Finding Genes

Today’s topic: programs that search for genes
  • background
  • two applications
  • current research

Reading:
Haussler, “Computational Genefinding” unpublished; expanded version of Trends article (1998)

About the Authors

Christopher Burge (MIT) developed Genscan, one of the widely used gene finding algorithms
  • developed at Stanford, in Karlin’s lab
  • freely available under academic license
  • several Genscan servers on the web
  • original paper:
  • project web site: http://genes.mit.edu/GENSCANinfo.html

Goals

A “genefinder” is a program that searches DNA strings to locate genes
  • mostly used for protein coding genes
  • tRNA genes are located by a related process

Genomic sequencing projects produce “raw DNA”
  • shotgun sequencing determines sequence of small fragments
  • fragments are assembled into larger scaffolds or complete chromosomes
  • genefinders try to locate the genes

Annotation

• The overall goal is semi-automatic annotation of a genome
• An annotation is a “map” indicating locations of
  • protein coding genes
  • genes for tRNA and other RNA sequences
  • ESTs and STS and other “markers”
  • repeats
  • other genomic features

Example from table of fugu genes:
<table>
<thead>
<tr>
<th>Chr</th>
<th>start</th>
<th>end</th>
<th>note</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>431925</td>
<td>431789</td>
<td>unknown</td>
</tr>
<tr>
<td>1</td>
<td>434147</td>
<td>435122</td>
<td>transcription factor elf</td>
</tr>
</tbody>
</table>
Terminology

- Recall from the intro lecture on molecular biology...
- A gene may have several intervening introns (non-coding regions)
- The parts that are translated (expressed) are exons

Terminology (cont’d)

- Transcription factors bind to the DNA, enable the process that generates the mRNA
- The DNA segments are binding sites

Terminology (cont’d)

- DNA strands are oriented
- Transcription proceeds from the 5’ (upstream) toward the 3’ (downstream) end

Terminology (cont’d)

- The reliability of a genefinder is measured by
  - sensitivity — the percentage of true genes it locates (i.e. minimize the number of “false negatives”)
  - specificity — the proportion of predicted genes that really are genes (i.e. minimize “false positives”)

Prokaryotic Genomes

- There are very few introns in archaea or bacteria genes
  - search in .gbk files for bacteria indicates many in tRNA
  - some “type II” introns (mobile elements)
  - one protein coding gene with a “self-splicing intron”
- Straightforward method for finding genes in prokaryotes: search for “open reading frames” (ORFs)

ORF

- An ORF is a DNA sequence that begins with a start codon and ends with a stop codon
- For a segment of DNA there are 6 frames:
  - 3 on the forward strand (stored in the DB)
  - 3 on the complementary (reverse) strand
- The stop codon should be in the same frame as the start codon
Many prokaryotes have non-standard genetic codes
- Example:
  Mycoplasma/Spiroplasma Code (transl_table=4)
  AAs = FFLLSSSY*CCMWWLL... alternate start codons
  Start = --NH----------M...
  Base1 = TTTTCCCCAAAAGG GTTTT...
  Base2 = TTTTCCCCAAAAGG GTTTT...
  Base3 = TCAATACGGACATCAATCAG... stop codon in eukaryotes

Finding Prokaryotic Genes (cont’d)
- Different organisms have different codon usage patterns
- B&K cite a study of E. coli genes:
  "typical" genes (70%) show a definite preference for one pattern of codon usage
  "highly expressed" genes (15%) show another pattern
  remainder: probably the result of horizontal transfer from another bacterial genome

Eukaryotic Genes
- B&K outline three basic methods for locating genes in eukaryotes
  1. search for "signals" — patterns that identify well-defined components
  2. stochastic models — train a HMM to recognize gene sequences
  3. similarity search — translate the DNA, do BLAST search in database of known genes

Signals
- B&K describe three types of signals used in genefinders
- Transcription signals are associated with DNA segments that are important for generating mRNA
  - transcription factors bind to control regions
  - other proteins are involved in initial processing of mRNA strings

mRNA Structure
- A newly transcribed sequence (pre-mRNA?) has
  - untranslated region — 5’ UTR
  - initial exon
  - introns and internal exons
  - terminal exon
  - untranslated region — 3’ UTR

ORFs (cont’d)
  - Mycoplasma/Spiroplasma Code (transl_table=4)
    AAs = FFLLSSSY*CCMWWLL... alternate start codons
    Start = --NH----------M...
    Base1 = TTTTCCCCAAAAGG GTTTT...
    Base2 = TTTTCCCCAAAAGG GTTTT...
    Base3 = TCAATACGGACATCAATCAG... stop codon in eukaryotes
Transcription Signals

- Regions at the front of a gene (upstream from the ORF):
  - “proximal elements”
  - wide variety, 6 to 20bp long
  - TATA box -- around 30bp before start of transcribed region
  - very highly conserved
  - in area rich in other A’s, T’s?
- transcription begins somewhere in front of start codon
- after transcription, a “cap” is added to mRNA
- other conserved regions help locate this site?

Transcription Signals (cont’d)

- A region at the end of the gene identifies a place where a “tail” is added to the mRNA
- polyadenylation adds ~250 A’s
- the new bases are added soon after AATAAA

Translation Signals

- B&K describe a small region immediately before the start codon
  - “Kozak signal”
  - elsewhere on net: descriptions of consensus sequence
  - ~12 bp
- Not enough information here to search raw DNA
- Use in conjunction with other methods to locate the initial exon

Splicing Signals

- Splicing signals define exon/intron boundaries for internal exons
  - Most introns in plants, vertebrates, invertebrates start with GT, end with AG
  - 3’ ends of introns also tend to have higher percentage of T and C
  - B&K cite papers on dependences -- correlations between separated bases -- and models to recognize them
  - These models do not locate introns in “bulk DNA” but do help locate them when integrated with other methods that find complete genes

Markov Process

- The second method described by H&K involves stochastic models
- A “Markov process” is
  - A system defined by a current state
  - state transition function defines next state as a function a finite number of previous states
  - first-order Markov process: only the current state defines the possible values of the next state
- Example: number of chips in your stack at a casino
  - bet $1 chip on each roll of a roulette wheel
  - stack will grow or shrink by one chip after each roll

Markov Model

- In sequence analysis, a Markov model is a stochastic finite state machine
  - Example: FSA to “generate” an intron string
  - From start state, write the symbol on the arc that leads to the next state
  - Where there is more than one arc leaving a state, attach a probability to each arc

Translation Signals

- B&K describe a small region immediately before the start codon
  - “Kozak signal”
  - elsewhere on net: descriptions of consensus sequence
  - ~12 bp
- Not enough information here to search raw DNA
- Use in conjunction with other methods to locate the initial exon

Splicing Signals

- Splicing signals define exon/intron boundaries for internal exons
  - Most introns in plants, vertebrates, invertebrates start with GT, end with AG
  - 3’ ends of introns also tend to have higher percentage of T and C
  - B&K cite papers on dependences -- correlations between separated bases -- and models to recognize them
  - These models do not locate introns in “bulk DNA” but do help locate them when integrated with other methods that find complete genes

Markov Process

- The second method described by H&K involves stochastic models
- A “Markov process” is
  - A system defined by a current state
  - state transition function defines next state as a function a finite number of previous states
  - first-order Markov process: only the current state defines the possible values of the next state
- Example: number of chips in your stack at a casino
  - bet $1 chip on each roll of a roulette wheel
  - stack will grow or shrink by one chip after each roll

Markov Model

- In sequence analysis, a Markov model is a stochastic finite state machine
  - Example: FSA to “generate” an intron string
  - From start state, write the symbol on the arc that leads to the next state
  - Where there is more than one arc leaving a state, attach a probability to each arc
Markov Model (cont’d)

- An FSA can recognize (parse) strings, also
  - Read one character on each state transition
  - Final product, after last symbol read, is a measure of the probability the string belongs to the set modeled by the FSA

![Diagram of an FSA with states G, T, A, and C, and arcs with probabilities 0.95 and 0.05.]

Hidden Markov Model

- In a Hidden Markov Model (HMM) the state transition probabilities are not known or not specified
  - Use a machine learning approach to set probabilities
    - select a training set of known genes
    - assign random probabilities
    - iterate:
      - have model analyze each gene, record score
      - adjust weights

Training Set

- An important issue in machine learning: the size and structure of the training set
  - big enough to include several examples of each type of sequence
  - focused enough so the system will converge on a solution
  - not too focused or the system will be “over-trained” to recognize only the examples from the training set

Tutorial

- For more information:
  - Stochastic Modeling Techniques: Understanding and Using Hidden Markov Models
  - L. Grate, R. Hughey, K. Karplus, K. Sjölander
  - Tutorial notes, ISMB’96
  - (PDF on class web site)

Tutorial (cont’d)

- Describes HMMs for remote homology
  - e.g. given a protein sequence s, what is the probability it belongs to the class of globins?

- \[ P(s \mid M, \text{path}) = \prod_i P(s_i \mid \text{path}_i) \prod_i P(\text{edge}_{i,i+1}) \]
  where path is the sequence of arcs traversed
  \[ i \] is the sequence index
  \[ P(s_i \mid \text{path}) \] is probability of finding letter \( s_i \) at this machine state
  \[ P(\text{edge}_{i,i+1}) \] is probability on the arc from \( i \) to \( i+1 \)

Markov Model of a Gene

- A simple Markov model of a gene:

![Diagram of a gene with exons and inter-genic DNA.]


Tutorial
**A More Realistic Model**

- From B&K: A composite model uses MMs for different exons, plus models of other features

**Similarity Search**

- The third class of gene finding methods described by H&K is remote homology search
  - Use BLAST to compare DNA with EST database
  - Use BLASTX to translate DNA (9 ways) and compare with protein database
  - Use TBLASTX to search translations of both the DNA and a DNA database
  - PROCRUSTES: does "spliced alignment" of tentative gene with known protein coding sequences
  - These methods work best in conjunction with the others, as a final verification step

**RepeatMasker**

- RepeatMasker scans a DNA string to look for repeats
  - several patterns in noncoding intergene regions
  - up to 50% of the human genome

- Use before gene finders are run on newly assembled chunks of DNA

- Filters out repeat elements, so gene finders can concentrate on the remainder

**Current Example: Ensembl**

- The Ensembl Genome Browser (www.ensembl.org) is a public repository of genomic sequence information
  - raw genomic sequences, assembled and unassembled
  - automated process for annotating assembled chromosomal scaffolds
  - web interface for searching for features
  - Perl CGI so remote applications can access the database

**Ensembl (cont’d)**

- Four step annotation process, described by Hubbard, et al, Nucl. Acids Research, 2002:
  - "First, 'best in genome' positions for all known human proteins from SPTREMBL are found using a fast protein to DNA matcher…"
  - "These positions are refined using genewise to provide an accurate gene structure. UTRs are also aligned to each gene structure using full-length cDNAs where known…”
### Ensembl (cont’d)

- Number of predicted genes as of Jan 2003:

<table>
<thead>
<tr>
<th></th>
<th>GenScan</th>
<th>Ensembl</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human</td>
<td>73,128</td>
<td>22,860</td>
</tr>
<tr>
<td>Mouse</td>
<td>110,379</td>
<td>22,444</td>
</tr>
<tr>
<td>Fugu</td>
<td>29,625</td>
<td>35,180</td>
</tr>
</tbody>
</table>