Las Vegas Algorithms for Gene Recognition: Suboptimal and Error-Tolerant Spliced Alignment

Sing-Hoi Sze[1] and Pavel A. Pevzner[1,2]

Departments of Computer Science[1] and Mathematics[2],
University of Southern California, Los Angeles, CA 90089.

Abstract

Recently, Gelfand, Mironov and Pevzner (*Proc. Natl. Acad. Sci. USA* (1996) **93**, 9061–9066) proposed a spliced alignment approach to gene recognition that provides 99% accurate recognition of human gene if a related mammalian protein is available. However, even 99% accurate gene predictions are insufficient for automated sequence annotation in large-scale sequencing projects and therefore have to be complemented by experimental gene verification. 100% accurate gene predictions would lead to a substantial reduction of experimental work on gene identification. Our goal is to develop an algorithm that either predicts an exon assembly with accuracy sufficient for sequence annotation or warns a biologist that the accuracy of a prediction is insufficient and further experimental work is required. We study suboptimal and error-tolerant spliced alignment problems as the first steps towards such an algorithm, and report an algorithm which provides 100% accurate recognition of human genes in 37% of cases (if a related mammalian protein is available). For 52% of genes, the algorithm predicts at least one exon with 100% accuracy.

1 Introduction

In 1995–1996, the area of gene prediction in eukaryotic DNA experienced a shift from statistical procedures to combinatorial algorithms based on similarity search. The similarity-based approach responds to the challenge of using GenBank for gene predictions and promises to be the method of choice in the future.

Conventional statistical approaches to gene recognition (Gelfand, 1990, Green and Hillier, 1991, Uberbacher and Mural, 1991, Guigo et al., 1992, Gelfand and Roytberg, 1993, Snyder and Stormo, 1993, Solovyev et al., 1994, Xu et al., 1994) use the facts that (i) coding function puts constraints on nucleotide sequences, and (ii) splicing machinery puts constraints on splicing sites. Thus one can measure these constraints using *coding potential* and *strength of splicing sites* (Fickett, 1995, Gelfand, 1995). However, none of these measures are sufficiently reliable: one either gets a lot of false exons or loses some true exons. Currently, correlation between predicted and actual genes is around 70%, with less than 50% of exons predicted correctly even for the best gene recognition programs which do not use similarity search (Burset and Guigo, 1996). Use of these predictions for sequence annotation in databases is hardly possible since it would create a clutter of unreliable feature tables. It indicates that conventional programs for gene recognition probably have reached the limit of accuracy. This not comforting conclusion was offset by a very important observation made by Burset and Guigo, 1996. They noticed that a similarity-based approach (Seely et al., 1990, Gish and States, 1993) greatly improves the accuracy of gene recognition: a simple use of sequence similarity in GeneParser (Snyder and Stormo, 1995) and GeneID led to an immediate increase in the *correlation coefficient* by 15% (see Burset and Guigo, 1996 for the definition of correlation coefficient).

Gelfand et al., 1996 suggested a *spliced alignment algorithm* for similarity-based gene recognition and demonstrated that many human genes can be accurately predicted even in the cases when only distantly related bacterial or yeast proteins are available. The spliced alignment algorithm provides 99% accurate recognition of human gene (i.e. average correlation coefficient of prediction is 99%) if a related mammalian protein is available. Although 99% accurate gene predictions look almost like an acme of perfection, they are not sufficiently reliable for sequence annotation. The question arises: *why do biologists want such inaccurate predictions if they cannot provide a reliable database annotation and protein sequence*? The answer to this question is simple: biologists use the predictions as clues to direct further experimental work. However, a biologist might prefer an accurate prediction of primers rather than an inaccurate prediction of entire genes. Since accurate prediction of primers can be done without accurate gene recognition (Stormo and Haussler, 1994, Roytberg et al., 1996), we feel that any gene predictions with less than 100% accuracy have rather limited applications. From this perspective, we are trying to develop an algorithm that either predicts an exon assembly with accuracy sufficient for sequence annotation (say in half of the cases) or warns a biologist that accuracy of a prediction is insufficient and further experimental work is required to complete the annotation (in this case, our goal is to provide biologists with accurate primer prediction). An 100% accurate gene prediction in half of the cases

This work was supported by Department of Energy Grant DE-FG02-94ER61919.

RECOMB 97, Santa Fe New Mexico USA

would greatly reduce experimental work on gene verification in large-scale sequencing projects.

Algorithms which provide a correct answer in some cases and have an option "no answer" in other cases are known as *Las Vegas algorithms* in computer science (Brassard and Bratley, 1996). The term "Las Vegas" was introduced to distinguish algorithms that "reply correctly when they reply at all" from *Monte Carlo algorithms* that occasionally make mistakes. Similar to many Las Vegas algorithms in computer science that benefit from the "no answer" option, Las Vegas algorithms for gene recognition use the "no answer" option to avoid unreliable predictions and benefit from reduction in experimental work in the "correct answer" cases. Conventional gene recognition algorithms may be compared to Monte Carlo algorithms with a very high error rate (Burset and Guigo, 1996), while the spliced alignment gene recognition algorithm (Gelfand et al., 1996) may be compared to Monte Carlo algorithms with a relatively low error rate. However, a biologist has no way to find out whether a given prediction is correct or not and therefore has to experimentally verify the Monte Carlo predictions anyway.

Although it is generally agreed that recognition of a human gene by a related yeast protein is a difficult problem, some people think that accurate recognition of a human gene by a related mouse protein is a simple problem. If a related mouse protein is available, a common practice in many sequencing centers is to look for local similarities between the mouse protein and the human gene and mark areas of high local similarity as potential exons. Although such approaches have proved to be successful for primer selection, it can hardly be used for the exact resolution of exon endpoints and reliable sequence annotation. Even worse, short exons frequently do not correspond to areas of best local similarities thus leading to inaccurate predictions. A number of examples from Gelfand et al., 1996 demonstrate that even for such close species as mouse and human, similarity may be distributed unevenly over the length of proteins and, as a result, exact predictions may be hard to obtain. Moreover, as Burset and Guigo, 1996 noted, even when an error-free cDNA encoded by a given genomic sequence is known, finding exons may be a non-trivial problem. For example, in the human gene HSTUBAG, an error-free cDNA does not allow one to find exon locations unambiguously by similarity search since the genomic DNA has two identical copies of the initial exon at positions 533 and 990 (with conventional splicing sites). As another example, in the human gene HUMNTRI, the initial exon (positions 2627–2801) can be replaced by two exons (positions 1934–1938 and positions 2632–2801) having the same concatenated sequence with conventional splicing sites.

We observed that in many cases of inexact predictions the score of the optimal spliced alignment almost coincides with the score of the alignment corresponding to the true structure. This naturally leads to the problem of finding *suboptimal spliced alignments*. We describe an algorithm to generate, evaluate and rank suboptimal spliced alignments. The key ingredient of our approach is an estimate of *prediction quality*.

Imagine that both human and mouse genomes have been sequenced and all mouse genes have been experimentally confirmed. Do we still need to verify predictions of human genes experimentally? Unfortunately, the answer is "yes" unless we can design an 100% accurate gene recognition algorithm. Below we describe a Las Vegas algorithm which provides extremely accurate gene recognition in 37% of cases (on a sample of all human genomic sequences with mammalian relatives). The key difference between the algorithm in Gelfand et al., 1996 and the new algorithm is that the new algorithm allows a biologist to avoid (or significantly reduce) experimental gene verification in these cases. For the new method, the accuracy of computational gene prediction is comparable with accuracy of experimental gene verification (experimental verification with further computational analysis would produce ambiguous results for some genes in the sample). The idea of our new algorithm comes from the observation that if the optimal spliced alignment is unique and has high prediction quality, the prediction is likely to be correct.

Performance of many gene recognition programs falls significantly in the presence of sequencing errors. We describe a version of the spliced alignment algorithm which uses alignment of nucleotide sequences instead of alignment of protein sequences. This version is insensitive to sequencing errors for close target proteins (although performance falls for distant target proteins). A simple but practically very important variant is the problem of searching for exon assembly given noisy cDNA or EST data (Boguski, 1995).

2 Spliced Alignment

2.1 Statement of Spliced Alignment Problem

The *gene recognition problem* is to find a gene encoded in experimentally determined *genomic sequence*. A *gene* is a set of *exons*, where each exon is a substring of the genomic sequence G. If $B = g_i \dots g_j$ and $B' = g_{i'} \dots g_{j'}$ are substrings of G (potential exons), denote $B \prec B'$ if $j < i'$ (i.e. if B ends before B' starts). A sequence $\Gamma = \{B_1, \dots, B_k\}$ of substrings of G is a chain if $B_1 \prec \dots \prec B_k$. Denote the concatenation of strings from the chain Γ by $\Gamma^* = B_1 \star \dots \star B_k$ and the score of an *optimal (global) alignment* between two strings G and T by $s(G, T)$ (see Waterman, 1995 for notion of alignment).

The *spliced alignment problem* is formulated as follows. Let $G = g_1 \dots g_n$ be a string called the *genomic sequence*, $T = t_1 \dots t_m$ be a string called the *target sequence*, and $\mathcal{B} = \{B_1, \dots, B_b\}$ be a set of substrings of G called *blocks*. Given G, T and $\mathcal{B}$, the spliced alignment problem is to find a chain Γ of strings from $\mathcal{B}$ such that $s(\Gamma^*, T)$ is maximum among all chains of blocks from $\mathcal{B}$. The genomic sequence G corresponds to a newly sequenced DNA fragment, the target sequence T corresponds to a related protein found by database search, and the set of blocks is defined based on analysis of splicing sites, open reading frames and codon usage.

2.2 Spliced Alignment Algorithm

Gelfand et al., 1996 decribed an algorithm for the spliced alignment problem using a dynamic programming approach. If $A = a_i \dots a_j \dots a_l$ is a string, denote its j-prefix by $A(j) = a_i \dots a_j$. For a block $B = g_i \dots g_j$, $first(B) = i$, $last(B) = j$. A chain $\Gamma = B_1 \star \dots \star B_k$ is said to end at $last(B_k)$. It ends before position i if $last(B_k) < i$. Define, for $1 \le i \le n+1$ and $0 \le j \le m$,

$$P(i, j) = \max_{\text{all chains } \Gamma \text{ ending before } i} s(\Gamma^*, T(j)).$$

$P(n+1, m)$ is the score of the optimal spliced alignment. If $first(B_k) \le i \le last(B_k)$, define the i-prefix of a chain

$\Gamma = B_1 \star \cdots \star B_k$ as $\Gamma^*(i) = B_1 \star \cdots \star B_{k-1} \star B_k(i)$. Define, for $first(B_k) \leq i \leq last(B_k) = l$ and $0 \leq j \leq m$,

$$BL(i,j,l) = \max_{\text{all chains } \Gamma \text{ ending at } l} s(\Gamma^*(i), T(j)).$$

P can be expressed in terms of BL as follows. For $0 \leq j \leq m$:

$$P(1,j) = j \cdot \Delta(-, t_j)$$
$$P(i,j) = \max \begin{cases} P(i-1,j) \\ BL(i-1,j,i-1) \end{cases} \quad \text{for } 1 < i \leq n+1$$

where $BL(i-1,j,i-1)$ is used when there exists chains ending at $i-1$. BL can be computed by the following recurrences:

$$BL(i,j,l) = \max \begin{cases} BL(i-1,j-1,l) + \Delta(g_i, t_j) \\ \quad \text{if } j \geq 1, \exists B : first(B) < i \leq last(B) = l \\ BL(i-1,j,l) + \Delta(g_i, -) \\ \quad \text{if } \exists B : first(B) < i \leq last(B) = l \\ P(i,j-1) + \Delta(g_i, t_j) \\ \quad \text{if } j \geq 1, \exists B : first(B) = i \leq last(B) = l \\ P(i,j) + \Delta(g_i, -) \\ \quad \text{if } \exists B : first(B) = i \leq last(B) = l \\ BL(i,j-1,l) + \Delta(-, t_j) \quad \text{if } j \geq 1 \end{cases}$$

where $\Delta(x,y)$ is the score of the alignment of letters x and y (symbol $-$ corresponds to an indel).

A block B is called *prime* if it contains all blocks ending at $last(B)$, i.e., for every other block B', $last(B) = last(B')$ implies $first(B) < first(B')$. Let c_p be $\frac{1}{n}\sum_{\text{prime } B} size(B)$. The described algorithm takes $O(mnc_p + mb)$ time and space to *compute* the spliced alignment (computing spliced alignment requires backtracking through the dynamic programming matrix and thus storing this matrix). This indicates that the major bottleneck in implementation is space (time is not a problem — for typical sizes it takes a few minutes on an UltraSparc).

2.3 Space-Efficient Spliced Alignment

To reduce space requirements we use Hirschberg's memory line technique (Hirshberg, 1975). Our approach follows the general idea of space-efficient algorithms for sequence alignment (Chao et al., 1994) but includes many complications. The algorithm consists of a forward pass scanning through the first half of the genomic sequence to the midpoint, followed by a backward pass through the second half to the same midpoint. The optimal score is computed at the midpoint. "Gluing" is performed at the midpoint and recursions on the left and right subproblems are performed as necessary.

In addition to P and BL for the forward pass in the first half of the genomic sequence, variables P' and BL' are used for the reversed backward computation in the second half. But, while computing BL', blocks starting at the same position are considered together instead of blocks ending at the same position as in BL. So, BL' is not exactly the backward correspondence that can be used to compute the optimal score at the midpoint. For technical reasons, we also define BF in the same way as BL' with the exception that blocks ending at the same position are considered.

Gluing at the midpoint is more complicated than in usual memory line approaches. If an optimal solution does not contain a block passing through the midpoint, values of P and P' are used. Either the solution is completely to the left of the midpoint (use P), or the solution is completely to the right of the midpoint (use P'), or the solution consists of a part to the left and another part to the right of the midpoint (use both P and P'). If an optimal solution contains a block passing through the midpoint, BL and BF are matched against each other. Recurrences for BL and BF are divided into subparts. For BL, BL_{align} gives the values of BL for spliced alignments ending in a match or mismatch, while BL_{indel} gives the values of BL for spliced alignments ending in an insertion or deletion. BL is the maximum of BL_{align} and BL_{indel}. The same is done for BF. There are two subcases in the gluing. When the genomic sequence at the midpoint is a match or mismatch, BL_{align} and BF_{align} at the midpoint are matched subtracting the extra match or mismatch counted respectively. When the genomic sequence at the midpoint is an insertion or deletion, BL_{indel} and BF_{indel} at the midpoint are matched subtracting the extra insertion or deletion counted.

2.4 Spliced Alignment with Amino Acid Sequence Comparison

The spliced alignment problem as described above captures the major computational challenges of the similarity search approach to exon assembly. However, in realistic situations, there exists important complications that do not seriously affect the running time of the algorithm although they greatly increase the complexity of software implementation.

It is well known that amino acid sequence comparisons can reveal similarities which can hardly be detected at the level of nucleotides. To achieve gene recognition with distant target proteins we use amino acid sequence comparison with the PAM120 matrix (Altschul, 1991) and $\Delta_{\text{indel}} = -3$. Since one codon specifies one amino acid, there are three possible sequences of codons for a given block, i.e., the first codon of a block can start from either the first, second or third nucleotide. These three variants correspond to three *reading frames* which need to be incorporated into the dynamic programming recurrences.

Instead of a single variable P, we need a set of variables, one for each prefix of a codon: null string, A, C, G, T, AA, AC, AG, AT, CA, CC, CG, CT, GA, GC, GG, GT, TA, TC, TG and TT. Each new block uses an appropriate set of variables for each reading frame. New BL values are computed from old values three positions before, instead of one. It is also necessary to detect in-frame stop codons and discard corresponding frames. We also take into account minimum intron size and require predicted exons in an assembly to be separated by at least 70 nucleotides.

2.5 Mosaic Effect

The dinucleotides AG and GT on the left and right sides of exons are highly conserved. We distinguish four different types of potential exons. The first type (*start block*) is of the form ATG...GT, where the underlined part forms an exon. The second type (*intermediate block*) is of the form AG...GT. The third type (*final block*) is of the form AG...(TAA|TAG|TGA). Finally, a *single exon gene* is of the form ATG...(TAA|TAG|TGA). Exons in a multi-exon gene are arranged as follows: a start block, followed by zero or more intermediate blocks, followed by a final block. This difference in block types has to be incorporated into the dynamic programming recurrences.

The simplest way to generate the set of blocks (potential exons) is to include all blocks of different types in $\mathcal{B}$ with the exception of blocks with stop codons in all three

frames. However, this approach creates a problem since a large number of short blocks is generated. First experiments with the spliced alignment algorithm revealed that incorrect predictions for distant targets are frequently associated with *mosaic effect* caused by very short potential exons. The problem is that these short exons can be easily combined together to fit whatever target protein. It is easier to "make up" a given sentence from a thousand random short strings than from a sample of the same number of longer strings. For example, with high probability, the phrase "filtration of candidate exons" can be made up from a sample of a thousand random two-letter strings ("fi", "lt", "ra", etc. are likely to be present in this sample). The probability that the same phrase can be made up from a sample of the same number of random five-letter strings is close to zero (even finding a string "filtr" in this sample is unlikely). This observation explains mosaic effect: if the number of short blocks is high, chains of these blocks can replace actual exons in spliced alignments, thus leading to predictions with unusually large number of short exons.

2.6 Filtration of Candidate Exons

Each candidate exon is assigned scores reflecting the *strength* of its donor and acceptor sites and *coding potential*. To offset mosaic effect we use strong filtration for short exons and weaker filtration for long exons. Separate cutoffs are used for each of the following block types: single, start, short intermediate (shorter than 100 nucleotides), long intermediate, and final; in each of the following measures: acceptor score (AC), donor score (DO), total site score (AC+DO), coding potential (CO), and total score (AC+DO+CO). A block is retained only if its scores are above the cutoffs, leaving 3% of candidate short intermediate exons and 15% of long intermediate exons. We also observed that coding potential can be ignored since it almost does not affect filtration efficiency. Since intermediate exons shorter than 15 nucleotides are never observed, very short intermediate blocks can be filtered out.

2.7 Spliced Alignment Quality

A serious obstacle on the way towards improving prediction accuracy is the lack of statistical theory for spliced alignment. Even for the relatively simple case of the score distribution of alignment with a *single* block, the situation is far from simple (Goldstein and Waterman, 1994). On the other hand, estimates for the statistical significance of local alignments are rather time-consuming and may not work well for short exons (Vingron and Waterman, 1994). Lack of statistical theory for spliced alignment forces us to look for an alternative estimate of spliced alignment quality.

Given sequences $A = a_1 \dots a_M$ and $B = b_1 \dots b_N$, a *fit* of A on a substring $b_i \dots b_j$ of B is simply a (global) alignment of A and $b_i \dots b_j$. Spliced alignment of a genomic sequence with k exons defines k fits of exons on a target protein called the *spliced alignment fit* of an exon on a target protein. An *optimal fit* of A on B is a fit with the highest score over all (i, j). Ideally, the optimal fit and the spliced alignment fit of each exon on the target protein should align the exon onto the same region. In this case,

$$\textit{exon quality} = \frac{\text{spliced alignment fit score of exon}}{\text{optimal fit score of exon}}$$

and

$$\textit{spliced alignment quality} = \frac{\text{spliced alignment score}}{\sum_{\text{exon}} \text{optimal fit score of exon}}$$

should be close to 100% for reliable gene predictions.

There are complications with the notion of spliced alignment quality. Since each exon should be aligned in the correct frame, there may be initial or terminal nucleotides that do not form codons. Such nucleotides are ignored in computing the optimal fit. An adjustment in the spliced alignment score is therefore necessary to correct for formation of codons from different blocks.

3 Suboptimal Spliced Alignment

We have observed that in many cases of imperfect predictions the score of the biologically correct solution is only slightly below the score of the optimal spliced alignment. Such cases may be caused by alternative splicing, mosaic effect, using inappropriate scoring matrices, etc. One way to compensate for these problems is to examine suboptimal spliced alignments. The suboptimal spliced alignment problem has a different flavor than the classical suboptimal alignment problem (Waterman, 1983, Naor and Brutlag, 1993). We are not interested in different high-score alignment paths through the dynamic programming matrix but rather in different blocks which make up suboptimal alignments. More precisely, for any block assembly, we compute the optimal alignment between the concatenated sequence and the target sequence, which we will refer to as *suboptimal spliced alignment*.

To find out whether a block B belongs to a suboptimal spliced alignment, we compute the score of an optimal alignment containing B (*block score* of B). A subset of blocks with block scores above a threshold (*suboptimal blocks*) are retained, and a search procedure is used to find all suboptimal assemblies of blocks from this subset with spliced alignment scores above threshold SC.

The algorithm for finding the score of an optimal alignment containing a given block uses a full forward pass and a full backward pass to accumulate P and P' values (similar to the forward and backward passes of the space-efficient algorithm for optimal alignment except that computation ranges over the entire genomic sequence for both passes). For each block B, appropriate P values are used to initialize forward alignment scores at the left of B, while appropriate P' values are used to initialize backward alignment scores at the right of B. A forward pass is performed from the left of B using simple recurrences until the right of B is reached. Gluing is then carried out at the right of B using these forward and backward scores.

Although we can find all chains of suboptimal blocks with spliced alignment scores above SC by exhaustive search, this approach is rather time-consuming. Even with some branch-and-bound optimization, suboptimal spliced alignments for genomic sequences 10,000 nucleotides long using 50 suboptimal blocks take a few hours on an UltraSparc.

To avoid generating *all* suboptimal block assemblies, we instead, for each suboptimal block B, find the best block assemblies containing B. Of course, when the real biological solution is far from optimal, it may not correspond to a best block assembly for any block. But, if this is the case, the real biological solution would be hard to find anyway.

For each block B with score SC (optimal score containing B), we outline a procedure to compute all optimal block assemblies containing B. Let $\mathcal{B}'$ be the set of blocks with scores at least SC. We use similar procedures as before except that blocks in conflict with B (blocks that cannot form a solution with B) are excluded. A forward pass and a backward pass are used to accumulate P and P' values using only

blocks in $\mathcal{B}'$ and requiring that B be in an alignment. Gluing is performed for each block B' in $\mathcal{B}'$ finding the score SC' of B' considering only solutions containing B (alternatively, it is the optimal score containing B and B'). Only blocks in $\mathcal{B}'$ that have the highest SC' $(= SC)$ can be in an optimal solution containing B. We subject these blocks to the exhaustive search procedure, giving all optimal solutions containing B.

We are now able to find a set of suboptimal solutions which is likely to contain the correct exon assembly. The problem is how to find the correct solution among the suboptimal ones. Among many potential ranking strategies, we have chosen to *rank* suboptimal solutions in decreasing order of *adjusted spliced alignment score* AS:

AS = spliced alignment score · spliced alignment quality.

4 Las Vegas Algorithms for Gene Recognition: 100% Accurate Prediction in 37% of Cases

We now describe a Las Vegas algorithm for gene recognition. The algorithm makes use of the notion of competing solutions, which is defined as follows. Let S be the set of suboptimal solutions having spliced alignment scores above a threshold. For a given parameter Q, define the set of competing solutions $CS(Q)$ to be all solutions in S in which all predicted exons have exon quality $\geq Q$. Intuitively, we want to find a parameter Q such that $CS(Q)$ is a small nonempty set and the true gene structure is guaranteed to be in $CS(Q)$. There are tradeoffs associated with the choice of Q. When Q is very high (i.e. 100%), $CS(100)$ will be empty in many cases. When Q is low (i.e. 80%), $CS(80)$ might be too large, but there will be far less cases when $CS(80)$ is empty.

In reality, there are always exceptional cases and the true gene structure is not always among competing solutions. Figure 1 shows the percentages of cases when the correct solution is among the top n competing solutions in $CS(100)$, $CS(95)$ or S on a sample of all complete human genomic sequences with mammalian relatives (see below). In 97 out of 253 cases (38%), the set $CS(100)$ is empty. In the remaining 62% of cases, the size of $CS(100)$ varies from 1 to 5 and the true solution is in $CS(100)$ in all but 8 cases. In 54 out of 253 cases (21%), the set $CS(95)$ is empty. In the remaining 79% of cases, the size of $CS(95)$ varies from 1 to 25 and the true solution is in $CS(95)$ in all but 14 cases. Finally, in 9 out of 253 cases (4%), the set S is empty. In the remaining 96% of cases, the size of S varies from 1 to 94 and the true solution is in S in all but 38 cases.

We develop a strategy to determine simple conditions for 100% accurate gene prediction. These conditions are (i) $CS(100)$ contains only one solution, and (ii) this solution is the only optimal spliced alignment. When a solution meets all these requirements, we declare it to be the actual gene structure. Under these conditions there are no mistakes in our sample (predicted experimentally proven alternative splicing does not count as a mistake). In 37% of 253 genes, the above conditions generate gene predictions which are perhaps at least as accurate as experimental gene verification.

In long multi-exon genes, it is frequently the case that some exons have exon quality below 100%. To complement the above approach, we would like to accurately predict some exons in cases when we cannot accurately predict entire gene structure. A natural approach is to find a set of solutions such that exons appearing in all solutions in the set are true exons. The following approach works for our sample: Find the lowest spliced alignment score LS among all solutions in $CS(100)$. Consider the set of *all* suboptimal solutions having spliced alignment scores $\geq LS$ and declare exons appearing in all such solutions to be true exons. This simple algorithm generates at least one sure exon in 132 genes (52%).

5 Error-Tolerant Spliced Alignment

Every practical system for gene recognition should be tolerant to DNA sequencing errors. The best currently available system for gene prediction in the presence of errors was described by Xu et al., 1995. Insertions and deletions of nucleotides cause major difficulties for conventional gene prediction programs since they trigger a frameshift. Substitutions and sequencing ambiguities are easier to handle unless they bring a stop codon in the middle of an exon or "destroy" the conservative donor/acceptor dinucleotides. However, these events are relatively rare.

Frameshift sequencing errors cause difficult problems for the spliced alignment algorithm with amino acid sequence comparison. Although algorithms for correcting sequencing errors (States and Botstein, 1991, Guan and Uberbacher, 1996) work well for detecting protein similarities, they can hardly be incorporated into the spliced alignment algorithm. On the other hand, spliced alignment with *nucleotide sequence comparison* ($\Delta_{\text{match}} = 1$, $\Delta_{\text{mismatch}} = -1$, $\Delta_{\text{indel}} = -2$) is extremely tolerant to sequencing errors. One can even view the nucleotide sequence of a target protein as cDNA of a genomic sequence with a very high rate of sequencing errors. Adding 1 to 3% of "real" sequencing errors on top of these "mutation-caused" errors almost does not affect accuracy of spliced alignment. However, prediction accuracy for distant targets is lower in the nucleotide version than in the amino acid version (since conservation of protein sequences is more pronounced than conservation of nucleotide sequences). Solutions found by the nucleotide version of the spliced alignment algorithm may not be biologically plausible since the concatenated sequence length may not be a multiple of three and a solution may contain stop codons in frame. However, error corrections of frameshifts in a solution can be found by associating frameshifts with indels in the spliced alignment.

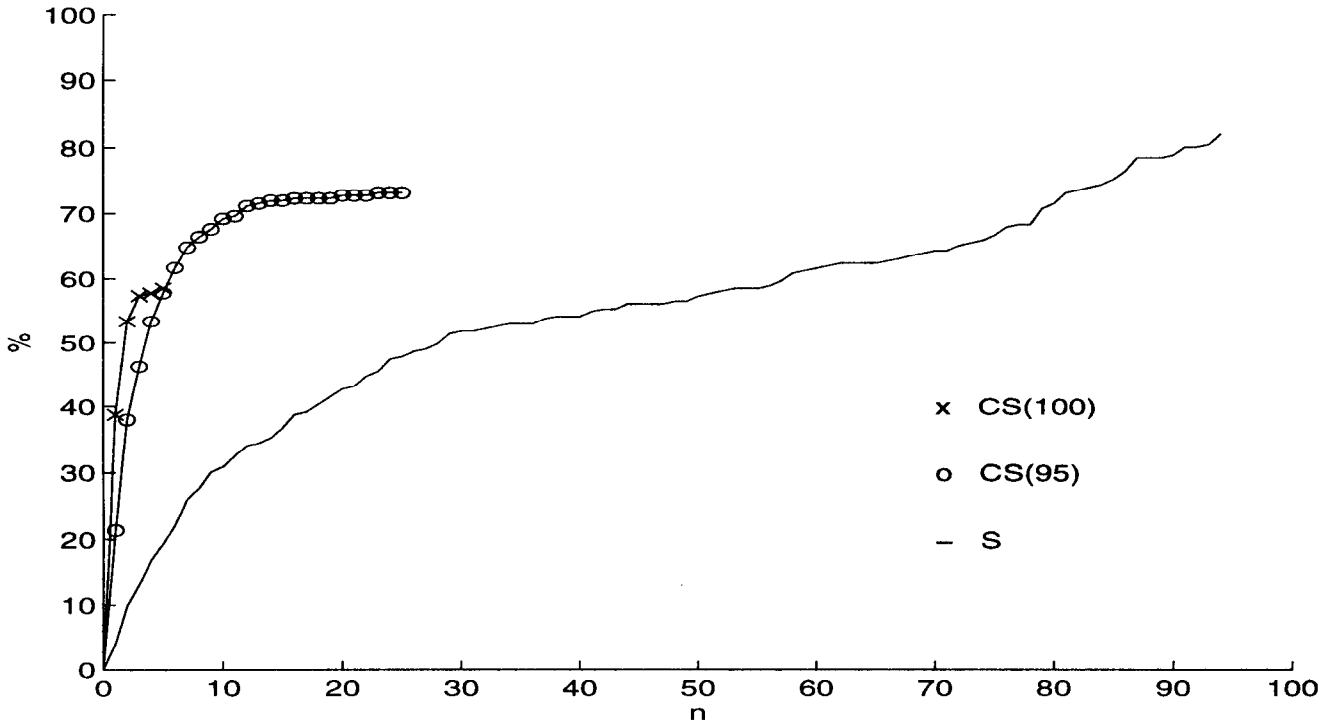
We also tested 171 genes with the nucleotide version of the Las Vegas algorithm. Sure gene structures are obtained in 28% of cases and at least one sure exon is found in 43% of genes.

6 Data Sets

We used a test sample of all complete human genomic sequences from GenBank which have related proteins from other species (Gelfand, 1996, personal communication) and a subsample of this sample from Gelfand et al., 1996. For each genomic sequence, a list of related proteins was constructed using the ENTREZ database of BLAST similarity scores. Proteins having the highest BLAST scores were retained in each of the following categories: mammals, warm-blooded organisms, cold-blooded organisms, chordates, insects, animals different from the above, plants, fungi including yeast and other unicellular fungi, eukaryotes different from the above, organelles, and finally bacteria including Archaea.

Four standard parameters were used to evaluate agreement between the predicted and true gene structures. Denote the number of correctly predicted coding (true posi-

Figure 1: Percentages of cases when the correct solution is among the top n competing solutions in sets $CS(Q)$ (on a sample of all complete human genomic sequences with mammalian relatives). Set S consists of all suboptimal spliced alignments with scores within 5% of optimal.



tive) and non-coding (true negative) nucleotides by TP and TN respectively. Denote the number of missed coding nucleotides (false negative) and the number of non-coding nucleotides predicted to be coding (false positive) by FN and FP respectively. Performance of gene recognition algorithms is characterized by the *correlation coefficient*:

$$CC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TN + FN)(TP + FN)(TN + FP)}}.$$

7 Results

For spliced alignment with mammalian targets using amino acid sequence comparison (Table 1), prediction accurary was similar to that in Gelfand et al., 1996. An important observation is that spliced alignment quality values strongly correlated with CC, thus indicating that *spliced alignment quality is a good measure of prediction accuracy.* There were 8 cases where the real solution did not coincide with the 1st ranked suboptimal solution, 4 cases where the real solution was not found among the first five solutions, and only one case where the real solution was not found among all suboptimal solutions. In the case of HUMEMBPA, the first exon is very short. In the cases of HUMI309 and HUMPPPA, the spliced alignment fit score of the final exon is very low. Finally, in the case of HUMPLPSPC, we find an experimentally proven alternative splicing.

High exon quality values of the 1st ranked suboptimal solution were associated with either true exons or exons having significant overlaps with true exons. Low exon qualities in the cases of HUMCAPG, HUMCBRG, HUMEMBPA and HUMI309 corresponded to wrongly predicted exons. There were also cases when exon quality values were low for true exons: in the case of HUMGHN, the first exon is very short (exon quality 75%); in the case of HUMIL8A, the last exon is short (exon quality 86%); in the case of HUMPPPA, the last exon fits poorly (exon quality negative).

For spliced alignment with non-mammalian targets using amino acid sequence comparison (Table 2), spliced alignment quality values also estimated CC well. Let $max5C$ and $maxC$ be the maximum correlation coefficient among the top five suboptimal alignments and among all suboptimal alignments respectively. The algorithm performed well in bringing good solutions into the set of suboptimal solutions (average optimal CC was 86%, while average $max5C$ was 89% and average $maxC$ was 92%).

Although exon quality values were not as high as before, there were usually at least a few exons that had high values. Exon qualities were less reliable for short exons (HUMBNPA, HUMEMBPA, HUMGAD45A, HUMUBILP, HUMG0S19B with target CHKCYTO, HUMHLL4G with target CELBGB, etc.).

For spliced alignment with mammalian targets using nucleotide sequence comparison (Table 3), only slight decreases in prediction accuracy were observed when compared to the amino acid version. Exon quality values were slightly lower in most cases. There were two cases when there were significant drops in CC: HUMIL4A and HUMPPPA. There was also one case (HUMI309) when there was slight increase in CC.

For spliced alignment with non-mammalian targets using nucleotide sequence comparison (data not shown), the results deteriorated when compared to the amino acid version and CC fell by 10 to 30% in many cases. However, the predictions were still at least as accurate as those given by conventional gene recognition programs (average CC was

72%). Although exon qualities deteriorated significantly, in many cases we obtained at least one correct exon. In terms of CC, there were still good predictions with $CC \geq 95\%$ in 12 cases.

8 Discussion

Gelfand et al., 1996 listed the suboptimal and error-tolerant spliced alignment problems as two important open problems on gene recognition. This paper addresses these problems and further attempts to estimate the quality of spliced alignment. Since no results on the statistical significance of spliced alignment are known, we introduce a combinatorial measure (spliced alignment quality) and demonstrate that predictions given by spliced alignment are very reliable when spliced alignment quality is high. Based on the suboptimal spliced alignment approach, we design the first Las Vegas type algorithm for gene recognition which offers the possibility of a significant reduction in the amount of experimental work for gene verification.

9 Acknowledgments

We are indebted to Mikhail Gelfand, Andrey Mironov and Mikhail Roytberg for many helpful discussions. Mikhail Gelfand provided us with the large sample of human genomic sequences. Andrey Mironov shared with us his experience in gene recognition algorithms and software. Mikhail Roytberg made important emphasis on the correlation between quality of predictions and distribution of suboptimal spliced alignments. We are also grateful to Sampath Kannan, George Komatsoulis and Sergei Rahmanov for many helpful comments.

References

Altschul S.F. (1991) Amino acid substitution matrices from an information theoretic perspective. *J. Mol. Biol.*, **219**, 555–565.

Boguski M.S. (1995) The turning point in genome research. *Trends in Biochemical Sci.*, **20**, 295–296.

Brassard G. and Bratley P. (1996) *Fundamentals of Algorithmics*. Prentice-Hall.

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

Chao K.-M., Hardison R.C. and Miller W. (1994) Recent developments in linear-space alignment methods: a survey. *J. Comp. Biol.*, **1**, 271–291.

Fickett J.W. (1995) The gene identification problem: an overview for developers. *Computers & Chemistry*, **20**, 103–118.

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

Gelfand M.S. (1995) Prediction of function in DNA sequence analysis. *J. Comp. Biol.*, **2**, 87–115.

Gelfand M.S., Mironov A.A. and Pevzner P.A. (1996) Gene recognition via spliced sequence alignment. *Proc. Natl. Acad. Sci. USA*, **93**, 9061–9066.

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

Gish W. and States D.J. (1993) Identification of protein coding regions by database similarity search. *Nature Genet.*, **3**, 266–272.

Goldstein L. and Waterman M.S. (1994) Approximations to profile score distribution. *J. Comp. Biol.*, **1**, 93–104.

Green P. and Hillier L. (1991) Gene identification using GeneFinder. *Genome Sequencing III* (Hilton Head, SC), p.22.

Guan X. and Uberbacher E.C. (1996) Alignments of DNA and protein sequences containing frameshift errors. *Comp. Appl. in Biological Sci.*, **12**, 31–40.

Guigo R., Knudsen S., Drake N. and Smith T. (1992) Prediction of gene structure. *J. Mol. Biol.*, **226**, 141–157.

Hirshberg D.S. (1975) A linear space algorithm for computing maximal common subsequences. *Comm. of ACM*, **18**, 341–343.

Naor D. and Brutlag D. (1993) On suboptimal alignments of biological sequences. *Lecture Notes in Computer Science, Combinatorial Pattern Matching*, **684**, 179–196.

Roytberg M.A., Astakhova T.V. and Gelfand M.S. (1996) Algorithm for highly specific recognition of protein-coding regions in higher eukaryote DNA sequences. *Molek. Biol.* (in press).

Seely O., Feng D.F., Smith D.W., Sulbach D. and Doolittle R.F. (1990) Construction of a facsimile data set for large genome sequence analysis. *Genomics*, **8**, 71–82.

Snyder E.E. and Stormo G.D. (1993) Identification of coding regions in genomic DNA sequences. *Nucleic Acids Res.*, **21**, 607–613.

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

Solovyev V.V., Salamov A.A. and Lawrence C.B. (1994) Predicting internal exons by oligonucleotide composition and discriminant analysis of spliceable open reading frames. *Nucleic Acids Res.*, **22**, 5156–5163.

States D.J. and Botstein D. (1991) Molecular sequence accuracy and the analysis of protein coding regions. *Proc. Natl. Acad. Sci. USA*, **88**, 5518–5522.

Stormo G.D. and Haussler D. (1994) Optimally parsing a sequence into different classes based on multiple types of evidence. *Proc. 2nd Int. Conf. on Intelligent Systems for Mol. Biol.*, 369–375.

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

Vingron M. and Waterman M.S. (1994) Sequence alignment and penalty choice: review of concepts, case studies and implications. *J. Mol. Biol.*, **235**, 1–12.

Waterman M.S. (1983) Sequence alignments in the neighborhood of the optimum with general application to dynamic programming. *Proc. Natl. Acad. Sci. USA*, **80**, 3123–3124.

Waterman M.S. (1995) *Introduction to Computational Biology*. Chapman & Hall.

Xu Y., Einstein J.R., Mural R.J., Shah M. and Uberbacher E.C. (1994) An improved system for gene recognition and gene modeling in human DNA sequences. *Proc. 2nd Int. Conf. on Intelligent Systems for Mol. Biol.*, 376–383.

Xu Y., Mural R.J. and Uberbacher E.C. (1995) Correcting sequencing errors in DNA coding regions using a dynamic programming approach. *Comp. Appl. Biosci.*, **11**, 117–124.

Table 1: Spliced alignment with mammalian targets (amino acid sequence comparison)

(no) Number. (genomic) Genomic sequence. (target) Target sequence. (optC) Correlation coefficient of optimal solution. (optQ) Spliced alignment quality of optimal solution. (subC) Correlation coefficient of 1st ranked suboptimal solution. (subQ) Spliced alignment quality of 1st solution. (subexonQ) Exon qualities of 1st solution in order: [+] All exon predictions are correct with exon quality 100% [*] Real exon [–] Overlap with real exon(s) [.] Low exon quality — spliced alignment fit score ≤ 0. (max5C) Maximum correlation coefficient among first five ranked solutions. (maxC) Maximum correlation coefficient among all exon assemblies considered. (Nasm) Total number of exon assemblies considered.

no	genomic	target	optC	optQ	subC	subQ	subexonQ	max5C	maxC	Nasm
1	humapexn	btbap1r	100	100	100	100	+	100	100	86
2	humazcdi	mmnel	92	91	100	97	(100*,94*,100*,99*,94*)	100	100	86
3	humbhsd	bt3bhsd	100	100	100	100	+	100	100	90
4	humbnpa	pigbnp	100	100	100	100	+	100	100	88
5	humcapg	mmcatheg	99	98	99	98	(100*,100*,100*,100*,98–,.)	100	100	81
6	humcbrg	pig20bhd	97	98	97	98	(100*,100*,100–,.)	100	100	90
7	humchymb	dogchamc	100	100	100	100	(100*,100*,100*,100*,99*)	100	100	87
8	humcox5b	ratdccovb	100	100	100	100	+	100	100	86
9	humcspa	musccpa	100	100	100	100	(97*,100*,99*,100*,99*)	100	100	83
10	humembpa	s33799	93	99	93	99	(20,100*,100*,100*,100*)	94	100	87
11	humfabp	ratfabpx	100	100	100	100	+	100	100	92
12	humg0s19a	mmscimip	100	100	100	100	+	100	100	87
13	humg0s19b	musstcpa	100	100	100	100	+	100	100	85
14	humgad45a	crugad45a	100	100	100	100	+	100	100	90
15	humgare	pytgcbr	99	100	99	100	(100*,100*,100*,100–,100*)	100	100	83
16	humghn	bovgrowp	100	99	100	99	(75*,100*,99*,100*,100*)	100	100	88
17	humhll4g	ratbpgal	100	100	100	100	+	100	100	91
18	humhmg2a	pighmg2	100	100	100	100	+	100	100	85
19	humi309	musstcpb	66	94	66	94	(100*,100*,9)	70	100	89
20	humibp3	ratigfbp3a	100	100	100	100	+	100	100	88
21	humigera	dogierac	100	100	100	100	(100*,100*,100*,100*,100*)	100	100	82
22	humil1b	rabil1b	100	100	100	100	(100*,96*,100*,100*,100*,100*)	100	100	80
23	humil4a	ssilk4	100	98	100	98	(98*,100*,100*,97*)	100	100	88
24	humil5a	b39881	100	99	100	99	(100*,100*,100*,98*)	100	100	89
25	humil8a	rabnap1	100	100	100	100	(100*,100*,100*,86*)	100	100	86
26	humil9a	musp40m	98	100	98	100	(100–,100*,100*,100*,100*)	100	100	90
27	humkal2	cfkallik	100	100	100	100	(100*,100*,100*,97*,100*)	100	100	84
28	hummif	musgia	100	100	100	100	+	100	100	89
29	hummis	bovmis	100	100	100	100	+	100	100	83
30	humops	cfopsin	100	100	100	100	+	100	100	88
31	humpald	oatthyre	100	100	100	100	+	100	100	96
32	humpf4v1a	ratpf4	100	100	100	100	+	100	100	53
33	humpgammg	rnpgmut	100	100	100	100	+	100	100	91
34	humplpspc	mvspc	98	100	98	100	(100*,100*,100*,100*,100–)	98	100	88
35	humpppa	bovsmplsm	89	93	89	93	(100*,90–,.*)	89	95	90
36	humrps17a	crurps17	100	100	100	100	+	100	100	89
37	humrps6b	ratrps6	100	100	100	100	+	100	100	86
38	humsaa	musamyaff	100	100	100	100	(98*,100*,100*)	100	100	89
39	humsftp1a	s48768	100	100	100	100	+	100	100	89
40	humtfpb	rabrtf	100	99	100	99	(99*,96*,100*,100*,100*,100*)	100	100	82
41	humthy1a	rnthycsg	100	100	100	100	+	100	100	87
42	humtnfba	muslta	100	100	100	100	(100*,100*,100*)	100	100	89
43	humtnfx	cattnfaa	100	100	100	100	+	100	100	85
44	humtpalbu	rnvegp2b	100	100	100	100	+	100	100	85
45	humtrpy1b	dogmctrpa	100	100	100	100	(97*,100*,100*,100*,100*)	100	100	88
46	humubilp	musubilp	100	100	100	100	+	100	100	92
47	humv2r	ssvrv2a	100	100	100	100	(96*,100*,100*)	100	100	84

Table 2: Spliced alignment with non-mammalian targets (amino acid sequence comparison)

(tt) Target type: 'V' — vertebrate, 'I' — invertebrate, 'E' — other eukaryote, 'P' — prokaryote.

no	genomic	tt	target	optC	optQ	subC	subQ	subexonQ	maxSC	maxC	Nasm
1	humapexn	I	drorrp	68	71	68	71	(.,.,.–,52–,99*,100*)	69	77	78
2	humazcdi	V	ggu15155	100	98	100	98	(100*,99*,88*,100*,100*)	100	100	78
3	humbhsd	P	noccdh	83	85	93	89	(100–,4,64,99–,95*)	93	93	76
4	humbnpa	V	anf_chick	100	97	100	97	(98*,100–,55*)	100	100	88
6	humcbrg	P	svpks	38	71	85	86	(93*,79*,93–,19)	85	91	87
10	humembpa	V	pwlec	65	82	78	85	(79,75–,47–,100–,100–,75*)	81	81	81
11	humfabp	V	xelifabp	100	100	100	100	+	100	100	92
11		E	scmfabp14	100	92	100	92	(100*,100*,65*,100*)	100	100	94
12	humg0s19a	V	chkcyto	97	95	97	95	(100–,98*,89*)	100	100	90
13	humg0s19b	V	chkcyto	68	91	68	91	(100,100,45,98*,90*)	68	87	81
14	humgad45a	V	xelrbl12x	78	84	77	90	(70*,100*,94–,42)	80	87	84
15	humgare	V	xelnpypyy	62	95	62	95	(12,100*,95–,97,98–)	79	83	83
15		I	dmdoprec	77	78	72	84	(24,38–,100–,93*,100,33–,97*)	72	85	81
17	humhll4g	V	leg6_chick	98	99	98	99	(50,100*,100*,100*)	100	100	91
17		V	xellbl	99	95	99	95	(36–,100*,97*,100*)	99	99	89
17		I	celbgb	58	77	58	77	(56–,81,.,93,81–,69,97*,51*)	58	66	78
18	humhmg2a	V	xelhmg2a	100	100	100	100	+	100	100	86
18		I	dmu13881	74	86	67	88	(.,61,97–,100*,100*,97*)	74	75	79
18		E	yscmaknhp	53	97	53	97	(100–,99–,62)	56	66	90
19	humi309	V	chkcyto	100	100	100	100	+	100	100	92
25	humil8a	V	chkrsvind	93	95	93	95	(100*,98*,99*,10)	93	100	85
28	hummif	V	chklmif	100	100	100	100	+	100	100	89
30	humops	V	chkrdpsn	100	100	100	100	+	100	100	86
30		V	s49004	100	100	100	100	+	100	100	89
30		I	s53494	98	98	98	98	(98*,100*,97*,98–,100*)	98	98	82
30		P	hhrhod	45	78	45	78	(80,49–,86,87–,100–,50,50–)	47	68	91
31	humpald	V	gdtrthy	100	99	100	99	(94*,100*,100*,100*)	100	100	97
31		V	trtranst	100	99	100	99	(95*,100*,100*,100*)	100	100	91
33	humpgammg	P	stmpgm	100	100	100	100	(100*,100–,100*)	100	100	93
35	humpppa	V	larpyy	100	89	63	96	(100*,100,30)	100	100	86
36	humrps17a	V	ggrps17	100	98	100	98	(.*,100*,100*,100*,100*)	100	100	90
36		I	drorps17	100	100	100	100	+	100	100	88
36		E	neucrp3	100	100	100	100	(100*,100*,100*,100*,97*)	100	100	89
37	humrps6b	V	xelrps6x	100	100	100	100	+	100	100	85
37		I	drorps6x	100	100	100	100	(100*,100*,100*,100*,100*,96*)	100	100	84
37		E	ysprps6a	100	100	100	100	+	100	100	83
37		P	ecrpsfri	47	78	47	78	(100*,69,100–,56–,60–,80–)	52	71	89
41	humthy1a	V	chkthy1gp	100	100	100	100	+	100	100	91
46	humubilp	I	tpubiexta	72	75	65	84	(55*,95*,76*,100–,100,67)	77	77	80

Table 3: Spliced alignment with mammalian targets (nucleotide sequence comparison)

no	genomic	target	optC	optQ	subC	subQ	subexonQ	max5C	maxC	Nasm
1	humapexn	btbap1r	100	100	100	100	+	100	100	90
2	humazcdi	mmnel	95	91	97	93	(89*,100*,100*,80*,70-,60)	97	97	80
3	humbhsd	bt3bhsd	100	100	100	100	+	100	100	91
4	humbnpa	pigbnp	100	100	100	100	+	100	100	91
5	humcapg	mmcatheg	97	99	97	99	(100*,100*,100*,100*,99-,25)	99	100	87
6	humcbrg	pig20bhd	97	97	97	97	(100*,100*,100 ,.)	98	100	92
7	humchymb	dogchamc	99	100	99	100	(100*,100*,100*,100*,100-)	100	100	90
8	humcox5b	ratdccovb	100	100	100	100	+	100	100	90
9	humcspa	musccpa	100	99	100	99	(100*,100*,99*,99*,100*)	100	100	80
10	humembpa	s33799	Nucleotide sequence not available							
11	humfabp	ratfabpx	100	100	100	100	(100*,100*,100*,96*)	100	100	94
12	humg0s19a	mmscimip	100	99	100	99	(98*,100*,100*)	100	100	91
13	humg0s19b	musstcpa	100	100	100	100	+	100	100	89
14	humgad45a	crugad45a	100	100	100	100	+	100	100	93
15	humgare	pytgcbr	99	100	99	100	(100*,100*,100*,100-,100*)	99	100	81
16	humghn	bovgrowp	98	99	98	99	(71,100*,100*,100*,100*)	100	100	88
17	humhll4g	ratbpgal	100	100	100	100	+	100	100	94
18	humhmg2a	pighmg2	100	100	100	100	+	100	100	88
19	humi309	musstcpb	70	84	70	84	(100*,95*,.,17-)	70	93	99
20	humibp3	ratigfbp3a	100	100	100	100	+	100	100	91
21	humigera	dogierac	100	97	100	97	(100*,82*,100*,94*,100*)	100	100	87
22	humil1b	rabil1b	100	100	100	100	+	100	100	85
23	humil4a	ssilk4	86	98	86	98	(98*,100*,100-,.,100*)	93	98	89
24	humil5a	b39881	Nucleotide sequence not available							
25	humil8a	rabnap1	100	99	100	99	(100*,100*,100*,40*)	100	100	93
26	humil9a	musp40m	98	100	98	100	(100-,100*,100*,100*,100*)	98	100	92
27	humkal2	cfkallik	100	100	100	100	+	100	100	85
28	hummif	musgia	100	100	100	100	+	100	100	90
29	hummis	bovmis	100	100	100	100	(97*,100*,100*,100*,100*)	100	100	85
30	humops	cfopsin	100	100	100	100	+	100	100	91
31	humpald	oatthyre	100	100	100	100	+	100	100	90
32	humpf4v1a	ratpf4	100	99	100	99	(100*,98*,100*)	100	100	91
33	humpgammg	rnpgmut	100	100	100	100	+	100	100	91
34	humplpspc	mvspc	98	100	98	100	(100*,100*,100*,100*,100-)	98	100	88
35	humpppa	bovsmplsm	75	98	75	98	(100*,92-)	85	86	90
36	humrps17a	crurps17	100	100	100	100	+	100	100	89
37	humrps6b	ratrps6	100	100	100	100	+	100	100	84
38	humsaa	musamyaff	100	100	100	100	+	100	100	91
39	humsftp1a	s48768	100	100	100	100	+	100	100	91
40	humtfpb	rabrtf	100	99	100	99	(100*,93*,100*,100*,100*,100*)	100	100	81
41	humthy1a	rnthycsg	100	100	100	100	+	100	100	92
42	humtnfba	muslta	100	100	100	100	+	100	100	91
43	humtnfx	cattnfaa	100	100	100	100	+	100	100	89
44	humtpalbu	rnvegp2b	100	100	100	100	+	100	100	85
45	humtrpy1b	dogmctrpa	100	100	100	100	+	100	100	90
46	humubilp	musubilp	100	100	100	100	+	100	100	90
47	humv2r	ssvrv2a	100	100	100	100	(100*,100*,99*)	100	100	89