**EXTRACTING KOZAK CONSENSUS SEQUENCE USING KLEISLI.**

JING CHEN1, NAM-HAI CHUA2, DAPHNA STRAUSS3, LIMSOON WONG4,5

1Kent Ridge Digital Labs, 21 Heng Mui Keng Terrace, Singapore 119613. e-mail: cjing@krdl.org.sg;

2The Rockefeller University, 1230 York Avenue, New York, NY 10021-6399. e-mail: chua@rockvax.rockefeller.edu;

3Kent Ridge Digital Labs, 21 Heng Mui Keng Terrace, Singapore 119613. e-mail: daphna@krdl.org.sg;

4Contact author;

5Kent Ridge Digital Labs, 21 Heng Mui Keng Terrace, Singapore 119613. e-mail: limsoon@krdl.org.sg

Keywords: context sequences, database integration, high-level query luanguage CPL

Consensus sequence for the context of translation initiation codon is useful to many molecular biologists in identifying the possible translation initiation codon in new genes. We have two objectives in this paper. First, we want to build a system that can extract context sequences of translation initiation codon from large DNA databases and compute a consensus sequence for these context sequences. Second, we want to demonstrate how the general bioinformatics database integration system called Kleisli can help build such a system easily. We achieve these two objectives by showing that short and clear programs can be written in Kleisli, using its high-level query language CPL, to build such a system by integrating WU-BLAST2.0, Entrez, and database-style indices.

**1. Introduction**

The recent explosion of genomic information, as gleaned from the Human Genome Project and other similar efforts, has been fueled by engineering and technological advances. However, as the amount of information grows the challenge becomes one of managing and making sense of the data. In fact, biological discovery is increasingly performed in a "dry and wet" paradigm, in which hypotheses are first formulated by analyzing existing data sources and subsequently confirmed by experiments. This paradigm relies on the development of a number of tools that require advances in the fundamentals of computer science as well as careful engineering. "Dry" experiments inevitably involve integrating and transforming data sources so that they can be analyzed or "mined" with the appropriate software. Integration of these data sources is complicated by the fact that they use different formats and require different languages for accessing them. One of the first challenges that must be addressed therefore is to provide the appropriate tools to scale this "tower of babel."

Kleisli [6, 2] is an advanced technology that attempts to scale this bioinformatics "tower of babel." Many bioinformatics problems (1) require access to data sources that are high in volume, highly heterogeneous and complex, constantly evolving, and geographically dispersed; (2) require solutions that involve multiple carefully sequenced steps; (3) require information to be passed smoothly between the steps; (4) require increasing amount of computation; and (5) require increasing amount of visualization. Kleisli is designed to handle the first three requirements directly. In particular, Kleisli provides the high-level query language CPL [3] that can be used to express complicated transformation across multiple data sources in a clear and simple way. In addition, while Kleisli does not handle the last two requirements directly, it is capable of distributing computation to appropriate servers and initiating visualization programs.

We demonstrate here the use of Kleisli to solve an interesting problem in bioinformatics: the extraction of Kozak sequences from large genomic databases and the derivation of a consensus for them. The context of the translation initiation codon in genomic sequences is called Kozak sequence [10]. Ten years ago, Joshi [7] proposed a consensus Kozak sequence for the translation initiation codon in higher plants on the basis of 79 genomic sequences. His result was useful to many plant researchers in identifying the possible translation initiation sites in new genes. The distribution of sequences in genomic databases such as GenBank [4] is skewed, with certain gene families heavily over represented. Fortunately, the size of GenBank is increasing rapidly. As the size increases, the coverage of GenBank of gene families becomes more extensive and less biased. Therefore, it is worthwhile to periodically extract Kozak sequences for various organisms and compute their consensus. Also, the coverage of GenBank on certain organism is now large enough that it is feasible to consider extracting Kozak sequences and computing their consensus with respect to functional classification of genes within one species or across entire family of species. The problem of extracting Kozak sequences and computing their consensus with respect to functional classification of genes is quite typical of many bioinformatics problems. We use for illustration here the energy-related genes of the model dicot*Arabidopsis,*whose entire genome is currently being sequenced [9].

On 25 April 1998, one of us used the specification "energy AND arabidopsis[Organism]" to search the GenBank portion of the popular Entrez system [13] at the National Center for Biotechnology Information (NCBI) in Washington DC to get these "energetic" Arabidopsis genomic sequences. One needed to get hold of them before one could proceed to extract and analyse their Kozak sequences. It returned a list of exactly one record. There were actually more "energetic" Arabidopsis genes than one in GenBank. There are two reasons associated with this poor recall performance. The first is that the specification is too simple minded. The second is that records in GenBank are poorly annotated and some are even unannotated. The first problem can be rectified by a more sophisticated specification that more fully exploit the biological meaning of "energy-related." However, no amount of fixing of the Entrez search specification can deal with the second factor of poor recall performance. So we can see that without the support of good tools, it is not even possible to start!

Let us try to describe what kind of tools are needed. The most straightforward way to find most of the energy-related Arabidopsis sequences is to rely on the fact that sequences having a high degree of homology are likely to have highly similar functions. So one should ask Entrez for all energy-related sequences, regardless of species. Even though some specific energy-related sequences of one species might be missed out, hopefully a similar one from another species would be picked out to fill the gap. Then we can compare all Arabidopsis sequences to this set of known energy-related sequences from other species. Those that exhibit strong homology would be the energy-related Arabidopsis sequences. In order to carry out this procedure smoothly, one has to integrate Entrez with a reliable homology search system such as WU-BLAST2.0 [1]. The procedure above can identify energy-related Arabidopsis sequences. However, we are still far from being able to extract their Kozak sequences. To do that we need to locate the start codon of each sequence. To locate the start codon we need to inspect the so-called feature table of each sequence record. The feature table gives the positions of various features in a sequence, including positions of its first exon and thus the location of its translation start codon. However, feature tables in GenBank reports are considerably more complicated than simple flat relational tables. So we would also need a system that can manipulate these tables with ease. Assume that we can manipulate feature tables and thus extract Kozak sequences, we would still need a way to analyse them in order to calculate their consensus using, for example, Cavener's 50/75 consensus rule [5].

The modern database integration system Kleisli possesses all the necessary attributes to address the requirements above. This article focuses on showing how Kleisli can be used to extract and analyse Kozak sequences from Entrez. We organize our presentation into the following sections. Section 2 describes the programming of the basic steps of extracting Kozak sequences of all Arabidopsis sequences from the GenBank portion of Entrez. Section 3 presents the calculation of consensus Kozak sequence using the popular 50/75 rule [5]. Then we proceed to Section 4 which shows we can integrate Entrez and WU-BLAST2.0 to extract and analyse Kozak sequences with respect to functional classification of sequences.

**2. Extracting Kozak Sequences**

In this section we explain the basic steps of extracting Kozak sequences of the model dicot*Arabidopsis*. We use the GenBank section of Entrez at Washington DC to be our source of genomic sequences.

**Step 1**. It is only possible to extract Kozak sequences from genomic sequences whose coding regions are explicitly annotated. The Entrez specification to download Arabidopsis DNA sequences having coding regions that are explicitly annotated is "Arabidopsis[0rganism] AND cds[Feature key]." The CPL function na-get-uid-general remotely queries Entrez for unique identifiers of GenBank reports satisfying a given specification. Thus to find out which Arabidopsis sequences have annotated coding regions, we just apply na-get-uid-general to the specification "Arabidopsis[0rganism] AND cds[Feature key]" and write the unique identifiers obtained to a file uids for later use, as shown in the CPL program below.

writefile na-get-uid-general

"Arabidopsis [Organism]+AND+cds [Featnre+key] "

to "uids" using stout;

readfile uids from "uids" using stdin;

**Step 2**. Our second step is to obtain all Arabidopsis DNA sequences whose coding regions are explicitly annotated. The CPL function na-get-seq-by-uid downloads from Entrez the DNA sequence uniquely identified by its input argument. So for each unique identifier u in uids, we apply na-get-seq-by-uid to u get the corresponding sequence s. These sequences are written to the file seq. A database-style index accn2seq is then created on the #accession field of the sequence file seq, so that we have fast access to a sequence given its accession number. The CPL program implementing all these is given below.

writefile

{ s I \u <- uids, \s <- na-get-seq-by-uid (u) }

to "seq" using stdout;

writefile "seq"

to "#accession" using setindex-create;

setindex-access (#name: "accn2seq",

#file: "seq", #key: "#accession");

**Step 3**. Before we proceed to obtain the feature table of each sequence, we need functions to implement the extraction of a Kozak sequence given position information. There are two functions, get-+ve-strand for extracting from the positive strand and get—ve-strand for extracting from the negative strand. One of them is implemented as shown below.

primitive get-+ve-strand == \p ==>

{ left " "ATG" ^ right

| \S <- process <#key: p.#accn> using accn2seq,

(\s,\ps,\pe) == (S. #sequence, p.#start,p.#end),

\exon === string-span (s, ps, pe),

exon string-islike "ATG%",

\left == string-span(s, ps-15, ps-1),

string-length (left) = 15,

\aug == string-span(s, ps, ps + 2),

\right == string-span(s, ps+3, ps+ 12),

string-length (right) = 10 };

We describe get-+ve-strand now. Its input p is a position record, specifying how the first exon is derived. The #accn field of p gives the accession number of the sequence of this exon. We fetch the sequence s from our index accn2seq. Then we use the #start and #end fields of p to extract from s the specific segment exon constituting the exon. We make sure that exon is indeed the first exon by testing that it begins with ATG. The upstream component of the Kozak sequence left is extracted from s. We test to make sure that it has exactly 15 bases. The downstream component of the Kozak sequence right is extracted from s. We test to make sure that it is 10 bases long. The Kozak sequence is then built by concatenating left, ATG, and right. The get—-ve-strand function is similar, except we need take care of reverse-complementation .

**Step 4**. We can now proceed to get our Kozak sequences. For each unique identifiers u in uids, we apply the CPL function na-get-feature-by-uid to fetch its feature table x from Entrez. We iterate through each feature f in x to look for feature that corresponds to coding regions; these are those feature whose name is CDS. We have to make sure that the feature is complete, hence the test f.#continuous. To ensure that the coding region is proper, we check to see if its annotations a contain a protein translation and if that translation begins with methionine (M). If all these tests pass, we set p to be the position record of the first exon of this feature. Then depending on whether the exon is on the positive or negative strand, we invoke get-+ve-strand or get--ve-strand to get the corresponding Kozak sequence z. The Kozak sequences, the unique identifiers, and the translations are all written to the file kozak for subsequent processing.

writefile

{ (#uid: x.#uid, #translation: a.#descr,

#fkozak: z)

| \u <- uids,

\x <- na-get-feature-by-ilid (u),

\f <- x.#feature,

f.#continuous, f.#name = "CDS",

\a <--- - f.#anno,

a.#anno\_name = "translation",

a.#descr string-islike "M%",

\p == f.#position .list-head,

\z <- if p.#negative

then get-ve-strand (p)

else get-+ve-strand (p) }

to "kozak" using stdout;

readfile kozak from "kozak" using stdin;

**3. Computing Sequences**

All the Kozak sequences of Arabidopsis are in the set kozak. We can now proceed to calculate a consensus sequence for them. We determine the consensus sequence using the 50/75 rule described by Cavener [5] and used in many other papers [8, etc.]

The consensus at a position is computed according to the following rules, with decreasing order of priority. (1) If a nucleotide at that position has a relative frequency greater than 50% and greater than twice the relative frequency of the second most frequent nucleotide, the nucleotide is given consensus status and is indicated in uppercase. (2) If the sum of the relative frequencies of a pair of nucleotides exceeds 75%, these two nucleotides are given co-consensus status are indicated in uppercase. (3) If there is a single most frequent nucleotide, it is given dominant status and is indicated in lowercase. (4) If two bases have the same highest frequency, they are given co-dominant status and are indicated in lowercase.

Due to space limitation, we omit short CPL programs of Steps 5-7 that derive consensus for Kozak sequences from the previous section. Instead, we show part of the output in a table below. The table contains the context information from -5 to +5 bases around the initiation codon ATG. The consensus is (t/a)aaaaaaaaaaaAaaLATGGc(t/g)actaata. It is consistent with the consensus on dicots in general[8]: aaaaaaaA(A/C)aATGGCtacta(c/t)ta; recall that Arabidopsis belongs to the Dicotyledon family! For example, the context upstream of ATG in dicots is predominantly a and t, as is our case here in Arabidopsis; and the -3 position upstream of ATG in dicots is A, as is our case here in Arabidopsis. However, there are also some differences between Arabidopsis and dicots in general. For example, at the +5 position downstream from ATG, the consensus for dicots is C at 60% frequencies [8], but it is only a c at 40% frequency for Arabidopsis.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| position | -5 | -4 | -3 | -2 | -1 | ATG | +4 | +5 |
| A-%  C-%  G-%  T-% | 32  25  17  26 | 44  14  23  19 | 51  10  25  14 | 43  30  9  18 | 44  20  24  12 | -  -  -  - | 20  7  58  15 | 30  40  15  15 |
| consensus | a | a | A | a | a | ATG | G | c |

**4. Classification by Function**

Genes of an organism have different functions and fall into general function classes such as energy, signal transduction, metabolism, and so on [II]. Having a good idea of the context sequences of translation initiation codon may be useful in understanding the function of novel genes. Now that we have gone through the basic steps of extracting and analysing Kozak sequences for an entire organism, let us discuss the detail of extracting and analysing Kozak sequences with respect to such function classes. Since we have already downloaded the sequences of Arabidopsis, we continue with them. As our example, we want to extract Kozak sequences from Arabidopsis sequences involved with energy-related functions.

**Step 8**. As discussed earlier, we need to first use known energy-related sequences from other organisms to identify energy-related Arabidopsis sequences in our collection. We accomplish this by downloading all energy-related protein sequences from Entrez and constructing a BLAST database on them. The CPL script for this step is shown below.

writefile

{ (#uid: x.#uid, #accession: x .#accession,

#title: x.#title, #seq: x.#sequence)

| \u <- aa-get-uid-general "energy",

\x <- aa-get-seq-by-uid (u), x.#uid = u }

to "energy-blast" using localblast-setdb;

localblast-blastp (#name: "energy-blast",

#db: "energy-blast");

The aa-get-uid-general function of CPL is used to obtained unique identifiers u of protein sequences in Entrez. The Entrez search specification used here is the string "energy". It can be easily replaced by a more refined specification; but this is outside the scope of this paper. The protein sequence x of u is then downloaded from Entrez using the CPL function aa-get-seq-by-uid. The downloaded sequences are written to the file energy-blast as a BLAST database using the localblast-setdb function. A BLASTP connection energy-blast is open to this file using the localblast-blastp function. Subsequently, we can perform BLASTP on a sequence against this database of energy-related sequences simply by processing the sequence using this connection.

**Step 9**. We can now identify which Kozak sequences in the file kozak obtained in Step 4 are from energy-related Arabidopsis genes. Recall that each record in Kozak has a #translation field. This field stores the protein sequence of the corresponding genes. So all we have to do now is to process this protein sequence using BLASTP to see if it has significant homology to some of the energy-related sequences. The CPL program is shown below.

writefile

{ x

| \x <- kozak,

set-thereis

@ (\h => h.#pscore < I.OE~6)

@ (process x .#translation using energy-blast)}

to "energy-kozak" using stdout;

Here we iterate the following for each x in kozak. BLASTP is performed on the protein translation of x.

We then check if there is a hit h whose pscore is significant. We use 0.000001 as our threshold here. If a significant hit is found, we include x in the output, which is written to the file energy-kozak. All that remains is to calculate their consensus as in the general case.

**5. Directions**

There are two main directions to proceed with respect to Kozak sequences. The first is to make use of our system to perform a comparative study of context sequences of translation initiation codon across different organisms and across different function classes within one organism. Such a study should be repeated at suitable interval because the size of genomic databases increases with respect to time and it is of interest to examine if conclusions based on current data will still be valid in the presence of more extensive data in future. Some preliminary results along this direction have already been obtained and are available at<http://adenine.krdi.org.sg:8080/~limsoon/kozak>.

The second direction is to make use of the Kozak sequences and consensus obtained to develop reliable systems to predict translation initiation site of DNA sequences from various organisms and function classes. The identification of translation initiation site is an important step in discovering new genes. Some systems already exist for predicting translation initiation sites. For example, in a 1997 study involving about 500 Arabidopsis sequences, NetStart [12] was able to correctly identify nearly 90%of translation initiation codons with negligible false positives. We should test it again using our considerably larger number of Arabidopsis sequences. If the test is promising, we should then use our Kozak sequences to train neural networks similar to NetStart to predict translation initiation sites of other species (especially amongst plants) and of selected function classes.

**References**

1. S. F. Altschul and W. Gish. Local alignment statistics.*Methods in Enzymology,*266:460-480, 1996.
2. P. Buneman, S. Davidson, K. Hart, C. Overton, and L. Wong. A data transformation system for biological data sources. In*Proceedings of 21st International Conference on Very Large Data Bases,*pages 158-169, August 1995.
3. P. Buneman, L. Libkin, D. Suciu, V. Tannen, and L. Wong. Comprehension syntax.*SIGMOD Record,*23(l):87-96, March 1994.
4. C. Burks, M.J. Cinkosky, W.M. Fischer, P. Gilna, J.E. Hayden, G.M. Keen, M. Kelly, D. Kristoffer-son, and J.D. Lawrence. GenBank.*Nucleic Acids Research,*20 Supplement:2065-9,1992.
5. D. R. Cavener. Comparison of the consensus sequence flanking translation start site in*drosophila*and vertebrates.*Nucleic Acids Research,*15:1353-1361,1987.
6. S. Davidson, C. Overton, V. Tannen, and L. Wong. BioKleisli: A digital library for biomedical researchers.*International Journal of Digital Libraries,*l(l):36-53, April 1997.
7. C. P. Joshi. An inspection of the domain between putative TATA box and translation start site in 79 plant genes.*Nucleic Acids Research,*15:6643-6653, 1987.
8. C. P. Joshi, H. Zhou, X. Huaag, and V. L. Chiang. Context sequences of translation initiation codon in plants.*Plant Molecular Biology,*35:993-1001, 1997.
9. J. Kaiser. First global sequencing effort begins.*Science,*274,1996.
10. M. Kozak. An analysis of 5'-noncoding sequences from 699 vertebrate messenger RNAs.*Nucleic Acids Research,*15:8125-8148,1987.
11. C. Ouzounis, G. Casari, C. Sander, J. Tamames, and A. Valencia. Computational comparisons of model genomes.*TIBTECH,*14:280-285, August 1996.
12. A. Gorm Pedersen and H. Nielsen. Neural network prediction of translation initiation sites in eukary-otes: Perspectives for EST and genome analysis.*Intelligent Systems for Molecular Biology,*5:226-233, 1997.
13. G. D. Schuler, J. A. Epstein, H. Ohkawa, and J. A. Kans. Entrez: Molecular biology database and retrieval system.*Methods in Enzymology,*266:141-162, 1996.
14. L. Wong.*The CPL Reference Manual.*Institute of Systems Science, Heng Mui Keng Terrace, Singapore 119597, October 1995. Available at<http://sdmc.iss.nus.sg/kleisli/psZ/cpl-defn.ps>.