Pergamon

PII: S0097-8485(96)00034-4

Computers Chem. Vol. 21, No. 4, pp. 229–235, 1997

Printed in Great Britain
0097-8485/97 $17.00 + 0.00

Combinatorial approaches to gene recognition*

M. A. Roytberg,[1] T. V. Astakhova[1] and M. S. Gelfand[2]†

[1]Institute of Mathematical Problems in Biology, Russian Academy of Sciences, Pushchino 142292, Russia and [2]Institute of Protein Research, Russian Academy of Sciences, Pushchino 142292, Russia.

(Received 3 May 1996; Accepted 1 November 1996)

Abstract—Recognition of genes via exon assembly approaches leads naturally to the use of dynamic programming. We consider the general graph-theoretical formulation of the exon assembly problem and analyze in detail some specific variants: multicriterial optimization in the case of non-linear gene-scoring functions; context-dependent schemes for scoring exons and related procedures for exon filtering; and highly specific recognition of arbitrary gene segments, oligonucleotide probes and polymerase chain reaction (PCR) primers.‡ © 1997 Elsevier Science Ltd

1. INTRODUCTION

Recognition of genes in eukaryotic DNA is seriously complicated by the exon–intron structure. Currently, the most popular approach is to consider a set of candidate exons weighed by some statistical parameters and then construct the optimal gene, defined as a consistent chain of exons, using dynamic programming (DP) (Gelfand and Roytberg, 1993; Snyder and Stormo, 1993, 1995; Stormo and Haussler, 1994; Xu *et al.*, 1994b; Gelfand *et al.*, 1996a). However, straightforward application of this approach meets difficulties of both conceptual and computational natures. Each candidate gene is characterized by several diverse statistical parameters, and it is not immediately clear how to combine them in a single scoring function. This is overcome partially by application of neural networks either for scoring individual exons (Xu *et al.*, 1994a, 1994b) or in combination with a DP procedure so that multiple rounds of network training and construction of the optimal genes are performed (Snyder and Stormo, 1993, 1995). However, the use of the standard DP implies that we consider only additive scores, although in various branches of biopolymer sequence analysis non-linear functions sometimes perform better (e.g. Piterbarg, 1992; Brodsky *et al.*, 1993). The opportunity to consider non-linear functions is provided by vector dynamic programming, which constructs the set of genes guaranteed to contain the optimal gene for each function satisfying some natural monotonicity conditions (Pareto set) (Gelfand and Roytberg, 1993).

Here we consider some new problems arising in this field. The plan of the paper is as follows. First, we state the problem of exon assembly in general graph terms and demonstrate that most existing approaches can be formulated using this language; the exposition in this section follows the line of Finkelstein and Roytberg (1993). Then we consider the vector dynamic programming algorithm and describe the results of prediction for one particular non-linear function.

In the traditional approaches to gene recognition an algorithm is trained so as to minimize some parameter dependent both on the number of false positive and false negative predictions (e.g. the correlation coefficient). Thus, the average performance is optimized. However, there exist situations where it is desirable to minimize either over-, or underprediction, while paying less attention to false negatives (respectively, false positives). Two such "extremal" problems are considered in Sections 5 and 6.

In Section 5 we suggest a procedure for filtration of candidate exons taking into account their combinatorial possibilities for linking with other exons. Such a procedure is useful, since the number of candidate exons is usually very large, whereas many algorithms are polynomial or even exponential on the number of exons; it is clear that this procedure should be highly sensitive, i.e. allow (almost) no underprediction.

Section 6 is devoted to the problem of the high-specificity prediction of coding segments (not necessarily complete exons), which arises from the

* Some topics included in this work have been discussed during the *Gene-Finding and Gene Structure Prediction Workshop*, Philadelphia, PA, 13–14 October 1995.

† Author for correspondence. E-mail: misha@imb.imb.ac.ru; Fax: [7]-(095)-135-9984.

‡ Described programs are available at http://www-hto.usc.edu/software/procrustes/

need to construct oligonucleotide probes and primers given genomic sequences for subsequent experimental analysis of cDNA libraries or total cellular RNA using hybridization or RT-PCR (reverse transcription-PCR) techniques (Parrich and Nelson, 1993; D'Esposito *et al.*, 1994).

Finally, we demonstrate that in many cases the general exon assembly graph can be reconstructed, providing a sharp increase in computational effectiveness.

When describing data structures and algorithms we will usually ignore technical complications such as accounting for the reading frame, keeping an open reading frame in the spliced gene, taking into account non-zero widths of donor and acceptor splicing sites, consideration of codons interrupted by exon–intron boundaries, etc. Usually, in the course of implementation, it is clear how to take care of these problems, although they can seriously complicate the exposition. We assume that the reader is familiar with basic biological facts about splicing (for a review see, e.g. Sharp, 1994). Necessary computer science definitions can be found in Aho *et al.* (1976).

In all of the implementations mentioned below we use only the simplest statistical parameters (codon usage and positional nucleotide frequencies in splicing sites).

2. GRAPH REPRESENTATION OF THE GENE RECOGNITION PROBLEM

We start with a nucleotide sequence with marked positions of candidate start and stop codons, and donor and acceptor sites (Fig. 1(a)). Each site is assigned a numerical weight dependent on the sequence around it and computed using some empirical procedure (reviewed, e.g. in Gelfand, 1995).

The sites generate candidate exons and introns, and their combinations form exon–intron structures, or genes. Formally, *candidate exon* is a sequence fragment whose left boundary is an acceptor site or a start codon, whilst the right boundary is a donor

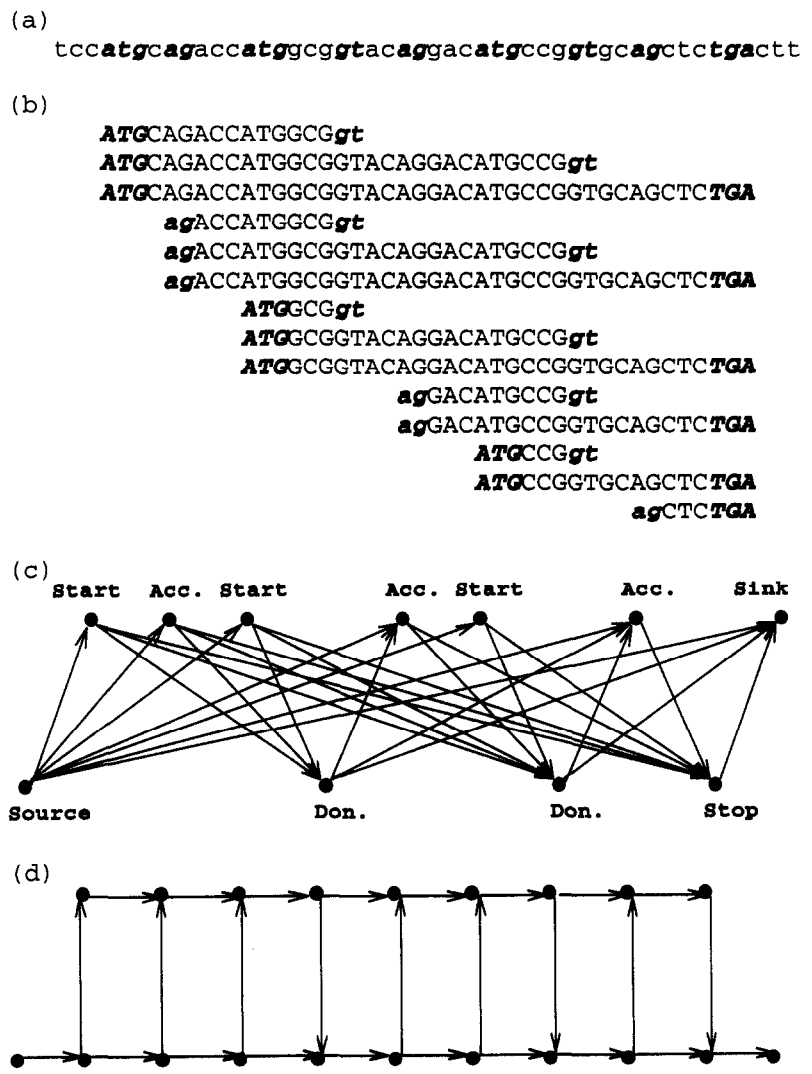

Fig. 1. A sample sequence and generated graphs. (a) Sequence with candidate sites marked with bold italics. (b) List of candidate exons. (c) Site graph. (d) Corresponding railway graph.

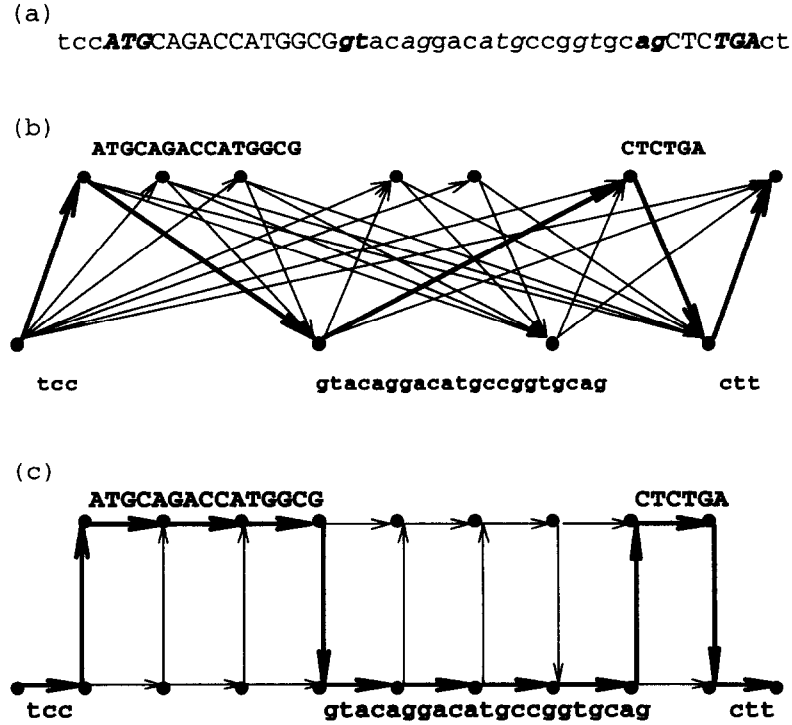

Fig. 2. A candidate gene and corresponding paths (bold arcs). (a) Candidate gene (where bold italics indicate generating sites and capitals indicate candidate exons). (b) Path on the site graph. (c) Path on the railway graph.

site or a stop codon. [Note that this definition slightly deviates from the biological reality, since we consider only translated exons or translated parts of exons (Fig. 1(b)).] Similarly, *candidate intron* is a fragment between a donor site and an acceptor site, or between the beginning of the sequence and a start codon, or between a stop codon and the end of the sequence. Like sites, candidate exons and introns can be supplied by weights scoring their statistical exon and intron likeness (Fickett and Tung, 1992; Gelfand, 1995).

Candidate gene is a chain of non-intersecting alternating exons and introns $(i_0, e_1, i_1, \ldots, e_n, i_n)$ (Fig. 2(a)) that cover the entire sequence and satisfy the natural consistency conditions:

(i) the total length of exons is divisible by 3;
(ii) in exons there are no in-frame stop codons;
(iii) the first intron–exon boundary (i_0, e_1) is a start codon; the last exon–intron boundary (e_n, i_n) is a stop codon.

However, most algorithms described below can be simply generalized to the case of incomplete genes violating condition (iii) and possibly condition (i).

Given a sequence with marked sites or a set of candidate exons and introns, it is possible to construct an acyclic-oriented graph G such that the set of complete paths on this graph is set in one-to-one correspondence with the set of complete genes. Thus the gene recognition problem reduces to analysis of paths in the graph G often performed using some form of dynamic programming (Finkelstein and Roytberg, 1993).

This reduction, often implicit, can be done in several ways dependent upon particular weighting functions, procedures for candidate exon filtering (e.g. Xu *et al.*, 1994b), etc. The choice of the graph G can seriously affect the effectiveness of the algorithm.

Below we describe *site graph*, which has vertices that are sites and arcs that are exons and introns (Snyder and Stormo, 1993, 1995). Other possibilities are *intron graph* with vertices that are introns and arcs that are exons (Gelfand and Roytberg, 1993; Xu *et al.*, 1994b; Gelfand *et al.*, 1996a), and *railway graph*, considered in Section 7. Note that the number of vertices in the intron graph is much larger than that in the site graph. The railway graph is the smallest one. However, it places some restrictions on the scoring procedures.

3. DYNAMIC PROGRAMMING ON SITE GRAPH

Consider an acyclic-directed graph G where vertices are splicing sites, start and stop codons, and arcs are exons and introns (Fig. 1(c)). In order to account for the reading frame, it is convenient to assign to each site three vertices corresponding to three possible positions of the site relative to the triplet reading phase. Further, trivial filtering allows one to retain only exons with no in-frame stop codons, and introns that link vertices with a consistent reading phase. Thus, each path corresponds to some candidate gene, and conversely, each

gene is represented by a path in G (Fig. 2(b)). The set of all paths in G is denoted by S_G.

Each arc $e = (v,v')$ is supplied by a (not necessarily scalar) weight $W(e) = W(v,v')$. Let $p = e_1 \odot \ldots \odot e_n$ be a path corresponding to some candidate gene. The path weight (gene score) is then defined as the formal product of weights of arcs (exons and introns) forming the path:

$$W(p) = W(e_1 \odot \ldots \odot e_n) = W(e_1) \otimes \ldots \otimes W(e_n) = \bigotimes_{i=1}^{n} W(e_i) . \quad (1)$$

If S is a set of paths, its total weight is the formal sum of path weights

$$\Phi(S) = \bigoplus_{p \in S} W(p) . \quad (2)$$

The objective is to compute $\Phi(S_G)$.

The operations $\otimes$ and $\oplus$ are not necessarily numerical multiplication and addition. It is required only that they satisfy the semi-ring axioms (Finkelstein and Roytberg, 1993). Associativity of $\otimes$ is used in the definition of path weight by equation (1), and associativity and commutativity of $\oplus$ is used in the definition of $\Phi(S)$ by equation (2), and this means that we can consider $\oplus$ as a multivalent operation. DP specifically uses the distributivity of $\otimes$ relative to $\oplus$:

$$W \otimes (W' \oplus W'') = (W \otimes W') \oplus (W \otimes W'') , \quad (3)$$
$$(W' \oplus W'') \otimes W = (W' \otimes W) \oplus (W'' \otimes W) .$$

This is the property that allows one to compute $\Phi(S_G)$ in an effective way that avoids explicit consideration of all paths from the set. Indeed, let $S(v_i)$ be the set of paths coming to some vertex v_i, and let e be an arc starting in v. Then

$$\bigoplus_{p \in S(v_i)} W(p \odot e) = \bigoplus_{p \in S(v_i)} (W(p) \otimes W(e)) = \left(\bigoplus_{p \in S(v_i)} W(p) \right) \otimes W(e) . \quad (4)$$

Now, if v is a vertex and $(v_1,v),\ldots,(v_k,v)$ are all vertices entering v, then

$$\Phi(S(v)) = \bigoplus_{p \in S(v)} W(p) = \bigoplus_{i=1}^{k} \left(\bigoplus_{p \in S(v_i)} W(p \odot (v_i,v)) \right) = \bigoplus_{i=1}^{k} \left(\left(\bigoplus_{p \in S(v_i)} W(p) \right) \otimes W(v_i,v) \right) = \bigoplus_{i=1}^{k} (\Phi(S(v_i)) \otimes W(v_i,v)) . \quad (5)$$

This formula provides the recurrency for computing $\Phi(S(v))$ given the set weights $\Phi(S(v_i))$, $i = 1, \ldots, k$, for predecessors of v.

In most algorithms the arc weights W are real numbers. If now $\otimes$ is the simple addition of path weights and $\oplus$ is the operation of taking the maximum (it is simple to prove that the semi-ring axioms are satisfied), we get the usual problem of finding the optimal path weight, which together with a backtracking procedure, constructs the highest scoring gene. This approach is employed by GeneParser (Snyder and Stormo, 1993, 1995) and GRAIL (Xu *et al.*, 1994b). If $\otimes$ is multiplication, $\oplus$ is addition, and arc weights are considered to be energies, then equation (2) transforms into the definition of the partition function (Stormo and Haussler, 1994)

$$\Phi(S_G) = \sum_{p \in S_G} \prod_{e \in p} \exp W(e) . \quad (6)$$

Then $W(p)/\Phi(S_G)$ is the probability of the path p. The duality between addition/maximum and multiplication/addition weight systems is well known in computational molecular biology. [See Finkelstein and Roytberg (1993) for other examples of this kind.] Some further results based on these definitions are described in Sections 5 and 6.

4. VECTOR DYNAMIC PROGRAMMING, SUBOPTIMAL GENES, MULTICRITERIAL OPTIMIZATION, AND NON-LINEAR SCORING FUNCTIONS

Since our knowledge of the statistical properties of genomic DNA sequences is far from complete, we cannot always find the correct exon assembly; we also do not have a standard way of weighing diverse statistical parameters and do not even know what types of scoring functions are reasonable. Thus, some sort of pattern recognition is required. One way to do this is to consider linear scoring functions, set coefficients for parameters by a neural network or a similar technique, and consider multiple suboptimal assemblies (Snyder and Stormo, 1993, 1995). Another possibility is provided by multicriterial optimization based on vector dynamic programming (Gelfand and Roytberg, 1993; Gelfand *et al.*, 1996a).

Assume that each arc e is weighed by a vector $(W_1(e),\ldots, W_m(e))$. Path weights are defined by the componentwise addition of weights of the constituent arcs: for $p = e_1 \odot \ldots \odot e_n$ we have

$$W_j(p) = \sum_{i=1}^{n} W_j(e_i),\ j = 1, \ldots, m . \quad (7)$$

We say that a vector U dominates over a vector V (denoted $U \succ V$) if $U_j \geq V_j$ for any $j = 1,\ldots, m$ and at least one inequality is strict. If H is a set of vectors, its *Pareto subset* $\Psi(H)$ is a subset such that

(i) for any $V \in H \setminus \Psi(H)$ there exists $U \in \Psi(H)$ such that $U \succ V$;

(ii) for any $U, U' \in \Psi(H)$ neither $U \succ U'$, nor $U \prec U'$.

The Pareto subset is a multidimensional analog of the maximum value (in the one-dimensional case $\Psi(H)$ is a set containing a single element $\max\{U \in H\}$). Allowing ourselves some latitude in our treatment, we will consider also Pareto sets of paths, meaning the set of paths the weights of which constitute the Pareto set of vectors. Formally, if S is a set of paths, its Pareto subset $P(S) = \{s \in S | W(s) \in \Psi(W(S))\}$, where $W(S) = \{W(s) | s \in S\}$ is the set of path weights. [We can also define domination conditions on paths, directly, as in Gelfand and Roytberg (1993) and Gelfand *et al.* (1996a).]

Without loss of generality we can assume that the gene score is a function of the components of the corresponding path weight increasing monotonically at each of its m variables. It is simple to demonstrate that the Pareto set contains an optimal gene for all scoring functions. Thus, rather than fixing the scoring function and constructing the set of suboptimal solutions, we instead construct the set containing an optimal solution for any scoring function satisfying the natural monotonicity conditions (Gelfand and Roytberg, 1993).

This procedure follows the general scheme of Section 3. Recall the problem of finding the maximum path weight. Then for each vertex v we considered the set of paths $S(v)$ ending in v and computed

$$\Phi(S(v)) = \bigoplus_{p \in S(v)} W(p) = \max \{W(p) | p \in S(v)\} . \quad (8)$$

The distributivity given by equation (4) allowed us to retain only one path coming to v.

Now $\Phi(S) = \oplus_{p \in S} W(p) = \Psi(W(S))$ is the Pareto subset of the set $W(S)$. So defined, $\oplus$ together with $\otimes$ given by equation (7) satisfy the semi-ring axioms [formally, the semi-ring elements are vector sets: $H \oplus H' = \Psi(H \cup H')$; $H \otimes H' = \Psi\{V + V' | V \in H, V' \in H'\}$]. Now since by definition $\Phi(S) = \Phi(P(S))$, we may retain only paths from the Pareto set $P(S)$.

Computer experiments demonstrated that non-linear functions indeed perform better than linear ones. We considered the following vector of arc weights: acceptor site score A and donor site score D defined as the Berg–von Hippel discrimination energy (Gelfand, 1989); coding potential C defined as the sum of codon weights equal to logarithms of codon frequencies; exon length L; and exon counter 1 (so that the path weight had the component N equal to the number of the constituent exons). The gene score was (recall that we use the same notation for the components of the arc weight and the path weight)

$$R = \frac{A - N\mu_A}{N\sigma_A} + \frac{D - N\mu_D}{N\sigma_D} + \frac{C - L\mu_C}{\sqrt{L}.\sigma_C} , \quad (9)$$

where μ_x and σ_x denote, respectively, the mean and the standard deviation of the parameter x on a learning sample.

The vector dynamic programming algorithm using this very simple set of statistical parameters was implemented in the Genome Recognition and Exon Assembly Tool (GREAT) (Gelfand *et al.*, 1996a). Its performance on an independent test sample was comparable to that of GRAIL II (Xu *et al.*, 1994a). Sensitivity of GREAT was 88%, specificity was 79%, and the exact prediction was obtained in 27% of cases. GRAIL II had 82% sensitivity, 90% specificity and 4% exact predictions. More detailed analysis of the results, including hard cases, confirmed the general impression that GREAT is more sensitive, but less specific than GRAIL II with the same overall quality of predictions (Gelfand *et al.*, 1996a).

5. CONTEXT-DEPENDENT FILTRATION OF EXONS

After the first combinatorial algorithms had been implemented, it was noted that true exons tend to appear in many high-scoring structures (Gelfand, 1990; Snyder and Stormo, 1993). This observation was formalized in the algorithm for computation of the partition function (Stormo and Haussler, 1994) described in Section 3. This algorithm is applicable only in the distributive case. However, the general idea of rescoring exons according to their context, more exactly, their ability to participate in high-scoring structures, can be used in more general situations as well. In particular, this technique can be used for exon filtering, where the aim is to decrease the number of candidate exons without losing more than some fixed fraction of actual exons.

We define new exon scores as

$$W'(e) = \sum_{p \in P} \exp(cR(p)) , \quad (10)$$

where c is some constant, the summation is taken over all genes p from the Pareto set P, and R is the gene-scoring function, e.g. the one defined by equation (9). Then we can order the exons by the decrease of their new scores and consider the given fraction of all exons. As an extreme case, we can use all exons that occur in Pareto-optimal genes. Exon filtration can be performed using arbitrary exon chains (not necessarily complete genes; in the experiment described below we used two-exon chains), and the filtered exons can then be used to construct longer chains. This allows one to perform the computationally intensive construction of the Pareto set of multi-exon genes on a smaller set of exons. The filtered set can also be input to the similarity-based algorithm of spliced alignment, increasing the quality of its predictions, which strongly depends on the number of candidate exons (Gelfand *et al.*, 1996b).

Table 1 presents results of testing the above approach on a set of 244 human genes of length up to 32000 nt. The statistical parameters (codon frequencies, positional nucleotide frequencies on the splicing sites) were taken from Gelfand *et al.* (1996a). The exon filtering procedure depends on a single adjustable parameter, the number of accepted candidate exons per 100 nt (without filtering there would be approximately 150 candidate exons per 100 nt). It can be seen that in the entire range of

Table 1. Filtering of candidate exons

Score	Exons per 100 nt									
	1	2	3	4	5	6	7	8	9	10
Equation (9)	89	129	152	169	183	194	205	212	216	220
Equation (10)	124	159	182	197	207	218	226	228	229	229

Exons are scored by equation (9) and rescored by equation (10); then the given number of candidate exons (1-10 per 100 nt) is retained. The values in each cell show the number of genes with no actual exons lost during the filtering.

Table 2. Results of prediction for long human genes

Coding segments	Candidate segments 1	2	3	4-9	Total
1	49	1	0	1	51
2	—	44	4	3	51

The values in each cell show the number of genes where the given number of candidate segments should be considered in order to get the given number of coding segments.

values the combinatorial procedure rescoring exons according to equation (10) outperforms the purely statistical filter without rescoring.

6. OLIGONUCLEOTIDE PROBE SELECTION FROM GENOMIC DNA

Analogously to equation (10) we can score an arbitrary sequence fragment. It can be useful when there is no necessity to predict complete genes or complete exons, but the prediction should be highly specific, that is, allowing almost no false positives. One of the possible experimental settings for such programs is construction of oligonucleotide probes or PCR primers.

We use the following approach. Let b be a position in the analyzed fragment, and let $S(b)$ be the set of genes in which b belongs to some exon. The nucleotide score is defined as

$$W'(b) = \sum_{p \in S(b)} \exp(cR(p)) . \qquad (11)$$

The score of a segment $B = 1 \ldots k$ is defined as the average nucleotide score:

$$W^*(B) = \frac{1}{k} \sum_{b=1}^{k} W'(b) . \qquad (12)$$

Then the segments are ordered by a decrease of their scores. If PCR primers are predicted, one can consider pairs of segments occurring at a distance exceeding some threshold.

Testing of the algorithm (Roytberg *et al.*, 1996, 1997) has demonstrated that it indeed can be used for highly specific recognition of coding segments. On a set of 51 human DNA fragments of length 10000–30000 nt (Table 2) the highest scoring segment of length 30 was coding in 49 cases. If a pair of coding segments was needed, two candidates were sufficient in 44 cases, whilst three candidates were sufficient in 48 cases. The algorithm was also tested on a sample of 124 *Arabidopsis* genes (Korning *et al.*, 1996) (Table 3). To have a single coding segment, it was sufficient to retain the highest scoring candidate in 120 cases, and the two best candidates in the remaining 4 cases. To have two segments, two candidates were sufficient in 116 cases and three candidates in 6 more cases.

Table 3. Results of prediction for *Arabidopsis*

Coding segments	Candidate segments 1	2	3	4	Impossible	Total
1	120	4	0	0	0	124
2	—	116	6	1	1	124

Notation as in Table 2.

7. RAILWAY GRAPH

The number of arcs in the graphs considered in previous sections is quadratic relative to the number of vertices (sites). Since the number of candidate sites is typically rather large (or we risk losing some actual sites and the corresponding exons), even the polynomial DP procedures become computationally intensive. However, if path weights (or their components in the vector situation of Section 4) can be defined as sums of arc weights as in equation (1) or equation (7), it is reasonable to assume that exon (or intron) weight can be defined as the sum of weights of individual codons or nucleotides. Indeed, if we can perform addition over segments, why restrict this possibility only to exons (introns), and not to arbitrary sequence segments.

If the above additivity holds, we can present the data in the form of a much smaller *railway graph* (Fig. 1(d)). Its vertices again are donor and acceptor sites (two vertices per site), whereas the arcs are of two types: *rails* corresponding to segments situated between sites and considered to be coding (upper rail on the figures) and non-coding (lower rail); and *ties* corresponding to transitions between states (coding → non-coding at donor sites and non-coding → coding at acceptor sites). Each path on the old graph (Fig. 2(b)) corresponds to a path on the new graph (Fig. 2(c)) and vice versa, whereas additivity of segment weights allows one to define path weights in the usual manner. The number of arcs in the railway graph is linear relative to the number of vertices, which makes the computation much more effective.

8. CONCLUSION

The main result of this work is that the combinatorial approach to gene recognition provides a flexible tool that can be simply tuned to different experimental situations. We have considered three particular examples of its use: recognition of genes with reasonable average sensitivity and specificity, and two extremal situations when one of these quality parameters is more important than the other. In all cases we saw that the developed algorithms provide predictions useful for experimental biologists.

Most exon assembly methods find the best gene candidate or several candidates, but do not specifically consider the situation when the analyzed fragments are entirely non-coding. This situation can be diagnosed by GREAT, which has three possible answers: "coding" (with a list of gene candidates), "non-coding", and "no opinion" (a list of candidates is produced, but no opinion is given about the certainty of prediction). Decrease of the "no opinion" zone requires the use of more complicated statistical parameters.

An important open problem is analysis of large genome fragments, covering many genes. It requires linking of the exon assembly procedures with modules for prediction of functional signals such as promoters and polyadenylation sites.

Finally, it might be useful to combine the gene recognition methods based on statistical analysis with the powerful similarity-based techniques. It has been demonstrated in (Snyder and Stormo, 1995) that

simple use of BLAST scores as one more exon characteristic seriously improves the recognition quality (see also Burset and Guigo, 1996); our results show that simple exon filtering enhances the performance of the spliced alignment algorithm (Gelfand *et al.*, 1996b). Thus it can be expected that merging of statistics with bank searches and similarity analysis within a uniform combinatorial frame will be the main direction of development in the field of gene recognition.

Acknowledgements—This work was supported by grants 94-04-12330 from the Russian Fund of Fundamental Research, 70/95 from the Russian State Scientific Program "Human Genome", MTW300 from ISF and the Russian Government, and DE-FG-94ER61919 from DOE (USA). We are grateful to Drs L. Brodsky, E. Koonin, A. Mironov, and V. Veiko for assistance in sample collection, M. Borodovsky and P. Pevzner for discussions, and O. Evgrafov for consultations about wet lab work. At different stages, this work has been done in collaboration with D. Fomin, L. Podolsky and M. Semionenkov. We are grateful to SmithKline Beecham and DOE for support of our trip to the *Gene-Finding and Gene Recognition Workshop* and to J. Fickett and D. Searls for help in arranging the trip.

REFERENCES

Aho, A., Hopcroft, J. and Ullman, J. (1976) *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, MA.

Brodsky, L. I., Drachev, A. L., Leontovich, A. M. and Feranchuk, S. I. (1993) A novel method of multiple alignment of biopolymer sequences. *BioSystems* **30,** 65–79.

Burset, M. and Guigo, R. (1996) Evaluation of gene structure prediction programs. *Genomics* **34,** 353–375.

D'Esposito, M., Mazzarells, R., Pengue, G., Jones, C., D'Urso, M. and Schlessinger, D. (1994) PCR-cased immortalization and screening of heirarchial pools of cDNAs. *Nucleic Acids Research* **22,** 4806–4809.

Fickett, J. W. and Tung, C.-S. (1992) Assessment of protein coding measure *Nucleic Acids Research* **20,** 6441–6450.

Finkelstein, A. V. and Roytberg, M. A. (1993) Computation of biopolymers: a general approach to different problems. *BioSystems* **30,** 1–19.

Gelfand, M. S. (1989) Statistical analysis of mammalian pre-mRNA splicing sites. *Nucleic Acids Research* **17,** 6369–6382.

Gelfand, M. S. (1990) Computer prediction of the exon–intron structure of mammalian pre-mRNAs. *Nucleic Acids Research* **18,** 5865–5869.

Gelfand, M. S. (1995) Prediction of function in DNA sequence analysis. *Journal of Computational Biology* **2,** 87–115.

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

Gelfand, M. S., Podolsky, L. I., Astakhova, T. V. and Roytberg, M. A. (1996a) Recognition of genes in human DNA sequences. *Journal of Computational Biology* **3,** 223–234.

Gelfand, M. S., Mironov, A. A. and Pevzner, P. A. (1996b) Gene recognition via spliced sequence alignment *Proceedings of the National Academy of Sciences of the U.S.A.* **93,** 9061–9066.

Korning, P. G., Hebsgaard, S. M., Rouze, P. and Brunak, S. (1996) Cleaning the GenBank *Arabidopsis thaliana* data set. *Nucleic Acids Research* **24,** 316–320.

Parrich, J. E. and Nelson, D. L. (1993) Methods for finding genes: a major rate-limiting step in positional cloning. *Gene. Analysis Techniques and Applications* **10,** 29–41.

Piterbarg, V. I. (1992) On the Distribution of the maximum similarity score for fragments of two random sequences. In *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, ed. S. Gindikin, Vol. 8, pp. 11–18. AMS, Providence, RI.

Roytberg, M. A., Astakhova, T. V. and Gelfand, M. S. (1996) An algorithm for highly specific recognition of protein coding regions. In *Genome Informatics 1996, Proceedings of the 7th Workshop on Genome Informatics*, Tokyo, December 1996, pp. 82–87.

Roytberg, M. A., Astakhova, T. V. and Gelfand, M. S. (1997) Algorithm for highly specific recognition of protein coding regions in DNA sequences of higher eukaryotes (in Russian). *Molekulyarnaya Biologiya* **31,** 25–31.

Sharp, P. A. (1994) Split genes and RNA splicing. *Cell* **77,** 805–815.

Snyder, E. E. and Stormo, G. D. (1993) Identification of coding regions in genomic DNA sequences: an application of dynamic programming and neural networks. *Nucleic Acids Research* **21,** 607–613.

Snyder, E. E. and Stormo, G. D. (1995) Identification of protein coding regions in genomic DNA. *Journal of Molecular Biology* **248,** 1–18.

Stormo, G. D. and Haussler, D. (1994) Optimally parsing a sequence into different classes based on multiple types of evidence. In *Proceedings of the 2nd International Conference on Intelligent Systems for Molecular Biology*, eds R. Altman, D. Brutlag, P. Karp, R. Lathrop and D. Searls, pp. 369–375. AAAI Press, Menlo Park, CA.

Xu, Y., Mural, R. J., Shah, M. and Uberbacher, E. C. (1994a) Recognizing exons in genomic sequence using GRAIL II. In *Genetic Engineering: Principles and Methods*, ed. J. Setlow, Vol. 16, pp. 241–253. Plenum Press, New York.

Xu, Y., Mural, R. J. and Uberbacher, E. C. (1994b) Constructing gene models from accurately predicted exons: an application of dynamic programming. *Computer Applications in the Biosciences* **10,** 613–623.