Gene Prediction by Pattern Recognition and Homology Search

Ying Xu and **Edward C. Uberbacher**

Informatics Group
Computer Science and Mathematics Division
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6364
Email: yingx@ornl.gov ube@ornl.gov

Abstract

This paper presents an algorithm for combining pattern recognition-based exon prediction and database homology search in gene model construction. The goal is to use homologous genes or partial genes existing in the database as reference models while constructing (multiple) gene models from exon candidates predicted by pattern recognition methods. A unified framework for gene modeling is used for genes ranging from situations with strong homology to no homology in the database. To maximally use the homology information available, the algorithm applies homology on three levels: (1) exon candidate evaluation, (2) gene-segment construction with a reference model, and (3) (complete) gene modeling. Preliminary testing has been done on the algorithm. Test results show that (a) perfect gene modeling can be expected when the initial exon predictions are reasonably good and a strong homology exists in the database; (b) homology (not necessarily strong) in general helps improve the accuracy of gene modeling; (c) multiple gene modeling becomes feasible when homology exists in the database for the involved genes.

Introduction

Identification of genes in anonymous DNA sequences involves recognizing coding regions and parsing recognized coding regions into gene models. With different coding recognition methods and different parsing strategies, a number of computer programs have been developed for the gene identification problem (Uberbacher and Mural, 1991; Xu *et al.*, 1994a; Fields and Soderlund, 1990; Gelfand, 1990; Guigo *et al.*, 1992; Hutchingson and Hayden, 1992; Snyder and Stormo, 1993; Dong and Searls, 1994; Krogh, Mian and Haussler, 1994). Though varying in how the information is processed and applied, common to all these gene prediction methods is the basic information used: (1) content statistics measuring the positional/compositional biases imposed on the DNA sequence in coding regions by the genetic code, and/or (2) homologous sequences existing in the database. While content-statistics based methods are more general and robust they may fail on "abnormal" or short DNAs; Also this type of method tends to be "less objective" in the gene parsing phase due to the lack of discernible biological constraints. Typically, a (single) gene structure is predicted by selecting a set of recognized coding regions that satisfy the adjacent-exon spliceability condition and also optimize some (simple) objective function, which in general does not guarantee a perfect correspondence with the actual gene structure (Xu *et al.*, 1994b). Homology based methods can provide more direct evidence of coding characters and possibly a reference model in the gene parsing phase, but they may not be generally applicable as more than 50% of the newly discovered genes have no detectable homologs in the database.

We previously developed a gene prediction system, GRAIL, based on content statistics measuring exon related properties of a DNA (Uberbacher and Mural, 1991; Xu *et al.*, 1994a). The system first extracts over a dozen features from each potential exon candidate. Each of these features exhibits some discriminating power between an exon and a non-exonic region. A neural network was trained to score the partial correctness of each exon candidate based on the extracted features. The result of the neural network evaluation is a set of scored candidates, each having a pair of boundaries and a fixed translation frame. Two exon candidates can be defined to be spliceable if a certain relationship holds among their boundaries and translation frames. The GRAIL gene structure prediction subsystem predicts a (single) gene model by selecting a subset of the scored candidates so that adjacent candidates are spliceable and the total score is maximized. Though enforcing the spliceability condition has increased the accuracy of (final) exon prediction and gene structure prediction, it does not guarantee to generate the correct (single) gene model even when all the components to form the correct model are present in the exon candidate pool. A typical example for such a case may be as follows. A high scoring false exon was predicted in between of two adjacent true exons. If this false exon happens to be spliceable to both of the true exons it may be included in the predicted gene model.

Though it may not be universally applicable, homology information, when available, can be used to provide reference models in both exon prediction and gene structure prediction. Some recent work has been done to incorporate homology information in the process of exon (re)evaluation (Snyder and Stormo, 1994; Guigo and Knudsen, manuscript in preparation).

We have developed a framework for incorporating homology information in the GRAIL gene prediction process. Our goal is to maximally use the available homology information in both exon prediction and gene structure prediction. The framework consists of three main steps: (1) exon candidate re-evaluation, (2) reference-based gene-segment construction, and (3) (multiple) gene structure prediction. The algorithm first uses the GRAIL exon prediction subsystem to predict a set of exon candidates. The predicted candidates form a set of clusters, each containing overlapping exon candidates. In general, each cluster represents different predictions of a presumed exon with different boundaries. The algorithm then selects a few high-scoring candidates from each cluster to do database homology search. If homology is found, the search results are processed to form a reference exon model for the cluster, and all the candidates of the cluster are re-scored according to this reference model. In the next step, the algorithm combines the search results for all the clusters to form a set of maximal reference models (each one covers more than one exon). An optimal partial gene model, or gene segment, is constructed based on each reference model. In the third step, gene models are constructed from the gene segments and exon candidates by optimizing an objective function more general than to the one used in Xu *et al.*, 1994b. In the actual implementation, steps 2 and 3 are combined into one single step.

Preliminary tests have been done on this algorithm. In general, as expected, incorporating homology information into the gene prediction process improves the accuracy of individual exon predictions (mainly boundaries of exons). By applying reference-based gene-segment construction, the algorithm significantly reduces the false positive rate by not including exon candidates that are obviously inconsistent with the reference models. Based on our limited tests, a perfect (single) gene model can be expected when the correct exon candidates are present in the candidate pool and a strong homology exists in the database. The database search also may provide information indicating the start and end of a gene, and hence makes automated multiple gene model prediction feasible.

Exon Prediction by Pattern Recognition

This section reviews the GRAIL exon prediction algorithm (Uberbacher and Mural, 1991; Xu *et al.*, 1994a; Uberbacher *et al.*, 1996). As in any pattern recognition problem, to recognize exons we need to select a set of features that are associated with exons, and to design a method to discriminate exons from non-exonic regions.

To determine the likelihood of a DNA segment being an exon involves determination of coding potentials of the region and evaluation of the potential splice junctions (or translation starts/stops) bounding the region. GRAIL uses a frame-dependent 6-tuple preference model (Uberbacher and Mural, 1991; Claverie *et al.*, 1990) and a 5^{th} order non-homogeneous Markov chain model (Borodovsky *et al.*, 1986) to calculate coding potentials of each candidate region and its two 60-base surrounding regions (as background signal). These coding measures are used as features in the exon discrimination process.

Recognition of coding regions using the 6-tuple (or in general k-tuple, for any fixed k) method is known to have strong dependence on the G+C composition, and is more difficult in G+C poor domains. If we estimate the frequencies of frame-dependent coding 6-tuples and noncoding 6-tuples in the high G+C domain, and use these frequencies to calculate coding measures for a set of coding regions and their 60-base flanks in all ranges of G+C composition, an unexpected pattern is shown in Figure 1. The coding measures for both the coding regions and their flanks are much lower in the G+C poor domain compared to the G+C rich domain. A very similar behavior is observed if the 6-tuple frequencies are collected from low G+C DNA sequences. Hence GRAIL uses the G+C compositions of both a candidate region and a 2000-base region centered around the candidate as correction factors in the exon discrimination process.

A number of measures including a 5-tuple preference model, long-distance correlations between single bases, etc. have been used in a separate process for evaluating the strength of a potential splice junction. The result of this evaluation is used as a feature in the exon discrimination process.

One interesting observation we recently made indicates that shorter exons tend to have stronger splice junction sites and hence higher scores. Also short false exon candidates may have better chance to accidentally have high coding measures. Based on these considerations, we have included the exon candidate length as another feature in the exon discrimination process.

The extracted features over each candidate region are fed to a neural network, which has been trained to score the partial correctness of a candidate. The result of the neural network evaluation is a set of scored candidates with each having a fixed translation frame. A clustering procedure divides the candidates into clusters of overlapping candidates, each of which represents a different prediction of a presumed exon.

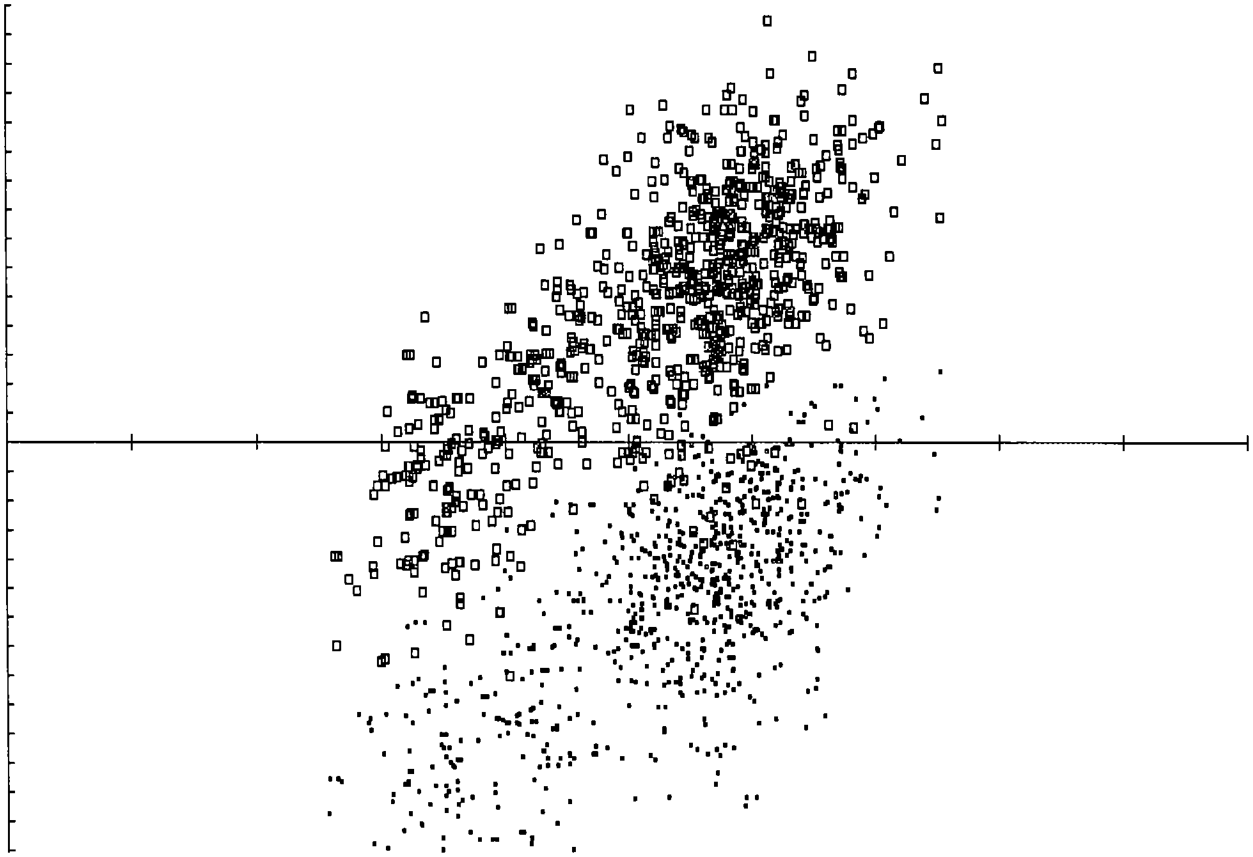

Figure 1: The X-axis represents the G+C composition of an exon candidate and Y-axis represents the 6-tuple scores measured by the frame-dependent preference model. Each tick mark on the horizontal axis represents 10% in G+C composition with 0% on the left and 100% on the right. The large squares represent the coding regions and the small dots represent the regions flanking coding regions.

Database Homology Search

By doing database search, we attempt to achieve the following goals: (1) to collect "sufficient" information to help locate where the corresponding coding region starts and ends for each candidate cluster; (2) to collect as much information as available to piece together protein segments to form long, hopefully complete protein sequence(s) (in the sense of a complete gene).

In the current implementation, we use Swissprot as the target database. The search for homology is done by the FASTA program version 2.0 (Pearson and Lipman, 1988) using the score matrix BLOSUM50. Experiments on other databases with different search programs are expected to be done in the near future.

To conduct a database search, an exon candidate is first translated into a protein sequence in its translation frame, and then this protein sequence is used as a query in the search. For each search, FASTA returns a number of hits from different proteins, possibly of different organisms. A typical FASTA hit is shown in Figure 2.

For each database hit, the following information can be extracted: (1) the starting and ending positions of the matched portion of the query sequence, from which we can calculate the starting and ending positions of the corresponding coding DNA segment; (2) the portion (subsequence and location) of the protein that the query sequence matches, from which we can further know if this portion is the beginning or end of the protein, or somewhere in between.

In the current implementation, we use the top five[1] highest-scoring candidates from each cluster to do the database search. The reason we use five instead of one or all candidates of a cluster is due to the consideration of (1) having a good representative set, and (2) the computation time constraint. The search results are sorted into different gene groups. For each group, the union of the matched protein portions is used as the reference model (this is a simplified statement but gives the basic idea). Thus each cluster has a number of reference models from different genes.

After the database search is done for all the clus-

[1] We use as many hits as returned by the search if less than five are found.

```
                10        20        30        40        50        60
query                              MADFIRGVVDSEDLPLNISREMLQQSKILKVIR
                                      :..X::::.:::::::::::::.:::::::
gene1   TRKKMNNIKLYVRRVFIMDNCEELIPEYLGFVKGVVDSDDLPLNISREMLQQNKILKVIR
                80        90        100       110       120       130

                70        80        90
query   KNIVKKCLELFSELAEDKENYKKFYEAFSKNLK
        ::.::::.:.:.:.::.::.:.::::::::::X
gene1   KNLVKKCIEMFNEIAENKEDYNKFYEAFSKNLKLGIHEDSQNRAKLADLLRYHSTKSGDE
                140       150       160       170       180       190
```

Figure 2: An example of FASTA search result.

ters, matched protein segments are sorted into different gene groups. A set of non-overlapping maximal gene segments are formed based on the protein segments obtained from the search for each gene group (note that each protein segment could be longer than the matched portion of the protein as can be seen in Figure 2). These gene segments will serve as reference models in the reference-based partial gene model construction. Labels are marked on gene segments that start and/or end a gene, which will be used in the multiple gene model construction.

Gene Modeling

Gene model construction is currently done in GRAIL by selecting a subset of non-overlapping exon candidates from the predicted candidate pool such that adjacent candidates (in their spatial relationship) satisfy the following spliceability condition and the total (normalized) neural-net score is maximized (Xu *et al.*, 1994b). We classify exons into three classes: (1) ***initial*** exons: the exons that start with a translation start *ATG*, (2) ***internal*** exons: the exons that start with an acceptor junction and end with a donor junction, and (3) ***terminal*** exons: the exons that end with a in-frame translation stop codon. Note that an exon can be both initial and terminal exon. An exon E_1 is said to be ***spliceable*** to exon E_2 if the following conditions hold. We use $l(E)$, $r(E)$ and $f(E)$ to represent the left boundary, right boundary and translation frame of E, respectively.

(1) E_1 is a non-terminal exon, and E_2 is a non-initial exon;

(2) $l(E_2) - r(E_1) \geq \mathcal{K}$, (in GRAIL, $\mathcal{K} = 60$);

(3) $f(E_2) = (l(E_2) - r(E_1) - 1 + f(E_1)) \bmod 3$;

(4) no in-frame stop are formed at the joint point when appending E_1 to E_2.

The basic assumption for such a mathematical model for gene modeling to be effective is that the score of an exon candidate is, in general, an accurate reflection of the partial correctness of it being a true exon. When this assumption is violated we may see that high-scoring false candidates are included in the gene model, or low-scoring true exons are excluded from the gene model. The problem is caused by a lack of detectable biological constraints used in this mathematical model.

In this section, we extend this model by applying homology information in addition to spliceability condition when appending exon candidates to form a gene model. Because of the markers of the start/end of a gene from the database search, we can further extend this model to construct multiple gene models.

Exon re-evaluation

For each cluster, all the candidates will be re-scored based on the reference models if the homology is above some threshold. Let E be an exon candidate, P be its corresponding protein, and R be a reference model. Recall the format of a FASTA output. We replace each identity match (":") between P and R by a value 1, each similar match (".") by 0.5 and a miss match by 0. The total of all these values is defined to be the match score between P and R, denoted by $match(P, R)$. The new score of E with respect to the reference model R is given by

$$score_R(E) = \frac{match(P, R)^2}{||R||},$$

where $||R||$ represents the cardinality of R.

To be consistent in the scoring scheme, we also rescore the exon candidates with no database homology as follows (note that the neural net score does not explicitly reflect the length of a candidate but the above does). For each such candidate E, let $net(E)$ represent E's neural net score (scaled to the range of [0,1]).

$$score_\emptyset(E) = net(E)^2 ||E||/3,$$

where $\emptyset$ indicates that the score does not depend on any reference model and $||E||/3$ gives the length of E's corresponding protein. We also define $score_R(E) = -\infty$ for all such E's.

Reference-based gene model construction

This subsection presents an algorithm for constructing a (multiple) gene model from a set of predicted exon candidates that maximizes the total exon candidate scores under the constraint that the model is consistent with a set of given reference models.

An example We first use an example, shown in Figure 3, to explain the basic idea of the algorithm. In this example, every cluster except clusters number 7 and number 11 has some homology in the database. To make our discussion simple, we assume that the homology between R_i's and the corresponding DNA segments is strong. We want to construct a gene model that are the most probable based on the given neural net scores and the homology information. Recall that the neural net scores of the candidates in Figure 3(b) represent the confidence level of a candidate being an exon without any knowledge of database homology. In our early work (Xu *et al.*, 1994b), the most probable gene model is totally determined by these scores. Our new algorithm has extended this to the following strategy: apply homology information whenever possible, otherwise use neural network scores.

Note that each of the 5 reference models in Figure 3(c) is part of a protein possibly from different organisms, and these reference models could be inconsistent. We need to determine, for each exon candidate, which reference model to use while constructing a gene model. Our strategy is to use as few reference models (of highest quality) as possible under the condition that the maximum number of clusters are covered by these reference models. The rationale is that fewer number of reference models implies fewer splicings between exons covered by different reference models, or put it differently, more splicings between exons covered by the same reference model.

Based on the above discussion, a possible optimal gene model for this example could be $\{E_1, E_2, E_3, E_4, E_5, E_6, E_{8,9}, E_{10}, E_{11}\}$, and the reference models are R_5 and R_3, where E_i is from cluster number i, for $i \in \{1,2,3,4,5,6,10,11\}$ and $E_{8,9}$ is a candidate from cluster 8 or 9. Cluster number 7 (marked by "*") is excluded because of its inconsistency with the reference model R_5. Inclusion of a candidate from cluster 11 will increase the total score and does not cause any inconsistency with any reference model. Thus E_{11} is part of the gene model.

The problem Our goal is to define and solve the reference-based multiple gene modeling problem. But first we define a simpler problem, the reference-based partial gene modeling problem, which models a single gene and does not require a gene model to start with an initial exon and to end with a terminal exon.

We first introduce some notations. Let C denote the set of all candidates and $\{R_1, ..., R_k\}$ be all the reference models. We add a special $R_0 = \emptyset$ to the reference model set to simplify the notations. For each $E \in C$ and each R_i, $M(E, R_i)$ represents the portion of R_i that matched E's corresponding protein (by FASTA version 2.0). $M(E, R_i) = \emptyset$ if there is no match.

A *reference-based partial gene modeling* problem is defined as follows. We want to select a subset $\{E_1, ..., E_n\}$ of non-overlapping candidates from C and a mapping R from $\{E_1, ..., E_n\}$ to $\{R_0, ..., R_k\}$ so that the following function is maximized. We assume that $r(E_1) < ... < r(E_n)$.

$$\begin{array}{ll} \text{maximize} & \sum_{i=1}^{n} score_{R(E_i)}(E_i) + \\ & \sum_{i=2}^{n} P(R(E_{i-1}), R(E_i)) \\ & \\ \text{subject to:} & (1)\ E_i \text{ is spliceable with } E_{i+1} \text{ for all} \\ & i \in [1, n-1], \\ & (2)\ R(E_i) = R(E_j) \text{ and } i < j \text{ implies} \\ & r(M(E_i, R(E_i))) < l(M(E_j, R(E_j))). \end{array}$$

where $P(X, Y) = \mathcal{P}$ if $X = Y$ and $X \neq \emptyset$ otherwise $P(X, Y) = 0$, and $\mathcal{P}$ is used to reward splicings between exons with the same reference model.

Informally, we want to select a number of exons $E_1, ..., E_n$ from C and a reference model for each E_i so that the total scores of these exons is maximized. Such a set of exons should satisfy the adjacent-exon spliceability condition, and also the relative order of exons should be kept in their matched portions in the reference model. To encourage to use the same reference model for adjacent exons, we also add a reward factor in the objective function for splicings between adjacent exons using the same reference model. In our current implementation, $\mathcal{P}$ is chosen to be larger than the score of one "average" false candidate.

In the general reference-based gene modeling problem, we also include the information about the start and the end of a gene. For each exon E and a reference model R_i, we define $B(E, R_i) = 1$ if $M(E, R_i)$ is a prefix of R_i's corresponding protein, and $B(E, R_i) = 0$ otherwise. Similarly, $L(E, R_i) = 1$ if $M(E, R_i)$ is a suffix of R_i's corresponding protein, and $L(E, R_i) = 0$ otherwise.

A *reference-based multiple gene modeling* problem is defined as follows. We want to find a list $\{E_1, ..., E_n\}$ of non-overlapping exon candidates from C, a mapping R from $\{E_1, ..., E_n\}$ to $\{R_0, ..., R_k\}$, and a partition of $\{E_1, ..., E_n\}$ into D sublists, $\{E_1^1, ..., E_{e(1)}^1\}$, $\{E_1^2, ..., E_{e(2)}^2\}$, ..., $\{E_1^D, ..., E_{e(D)}^D\}$, so that the following function is maximized, where $e(d)$ represents the last exon of the d^{th} sublist. We assume that $r(E_1) < ... < r(E_n)$.

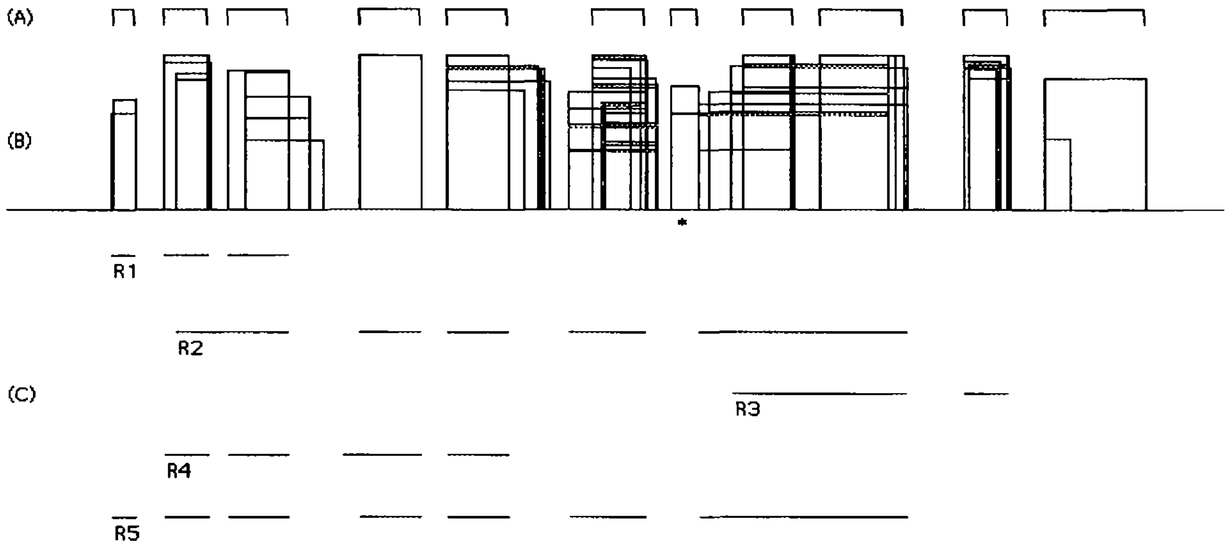

Figure 3: A schematic of a candidate cluster and database homology. The X-axis represents the sequence axis. (a) and (b) represent a set of clusters. Each hollow rectangle in (b) represents an exon candidate. The width of a rectangle represents the size of the candidate and the height represents the "probability" it being a true exon, scored by the neural net. The symbols in (a) indicate the eleven clusters these candidates form. Each R_i in (c) represents a continuous gene segment or reference model. A short line of each R_i represents the matched portion with the corresponding DNA segment, and the broken gaps are only used to indicate the reference model matches a number of DNA segments. There are five reference models in this example.

$$\begin{array}{ll}
\text{maximize} & \sum_{g=1}^{D}(\sum_{i=1}^{e(g)} score_{R(E_i^g)}(E_i^g)+ \\
& \sum_{i=2}^{e(g)} P(R(E_{i-1}^g), R(E_i^g))+ \\
& \mathcal{P}_f(E_1^g) + \mathcal{P}_t(E_{e(g)}^g)) \\
& \\
\text{subject to:} & \text{for all } g \in [1, D], \\
& (1)\ l(E_{e(g+1)}^1) - r(E_{e(g)}^g) > \mathcal{L}, \text{ for} \\
& g < D, \\
& (2)\ L(E_{e(g)}^g, R(E_{e(g)}^g)) = 1, \text{ or} \\
& B(E_1^{g+1}, R(E_1^{g+1})) = 1, \text{ for } g < D, \\
& (3)\ B(E_p^g, R(E_p^g)) = 1 \text{ implies } p = 1, \\
& \text{and } L(E_q^g, R(E_q^g)) = 1 \text{ implies } q = e(g), \\
& (4)\ E_i^g \text{ is spliceable with } E_{i+1}^g \text{ for all} \\
& i \in [1, e(g) - 1], \\
& (5)\ R(E_i^g) = R(E_j^g) \text{ and } i < j \text{ imply} \\
& r(M(E_i^g, R(E_i^g))) < l(M(E_j^g, R(E_j^g))).
\end{array}$$

where penalty factor $\mathcal{P}_f(E)$ is a fixed negative value if E is not an initial exon otherwise it is zero, similarly $\mathcal{P}_t(E)$ is a fixed negative value if E is not a terminal exon, otherwise it is zero, and $\mathcal{L}$ is the minimum distance between two genes ($\mathcal{L} = 1000$ in our current implementation). Note that D is not a predetermined value, but a part of the optimal solution. In the following, we say $\{E_1, ..., E_n\}$ form a *gene model* under mapping R and the partition given above if conditions (1) - (5) hold.

The main difference between the general gene modeling problem and the partial gene modeling problem is the treatment of the start and the end of a gene. By utilizing the information about the start/end of a gene from the database search, we are able to deal with multiple genes in a DNA sequence. By requiring conditions (2) and (3), a list of exons will be divided into two genes if and only if there is a start or end of a gene based on the database search information. To model a complete gene, we penalize gene models missing the translation start in its first exon or the translation stop in its last exon by using the two penalty factors $\mathcal{P}_f$ and $\mathcal{P}_t$.

The algorithm We now present a dynamic programming algorithm to solve the reference-based multiple gene modeling problem defined above. The partial gene modeling problem can be solved as a special case.

The input to the algorithm is a set of exon candidates sorted in the increasing order of their right boundaries. The algorithm scans through the exon candidates from left to right and constructs optimal solutions for the subset containing all candidates from the first to the current one, based on optimal solutions for previous subsets. We call these solutions the optimal solutions for *this* candidate. For each candidate, at most $k+1$ optimal solutions are constructed, i.e., at most one for each of the $k+1$ reference models $\{R_0, R_1, ..., R_k\}$. To construct an optimal solution for the candidate and a reference model, the algorithm tries to splice this candidate with all the previous candidates, and to find the one giving the highest total score with respect to the reference model. $\mathcal{P}$ is rewarded to each splicing between candidates using the same reference model. Conditions (1) - (5) are checked while trying to splice two candidates. The algorithm stops when all candidates are processed. The model having the highest total score is output as the solution. As we give more details in the following, it can be seen that this output corresponds to the solution to the reference-based multiple gene modeling problem.

Let $\{E_1, ..., E_{||C||}\}$ be the set of given exon candidates sorted in the increasing order of $r(E_i)$'s. We use $model(E_i, R_j)$ to denote the value of the objective function of the optimal gene model, for the subset $\{E_1, ..., E_i\}$, that ends with E_i using reference model R_j. By definition,

$$\max_{i \in [1,n], j \in [0,k]} model(E_i, R_j)$$

corresponds to the solution of the reference-based multiple gene modeling problem.

To calculate $model(E_i, R_j)$, the following recurrences can be proved using inductive proofs, which we omit here. To simplify the recurrences, we introduce another quantity $model_0(E_i, R_i)$, which is defined the same as $model(E_i, R_j)$ except that the $\mathcal{P}_t()$ term (in the objective function) is ignored for the last sublist in the partition of $\{E_1, ..., E_i\}$.

There are two cases we need to consider in calculating both $model(E_i, R_j)$ and $model_0(E_i, R_j)$.

Case # 1: When E_i is the first exon of a gene,

$$model(E_i, R_j) = \max_{p \in [1, i-1], q \in [0,k]} \{model(E_i, R_j), model(E_p, R_q) + score_{R_j}(E_i) + \mathcal{P}_f(E_i) + \mathcal{P}_t(E_i), \text{when (1) } L(E_p, R_q) = 1 \text{ or } B(E_i, R_j) = 1, \text{(2) } l(E_i) - r(E_p) > \mathcal{L}.\}$$

and

$$model_0(E_i, R_j) = \max_{p \in [1, i-1], q \in [0,k]} \{model_0(E_i, R_j), model(E_p, R_q) + score_{R_j}(E_i) + \mathcal{P}_f(E_i), \text{when (1) } L(E_p, R_q) = 1 \text{ or } B(E_i, R_j) = 1, \text{(2) } l(E_i) - r(E_p) > \mathcal{L}.\}$$

Case # 2: When E_i is not the first exon of a gene,

$$model(E_i, R_j) = \max_{p \in [1, i-1], q \in [0,k]} \{model(E_i, R_j), model_0(E_p, R_q) + score_{R_j}(E_i) + P(R_q, R_j) + \mathcal{P}_t(E_i), \text{when (1) } E_p \text{ is spliceable to } E_i, \text{(2) } L(E_p, R_q) = 0 \text{ and } B(E_i, R_j) = 0, \text{(3) } r(M(E_p, R_q)) < l(M(E_i, R_j)) \text{ if } R_q = R_j.\ \}$$

and

$$model_0(E_i, R_j) = \max_{p \in [1, i-1], q \in [0,k]} \{model_0(E_i, R_j), model_0(E_p, R_q) + score_{R_j}(E_i) + P(R_q, R_j), \text{when (1) } E_p \text{ is spliceable to } E_i, \text{(2) } L(E_p, R_q) = 0 \text{ and } B(E_i, R_j) = 0, \text{(3) } r(M(E_p, R_q)) < l(M(E_i, R_j)) \text{ if } R_q = R_j.\ \}$$

In the general case, $model(E_i, R_j)$ equals the highest value of the two cases. The same is true for $model_0(E_i, R_j)$. The initial values of $model(E_i, R_j)$ and $model_0(E_i, R_j)$ are defined as follows.

$$model(E_i, R_j) = score_{R_j}(E_i) + \mathcal{P}_f(E_i) + \mathcal{P}_t(E_i),$$
$$model_0(E_i, R_j) = score_{R_j}(E_i) + \mathcal{P}_f(E_i).$$

Using these recurrences, $model(E_i, R_j)$ can be calculated in the increasing order of i for all $j \in [0, k]$. It is easy to see that these quantities can be calculated in $O(||C||^2 k^2)$ time and $O(||C||k)$ space. To recover the set of candidates that achieves $\max_{i,j} model(E_i, R_j)$ some simple bookkeeping needs to be done, which can be accomplished in $O(||C||^2 k^2)$ time and $O(||C||k)$ space. We omit further details.

Figure 4 shows an example of reference-based gene modeling. An interesting thing is that the best reference models for the first two exon clusters are not Human but Mouse proteins while the best reference models for all the other clusters are Human proteins. Database search results show that the matches with Mouse proteins are 100% but only 96.6% with Human proteins for both clusters.

Results and Discussions

We have presented a framework for using homology information to guide gene structure predictions. The framework uses exons predicted by content-statistics based methods as basic building blocks and database homology information as references in constructing gene models. The mathematical model we used for the gene modeling problem rewards any application of homology information in the gene modeling process as an attempt to maximally use the known homology. Minimal "inconsistency" between predicted gene structures and database homology is the basic rule used in this gene modeling framework.

Preliminary tests have been done to test the effectiveness of applying homology information in gene modelings. Based on the test results on 59 genes, we conclude that (1) homology information has helped improve the prediction accuracy of exon boundaries in the (single) exon re-evaluation step, (2) homologs corresponding to a series of exons has helped greatly in eliminating false exons, and also has further helped improve the exon boundary predictions, and (3) the availability of information about the start/end of a protein makes it feasible to do multiple gene modeling.

The following table lists the test result on 59 single genes. While conducting these tests, the gene tested

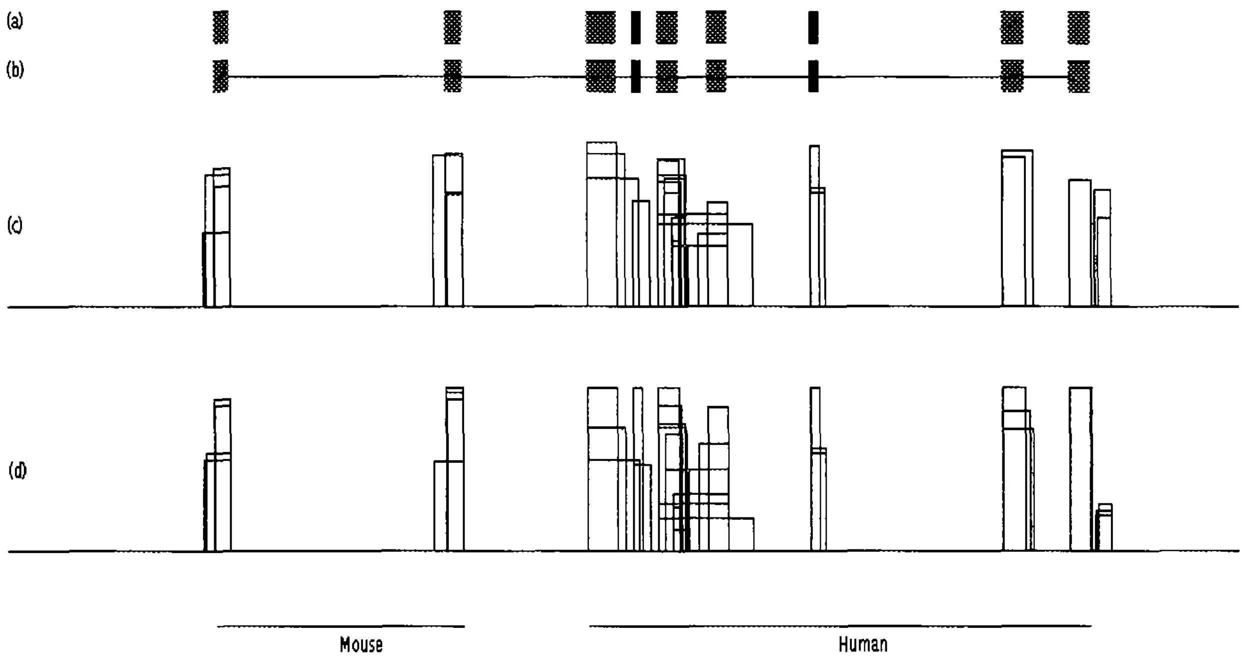

Figure 4: Reference-based gene modeling. The X-axis represents the sequence axis. (a) Each solid bar represents an exon. (b) The predicted exons and gene structures. The lines between solid bars represent splicings between exons of the same gene. (c) The neural net predictions of exon candidates. The Y-axis represents the axis of exon scores. (d) The re-scored exon candidates using homology information. The lines in the bottom indicate the reference models used in gene segment construction. The sequence is HUMIFNRF1A.

on is removed from the database. The first column in the table gives the sequence name and the number of exons in this sequence. The **Exons** columns give the prediction performance in terms of number of exons that are "correctly" and falsely predicted; We list the number of missing exons and false exons if there is any (a blank means no missing and false exons). Similarly the **Edges** columns give the number of exon boundaries that are incorrectly predicted.

From Table I, we can see that the reference-based gene modeling program has improved the performance of the GRAIL gene prediction system. This program has reduced the number of false exons, missing exons and off-edges from 22 to 3, from 27 to 19, and from 57 to 17, respectively. There are a few cases where the GRAIL gene prediction subsystem misses more exons than the reference-based gene modeling program does. The reason for this is that these missed exons are predicted by the GRAIL exon prediction program but not included in the gene models due to the incorrect exon boundary predictions and the enforcement of spliceability condition. The exon re-evaluation program corrected these exon boundary predictions based on the database search results, and hence these exons are included in the reference-based gene modeling.

Tests are also done on a number of multiple gene sequences. Figure 5 shows one example of multiple gene modeling on a DNA sequence artificially formed by appending three sequences HUMCYPIIE, HUMRASH, HUMACTGA.

While we are planning to conduct more extensive tests on the algorithm, the preliminary test results have pointed to possible directions for further improvement on the algorithm. We mention a few here. While our current reference-based gene modeling framework allows effectively removing falsely predicted exons and correcting exon boundary predictions, it does not support mechanism to generate exons missed by the neural network exon predictor. Some work is currently under way to develop effective methods to generate those missed exons based on the information provided by database search. We are also working on schemes to include even more biological constraints in the multiple gene modeling process, for example, indications of promoters, CpG islands, PolyA sites, etc.

In conclusion, we have generalized our previous algorithm for single gene model constructions and developed a reference-based multiple gene modeling framework. This framework attempts to maximally use the available homology information from existing databases in constructing gene models. By combining content-statistics based pattern recognition methods

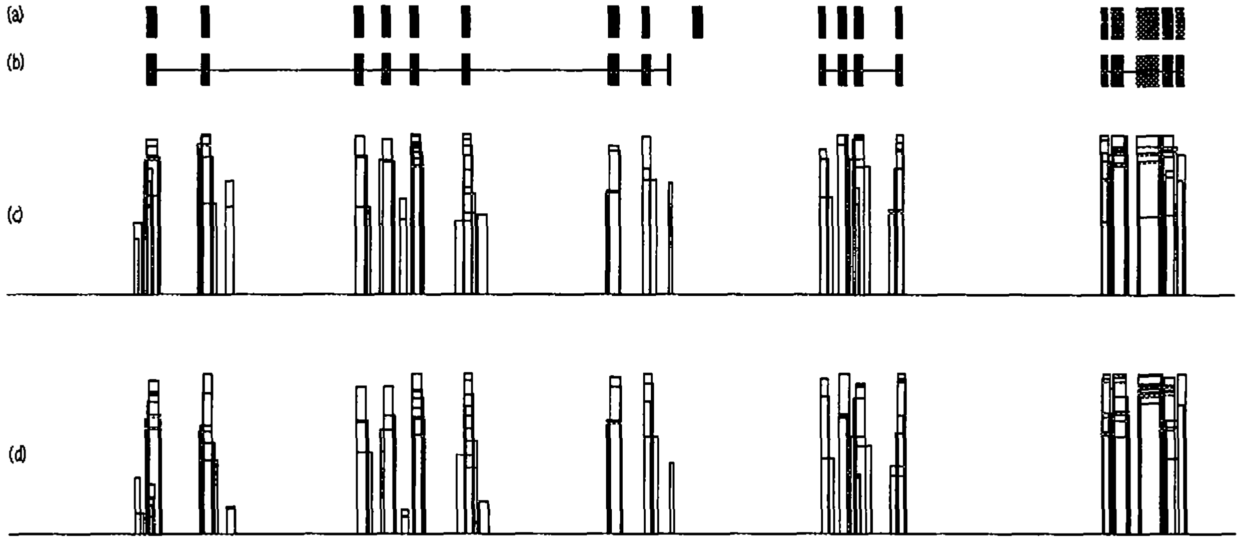

Figure 5: Multiple gene modeling. The X-axis represents the sequence axis. The DNA sequence is artificially formed by appending HUMCYPIIE, HUMRASH and HUMACTGA. (a) Each solid bar represents an actual exon. Exons # 1 through # 9 are the exons of HUMCYPIIE, exons # 10 to # 13 are the exons of HUMRASH, and exons # 14 to # 18 are the exons of HUMACTGA. (b) The predicted exons and gene structures by the reference-based gene modeling program. The lines between solid bars represent splicings between exons of the same gene. (c) The neural net predictions of exon candidates. The Y-axis represents the axis of exon scores. (d) The re-scored exon candidates using homology information.

and homology information, this reference-based gene prediction program should provide molecular biologists a more powerful and convenient tool in gene identification.

Acknowledgements

This research was supported by the United States Department of Energy, under contract DE-AC05-84OR21400 with Lockheed Martin Energy Systems, Inc.

References

M. Borodovsky, Yu. Sprizhitskii, E. Golovanov and A. Aleksandov, "Statistical Patterns in the Primary Structures of Functional Regions in E. Coli.", *Molekulyainaya Biologiya*, Vol. 20, pp. 1390 - 1398, 1986.

M. Burset and R. Guigo, "Evaluation of Gene Structure Prediction Programs", *Preprint*, 1996.

J. M. Claverie, I. Sauvaget and L. Bougueleret, "k-tuple Frequency Analysis: From Intron/Exon Discrimination to T-cell Epitope Mapping", *Methods in Enzymology*, Vol. 183, pp. 237 - 252, 1990.

S. Dong and D. B. Searls, "Gene Structure Prediction by Linguistic Methods", *Genomics*, Vol. 23, pp. 540 - 551, 1994.

C. A. Fields and C. A. Soderlund, "GM: A Practical Tool for Automating DNA Sequence Analysis", *Comput. Appl. Biol. Sci.*, Vol. 6, pp. 263 - 270.

M. S. Gelfand, "Computer prediction of Exon-Intron Structure of Mammalian pre-mRNAs", *Nucleic Acids Res.*, Vol. 18, pp. 5865 - 5869, 1990.

R. Guigo, S. Knudsen, N. Drake and T. Smith, "Prediction of Gene Structure", *J. Mol. Biol.*, Vol. 226, pp. 141 - 157, 1992.

G. B. Hutchinson and M. R. Hayden, "The prediction of Exons Through an Analysis of Spliceable Open Reading Frames", *Nucleic Acids Res.*, Vol. 20, pp. 3453 - 3462, 1992.

A. Krogh, I. S. Mian, and D. Haussler, "A Hidden Markov Model That Finds Genes in *E. Coli* DNA", *Preprint*, 1994.

V. R. Pearson and D. J. Lipman, "Improved Tools for Biological Sequence Comparison", *Proc. Natl. Acad. Sci. USA*, Vol. 85, pp. 2444 - 2448, 1988.

E. E. Snyder and G. D. Stormo, "Identification of Coding Regions in Genomic DNA Sequences: An Application of Dynamic Programming and Neural Networks", *Nucleic Acids Res.*, Vol. 21, pp. 607 - 613, 1993.

E. E. Snyder and G. D. Stormo, "Identification of Protein Coding Regions in Genomic DNA", *J. Mol. Biol.*, Vol. 248, pp. 1 - 18, 1995.

E. C. Uberbacher and R. J. Mural, "Locating Protein-coding Regions in Human DNA Sequences by a Multiple Sensors-neural Network Approach", *Proc. Natl. Acad. Sci. USA*, Vol. 88, pp. 11261 - 11265, 1991.

E. C. Uberbacher, Y. Xu and R. J. Mural, "Discovering and Understanding Genes in Human DNA Sequence using GRAIL", *Methods in Enzymology*, in press, 1996.

Y. Xu, J. R. Einstein, R. J. Mural, M. Shah and E. C. Uberbacher, "An Improved System for Exon Recognition and Gene Modeling in Human DNA Sequences", *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, Altman, Brutlag, Karp, Lathrop and Searls, Eds., pp. 376 - 384, 1994a.

Y. Xu, R. Mural and E. C. Uberbacher, "Constructing Gene Models from a Set of Accurately-predicted Exons: An Application of Dynamic Programming", *Computer Applications in the Biosciences*, Vol. 10, pp. 613 - 623, 1994b.

Table I: Test results

	GRAIL		GRAIL with db search	
Sequence	**Exons**	**Edges**	**Exons**	**Edges**
HUMALIFA (3)	1 false	1 off		
HUMALPHA (11)		1 off		
HUMAPOE4 (3)	1 miss	1 off	1 miss	
HUMAPRTA (5)		2 off		
HUMATPGG (22)	1 miss	1 off	1 miss	
HUMATPSYB (10)		1 off		
HUMBMYH7 (38)	3 miss 1 false	2 off	3 miss	2 off
HUMCACY (2)	1 miss			
HUMCAPG (5)		1 off		
HUMCYPIIE (9)	1 miss 1 false	1 off	1 miss	
HUMEDHB17 (6)				
HUMEF1A (7)				
HUMFESFP (18)	1 miss	1 off		1 off
HUMGLUT4B (11)				
HUMGOS24 (4)				
HUMHAP (4)				
HUMHOX4A (2)		1 off		
HUMHSD3BA (3)				
HUMHSKPQZ7 (6)				
HUMHSP90B (11)				
HUMIBP3 (4)	1 false	2 off		
HUMIFNRF1A (9)		2 off		
HUMIL1B (6)				
HUMIL2 (4)				
HUMIL4A (4)	1 false	1 off		
HUMIL5 (4)	1 miss		1 miss	
HUMKER18 (7)	1 miss	2 off		1 off
HUMMETIII (3)	1 miss		1 miss	
HUMMHB27D (7)		1 off		

HUMMHCP42 (10)	1 false	3 off		
HUMMKXX (4)	1 miss	1 off	1 miss	
HUMMLHDC (12)	2 miss 5 false	8 off	2 miss 2 false	3 off
HUMMRP8A (2)				
HUMMYCC (2)	1 false			
HUMOSTP (6)	2 miss	1 off	2 miss	
HUMPALD (4)				
HUMPCNA (6)				
HUMPDS02 (5)	1 miss	2 off	1 miss	2 off
HUMPF4V1A (3)				
HUMPGAMMG (3)				
HUMPIM1A (6)				
HUMPNMTA (3)		1 off		
HUMPOMC (2)				
HUMPRF1A (2)		2 off		
HUMPSAA (5)				
HUMSERG (2)	1 miss 2 false	2 off		1 off
HUMSFTP1A (4)	1 miss 1 false	1 off	1 miss 1 false	
HUMSODA (10)	1 false	1 off		
HUMSPERSYN (8)		2 off		
HUMTBB5 (4)	1 false			
HUMTCRBRA (2)	1 miss		1 miss	
HUMTHB (14)	1 miss	2 off	1 miss	
HUMTNC2 (5)	1 miss	2 off	1 miss	1 off
HUMTNFBA (3)				
HUMTROC (6)	1 false	3 off		
HUMTRPM2A (3)	1 miss	1 off	1 miss	
HUMTRPY1B (5)		2 off		2 off
HUMUBILP (4)	1 miss 2 false	2 off	1 miss	1 off
HUMVITBP (12)	3 miss 2 false	3 off	3 miss	3 off