An Improved System for Exon Recognition and Gene Modeling in Human DNA Sequences[1]

Ying Xu, J. Ralph Einstein, Richard J. Mural[†], Manesh Shah and Edward C. Uberbacher

Informatics Group
Engineering Physics and Mathematics Division and [†]Biology Division
Oak Ridge National Laboratory, Oak Ridge, TN 37831-6364
Telephone: 615-574-6640, Fax: 615-574-7860

Abstract

A new version of the GRAIL system (Uberbacher and Mural, 1991; Mural *et al.*, 1992; Uberbacher *et al.*, 1993), called GRAIL II, has recently been developed (Xu *et al.*, 1994). GRAIL II is a hybrid AI system that supports a number of DNA sequence analysis tools including protein-coding region recognition, PolyA site and transcription promoter recognition, gene model construction, translation to protein, and DNA/protein database searching capabilities. This paper presents the core of GRAIL II, the coding exon recognition and gene model construction algorithms. The exon recognition algorithm recognizes coding exons by combining coding feature analysis and edge signal (acceptor/donor/translation-start sites) detection. Unlike the original GRAIL system (Uberbacher and Mural, 1991; Mural *et al.*, 1992), this algorithm uses variable-length windows tailored to each potential exon candidate, making its performance almost exon length-independent. In this algorithm, the recognition process is divided into four steps. Initially a large number of possible coding exon candidates are generated. Then a rule-based prescreening algorithm eliminates the majority of the improbable candidates. As the kernel of the recognition algorithm, three neural networks are trained to evaluate the remaining candidates. The outputs of the neural networks are then divided into clusters of candidates, corresponding to presumed exons. The algorithm makes its final prediction by picking the best candidate from each cluster. The gene construction algorithm (Xu, Mural and Uberbacher, 1994) uses a dynamic programming approach to build gene models by using as input the clusters predicted by the exon recognition algorithm. Extensive testing has been done on these two algorithms. The exon recognition algorithm is a significant improvement over the original GRAIL system, and the gene construction algorithm further improves the prediction results.

[1]This research was supported by the Office of Health and Environmental Research, United States Department of Energy, under contract DE-AC05-84OR21400 with Martin Marietta Energy Systems, Inc.

Introduction

The recent identification of genes for major genetic diseases through the use of genomic DNA sequencing and informatics analysis (Mosser *et al.*, 1993; Legouis *et al.*, 1991) has underscored the importance of genomic sequencing for gene discovery and elucidation of genome function. Essential for such discovery is the ability to recognize features in genomic sequences, such as exons, splice junctions and promoters using pattern recognition methodology.

Despite recent success, developing the technology to accurately recognize the components of genes and to construct gene models from anonymous genomic DNA sequence data remains a significant challenge. We previously developed an e-mail server system for coding exon recognition called GRAIL (Gene Recognition and Analysis Internet Link) (Uberbacher and Mural, 1991; Mural *et al.*, 1992). Recently we have upgraded the system to provide recognition capabilities for a variety of sequence features using artificial intelligence-based pattern recognition and combinatorial optimization methods. In this paper, we describe the subsystems of GRAIL II (Xu *et al.*, 1994) that facilitate coding exon recognition and gene model construction.

The GRAIL II coding exon recognition algorithm recognizes a coding exon (in the following, we simply call it an *exon*) by combining the coding signals, edge signals (acceptor/donor/translation start sites) and domain information. Three models are used to recognize coding signals; they are a frame-dependent preferred 6mer model (Uberbacher and Mural, 1991; Uberbacher *et al.*, 1993), a 6mer coding preference model (Uberbacher and Mural, 1991; Uberbacher *et al.*, 1993) and a non-homogeneous Markov chain model (Uberbacher *et al.*, 1993; T. Mitchell (personal communication), 1991; Borodovsky *et al.*, 1986). To better determine the boundaries of an exon, measures for splice junctions and translation initiation are also used in the exon discrimination process. The core of the exon discrimination algorithm consists of three pre-trained neural networks, which are used to evaluate three different types of exon candidates, i.e., initial, internal and terminal, respectively.

The original GRAIL used a fixed-length sliding window in evaluating the coding potential of regions in a DNA sequence. In this form, the system has difficulty in locating exons that are much shorter than the window size, because of the inclusion of non-coding segments within the window being evaluated. A scheme of variable-length windows is implemented in the current algorithm. Our basic implementation of variable-length windows considers every possible exon candidate, and evaluates each using a window that matches the candidate region exactly. Using variable-length windows has particularly improved the system's performance on short exons, which was one of the main motivations for the current study. To fully utilize the variable-length window scheme, we have also used the coding (or noncoding) signals from areas surrounding an exon candidate in the discrimination process. This is done mainly to distinguish an actual exon from a partially correct exon candidate, i.e., one that partially overlaps one or more actual exons.

Improving the sensitivity to the exons in the A/T rich regions without increasing the background noise level was another motivation of this research. To make the predictions more sensitive to coding signals in regions of different G/C-content, particularly in low G/C (or high A/T) regions, we have used G/C content-dependent preferred 6mer models and Markov chain model.

Making use of these considerations allows the GRAIL II coding recognition algorithm to significantly improve the sensitivity and specificity of the coding exon recognition in human DNA sequences for both short and long exons, in low and high A/T regions, compared to the the original version of GRAIL.

The GRAIL II coding exon recognition algorithm divides the process of coding exon recognition into four main steps. It first generates a large candidate pool consisting of all possible exon candidates within all open reading frames (this is similar to the procedure used in (Hutchinson and Hayden, 1992)). A series of heuristic rules, each of which defines some necessary conditions a *probable* exon candidate should satisfy, then eliminates majority of the improbable candidates. The candidates which have passed the rules are then evaluated by three pre-trained neural networks. Using heuristic rules to eliminate improbable candidates simplifies the learning process for the neural networks and also gains computational efficiency. The scored candidates (by the neural networks) are then divided into clusters of candidates, corresponding to presumed exons, by a clustering algorithm based on the candidates' relative locations in the DNA sequence. The best scoring candidate from each cluster is finally selected as the initial exon prediction. During gene modeling, an alternative candidate from the cluster may be used to construct the best overall gene model.

Constructing an accurate gene model from a given set of predicted exons is a nontrivial matter. When appending exon candidates to form a gene model a number of conditions need to be satisfied. In a gene model, adjacent exons have to be reading-frame compatible, and no in-frame stop codons can be formed when appending the two adjacent exons. We formulate the gene model construction problem as a constraint optimization problem. The problem is solved by a fast dynamic programming algorithm.

Coding-exon Recognition

One of the main motivations of the current study was to improve the performance of the GRAIL system on short exons. Using fixed-length sliding windows is one of the reasons for the poor performance on short exons. Our current algorithm uses a window that exactly fits a candidate to evaluate its coding potential. To do so, the algorithm first generates all possible initial, internal and terminal exon candidates within every open reading frame. Each candidate has an assigned translation frame, two potential edge signals and must be at least 11 bases long. For initial exon candidates, a possible translation start (ATG) and splice donor junctions (GT) are required. For an internal exon candidate, a possible splice acceptor junction (YAG) and a splice donor junction are the minimal requirements. Similarly the minimal requirements for a terminal exon candidate are a possible acceptor junction and a stop codon (TAA, TAG, and TGA).

As in any pattern recognition problem, to recognize exons we need to find a set of features that are associated with exons, and to design an effective method to discriminate exons from non-exonic regions based on these features. Three types of information are used in our exon discrimination process. They are statistical measures of frequencies of different "vocabularies", like 6mers, in exons versus non-exonic regions, measures of edge signals for splicing sites and/or translation initiation sites, and measures of domain information including local G/C composition and exon length probability profile. The first type measures the coding potential of an exon candidate, the second type indicates the possibility of an exon candidate having correct boundaries, and the third type provides domain information to the discrimination process in evaluating the significance of the coding scores and edge signals. Our discrimination algorithm uses three feed-forward neural networks, which have been trained using a back-propagation algorithm, to evaluate the *degree* of correctness of an exon candidate.

Sensors

Features are computed by different algorithms, called *sensors*, a term borrowed from robotics, in keeping with the tradition of the original GRAIL system (Uberbacher and Mural, 1991; Mural *et al.*, 1992). Each sensor "recognizes", with some probability, some property related to the existence of an exon.

Protein-coding sensors Two independent hypotheses are made about a DNA sequence. The first hypothesis is that an exon in a certain reading frame can be decomposed into a set of components which possess similar properties. In particular, an exon in a certain reading frame can be decomposed into a set of 6mers in that frame, which possess coding potential. We use the ratio of the normalized frequencies of a 6mer in coding and non-coding regions to measure its coding potential. The second hypothesis is that a DNA sequence forms a stochastic process, in which the probability of the next event (base) being in a certain state (A, C, G, T) depends only on a finite number of previous events, and hence the sequence forms a Markov chain. To distinguish three possible reading frames of an exon a non-homogeneous Markov chain is assumed (T. Mitchell (personal communication), 1991; Borodovsky *et al.*, 1986).

Based on these hypotheses, we have implemented a frame-dependent preferred 6mer model, a 6mer coding preference model and a 5th order non-homogeneous Markov chain as sensors of coding potential. The formulation of the three models is illustrated by the following example.

Let $a_1a_2....a_{3k+2}$ be an exon candidate and each a_i represent a base. We want to evaluate the coding potential of the candidate in reading frame 0 using the three models. Let $P(X)$, $P_0(X)$, $P_1(X)$, $P_2(X)$, and $P_n(X)$ denote the probabilities of a 6mer X (with respect to all 6mers) appearing in bulk DNA sequence, in a coding region with reading frame 0, 1, 2, and in a non-coding region, respectively. The coding potential of $a_1a_2...a_{3k+2}$ in reading frame 0, using the frame-dependent preferred 6mer model, is evaluated by the following expression:

$$\Pi_{i=0}^{k-2}\frac{P_0(a_{3i+1}...a_{3i+5}a_{3i+6})}{P_n(a_{3i+1}...a_{3i+5}a_{3i+6})}\frac{P_1(a_{3i+2}...a_{3i+6}a_{3i+7})}{P_n(a_{3i+2}...a_{3i+6}a_{3i+7})}\times$$

$$\frac{P_2(a_{3i+3}...a_{3i+7}a_{3i+8})}{P_n(a_{3k+3}...a_{3i+7}a_{3i+8})},$$

and the coding potential of $a_1a_2...a_{3k+3}$ in reading frame 0, using the 5th order Markov chain model, is estimated as follows:

$$C\times\frac{P_0(a_1...a_5)}{P(a_1...a_5)}\times\Pi_{i=0}^{k-2}\frac{P_0(a_{3i+6}|a_{3i+1}...a_{3i+5})}{P(a_{3i+6}|a_{3i+1}...a_{3i+5})}\times$$

$$\frac{P_1(a_{3i+7}|a_{3i+2}...a_{3i+6})}{P(a_{3i+7}|a_{3i+2}...a_{3i+6})}\frac{P_2(a_{3i+8}|a_{3i+3}...a_{3i+7})}{P(a_{3i+8}|a_{3i+3}...a_{3i+7})}.$$

where C is our estimate of the ratio of coding versus non-coding bases in a DNA sequence (we set $C = 9$ in our implementation).

The 6mer coding preference model, which is mainly useful in situations where frame-dependent characteristics are weak or where sequence errors violate frame information, measures the coding potential (in any frame) as follows:

$$\Pi_{i=0}^{k-2}\frac{P_c(a_{3i+1}...a_{3i+5}a_{3i+6})}{P_n(a_{3i+1}...a_{3i+5}a_{3i+6})}\frac{P_c(a_{3i+2}...a_{3i+6}a_{3i+7})}{P_n(a_{3i+2}...a_{3i+6}a_{3i+7})}\times$$

$$\frac{P_c(a_{3i+3}...a_{3i+7}a_{3i+8})}{P_n(a_{3k+3}...a_{3i+7}a_{3i+8})},$$

where $P_c(X) = P_0(X) + P_1(X) + P_2(X)$.

In essence, all three models recognize coding potential by comparing the *a priori* probabilities of 6mers of the sequence appearing in coding regions and non-coding regions. We can see that the frame-dependent preferred 6mer model and the 5th order Markov model provide similar information except that each term in of the Markov model is a conditional probability rather than the probability of a 6mer. Our test results suggest that each model has its own prediction strengths and weaknesses.

To make the coding models more sensitive to coding signals in regions of different G/C composition, our current algorithm uses G/C content-dependent 6mer probabilities in all three models. These probabilities are functions of the G/C content of the region surrounding the exon candidate. We use the preferred 6mer model as an example to illustrate this basic idea. When estimating coding potential, different *6mer preference values*, the ratio of a 6mer's probabilities in coding regions of frame 0 and non-coding region, are used depending on the local G/C-content (measured in the surrounding region of 2kb long) of the sequence under evaluation. To avoid abrupt behavior, we have interpolated the 6mer preference values derived in discrete G/C regions using a simple function. In our current implementation, we have divided the whole region into two, i.e., low and high G/C regions. 6mer preference values are estimated in both regions and are interpolated by a piecewise linear function.

The goal of the exon discrimination process is not just to discriminate exons from non-exonic regions but also to score the degree of correctness of a candidate that overlaps actual exons. To achieve this, we have used coding (or non-coding) signals from the surrounding areas in addition to the coding signals of the candidate. The rationale is that strong coding signals from the neighboring areas indicate that the candidate may be just a portion of an exon. As the candidate more closely approximates an actual exon, more non-coding elements will be included in its surrounding areas and hence the surroundings will exhibit a weaker coding signal. In our current implementation, we have used 60 bases on each side of an exon candidate as the surrounding area. In addition to the sensors described above, we have also used the following four values, two for each surrounding areas, as features in our discrimination process: the coding potentials obtained by the frame-dependent preferred 6mer model and by the Markov chain model.

Edge-signal sensors Recognizing exon boundaries is accomplished by using a splice acceptor junction sensor, a splice donor junction sensor and a translation start sensor (Uberbacher *et al.*, 1993).

Domain sensors Though our coding models are "normalized" with respect to the G/C content by separately estimating (6mer and Markov) preference values in different G/C regions, to help the neural networks refine the significance of the coding measures in different compositional regions, we have included the G/C content of a 2 kb region surrounding each exon candidate as an additional sensor.

Since the exon candidates evaluated by the system vary in length, we have also included the length of a candidate and an exon length probability profile, which is derived from a length histogram for exons, as additional factors (sensors) in the discrimination process to help evaluate the significance of the coding scores.

Heuristic rules

On average, about 5,200 candidates are generated for a DNA sequence of 10 kb, and about 85% of these candidates do not overlap any actual exons. The vast majority of the false candidates show very weak or no coding signals and/or poor edge signals. Filtering out most of these candidates can greatly simplify the discrimination process facilitated by neural networks.

We have developed a set of heuristic rules, based on existing knowledge and statistical analysis. Each of the rules defines some necessary conditions that a probable candidate should satisfy. On average, about 130 candidates per 10 kb pass the rules, which account only for 2.5% of the generated candidates, and about 40% of the surviving candidates do not overlap any actual exons. Only 2% of the actual exons are lost through the use of heuristic rules.

Neural networks

The core of the discrimination algorithm consists of three feed-forward neural networks. The neural network for internal exons, for example, is represented mathematically by the following formula:

$$output = g(\sum_{k=1}^{3} W_k^3 g(\sum_{j=1}^{6} W_{kj}^2 g(\sum_{i=1}^{11} W_{ji}^1 (input_i)))),$$

where

$$g(x) = \frac{1}{1 + exp(-x)}.$$

The network, as shown schematically in Figure 1, has twelve input nodes and two hidden layers consisting of 6 and 3 nodes. The parameters W's are "learned" using the back-propagation method. In training the network, our goal is to develop a network which can score the "partial correctness" of each potential candidate. A simple matching function $M()$ is used to represent the correspondence of a given candidate with the actual exon(s):

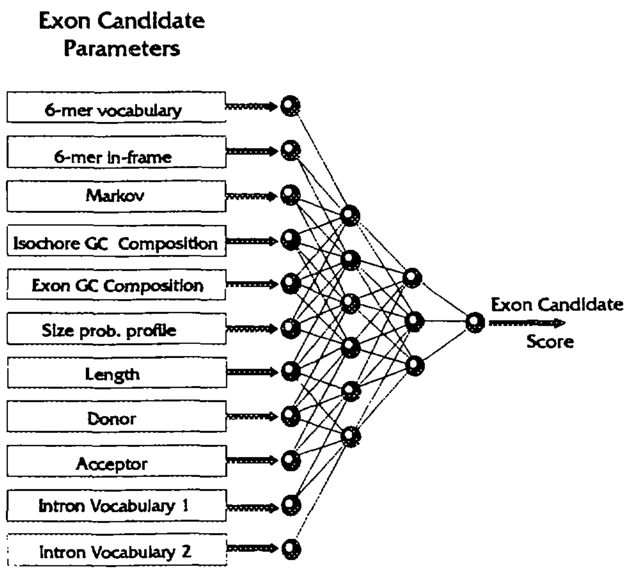

Figure 1: Neural network for evaluating exon candidates

$$M(\text{candidate}) = (\sum_i m_i/\text{length}(\text{candidate}))\times$$
$$(\sum_i m_i/\sum_j \text{length}(\text{exon}_j))$$

where $\sum_i m_i$ is the total number of bases of the candidate that overlap some actual exons (in the same reading frame), and $\sum_j$ length(exon$_j$) is the total length of all the exons that overlap the candidate. Using such a function helps "teach" the neural network to discriminate between candidates with different degrees of overlap with actual exon(s).

The network for internal exons has been trained on a set containing 335 internal exons and 2000 (partially) true or false candidates. Each training example includes twelve normalized features, as described in the previous subsections, and a $M()$ value. All sequences used for training were from the Genome Sequence Database (GSDB) (Bilofsky and Burks, 1988).

Figure 2(a) shows the prediction of exon candidates from all three neural networks combined. We can see that candidates fall into non-overlapping natural "clusters". In most cases, a "cluster" corresponds to one actual exon. In some cases, a "cluster" may correspond to two or even more actual exons as shown in Figure 2(a). In the next subsection, we will present a clustering algorithm which resolves this clustering problem so that candidates are divided into more "accurate" clusters, those having better one-to-one correspondence with actual exons.

In the output of the neural network (after deleting candidates with very low scores), about 94% of the ac-

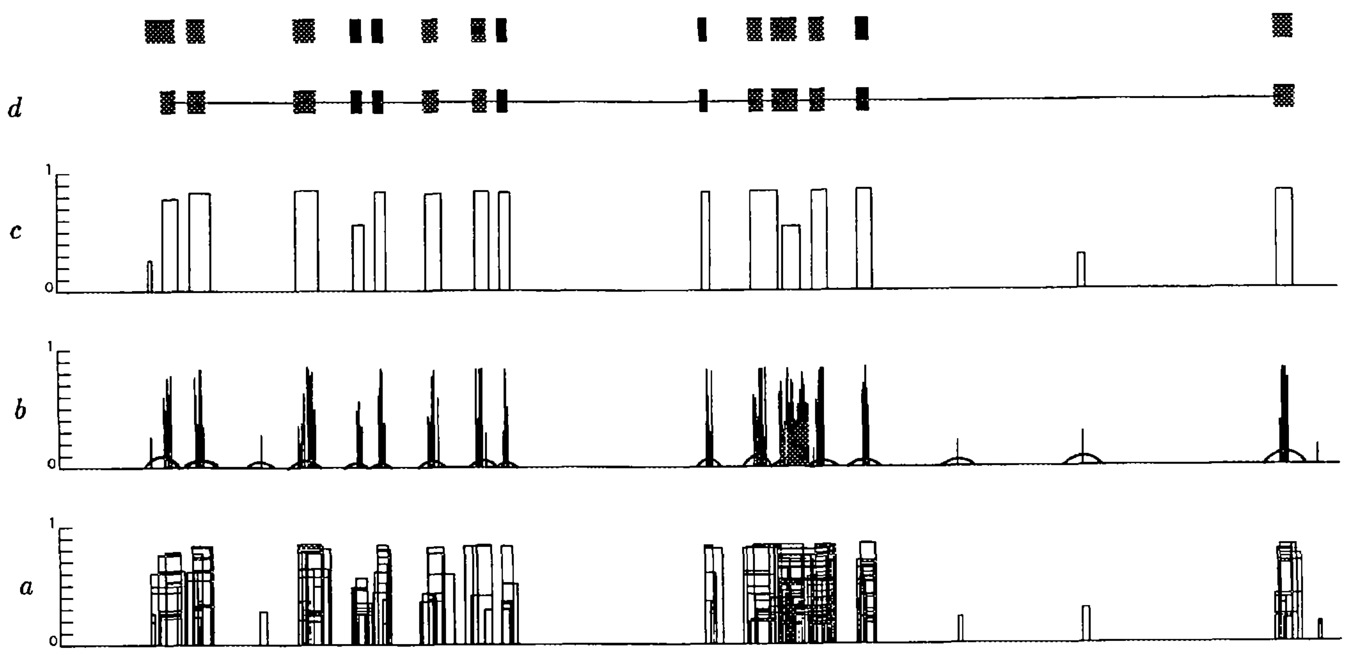

Figure 2: Each rectangle in Figure 2(a) represents an exon candidate, where the width and height of the rectangle represent its length and score, respectively. The vertical lines in Figure 2(b) are the centers of the corresponding exon candidates in part (a), and the arcs indicates the algorithmically defined clusters. Figure 2(c) shows the best scoring candidate picked from each cluster. Figure 2(d) is the gene model constructed by our algorithm. Note that the best scoring candidate in (c) is not necessarily used in the best gene model. The solid bars on the top represent the actual exons. The sequence used here is HUMKERTRA.

tual exons are overlapped by some clusters, and about 10% of the clusters do not overlap any actual exons.

Clustering

The candidates scored by the neural networks form a set of "clusters" of overlapping candidates. In the ideal situation, each "cluster" would correspond to one actual exon. However, as we have seen from Figure 2(a), in sequences with long open reading frames, one cluster may cover a region corresponding to more than one exon. For clarity, we use "cluster" to denote a natural cluster observed in Figure 2(a). Clustering in this subsection means to divide a "cluster" corresponding to more than one exon into smaller groups so that each group has a better one-to-one correspondence with a single exon.

Figure 2(b) shows that centers of the candidates form groups that have better one-to-one correspondence with actual exons. We have implemented the following algorithm that separates each natural "cluster" into one or more groups using the centers of the candidates. The algorithm has the following two steps.

- **Separate Step**: Divides each "natural" cluster into smaller groups so a given objective function is optimized.
- **Merge Step**: Selects the highest scoring candidate from each group, and merges groups which have their highest scoring candidates overlapping.

The **Separate Step** divides candidates' centers into groups so that the distance between two adjacent groups is "significantly" larger than the average distance between adjacent centers within each group, and the total number of groups formed from the "cluster" is "reasonably small". Specifically, it uses two application-specific parameters R and G, and guarantees the ratio of the distance between two adjacent groups and the average distance between two adjacent centers within each group to be bigger than R, and the number of partitioned groups to be less than G. The algorithm finds a partition of the "cluster" that satisfies these conditions and furthermore minimizes the sum of the average distances between two adjacent centers of all groups.

Gene Model Construction

The goal of the gene model construction is to linearly append the predicted exons (not necessarily including all non-overlapping exons) in such a way that a series of constraints are satisfied. These constraints include the following: (1) adjacent exons in the gene model are *reading-frame compatible*; (2) the distance between two adjacent exons is bigger than some given constant – the minimum intron size; (3) no in-frame stop codons can be formed when appending two adjacent exons; (4) the gene model should include as many high scoring candidates as possible.

Statement of the problem

Formally, a *DNA* sequence S is a sequence consisting of four letters $\{A, C, G, T\}$. An *exon candidate* is a subsequence $S[i, j]$ of S that starts with a start codon (ATG) or on the base following an acceptor junction site (CAG or TAG), and ends with a stop codon (TAA, TAG, or TGA) or on the base preceding a splice donor site (GT), where i and j are indices of edges. It has an assigned *reading frame* $\alpha \in \{0, 1, 2\}$ and has no *in-frame* stop codons, i.e., no stop codons $S[k, k+2]$ ($= TAA$, TAG, or TGA) with $(k-1) \bmod 3 = \alpha$ and $i \le k \le j-2$. Each exon candidate has a non-negative score $p() \in [0..1]$. An exon candidate is an *initial* exon candidate if it starts with a start codon, or is a *terminal* exon candidate if it ends with a stop codon, otherwise it is an *internal* exon candidate. Two non-overlapping exon candidates $S[i, j]$ and $S[m, n]$, $m > j$, are said to be *reading-frame compatible* if

$$\beta = (m - j - 1 + \alpha) \bmod 3, \tag{1}$$

where $S[i, j]$ and $S[m, n]$ are in reading-frames α and β, respectively.

Let $S[b_1, e_1], S[b_2, e_2], ..., S[b_k, e_k]$ be a set of k non-overlapping exon candidates (within a region assumed to contain a gene), with $b_1 < b_2 ... < b_k \in [1..|S|]$. $S[b_1, e_1]S[b_2, e_2]...S[b_k, e_k]$ forms a *partial gene model* if (1) $S[b_i, e_i]$ and $S[b_{i+1}, e_{i+1}]$ are reading-frame compatible, (2) no in-frame stop codon is formed when appending $S[b_{i+1}, e_{i+1}]$ to $S[b_i, e_i]$, (3) $b_{i+1} - e_i > K$, where K is a constant, the minimum intron size (in GRAIL II, $K = 60$), for all $i \in [1..k-1]$, and (4) all $S[b_i, e_i]$'s are internal exons except possibly for $S[b_1, e_1]$ and $S[b_k, e_k]$; they can be an initial exon and a terminal exon, respectively.

Let $\{C_1, C_2, ..., C_n\}$ be a set of clusters and each C_i contain a number of exon candidates, $i \in [1..n]$. Our goal is to select a set of non-overlapping exon candidates $E_1, E_2, ..., E_m$, at most one, from each cluster, $m \le n$, to form a partial gene model that maximizes the following function:

$$\sum_{i=1}^{m} p(E_i) + \mathcal{P}_f + \mathcal{P}_t,$$

where $p(E_i)$ is the score of exon candidate E_i, and $\mathcal{P}_f$ and $\mathcal{P}_t$ are two penalty factors. $\mathcal{P}_f$ (or $\mathcal{P}_t$) is a fixed negative real number when a partial gene model does not have an initial (or terminal) exon, otherwise it is zero.

Dynamic programming algorithm (GAP III)

A dynamic programming approach is used to solve the optimization problem defined in the previous subsection. Dynamic programming approachs have been used to solve the gene assembly problem in different settings (Snyder and Stormo; 1993; Gelfand and Roytberg, 1993). Because of the limited space, we give only an informal introduction to our solution to this problem.

For each cluster, the algorithm builds (at most) 18 best partial gene models that end with exon candidates of the current cluster, based on the best partial gene models which end with exon candidates of the previous clusters, counted from left to right. When extending a current partial gene model to the right to include one more exon, the algorithm checks if the conditions for a partial gene model are satisfied. It repeatedly does this until all clusters are processed. By doing so, the algorithm finds a partial gene model that optimizes the objective function given in the previous subsection.

To check if the conditions for a gene model are satisfied when extending gene models from left to right, some information needs to be provided about the reading frames and ending edges of (the last exons of) the previous models. We do this as follows. For each cluster under consideration, we construct a best partial gene model that ends with an exon of this cluster for each of the following possible situations. The exon can be in any reading frame $\alpha \in \{0, 1, 2\}$ and its right edge modulo 3 can be any $\beta \in \{0, 1, 2\}$. To also take into consideration the possibility of forming an in-frame stop codon when appending two adjacent exons, we, for each possible α and β, distinquish the following situations. For each α, when $\mathcal{D} = (\alpha + 3 - \beta) \bmod 3 = 0$ the exon can end with a T or a non-T letter, and when $\mathcal{D} = 1$, the exon can end with TA, TG, or any other doublet. We can show that considering all these 18 possible situations provides sufficient and necessary information for our optimal partial gene model construction algorithm. Figure 2(d) gives an example of the results of gene model construction.

The following gives the dynamic programming recurrence we have used to solve the gene assembly problem. Let $N_n(\beta, e, j)$ denote the total score of the best (partial) gene model that ends with a non-terminal exon candidate in cluster j, whose reading frame is β and right-edge modulo 3 is e, and similarly $N_t(\beta, e, j)$ denote the total score of the best (partial) gene model that ends with an exon in cluster j, whose reading frame is β and right edge modulo 3 is e.

$$N_n(\beta, e, j) = \max_{E_i^j \in C_j} \left\{ \begin{array}{l} p(E_i^j) + \mathcal{P}_f(E_i^j), \\ \\ \max_{\alpha, e', j'} \{ N_n(\alpha, e', j') + p(E_i^j), \\ \quad \text{for } rf(E_i^j) = \beta,\ 1 \leq j' < j, \\ \quad \beta = (\text{left}(E_i^j) - e' - 1 + \alpha) \bmod 3, \\ \quad \text{right}(E_i^j) = e, \text{ and } E_i^j \text{ is a} \\ \quad \text{non-initial exon. } \} \end{array} \right\},$$

$$N_t(\beta, e, j) = \max_{E_i^j \in C_j} \left\{ \begin{array}{l} p(E_i^j) + \mathcal{P}_f(E_i^j) + \mathcal{P}_t(E_i^j), \\ \\ \max_{\alpha, e', j'} \{ N_n(\alpha, e', j') + p(E_i^j) + \\ \quad \mathcal{P}_t(E_i^j), \\ \quad \text{for } rf(E_i^j) = \beta,\ 1 \leq j' < j, \\ \quad \beta = (\text{left}(E_i^j) - e' - 1 + \alpha) \bmod 3, \\ \quad \text{right}(E_i^j) = e, \text{ and } E_i^j \text{ is a} \\ \quad \text{non-initial exon. } \} \end{array} \right\},$$

where left(E_i^j) and right(E_i^j) are the indices of the left and right edges of E_i^j, respectively.

Note that the best gene model corresponds to

$$\max_{\beta, e, j} N_t(\beta, e, j).$$

Our algorithm for solving the above recurrence runs in time $O(E \times C)$ and space $O(E)$, where C is the number of clusters and E is the total number of exon candidates. In situations where a limit can be placed on the number of clusters that can potentially be excluded from the optimal gene model the running time is further reduced to $O(E + C)$.

Results and Discussion

The performance of the algorithms presented in this paper and implemented in the GRAIL II system show a significant improvement compared to the original GRAIL by several different measures. On the training set consisting of 61 Human DNA sequences, our exon recognition algorithm recognizes about 94% of all exons with about 10% false positives; On an independent test set consisting of 110 Human and Mouse DNA sequences, the algorithm recognizes 90% of all the exons, with about 8% false positives, compared to about 80% for the original GRAIL. The original GRAIL recognizes about 50% of exons less than 100 bases in length (Uberbacher and Mural, 1991) while our current algorithm located about 75% in the test set. Most of these statistics are further improved by the gene construction program (GAP III), particularly for false positive rate and edge accuracy. This is achieved mainly because of the enforcement of the reading-frame compatibility between consecutive exons.

Part of the performance improvement in the GRAIL II exon recognition algorithm is due to additional and more accurate information used in making coding-exon prediction. The 6mer probabilities are estimated based on a data set consisting of about 140,000 coding bases and 1,200,000 non-coding bases. In addition to several strong coding measures, scores at the specific edges of each candidate (splice junctions, translation starts where appropriate) and information related to the non-coding character of sequences adjacent to potential exon candidate are considered in the discrimination process. Essentially the algorithm uses the character of the expanded sequence context of the potential coding region to make its decision. However the algorithm may not always function well if coding segments do not have splice junctions, and neighboring intronic or non-coding DNA, as in cDNAs. Such regions do not meet the basic genomic context requirements for the GRAIL II coding exon recognition and the original GRAIL coding analysis is more appropriate for such sequences.

Tables I and II summarize the performance of the exon recognition and gene model construction algorithms on an independent test set consisting 110 sequences.

Figure 3 gives three examples of exon prediction and gene assembly by GRAIL II.

The high sensitivity and specificity of the GRAIL II exon recognition and gene construction program and its availability through e-mail and client/server mechanism greatly increases the viability of the gene hunting strategies based on genomic sequencing and informatics analysis. We have shown that the detailed structure of genes can be characterized with considerable fidelity, and expect that, in terms of providing relatively complete information about uncharacterized regions of the genome, this overall technology will fair well when compared to alternatives such as exon trapping and cDNA based methods. Computational characterization of genes in their genomic sequence context will increasingly provide an important framework for understanding aspects of gene regulation and larger questions related to the functional organization of the genome.

References

H. S. Bilofsky and C. Burks (1988), "The GenBank Genetic Sequence Data Bank", *Nucleic Acids Res.*, Vol. 16, pp. 1861 - 1864.

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

M. S. Gelfand and M. A. Roytberg (1993), "Prediction of the exon-intron structure by a dynamic programming approach", *Biosystems*, 30, pp. 173-182.

G. B. Hutchinson and M. R. Hayden (1992), "The Prediction of Exons Through an Analysis of Splice-

Table I

	DNA	Predictions				Gene Models			
	# Exons	TP	%	FP	%	TP	%	FP	%
Short	229	171	74.7%	39	18.6%	167	73.0%	16	8.7%
Long	600	575	95.8%	30	4.9%	564	94.0%	13	2.3%
Total	829	746	90.0%	69	8.5%	731	88.2%	29	3.8%
	# Bases								
Total	134814	122885	91.2%	13048	9.6%	122404	90.8%	5972	4.7%

TP and FT are the true and false positives, respectively. Short: 100 bases or less; Long: otherwise.

Table II

	# Exons	# Exons found	%	# Bases	# Bases found	%
Initial	110	91	82.7%	15936	14023	88.0%
Internal	611	566	92.6%	88703	81961	92.4%
Terminal	108	89	82.4%	30175	26420	87.6%

able Open Reading Frames", *Nucleic Acids Res.*, 20, pp. 3453 - 3462.

R. Legouis, *et al.* (1991), "The Candidate gene for the X-linked Kallmann Syndrome Encodes a Protein Related to Adhesion Molecules", *Cell*, 67, 10, 18, pp. 423 - 435.

J. Mosser, A. M. Douar, C. O. Srade, P. Kioschis, R. Feil, H. Moser, A. M. Poustka, L. J. Mandel, and P. Aubourg (1993), "Putative X-Linked Adrenoleukodystrophy Gene Shares Unexpected Homology with ABC transporters", *Nature*, 361, pp. 726 - 730.

R. J. Mural, J. R. Einstein, X. Guan, R. C. Mann and E. C. Uberbacher (1992), "An Artificial Intelligence Approach to DNA Sequence Feature Recognition", *Trend in Biotechnology*, 10, pp. 66 - 69.

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

E. C. Uberbacher, J. R. Einstein, X. Guan, and R. J. Mural (1993), "Gene recognition and assembly in the GRAIL system: progress and challenges", *Proceedings of The 2nd International Conference on Bioinformatics, Supercomputing and Complex Genome Analysis*, H. A. Lim, J. W. Fickett, C. R. Cantor and R. J. Robbins, Eds, World Scientific. pp. 465 - 476.

E. C. Uberbacher and R. J. Mural (1991), "Locating protein-coding regions in human DNA sequences by a multiple sensor-neural network approach", *Proc. Natl. Acad. Sci. USA*, Vol. 88, pp. 11261 - 11265.

Y. Xu, R. J. Mural, M. Shah, and E. C. Uberbacher (1994), "Recognizing exons in genomic sequence using GRAIL II", *Genetic Engineering: Principles and Methods*, Jane Setlow (Ed.), Plenum Press, Vol. 16, (in press).

Y. Xu, R. J. Mural, and E. C. Uberbacher (1994), "Constructing Gene Models from Accurately-predicted Exons: An Application of Dynamic Programming", To appear in *CABIOS*.

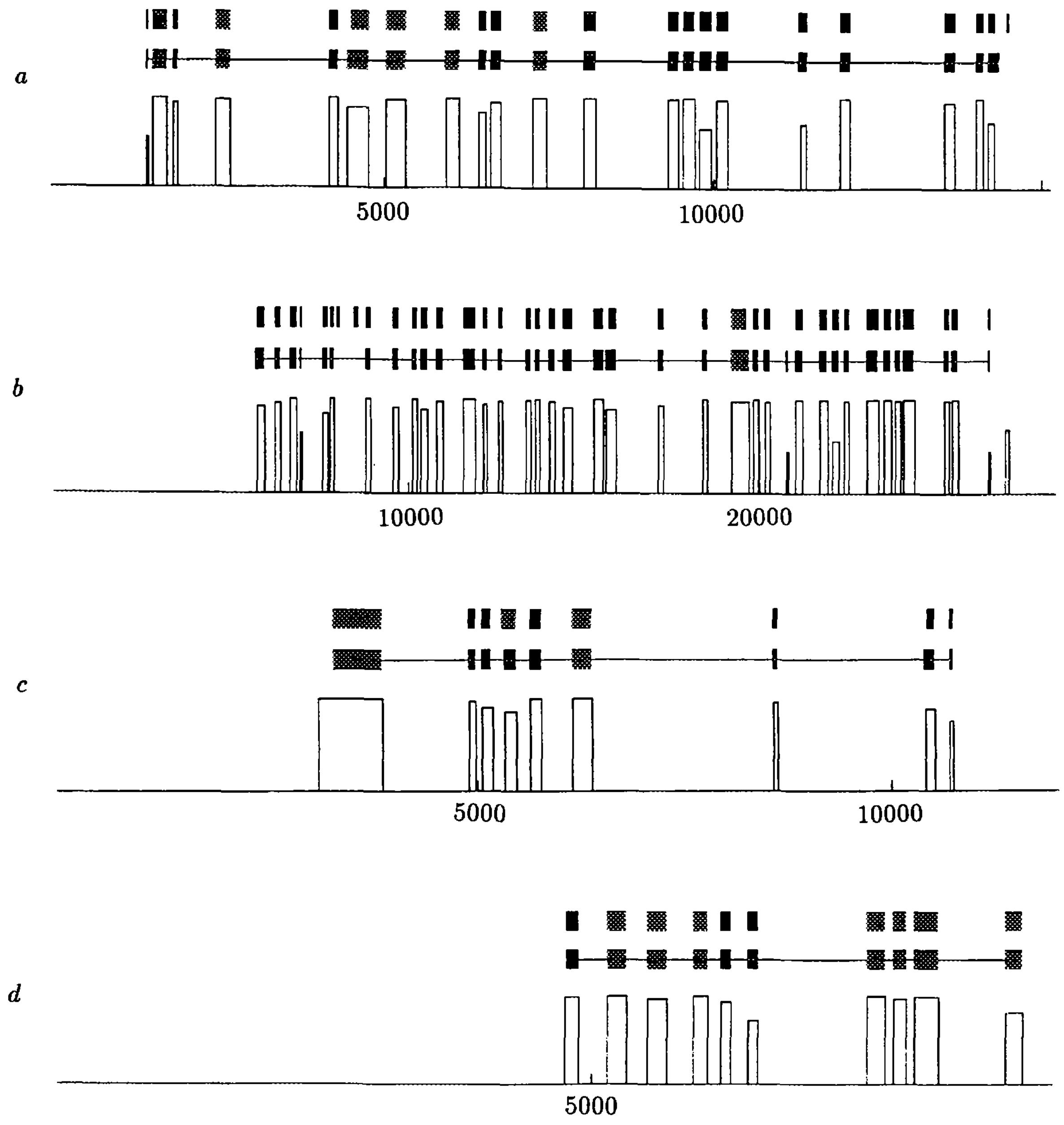

Figure 3: (a) HUMATPGG. (b) HUMBMYH7. (c) HUMDES. (d) HUMODC1A. Each rectangle represents a GRAIL exon prediction; The width and the height represent its length and score, respectively. The bars connected represent the assembly result. The bars on the top are the actual exons.