, Q).
bealign(X:MT,Y:WT,S,P,P and be2/X/Y/S):- %% TA-020312
      bottom(MT,Program),
      bottom(WT,Person),
      alignable0(Program,Person),
      !.
bealign(X,Y,S, P, Q):-
      align(X,Y,S, P, Q).
align(X:MT,Y:WT,S,P,P and be2/X/Y/S):- %% TA-020312
      bottom(MT,Program),
      bottom(WT,Person),
      alignable(Program,Person),
      !.
align_noun_name(XT,YT,P,XT,Q):-  %% The person John %% TA-020527
      align4(XT,YT,P,Q).
/* %%TA-020527
align_noun_name(X:Man,John:Man, John isa Man,John:Man,John isa Man):-
```

```
        John=X.
*/
%% TA-020313
align5(X:MT,Y:WT,P,P):-
        bottom(MT,Program),
        bottom(WT,Person),
        alignable(Program,Person),
         ( \+ flounder(X,Program)
                  ;                    % NOT both are variables
           \+ flounder(Y,Person)),  % is (bus)4 a number ?
        !,
        X=Y. %% both will be instantiated
worldvalue(W):-
        value(world,W),!;W=real. %% TA-020129
latin(or,X:T1,Y:T2,(X;Y):T):-
        joinclass(T1,T2,T),
        T \== thing.  %% They must have something in common %% TA-020107
```

B.6 tuc/gram_e.pl

```
be_compl(X,S,Com,P2) --->
        the0,              %% TA-020522
        ap(A,X,S,Com2,P1),
        verb_complements0(adj/A,X,S,Com2:P1,Com:P2).
        %% identical to Adj Compls NEW
whodidit(_V,_Y,X,B,C)--->     %% tore ws killed
        by,
        !,accept,
        lock,                   %% TA-020519
                  noun_phrase(X,B,C),   %% by john and mary
        unlock.
```

C Extracted facts

This appendix shows how GeneTUC semantically understood the given sentences. For an explanation of the TUC Query Language (TQL) syntax used here, see [Bra97].

This list was made with the command "*\r abs*" in GeneTUC mode:

```
the crem gene encodes both activators and repressors of
camp - induced transcription.
   (1)    crem isa gene
   (2)    sk(1)isa activator
   (3)    encode/crem/(sk(1),sk(3))/sk(2)
   (4)    event/real/sk(2)
   (5)    sk(3)isa repressor
   (6)    sk(5)isa transcription
   (7)    adj/camp_induced/sk(5)/real
   (8)    has/process/repressor/sk(5)/sk(3)
   (9)    event/real/sk(4)

icer ( inducible camp early repressor ) isoforms are
generated upon activation of an alternative intronic
promoter within the crem gene.
   (10)   sk(7)isa isoform
   (11)   adj/icer/sk(7)/A
   (12)   sk(8)isa agent
   (13)   sk(9)isa activation
   (14)   sk(10)isa promoter
   (15)   adj/alternative/sk(10)/real
   (16)   adj/intronic/sk(10)/real
   (17)   nrel/within/role/gene/sk(10)/crem
   (18)   nrel/of/activity/thing/sk(9)/sk(10)
   (19)   generate/sk(8)/sk(7)/sk(11)
   (20)   srel/upon/activity/sk(9)/sk(11)
   (21)   event/real/sk(11)

icer is proposed to down - regulate both its own
expression and the expression of other genes that
```

```
contain camp responsive elements ( cres ) such as a
number of growth factors.
   (22)   icer isa protein
   (23)   sk(13)isa agent
   (24)   propose/id/that/sk(13)/sk(14)/sk(15)
   (25)   event/real/sk(15)
   (26)   srel/down/place/nil/sk(15)
   (27)   sk(16)isa expression
   (28)   has/gene/expression/sk(20)/sk(16)
   (29)   regulate/icer/(sk(16),sk(18))/sk(17)
   (30)   event/sk(14)/sk(17)
   (31)   sk(18)isa expression
   (32)   sk(20)isa gene
   (33)   adj/other/sk(20)/real
   (34)   sk(22)isa factor
   (35)   adj/growth/sk(22)/A
   (36)   sk(24)isa element
   (37)   adj/camp/sk(24)/real
   (38)   adj/responsive/sk(24)/real
   (39)   adj/such/sk(24)/sk(25)
   (40)   event/real/sk(25)
   (41)   contain/sk(20)/sk(24)/sk(21)
   (42)   srel/as/thing/sk(23)/sk(21)
   (43)   srel/of/thing/sk(22)/sk(21)
   (44)   event/real/sk(21)
   (45)   has/gene/expression/sk(20)/sk(18)
   (46)   event/real/sk(19)

thus icer has been postulated to play a role in
proliferation and differentiation.
   (47)   adj/said/icer/sk(28)
   (48)   event/real/sk(28)
   (49)   srel/in_order_to/thing/sk(27)/sk(28)
   (50)   sk(30)isa role
   (51)   sk(31)isa proliferation
   (52)   nrel/in/role/thing/sk(30)/(sk(31),sk(32))
   (53)   sk(32)isa differentiation
   (54)   play/icer/sk(30)/sk(29)
   (55)   event/real/sk(29)
   (56)   srel/being_the/reason/sk(27)/sk(29)

here we show that icer gene expression is induced by
gastrin cholecystokinin ( cck ) and epidermal growth
factor ( egf ) in ar42j cells.
   (57)   'I' isa self
   (58)   show/id/that/'I'/sk(34)/sk(35)
   (59)   event/real/sk(35)
   (60)   srel/here/place/nil/sk(35)
```

(61) sk(36)isa expression
(62) adj/icer/sk(36)/A
(63) gastrin isa substance
(64) ar42j isa cell
(65) induce/(gastrin,cholecystokinin,epidermal_growth_factor)/sk(36)/sk(37)
(66) srel/in/thing/ar42j/sk(37)
(67) event/sk(34)/sk(37)
(68) cholecystokinin isa substance
(69) epidermal_growth_factor isa substance

the time course of gastrin - and cck - mediated icer induction is rapid and transient similar to forskolin - and pma - induced icer expression.
(70) sk(39)isa course
(71) adj/time/sk(39)/A
(72) sk(41)isa induction
(73) adj/icer/sk(41)/A
(74) adj/gastrin_cck_mediated/sk(41)/real
(75) has/activity/course/sk(41)/sk(39)
(76) event/real/sk(40)
(77) adj/rapid/sk(39)/sk(42)
(78) event/real/sk(42)
(79) adj/transient/sk(39)/sk(42)
(80) sk(43)isa expression
(81) adj/icer/sk(43)/A
(82) adj/forskolin_pma_induced/sk(43)/real
(83) comp/thing/eq/sk(39)/sk(43)

the specific cck - b receptor antagonist l740093 blocks the gastrin – but not the cck - response. (egentlig ‘gastrin-response' !)
(84) l740093 isa cckbr_antagonist
(85) sk(45)isa response
(86) cck isa gene
(87) (block)/l740093/gastrin/sk(46)
(88) event/real/sk(46)
(89) srel/in/thing/sk(45)/sk(46)
(90) srel/in/thing/cck/sk(46)
(91) srel/not/mode/nil/sk(46)
(92) srel/but/mode/nil/sk(46)
(93) adj/specific/l740093/real

this indicates that both the cck - b receptor and the cck - a receptor can mediate icer gene activation.
(94) indicate/id/that/it/sk(48)/sk(49)
(95) event/real/sk(49)
(96) cckbr isa gene
(97) sk(50)isa activation
(98) adj/icer/sk(50)/A
(99) mediate/(cckbr,cckar)/sk(50)/sk(51)
(100) event/sk(48)/sk(51)

```
   (101)  cckar isa gene

noteworthy creb is constitutively phosphorylated at ser
133 in ar42j cells and icer induction proceeds in the
absence of increased creb ser 133 - p.
   (102)  creb isa protein
   (103)  sk(53)isa agent
   (104)  serin isa amino_acid
   (105)  phosphorylate/sk(53)/creb/sk(54)
   (106)  srel/at/amino_acid/serin/sk(54)
   (107)  srel/in/thing/ar42j/sk(54)
   (108)  event/real/sk(54)
   (109)  srel/constantly/mode/nil/sk(54)
   (110)  sk(55)isa induction
   (111)  adj/icer/sk(55)/A
   (112)  sk(56)isa absence
   (113)  proceed/sk(55)/sk(54)
   (114)  srel/in/thing/sk(56)/sk(54)
   (115)  srel/of/thing/creb/sk(54)
   (116)  adj/increased/creb/real

gastrin - mediated icer induction was not reduced in
the presence of the pka inhibitor h - 89.
   (117)  (h_89 isa pka_inhibitor, A isa induction, B
          isa agent, C isa presence, adj/icer/A/D,
          adj/gastrin_mediated/A/real, reduce/B/A/E,
          srel/in/thing/C/E, srel/of/thing/h_89/E,
          event/real/E)=>false

this indicates a pka independent mechanism.
   (118)  sk(59)isa mechanism
   (119)  adj/pka_independent/sk(59)/real
   (120)  indicate/it/sk(59)/sk(60)
   (121)  event/real/sk(60)

this is the first report on icer inducibility via gq -
g11 protein coupled receptors.
   (122)  sk(62)isa inducibility
   (123)  gq_g11_pcr isa gpcr
   (124)  nrel/via/thing/thing/sk(62)/gq_g11_pcr
   (125)  sk(63)isa report
   (126)  adj/first/sk(63)/real
   (127)  nrel/on/information/thing/sk(63)/icer
   (128)  be1/it/sk(64)
   (129)  event/real/sk(64)
   (130)  srel/in/thing/sk(62)/sk(64)
   (131)  srel/in/thing/sk(63)/sk(64)
```

D TUC commands

Here is a list over the most common commands you need to know, in order to work with the GeneTUC (and other TUC) systems:

D.1 Sicstus Prolog

These are commands that can be typed at the "| ?- " prompt:

- [*Filename*]. Compiles (or re-compiles) the file called "*Filename.pl*" into the memory.
- hi. Homemade command to start the debugger the right way.
- listing (pred). Lists all occurrences of the predicate "*pred*" from the memory.
- run. Starts the GeneTUC mode.
- spy (pred). Put a spypoint on the predicate, for debugging purposes.
- testgram. Set some appropriate standard spypoints in the grammar predicates.

All these commands can also be written in GeneTUC mode, but then you need to add a \ (backslash) in front of the command.

D.2 GeneTUC mode

These are commands that can be typed at the "E: " prompt:

- *Any Sentence.* Will parse the *sentence* and write output.
- \r abs. Will parse sentences from the file "*abs.e*" and write output for each sentence.
- \set traceprog 1 Will give standard output (only facts)
- \set traceprog 2 Will give verbose output (including text tagging and anaphora candidates)

- \set traceprog 3 Will give extra verbose output (including the parse tree).
- \spyg *rule* Put a spypoint on the given grammar *rule*.
- *pred* Any prolog predicate can be executed in GeneTUC mode.

E Dictionary

E.1 Biological

cAMP	= cyclic Adenosine Mono-Phosphate
CCK	= Cholecystokinin
CRE	= cAMP Responsive Elements
CREM	= (a gene name)
GO	= Gene Ontology
ICER	= Inducible cAMP Early Repressor
MTFS	= Medical Research Center, NTNU
PKA	= cAMP-dependent Protein Kinase

E.2 Natural Language Processing

Biolinguistics	= NLP on biomedical texts
Consensical	= Context Sensitive Categorial Attribute Logical
DB	= Database
EoS	= End-of-Sentence
GUI	= Graphical User Interface
IR	= Information Retrieval
KS	= Knowledge System
KW	= Key Word
NCBI	= National Center for Biotechnology Information
NIH	= National Institutes of Health
NLM	= National Library of Medicine
NLP	= Natural Language Processing
NP	= Noun Phrase
POS	= Part Of Speech
RDF	= Resource Description Framework
RTF	= Rich Text Format

TFOL	= Temporal First Order Logic
TQL	= TUC Query Language (A simplified and skolemised version of TFOL)
TUC	= The Understanding Computer