Inferring Gene Structures in Genomic Sequences Using Pattern Recognition and Expressed Sequence Tags

(Extended Abstract)

Ying Xu, Richard J. Mural†, and Edward C. Uberbacher
Computer Science and Mathematics Division and †Life Sciences Division
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6364, USA
Email address: {xyn,m9l,ube}@ornl.gov
Tel. 423-574-7263. Fax: 423-574-7860

Abstract

Computational methods for gene identification in genomic sequences typically have two phases: coding region prediction and gene parsing. While there are many effective methods for predicting coding regions (*exons*), parsing the predicted exons into proper gene structures, to a large extent, remains an unsolved problem. This paper presents an algorithm for inferring gene structures from predicted exon candidates, based on Expressed Sequence Tags (*ESTs*) and biological intuition/rules. The algorithm first finds all the related ESTs in the EST database (dbEST) for each predicted exon, and infers the boundaries of one or a series of genes based on the available EST information and biological rules. Then it constructs gene models within each pair of gene boundaries, that are most consistent with the EST information. By exploiting EST information and biological rules, the algorithm can (1) model complicated multiple gene structures, including embedded genes, (2) identify falsely-predicted exons and locate missed exons, and (3) make more accurate exon boundary predictions. The algorithm has been implemented and tested on long genomic sequences with a number of genes. Test results show that very accurate (predicted) gene models can be expected when related ESTs exist for the predicted exons.

Keywords: multiple gene structure prediction, expressed sequence tags, sequence comparison and analysis, pattern recognition, and dynamic programming.

1. Introduction

Identification of genes in anonymous DNA sequences involves predicting exons and parsing the predicted exons into gene models. Typically, exons are predicted based on content statistics measuring the positional or compositional biases imposed on the coding DNA sequence by the genetic code. Using such statistical information, a number of computer programs have been developed for coding exon identification (Uberbacher and Mural, 1991; Guigo *et al.*, 1992; Snyder and Stormo, 1993; Gelfand and Royberg, 1993; Dong and Searls, 1994; Solovyev *et al.*, 1994; Xu *et al.*, 1996; Kulp, 1996; Salzberg *et al.*, 1996). While the effectiveness of these exon recognition programs has been shown through actual applications (Burset and Guigo, 1996), parsing the predicted exons into proper gene models remains an unsolved problem for genomic sequences with multiple genes. This is mainly due to the lack of effective methods for separating one gene from another. In a typical gene identification algorithm/program, predicted exons are parsed into a single gene structure that maximizes the total confidence-level of the predicted exons while maintaining the adjacent-exon spliceability condition (Xu, Mural and Uberbacher, 1994).

To analyze a long genomic sequence with multiple genes using such a gene identification program, a molecular biologist may need to first "manually" partition the genomic sequence into "single-gene" regions on both strands of the DNA based on the predicted exons, information gathered from EST/cDNA/protein databases, and his/her biological intuition. Such a process is very time-consuming and difficult when the genomic sequence is long (e.g., a few hundred thousand bases long) and has complicated gene structures. With the expectation of about 2 million bases of DNA being sequenced and released daily from the major genome centers from 1997 to 2003, more automatic methods for multiple gene structure identification will be necessary.

We have developed a multiple-gene structure identification algorithm for large-scale genomic sequences. The algorithm automatically identifies the boundaries of one or a series of genes (depending on the available information) on both strands of a genomic sequence based on GRAIL exon predictions (Uberbacher and Mural, 1991; Xu *et al.*, 1996), available EST information and biological rules/intuition. Then it constructs gene models within each pair of boundaries, that are most consistent with the exons predicted by GRAIL, and the related EST information.

ESTs have been used for gene discovery in genomic sequences for a number of years (Costanzo *et al.*, 1983; Adams *et al.*, 1991; Wilcox *et al.*, 1991; Houlgatte *et al.*, 1995; Adams *et al.*, 1995). With the significant

effort which has gone into EST sequencing in the past few years, it is estimated that about 60% of Human genes are partially represented by ESTs (Aaronson *et al.*, 1996). Though ESTs alone often are insufficient to make complete gene structure predictions due to their incomplete coverage, they can provide useful information to help determine the 3′ end of a gene, the minimal extent of a gene, to help identify falsely-predicted exons, identify and locate missed exons, and correct predicted exon boundaries.

Practical biological intuition/rules also add additional power in modeling gene structures. In our gene modeling process, we assume that (1) coding regions in opposite strands do not overlap, and (2) overlapping genes can only appear in opposite strands, and one has to be embedded inside another. By combining these heuristics with the EST information, the algorithm can model multiple genes with embedded gene structures.

The algorithm has four main steps: (1) Gene Boundary Identification – related ESTs from the dbEST database (Boguski, Lowe and Tolstoshev, 1993) are collected for every predicted exon candidate using the BLAST sequence alignment program (Altschul *et al.*, 1990), and potential gene boundaries are marked based on the identified 3′ ESTs, the minimal extent of a gene which is determined by overlapping ESTs, and the biological rules; (2) Exon Candidate Re-evaluation – GRAIL-predicted exons are rescored based on the EST information; (3) Gene Structure Prediction – a number of long stretches of "high-scoring" ESTs are selected as reference models and "optimum" gene models are built with respect to these reference models within each pair of gene boundaries; (4) Post Processing – missed exons and 5′ and 3′ untranslated regions are located based on the underlying reference models, and are added into the predicted gene models.

The algorithm has been implemented to analyze Human DNA sequences. While extensive tests are currently under way, preliminary test results have shown that (a) ESTs in general help make more accurate exon predictions, and (b) very accurate gene models can be expected when related EST information (not necessarily with perfect matches with the predicted exons) exist.

2. Reference Model Construction

This section describes the information extracted from the ESTs aligned with the predicted exons and how biological rules/intuition are implemented in the gene modeling system. The goal is to maximally use the available EST information to build a reference model for gene structure predictions from the predicted exons.

2.1. Information extraction from ESTs

As the first step of the gene modeling process, the algorithm finds all ESTs from the dbEST database (release of Feb. 1997) which match each exon candidate predicted by GRAIL II (version 1.3), using the BLAST search program (version 1.4.9). The search results are a list of alignments between exon candidates and EST segments. For each alignment, the name of the EST, the positions of the matched portion in the genomic sequence and in the EST, the alignment score and identity-match ratio, and the 3′/5′ label of the EST are recorded. Only alignments with alignment scores and identity-match ratios above specified thresholds are kept. A simple post-processing procedure is used to merge the broken segments of an alignment (caused by the gapless nature of BLAST alignments) between an exon candidate and an EST. For an exon candidate, the number of matching ESTs may range from zero to a few hundred. From this list of alignments, the following information is extracted, and used to build a reference model for the gene of interest.

2.1.1. Calculation of exon boundaries

Typically, GRAIL predicts a cluster of overlapping exon candidates with different boundary predictions for each presumed exon (see Figure 1). The matched ESTs can be used to better determine the boundaries of each predicted exon. Consider a predicted exon cluster C. Let $\{L_1, ..., L_n\}$ be the list of all left boundaries of C's aligned segments in the genomic sequence, and $s(L_i)$ be the summed scores of all the alignments that have L_i as the left boundary. For each L_i, we calculate a quantity[1]:

$$Q(C, L, L_i) = \frac{s(L_i)}{\max_k\{s(L_k)\}},$$

which ranges from 0 to 1, with 0 and 1 representing the lowest and highest confidence level of L_i being the correct boundary, respectively. Similarly for each right boundary R_j of C, we calculate

$$Q(C, R, R_j) = \frac{s(R_j)}{\max_k\{s(R_k)\}}.$$

2.1.2. Re-evaluation of predicted exons

GRAIL-predicted exons are rescored based on the matched ESTs. We have applied three practical rules to rescore predicted exons: (1) scores for exons with EST matches are increased; (2) exons that are inconsistent with the EST information are labeled as false exons; (3) scores for the rest of the exons are unchanged.

For each exon cluster C with EST matches, the algorithm rescores an exon by combing the exon's GRAIL score, EST alignment score and the scores for the left and right boundaries as defined above. More specifically, for each exon E of C, E's new score with respect to an EST *reference model* R is given by

[1] In the actual implementation, we have used a smoothed version of this function using a 11-base window.

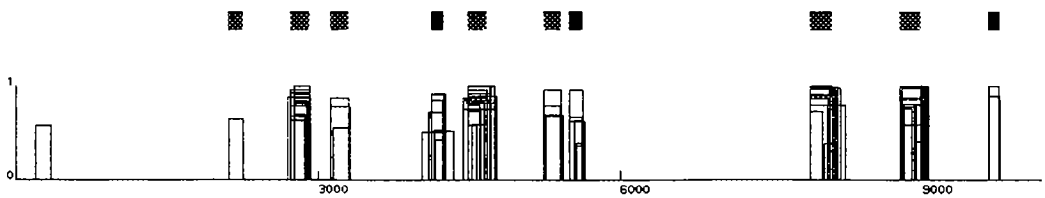

Figure 1: GRAIL exon predictions. The X-axis is the sequence axis. The solid bars on the top represent the positions of the actual exons, and the hollow rectangles represent the predicted exon candidates with different edge assumptions. The Y-axis represents the confidence-level of the predicted exons.

$$score(E,R) = w_1 \times score_G(E) + w_2 \times score_A(E,R) \times \mathcal{Q}(C,L,l(E)) \times \mathcal{Q}(C,R,r(E)),$$

where $score_G(E)$ is E's GRAIL predicted score, $score_A(E,R)$ is the alignment score between E and R, w_1 and w_2 are two empirically determined scaling factors, and $l(E)$ and $r(E)$ represent E's left and right boundary positions, respectively.

A falsely-predicted exon can be identified when exons to both its left and right match the same ESTs while this exon itself doesn't match any of them. To avoid falsely labeling/removing true exons in such a way due to the poor quality of the ESTs (e.g. ESTs with high sequencing errors) or use of ESTs from other species, the algorithm assigns a negative score to such exons, which is inversely proportional to the number of matched ESTs to its neighboring exons and their alignment scores. Removal of such an exon will be determined conditionally in the gene modeling process using more global information.

2.1.3. Determination of a gene's minimal extent

Two ESTs are said to be overlapping if their matched portions with exons in the genomic sequence overlap (in the same DNA strand). A cluster of all overlapping ESTs determines the minimal extent of a gene, called a *gene segment*. Figure 2 shows an example of gene segments. Two gene segments are merged into one if they contain ESTs with the same clone names (they correspond to the 5′ and 3′ ends of the same clone, respectively). Two exons are said to be *related* if they belong to the same gene segment. In the gene modeling process, related exons can only belong to the same gene model.

2.1.4. Determination of a gene's 3′ end

Each gene segment (the minimal extent of a gene) may contain both 5′ and 3′ ESTs. While 5′ EST may not provide much information regarding to the 5′ end of a gene, 3′ ESTs, in general, are indications of a gene's 3′ end. The algorithm labels a gene segment as 5′ gene segment if it contains only 5′ ESTs; And it labels a gene segment as a 3′ gene segment if it contains any 3′ EST and possibly 5′ ESTs. In the gene modeling process, a 3′ gene segment represents the end of a gene.

2.2. Implementation of biological rules

One of the key problems in the gene modeling process is to identify the boundaries of one or a series of genes depending on the available information. To help achieve this, we have implemented the biological rule that coding regions in opposite strands do not overlap, as follows.

Let $coding_f(i)$ and $coding_r(i)$ represent the highest exon score by GRAIL at position i in the forward and reverse strand, respectively. We define a *strand assignment* function $\mathcal{S}()$ as in (1), where W is one-half of the measuring window size ($W = 500$) and ϵ is a small positive number used to prevent the denominator from having a value zero. The coding strand is forward strand if $\mathcal{S}(i) \geq 0$, otherwise is reverse strand. A predicted exon in the opposite strand of the coding strand is considered as a false exon. Figure 3 shows a plot of this function for a genomic sequence with several genes.

The strand-assignment function $\mathcal{S}()$ partitions a genomic sequence into coding domains alternately in the forward and reverse strand. Based on these coding domains, the identified minimal gene extent information and 3′ gene segments, we can identify the boundaries of one or a series of genes, as follows.

Consider the coding domains in the forward (similarly reverse) strand. Two adjacent coding domains are merged into one if a gene segment straddles them (we assume that overlapping genes can be only in opposite strands, and hence a minimal gene extent can be a part of only one gene). After the merge-step is done, a coding domain is split at the end of every 3′ gene segment it contains, to form a number of new (smaller) coding domains. The boundaries of these coding domains are considered as boundaries of one or possibly a number of genes. The gene modeling procedure builds gene models within each pair of boundaries. Figure 4 shows an example of gene boundary identification.

3. Reference-based Gene Prediction

This section presents the gene modeling algorithm for constructing gene models within each pair of gene boundaries that are most consistent with the available EST information. We first review some terminology introduced in our previous work (Xu, Mural and

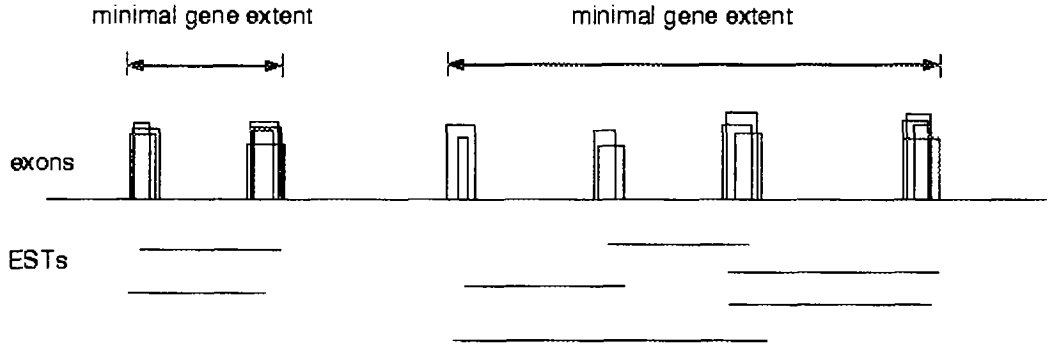

Figure 2: Minimal gene extent. The X-axis is the sequence axis. The hollow rectangles represent GRAIL-predicted exons with their height representing the prediction confidence level. The starting and ending positions of each EST-representing line segment represent its leftmost and rightmost matched positions in the genomic sequence.

$$\mathcal{S}(i) = \frac{\sum_{j=-W}^{W} coding_f(i+j)(1-|j|/W)+\epsilon}{\sum_{j=-W}^{W}(coding_f(i+j)+coding_r(i+j))(1-|j|/W)+2\epsilon} - \frac{1}{2} \quad (1)$$

Uberbacher, 1994; Xu and Uberbacher, 1996). Exons are predicted by GRAIL with a fixed *type* $\in$ {initial, internal, terminal}, and a fixed *translation frame* $\in \{0,1,2\}$. In a single-gene modeling process, an exon E is said to *spliceable* to exon E' if

1. E is a non-terminal exon, and E' is a non-initial exon,
2. $l(E') - r(E) \geq \mathcal{I}$,
3. $f(E') = (l(E') - r(E) - 1 + f(E)$ modulo 3,
4. no in-frame stop is formed at the joint point when appending E and E',

where $f(E)$ represents E's translation frame, and $\mathcal{I}$ represents the minimal intron size and $\mathcal{I} = 60$ in GRAIL. We can extend this spliceability condition to the multiple gene modeling process as follows. Two exons are *spliceable* if they are spliceable in the single-gene modeling process, and they are either related or if E belongs to a 3′ gene segment then E' belongs to the same gene segment.

A list of non-overlapping exons $\{E_1, ..., E_k\}$ form a *gene* model if (a) E_i is spliceable to E_{i+1} for all $i \in [1, k-1]$, and (b) E_1 is an initial exon, and E_k is a terminal exon. More generally, $\{E_1, ..., E_k\}$ form a *partial* gene model if condition (a) holds.

The goal of gene modeling is to parse the predicted exons into a series of gene models that are most consistent with the available EST information and the GRAIL-predicted exons. This problem can be modeled as a optimization problem with the goal of maximizing the total exon scores in the gene models. To encourage using the long stretches of ESTs as reference models in the gene modeling process and hence to provide necessary information for possible missing exon identification, we reward using the same EST as reference models for adjacent exons in a gene model.

A *reference-based multiple gene modeling problem* is defined as follows. Given are a set of N predicted exons and a list of M EST-reference models $\{R_1, ..., R_M\}$. Each exon E has a score $score(E, R)$ with respect to each of its EST reference model R. For the simplicity of discussion, we define $score(E, \emptyset) = score_G(E)$ and always use R_0 to represent $\emptyset$ as a special reference model. The goal is to select a list $\{E_1, ..., E_n\}$ of nonoverlapping exon candidates from the given exon set, a mapping $\mathcal{M}$ from $\{E_1, ..., E_n\}$ to the (extended) EST reference model list $\{R_0, R_1, ..., R_M\}$, and a partition of $\{E_1, ..., E_n\}$ into D (not predetermined) sublists $\{E_1^1, ..., E_{n_1}^1\}, ..., \{E_1^D, ..., E_{n_D}^D\}$ (corresponding to D (partial) gene models) in such a way that the following function[2] is maximized,

$$\text{maximize} \quad \sum_{g=1}^{D}(\sum_{i=1}^{n_g} score(E_i^g, \mathcal{M}(E_i^g)) + \sum_{i=2}^{n_g} link(\mathcal{M}(E_{i-1}^g), \mathcal{M}(E_i^g)) + \mathcal{P}_f(E_1^g) + \mathcal{P}_t(E_{n_g}^g)) \quad (P)$$

$$\text{subject to:} \quad (1)\ l(E_1^{g+1}) - r(E_{n_g}^g) \geq \mathcal{L}, \text{ for } g < D,$$
$$(2)\ E_{n_g}^g \text{ and } E_1^{g+1} \text{ are not related, for } g < D,$$
$$(3)\ E_i^g \text{ is spliceable to } E_{i+1}^g, \text{ for all } i \in [1, n_g - 1] \text{ and } g \leq D$$

where $link(X, Y)$ is a reward factor when $X = Y$ and $X \neq R_0$, and is zero otherwise, $\mathcal{P}_f()$ and $\mathcal{P}_t()$ are two

[2]One possible variation of this optimization problem is to relax the hard constraint that adjacent exons have to be spliceable, instead to add a penalty factor in the objective function for cases where the spliceability condition is violated. By doing so, the algorithm will not remove high-scoring exons from a gene model simply because it is not spliceable to other exons, probably caused by other missing exons.

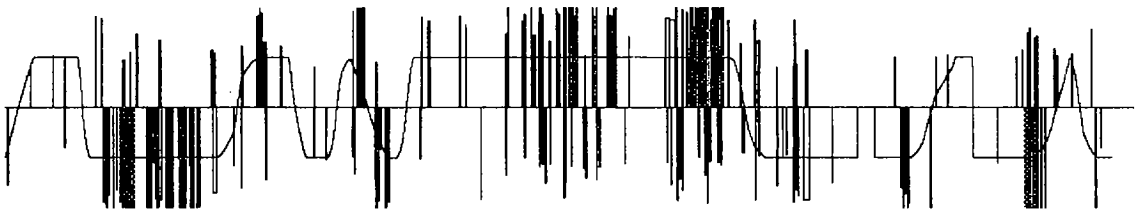

Figure 3: Coding strand assignment on a ($\geq$) 250 kb sequence. Each rectangle represents a predicted exon with width representing the size of the exon and the height representing the predicted score. The curve is the strand-assignment function.

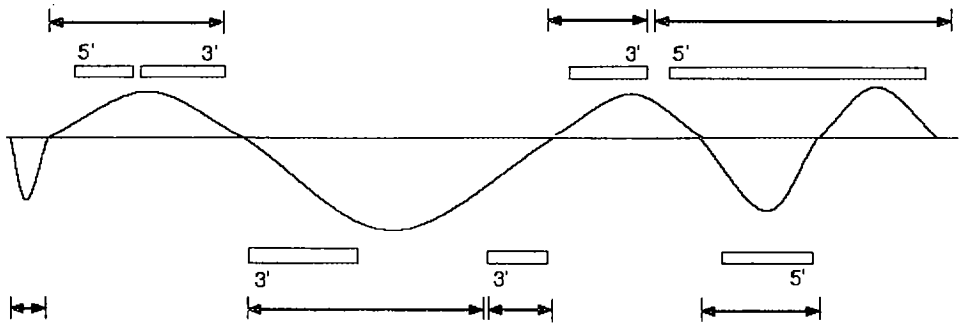

Figure 4: Gene boundary identification. The curve in the middle is a plot of the strand-assignment function. Each hollow rectangle represents minimal gene extent with its type (3′ or 5′) labeled. The bounded regions in the top and the bottom represent gene boundaries in both the forward and reverse strands, respectively. Multiple gene models are considered within each gene boundaries.

penalty factors for a gene missing the initial or terminal exon, respectively, and $\mathcal{L}$ is the minimum distance between two genes ($\mathcal{L} = 1000$ in our current implementation).

3.1. Gene modeling by dynamic programming

This subsection presents a fast dynamic programming algorithm for solving the reference-based gene modeling problem defined above. This algorithm improves the computational efficiency of our previous algorithm for solving the protein-based gene modeling problem (Xu and Uberbacher, 1996).

Let $\{E_1, ..., E_N\}$ be the set of all exon candidates (within a pair of gene boundaries) sorted in the increasing order of their right boundaries. The core of the algorithm is a set of recurrences that relate the optimum (partial) solutions ended at an exon E_i to the optimum partial solutions ended at some exons to the left of E_i. We define $model(E_i, R_j)$ to be the optimum value of the objective function (P) under the constraint that the rightmost exon in the optimum gene models is E_i and E_i's reference model is R_j. By definition,

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

corresponds to the optimum solution to the optimization problem (P). For the simplicity of discussion, we introduce an auxiliary notation $model_0(E_i, R_j)$, which is defined the same as $model(E_i, R_j)$ except that the $\mathcal{P}_t()$ term (in the objective function (P)) is removed for the rightmost sublist $\{E^D_1, ..., E^D_{n_D}\}$.

Now we give the recursive equations of $model(E_i, R_j)$ and $model_0(E_i, R_j)$. There are two cases we need to consider in calculating these two quantities. We define $model(E_0, R) = 0$ and $model_0(E_0, R) = 0$ for any $R \in \{R_0, ..., R_M\}$, for the simplicity of discussion.

Case 1: E_i is the first exon of a gene.

$model(E_i, R_j) = \max_{p\in[0,i-1],q\in[0,M]}$
$\{model(E_p, R_q) + score(E_i, R_j) + \mathcal{P}_f(E_i) + \mathcal{P}_i(E_i)$
when $l(E_i) - r(E_p) \geq \mathcal{L}$, and E_i and E_p are not related.$\}$

and

$model_0(E_i, R_j) = \max_{p\in[0,i-1],q\in[0,M]}$
$\{model(E_p, R_q) + score(E_i, R_j) + \mathcal{P}_f(E_i)$
when $l(E_i) - r(E_p) \geq \mathcal{L}$, and E_i and E_p are not related.$\}$

Case 2: E_i is not the first exon of a gene.

$model(E_i, R_j) = \max_{p\in[0,i-1],q\in[0,M]}$
$\{model_0(E_p, R_q) + score(E_i, R_j) + link(R_q, R_j) +$
$\mathcal{P}_t(E_i)$ when E_p is spliceable to E_i.$\}$

and

$model_0(E_i, R_j) = \max_{p\in[0,i-1],q\in[0,M]}$
$\{model_0(E_p, R_q) + score(E_i, R_j) + link(R_q, R_j)$
when E_p is spliceable to E_i.$\}$

These recurrences can be proved using a simple induction on i, which we omit in this abstract. In the general case, $model(E_i, R_j)$ equals to the highest value of the two cases, similarly for $model_0(E_i, R_j)$. Using the initial conditions $model(E_0, R) = 0$ and $model_0(E_0, R) = 0$, any $model(E_i, R_j)$ and $model_0(E_i, R_j)$ can be calculated in the increasing order of i using the above recurrences. In the following, we give an efficient implementation of these recurrences.

To implement the recurrences in **case 1** efficiently, we keep a table defined as follows. The table keeps the following information for the optimum (partial) gene model ended at each exon: the right boundary of the exon (**position**), the name of the exon, the score of the (partial) gene model, and the index to the entry that has the maximum **gene_score** among all entries from the top to the current one. The table is listed in the nondecreasing order of **positions**.

It can be seen that to calculate $model(E_i, R_j)$ (similarly $model_0(E_i, R_j)$), we only need to find the entry in this table that is the closest to the bottom under the condition that its distance to E_i is at least $\mathcal{L}$ and its corresponding exon is not related to E_i. To do this, we first get the left boundary L of E's gene segment (it is defined to be $l(E)$ if E does not belong to any gene segment), and search the table for the entry that is closet to the bottom and its **position** is $\leq \min\{L, l(E_i) - \mathcal{L}\}$. Obviously this can be done in $O(\log(table\_size))$, i.e., $O(\log N)$ time. After $model(E_i, R_j)$ is calculated, we need to update E_i's entry for each R_j. Each update takes only $O(1)$ time. So the total time used on **case 1** throughout the algorithm is $O(N \log(N) + NM)$.

To implement the recurrences in **case 2** efficiently using a similar technique is a little more involved due to the requirement of checking for spliceability. Recall that two exons are spliceable if they are at least $\mathcal{I}$ bases apart, their translation frames are consistent, and no in-frame stop can be formed at the joint point when they are appended (the extra conditions for the multiple-gene case can be easily checked and are omitted in our discussion). All these three conditions have to be checked and satisfied when calculating the recurrences in **case 2**.

Note that the translation frame consistency condition between two exons E and E'

$$f(E') = (l(E') - r(E) - 1 + f(E)) \text{ modulo } 3$$

can be rewritten as

$$f(E') = (l(E') - 1 - (r(E) - f(E)) \text{ modulo } 3) \text{ modulo } 3.$$

Hence we can classify all exons into three classes based on the value of $\mathcal{F}(E) = (r(E) - f(E))$ modulo 3. If forming in-frame stops was not a concern, we could have three tables for the three classes of exons like the one for **case 1**, and calculate $model(E_i, R_j)$ and $model_0(E_i, R_j)$ through searching the table that satisfies the adjacent-exon frame consistency.

To deal with the in-frame stop problem, we need to further divide these three classes. Note that when $\mathcal{F}(E) = 0$, E ends with the first base of a codon, and an in-frame stop can be possibly formed when appending E to some exon to its right only if E ends with the letter T (recall the three stops TAA, TAG, TGA); similarly when $\mathcal{F}(E) = 1$, stops can be possibly formed when E ends with either TA or TG; and when $\mathcal{F}(E) = 2$, in-frame stops can be formed only if E ends with a stop codon (E is a terminal exon). Knowing this, we classify all exons E into 7 classes: two classes (T and non-T) for $\mathcal{F}(E) = 0$; three classes (TA, TG and others) for $\mathcal{F}(E) = 1$; and two classes (stop codon and non-stop) for $\mathcal{F}(E) = 2$.

We maintain a table similar to the one in **case 1** for each possible triple $(R, \mathcal{F}, tail)$, where $R \in \{R_0, ..., R_M\}$, $\mathcal{F} \in \{0, 1, 2\}$, and $tail$ is one of the 7 cases addressed above. The table keeps an entry for each exon E of this class, and each entry contains three values: E's right boundary (**position**), the score of optimum partial gene ended at E (i. e. $(model_0(E.R))$, and the index to the entry that has the maximum score among all entries in the table from the top to the current one. The table is listed in the increasing order of its exon's right boundaries (for exons having the same right boundary, we only keep the one with the highest score). We define that each entry of the table for $(R_0, \mathcal{F}, tail)$ keeps the maximum score of the corresponding entries of tables for $(R_i, \mathcal{F}, tail)$ for all $i \in [0, M]$ (recall $R_0 = \emptyset$).

It can be seen that to calculate $model(E_i, R_j)$ (similarly $model_0(E_i, R_j)$), we only need to look at R_j's tables and R_0's tables that satisfy the frame consistency condition and the condition that no in-frame stop is formed. For each such table, we find the right entry in the same way as in **case 1** except that this time a search can be done in $O(\mathcal{I})$ time, which is a small constant, i.e., $O(1)$. Note that for each $model(E_i, R_j)$, there are at most 2×3 tables to look up. After $model(E_i, R_j)$ is calculated, we need to update the corresponding entries in both R_j and R_0's tables. and each of these updates takes $O(1)$ time. So the total time used on **case 2** is $O(NM)$.

To recover the gene models that achieve $\max_{i\in[1,N],j\in[0,M]} model(E_i, R_j)$, some simple bookkeeping needs to be done, which can be accomplished within the same asymptotic time bound of calculating $model(E_i, R_j)$. Hence we conclude that the optimization problem (P) can be solved in $O(NM + N \log(N))$

time, which improves the computational time of our previous algorithm for the protein-based gene modeling problem (Xu and Uberbacher, 1996).

3.2. Post-processing: missing exon location

The *Post Processing* step identifies and locates the missed exons based on the underlying EST reference models of the optimum gene model. The identified missing parts are searched and located in the genomic sequence using the FASTA search program (version 2.0) (Pearson and Lipman, 1988). In this step, also located are the 3′ and 5′ untranslated regions if the corresponding information exist in the underlying EST reference models of the optimum gene model. The located missing exons and the untranslated regions are added to the predicted gene models using a single-gene prediction algorithm.

Figures 5 and 6 show two examples of the predicted gene models in two long genomic sequences (HUMFLNG6PD and HUB384D8). In both Figures 5 and 6 (figure legend), the solid bars in both the top and the bottom represent the positions of the known genbank exons in the forward and reverse strand, respectively. The solid bars in the next-to-top and next-to-bottom rows represent the exons in the predicted gene models. Each set of bars connected through a line represent one gene. The hollow rectangles represent the predicted GRAIL exons. The short lines (or dots) represent the matched ESTs. The boxes (alternately hollow and solid) in the middle represent boundaries of one or a number of genes.

4. Summary

By combining the complementary nature of pattern recognition based exon prediction and EST gene structural information, we have developed a computer algorithm/system for automatic identification of gene structures in long and complex genomic sequences. With extensive tests being under way, preliminary test results have been very encouraging. Tests have shown that the predicted gene models are always very consistent with the available EST information. With its reliable gene structure prediction supported by the EST information, this computer system should provide molecular biologists a powerful and convenient tool in analyzing complex genomic sequences.

References

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.

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

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.

M. S. Gelfand and M. A. Royberg, "Prediction of the Exon-Intron Structure by a Dynamic Programming Approach", *Biosystem*, Vol. pp. 173 - 182, 1993.

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

V. V. Solovyev, A. A. Salamov, and C. B. Lawrence, "Predicting Internal Exons by Oligonucleotide Composition and Discriminant Analysis of Spliceable Open Reading Frames", *Nucleic Acids Res.*, Vol. 22, pp. 5156 - 5163, 1994.

Y. Xu, R J. Mural, J. R. Einstein, M. B. Shah, and E. C. Uberbacher, "GRAIL: A Multi-Agent Neural Network System for Gene Identification", *Proceedings of The IEEE*, Vol. 84, pp. 1544 - 1552, 1996.

D. Kulp, D. Haussler, M. G. Reese and F. H. Eckerman, "A Generalized Hidden Markov Models for the Representation of Human Genes in DNA", *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology*, AAAI Press, pp. 134 - 142, June 1996.

S. Salzberg, X. Chen, J. Henderson, and K. Fasman, "Finding Genes in DNA using Decision Trees and Dynamic Programming", *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology*, AAAI Press, pp. 201 - 210, June 1996.

M. Burset and R. Guigo, "Evaluation of Gene Structure Prediction Programs", *Genomics*, Vol. 34, pp. 353 - 375, 1996.

Y. Xu, R. J. 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, 1994.

F. Costanzo, L. Castagnoli, L. Dente, P. Arcari, M. Smith, P. Costanzo, G. Raugel, P. Izzo, T. C. Pietronaolo, L. Bougueleret, F. Cimino, F. Salvatore, and R. Cortese, "Cloning of Several cDNA Segments coding for Human Liver Proteins", *EMBO. J.*, Vol. 2, pp. 57 - 61, 1983.

M. D. Adams, J. M. Kelley, J. D. Gocayne, M. Dubnick, M. H. Polymeropolous, H. Xiao, C. R. Merril, Wu, B. Olde, R. F. Moreno, A. R. Kerlavage, W. R. McCombie, and J. C. Venter, "Complementary DNA Sequencing: Expressed Sequence Tags and Human Genome Project", *Science*, Vol. 252, pp. 1651 - 1656, 1991.

A. S. Wilcox, A. S. Khan, J. A. Hopkins, and J. M. Sikela, "Use of 3′ Untranslated Sequences of Human

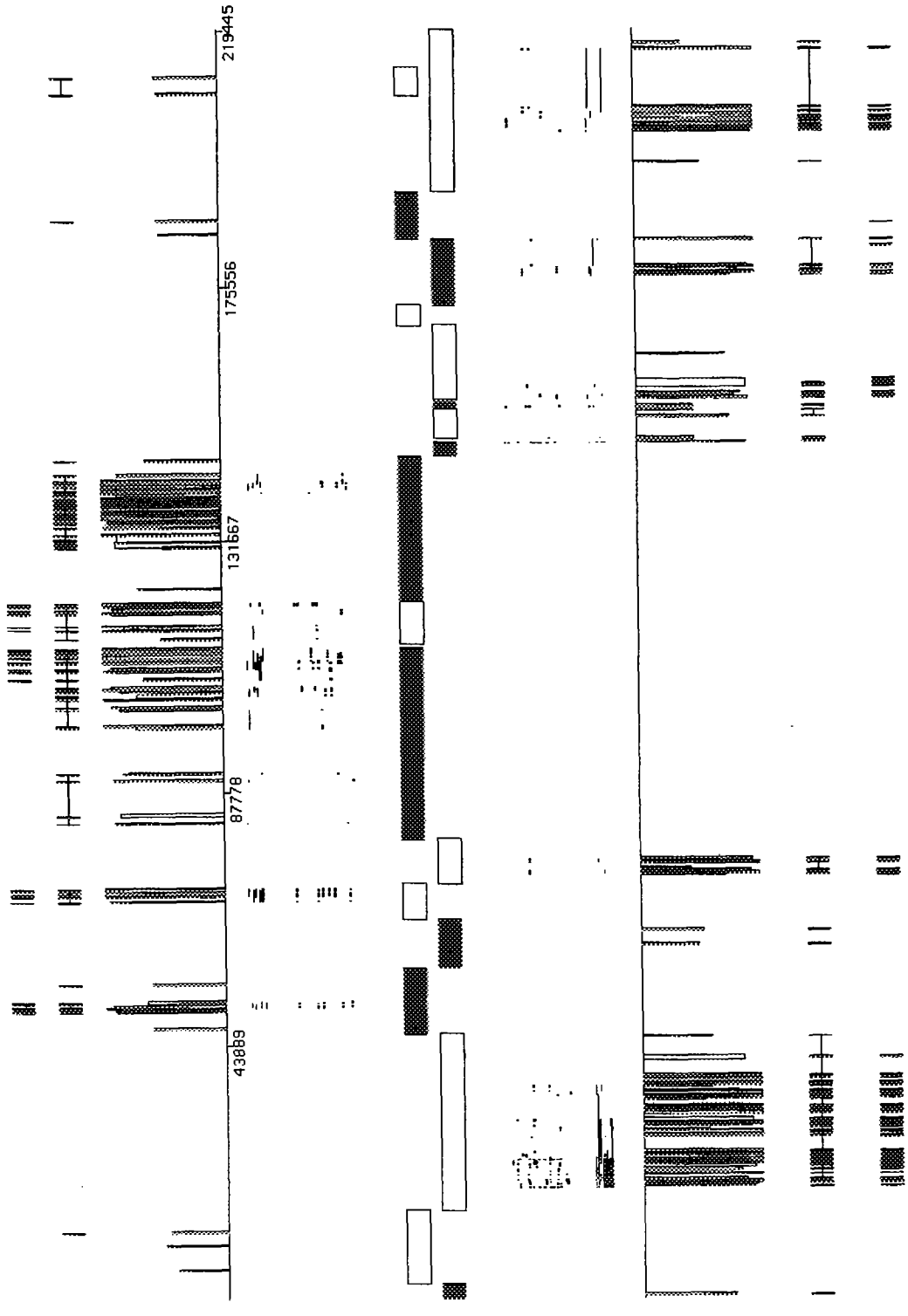

Figure 5:

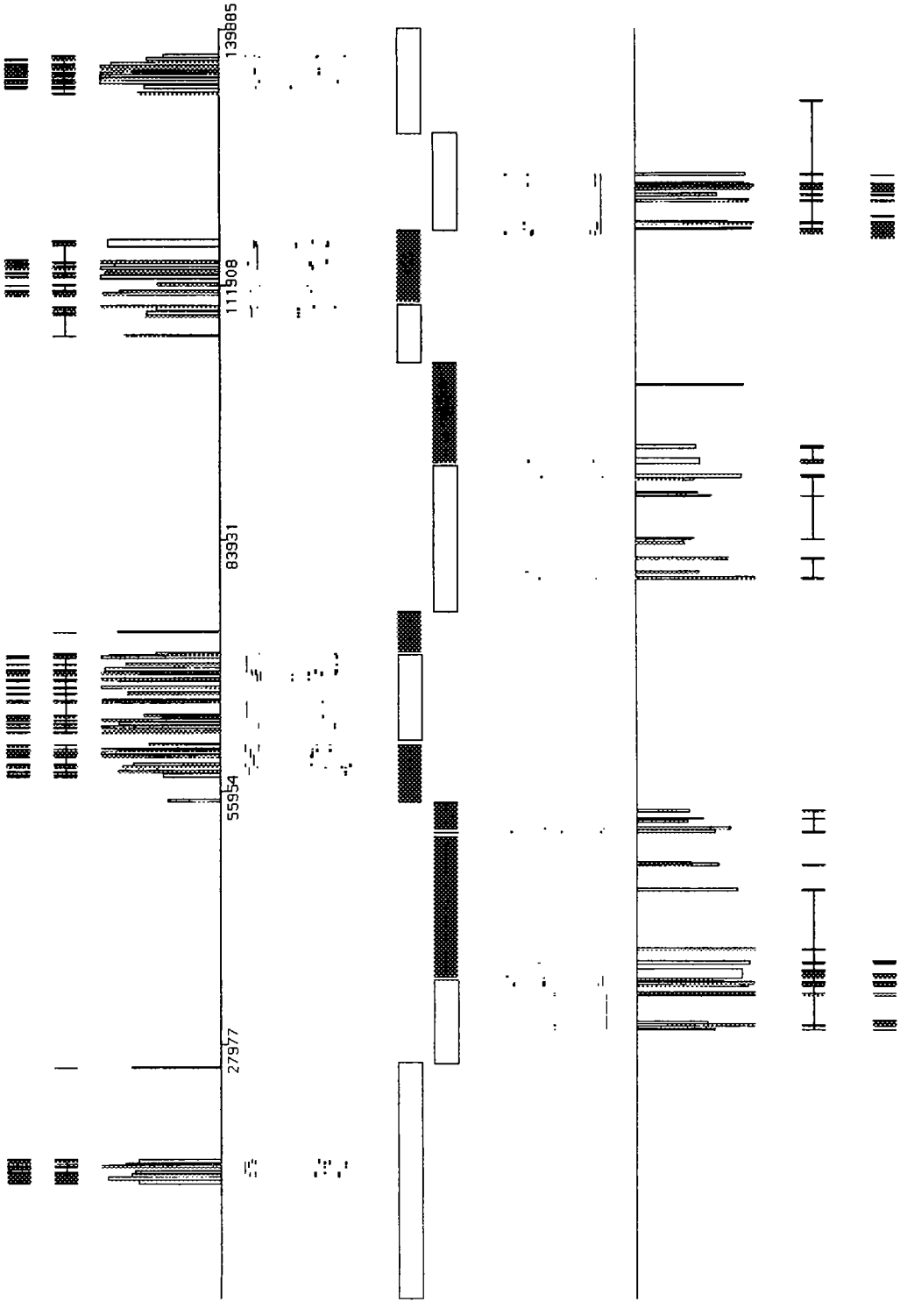

Figure 6:

cDNAs for Rapid Chromosome Assignment and Conversion to STSs: Implications for an Expression Map of the Genome", *Nucleic. Acids Res*, Vol. 19, pp. 1837 - 1843, 1991.

R. Houlgatte, R. Mairage-Samson, S. Duprat, A. Tessier, S. Bentolila, B. Lamy, and C. Auffray, "The Geneexpress Index: A Resource for Gene Discovery and the Genic Map of the Human Genome", *Genome Res.*, Vol. 5, pp. 272 - 304, 1995.

M. D. Adams, *et al.*, "Initial Assessment of Human Gene Diversity and Expression Patterns Based Upon 83 Million Nucleotides of cDNA Sequence", *Nature*, Vol. 377, pp. 3 - 174, 1995.

J. S. Aaronson, B. Eckman, R. A. Blevins, J. A. Borkowski, J. Myerson, S. Imran, and K. O. Elliston, "Towards the Development of a Gene Index to the Human Genome: An Assessment of the Nature of High-throughput EST Sequence Data", *Genome Research*, Vol. 6, pp. 829 - 845, 1996.

M. S. Boguski, T. M. Lowe, and C. M. Tolstoshev, "dbEST – Database for Expressed Sequence Tags", *Nat. Genet.*, Vol. 4, pp. 332 - 333, 1993.

S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic Local Alignment Search Tools", *J. Mol. Biol.*, Vol. 215, pp. 403 - 410, 1990.

Y. Xu and E. C. Uberbacher, "Gene Prediction by Pattern Recognition and Homology Search", *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology*, AAAI Press, pp. 241 - 251, 1996.

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