

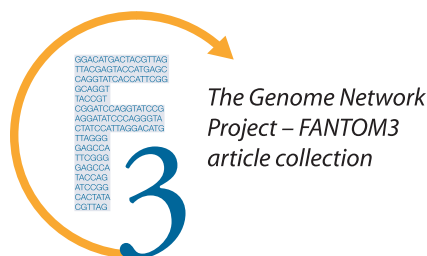
SPA: A Probabilistic Algorithm for Spliced Alignment

Erik van Nimwegen^{1,2*}, Nicodeme Paul^{1,2}, Robert Sheridan^{3‡}, Mihaela Zavolan^{1,2}

1 Biozentrum, University of Basel, Basel, Switzerland, **2** Swiss Institute of Bioinformatics, Switzerland, **3** The Rockefeller University, New York, New York, United States of America

Recent large-scale cDNA sequencing efforts show that elaborate patterns of splice variation are responsible for much of the proteome diversity in higher eukaryotes. To obtain an accurate account of the repertoire of splice variants, and to gain insight into the mechanisms of alternative splicing, it is essential that cDNAs are very accurately mapped to their respective genomes. Currently available algorithms for cDNA-to-genome alignment do not reach the necessary level of accuracy because they use ad hoc scoring models that cannot correctly trade off the likelihoods of various sequencing errors against the probabilities of different gene structures. Here we develop a Bayesian probabilistic approach to cDNA-to-genome alignment. Gene structures are assigned prior probabilities based on the lengths of their introns and exons, and based on the sequences at their splice boundaries. A likelihood model for sequencing errors takes into account the rates at which misincorporation, as well as insertions and deletions of different lengths, occurs during sequencing. The parameters of both the prior and likelihood model can be automatically estimated from a set of cDNAs, thus enabling our method to adapt itself to different organisms and experimental procedures. We implemented our method in a fast cDNA-to-genome alignment program, SPA, and applied it to the FANTOM3 dataset of over 100,000 full-length mouse cDNAs and a dataset of over 20,000 full-length human cDNAs. Comparison with the results of four other mapping programs shows that SPA produces alignments of significantly higher quality. In particular, the quality of the SPA alignments near splice boundaries and SPA's mapping of the 5' and 3' ends of the cDNAs are highly improved, allowing for more accurate identification of transcript starts and ends, and accurate identification of subtle splice variations. Finally, our splice boundary analysis on the human dataset suggests the existence of a novel non-canonical splice site that we also find in the mouse dataset. The SPA software package is available at <http://www.biozentrum.unibas.ch/personal/nimwegen/cgi-bin/spa.cgi>.

Citation: van Nimwegen E, Paul N, Sheridan R, Zavolan M (2006) SPA: A probabilistic algorithm for spliced alignment. *PLoS Genet* 2(4): e24. DOI: 10.1371/journal.pgen.0020024



Introduction

Recent large-scale sequencing projects such as FANTOM3 [1] have started to unveil the complexity of the mammalian transcriptome. Data from such projects provide a unique opportunity for an in-depth investigation of the mechanisms that lead a single region in the genome to produce a myriad of transcript forms through variation in the transcription initiation site, the transcription termination site, and the combination of splice sites used. In order to perform such an analysis it is essential that the mapping of the observed transcripts to the genome from which they derive is highly accurate. For instance, the study of the structure of basal promoters and of the mechanism of alternative transcription initiation requires that the starts of transcripts are correctly mapped. Similarly, the study of important regulatory elements such as microRNA binding sites requires that the ends of the transcripts are accurately mapped and the 3' UTRs are

correctly identified. For learning more about the mechanism and regulation of alternative splicing, it might well be that the most informative transcripts are those with rare splice variations, potentially including errors of the splicing machinery. The mapping program thus should reliably identify these cases as well.

Currently available algorithms for mapping cDNAs to the genome fail to reach the required level of accuracy for various reasons. Structurally, the most significant problem with existing mapping algorithms is that they use ad hoc scoring schemes for defining the quality of different alignments. Some programs, e.g., Sim4 [2], cannot properly deal

Editors: Judith Blake (The Jackson Laboratory, US), John Hancock (MRC-Harwell, UK), Bill Pavan (NHGRI-NIH, US), and Lisa Stubbs (Lawrence Livermore National Laboratory, US), together with *PLoS Genetics* EIC Wayne Frankel (The Jackson Laboratory, US).

Received August 15, 2005; **Accepted** January 18, 2006; **Published** April 28, 2006

DOI: 10.1371/journal.pgen.0020024

Copyright: © 2006 van Nimwegen et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Abbreviations: CCDS, Consensus CDS; ICAM, intercellular adhesion molecule; UCSC, University of California Santa Cruz

* To whom correspondence should be addressed. E-mail: erik.vannimwegen@unibas.ch

‡ Current address: Memorial Sloan-Kettering Cancer Center, New York, New York, United States of America

Synopsis

A prerequisite for the identification and analysis of splice variation in the transcriptomes of higher eukaryotes is the very accurate mapping of cDNAs to their genomes. However, current algorithms use ad hoc scoring schemes that cannot correctly trade off the likelihoods of different sequencing errors against the likelihoods of different gene structures.

In this paper the authors develop a Bayesian probabilistic approach to cDNA-to-genome mapping that combines explicit models for the prior probabilities of different gene structures with the likelihoods of different sequencing errors. The parameters of these probabilistic models can be estimated automatically from the input such that the mapping procedure is automatically adapted to the organism and sequencing technology of the data under study.

The authors implement their approach in a fast mapping algorithm called SPA and apply it to a dataset of human full-length cDNAs and the FANTOM3 dataset of mouse full-length cDNAs. Comparisons with four other mapping algorithms show that SPA produces mappings that are significantly more accurate, with the largest improvements in the mappings of the 5' and 3' ends of the cDNAs, and the mappings around splice boundaries. The authors also identify a novel set of putative splice sites in the human dataset.

with non-canonical splice boundaries and sometimes introduce multiple errors in the alignment around the splice boundary to force the canonical, GT-AG, pair of splice signals. Other programs, e.g., BLAT [3], do not explicitly distinguish between small introns and small deletions caused by sequencing errors. Finally, a significant fraction of all mapping errors are caused by the general inability of the algorithms to correctly map the starts and ends of the transcripts.

Here we introduce SPA, a novel algorithm for spliced alignment that is based on a Bayesian probabilistic model for mapping cDNAs to the genome and identifying their gene structures. Using a set of parameters that can be estimated in a dataset- and organism-specific way, SPA searches for the mapping with maximal posterior probability. Extensive comparisons between SPA, Sim4 [2], GMAP [4], BLAT [3], and Spidey [5] on mappings of full-length human cDNAs show that SPA performs significantly better than each of the other algorithms. Our main application is the mapping of the more than 100,000 full-length cDNAs of the FANTOM3 dataset [1]. We show that SPA's alignments are significantly more accurate than those that were obtained for the FANTOM3 project [1] using the BLAT program [3] with the -fine option and post-processing of the output to identify the intron/exon structure.

Results

Trading Off Sequencing Errors and Gene Structures: Bayesian Probabilistic Model

For a given cDNA we want to infer the set of exons in the genome from which it derives. That is, we want to align the cDNA with the genome and indicate in this alignment where exons start and end. For simplicity, we will refer to such a combination of an alignment and identification of exon boundaries as a mapping of the cDNA. To evaluate the quality of different mappings it is essential to take into account both

the sequencing errors that they imply as well as the gene structures that they imply. On the one hand, we know that sequencing errors are quite rare and that, roughly speaking, the fewer mismatches, insertions, and deletions there are in the alignment, the more likely the alignment is. On the other hand, one can always “perfectly” align a cDNA to the genome as long as one allows an arbitrary number of arbitrarily small exons separated by arbitrarily small introns, e.g., by letting each nucleotide form an exon by itself. However, it is clear that the resulting gene structure would be extremely unlikely. To correctly evaluate the quality of different mappings one thus has to trade off the likelihood of various sequencing errors against the likelihood of different gene structures.

To combine the probabilities of the gene structure and the sequencing errors in a systematic way we use a Bayesian approach. Depending on the organism under study, we assign different prior probabilities to different gene structures. In our model the prior probabilities of gene structures depend on the lengths of their exons and introns, and on the sequences occurring at their splice boundaries. This prior over gene structures is then combined with a likelihood model for different sequencing errors. Each possible mapping implies a transcript, and this transcript generally differs from the observed cDNA by a number of single-base mutations, insertions of bases into the cDNA that were not present in the transcript, and deletions of bases from the cDNA that were present in the transcript. The likelihood model assigns the probability that, starting from a given transcript, sequencing errors would lead to the observed cDNA. In this model we assume that bases are mutated at a constant rate during sequencing, and that insertions and deletions of different lengths occur at different rates.

Formally, given a cDNA c , the genome g , and a hypothesized mapping m , we have a prior probability $P(m|g)$ for the gene structure that the mapping implies, and a probability $P(c|m)$ for the sequencing errors that the mapping implies. Using Bayes's theorem the posterior probability $P(m|c,g)$ for a mapping given both cDNA and genome is

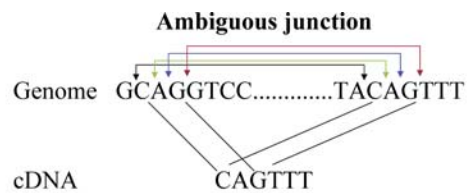
$$P(m|c,g) = \frac{P(c|m)P(m|g)}{\sum_{m'} P(c|m')P(m'|g)}, \quad (1)$$

Our alignment algorithm attempts to find, for a given cDNA c , the mapping m that maximizes this posterior probability $P(m|c,g)$.

Estimating Model Parameters

The likelihood of different gene structures generally varies between organisms. Similarly, the rate at which different errors occur during sequencing generally varies between different sequencing technologies. In contrast to existing algorithms that use a single scoring function for all organisms and experimental technologies, our algorithm adapts its scoring function by adapting the prior $P(m|g)$ over gene structures to the organism from which the cDNAs derive, and the likelihood model of sequencing errors $P(c|m)$ to the experimental technology used.

As detailed in the Materials and Methods, in our model the distributions $P(m|g)$ and $P(c|m)$ are specified by a number of parameters such as the distribution of intron lengths, the probabilities of different sequences at the splice boundaries, and the rates at which mutations, insertions, and deletions of various lengths are introduced during sequencing. These



Possible boundary sequences: CA-TA, **AG-AC**, **GG-CA**, **GT-AG**

Figure 1. Example of an Ambiguous Splice Boundary

Because a few nucleotides at the start of the intron match the nucleotides at the start of the neighboring exon, the splice boundary can be positioned in multiple ways without introducing mismatches. The different possible boundaries are indicated in different colors.

DOI: 10.1371/journal.pgen.0020024.g001

parameters are either estimated directly from the dataset under study or estimated from an external set of reference alignments. In the first approach, we start by mapping the dataset of cDNAs with a default set of parameters and then estimate the parameters of the distributions $P(m|g)$ and $P(c|m)$ from the resulting “reference” alignments. A second round of alignments is then performed with these new parameters. If necessary this procedure can be iterated more than once. If an external set of reference alignments is provided, we estimate the parameters of the distributions $P(m|g)$ and $P(c|m)$ directly from these reference alignments.

Part of the estimation procedure involves the estimation of the splice boundary probabilities $P(s_1s_2s_3s_4)$, i.e., the probability that an intron will start with bases s_1s_2 and end with bases s_3s_4 . To this end we could of course simply count the frequency with which the boundaries $s_1s_2s_3s_4$ occur in the set of reference alignments. However, as illustrated in Figure 1, many of these boundaries are ambiguous. That is, the boundary can be put in multiple places without introducing any mismatches or gaps into the alignment. Since the placement of these ambiguous boundaries in the reference alignments is very sensitive to the details of the scoring function that produced the reference alignments, the frequency of boundaries $s_1s_2s_3s_4$ in the reference alignments might be highly biased.

As detailed in the Materials and Methods we calculate a probability for all possible ways in which ambiguous splice boundaries in the reference alignments can be assigned (i.e., shifted left or right) and use a Monte-Carlo Markov chain to sample the space of all possible boundary assignments in proportion to their probability. We set $P(s_1s_2s_3s_4)$ equal to the frequency with which each boundary $s_1s_2s_3s_4$ occurs during this sampling. Roughly speaking, the assignments that contribute most to the average are those that lead to distributions $P(s_1s_2s_3s_4)$ that have low entropy. We confirm that our sampling has converged by performing multiple sampling runs.

Finally, we have observed that in both the human and mouse full-length cDNA datasets, a small fraction of the cDNAs show a much higher rate of misincorporation than the others. To optimize the mappings of these cDNAs, our parameter estimation procedure also identifies the subset of cDNAs that belong in this high misincorporation-rate class, estimates their average misincorporation rate, and remaps them with the misincorporation-rate parameter set to this higher value.

Alignment Algorithm

As shown in the Materials and Methods, finding the optimal mapping m with maximal posterior probability $P(m|c,g)$ reduces to finding the optimal alignment of the cDNA c to the genome g under an appropriate scoring scheme derived from our probabilistic model. In addition, the maximally scoring alignment can be determined by dynamic programming.

Formally, with $s(i,j)$ being the score of an optimal alignment that ends with position i in the cDNA being mapped to position j in the genome, this score can be determined by the recursion relation

$$s(i,j) = \max_{i' < i, j' < j} (s(i',j') + s[(i,j)|(i',j')]), \quad (2)$$

where the score $s[(i,j)|(i',j')]$ is the change in alignment score associated with extending the optimal alignment ending at (i',j') by adding the single pair of mapped bases (i,j) . Apart from the score associated with the matching or mismatching of the bases at i and j , the score $s[(i,j)|(i',j')]$ also incorporates the contribution of the (possible) gaps associated with leaving bases $i' + 1$ through $i - 1$ of the cDNA unmapped, and skipping bases $j' + 1$ through $j - 1$ in the genome. The latter may involve an intron.

Note that, in contrast to scoring models that use affine gap penalties, our much more general scoring of insertion, deletion, and intron lengths forces us to consider all pairs of positions (i',j') with $i' < i$ and $j' < j$. Therefore, although we can theoretically determine the optimal score rigorously using the dynamic programming recursion (Equation 2) on the full dynamic programming matrices formed by the cDNA and each of the chromosomes, it is computationally infeasible to map cDNAs using this scheme and we therefore developed a heuristic approach, which is illustrated in Figure 2.

We first use the BLAT gfServer [3] to determine a set of genomic loci to which the cDNA might map. We separately find the best alignment to each of these loci and choose the overall optimal alignment at the end. For each of the loci we then find a set of “defined positions.” The defined positions (i,j) consist of all areas in the alignment matrix that show significant homology between the cDNA and the genomic locus and are shown as the red diagonals in the top alignment matrix on the right in Figure 2. The defined positions are identified by first finding all k -mer matches between locus and cDNA, extending these diagonally up to the first mismatch on each side, and adding a “fuzz” of defined positions at both ends of the diagonal. In the next step an optimal alignment is found by applying the dynamic programming recursion (Equation 2) only to the set of defined positions. For each defined position (i,j) , shown as a green square in the second diagram on the right in Figure 2, we find the maximum alignment score obtained by extending the optimal alignment ending at any of the previous defined positions (i',j') , shown as the red squares in the same diagram. We also record the position (i',j') that leads to the optimal mapping at (i,j) . When all $s(i,j)$ are calculated we can trace back the optimal alignment as usual.

We then iteratively check the alignment for problem areas such as unmapped cDNA nucleotides and non-canonical splice boundaries, and add defined positions in the problem areas. When there are unmapped cDNA bases at the 5' or 3' end we also extend the genomic locus at the corresponding

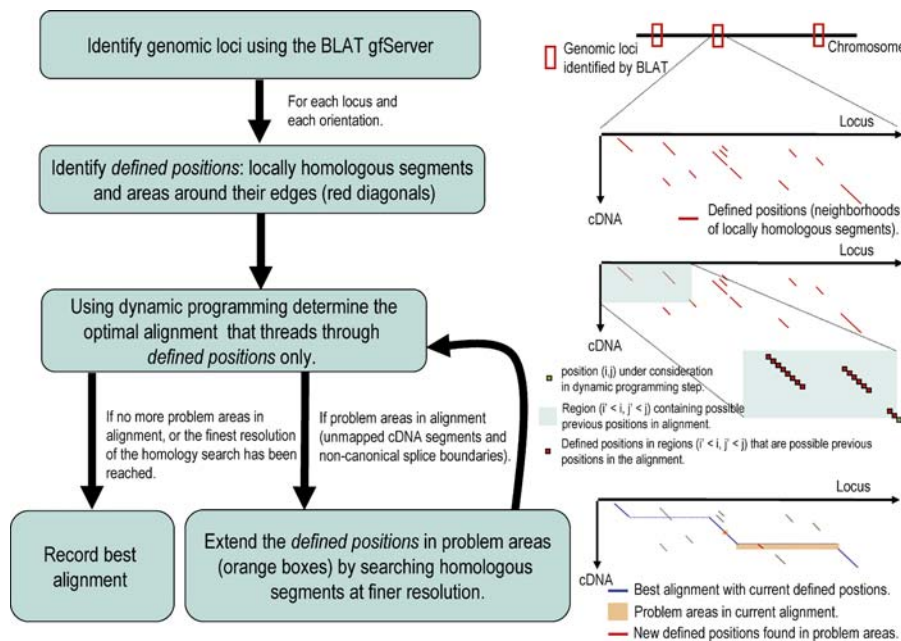


Figure 2. Schematic Summary of the SPA cDNA-to-Genome Mapping Algorithm

DOI: 10.1371/journal.pgen.0020024.g002

end or ends and search for defined positions between the unmapped 5' or 3' end of the cDNA and the added genomic segment. This “re-tiling” procedure is illustrated in the diagram at the bottom right in Figure 2. The blue line shows the best alignment through the current set of defined positions. There are two segments of the cDNA that remain unmapped in this alignment and these define the “problem areas” shown as orange boxes in the diagram. We then identify additional defined positions in these areas at a finer resolution than before, i.e., we decrease k . The new defined positions are shown as the two red diagonals. Every time new defined positions are added to the matrix we recalculate the globally optimal alignment afresh and check it for problem areas. This iterative procedure ends when no more problem areas exist, when no more defined positions can be added to the matrix, or when the number of defined positions exceeds a prespecified maximum. The last condition guarantees that the time that the algorithm spends on a given cDNA stays within a strict upper bound.

Finally, since a small fraction of the cDNAs are erroneously reverse-complemented with respect to the transcript from which they derive we also determine the optimal alignment for the reverse-complement of the cDNA, and we take into account the small prior probability that the cDNA was reverse-complemented in the sequencing process. SPA reports the alignment with maximal posterior probability over all loci and orientations. Details of the procedure are described in the Materials and Methods.

Running Times

The speed of the mapping is a considerable challenge. For example, to map 1 million cDNAs within a week on a cluster with 100 CPUs, the algorithm cannot take more than 1 min per cDNA. Ideally the mapping should take no more than seconds per cDNA on average. However, some transcripts are very long, derive from large paralogous gene families, and can

therefore be mapped to many different loci in the genome, some of which may be as long as 1,000,000 nucleotides. In addition, some loci may contain highly repetitive areas that give a very large number of significant local alignments that all need to be checked in order to determine which alignment gives the globally optimal score. When multiple sequencing errors fall within a small exon that is flanked by large introns in the genome the algorithm will have to search the entire genomic region at a fine resolution to discover the exon. Given these complications, it is simply impossible to produce accurate alignments in a matter of seconds for all transcripts. However, there are also many short single-exon transcripts that map without any, or with very few, errors to only one place in the genome. These transcripts can obviously be accurately mapped in much less than a second.

An efficient alignment algorithm should thus take into account the large variation in mapping difficulty for different cDNAs, quickly dispensing with cDNAs that are easy to map and detecting automatically when more time is needed for more complicated cases. As illustrated in Figure 3, this is naturally achieved by our iterative scheme that checks the current best alignment for problem areas and extends the number of defined positions in these areas by performing a finer resolution homology search. The figure shows the distribution of SPA's running time on all cDNAs of the FANTOM3 dataset. The average time per cDNA for this set of 102,793 cDNAs was 23.6 s. This is comparable to the time BLAT takes when run with the *-fine* option. The running time of individual cDNAs varies over six orders of magnitude from less than 0.01 s to several hours. The vast majority of cDNAs map in less than a second, a few percent take on the order of minutes, and a very small number of cDNAs take more than an hour. More detailed analysis of a number of examples shows that there are several reasons why some cDNAs take such a long time to map. Most of these cDNAs can be mapped to multiple locations in the genome (often

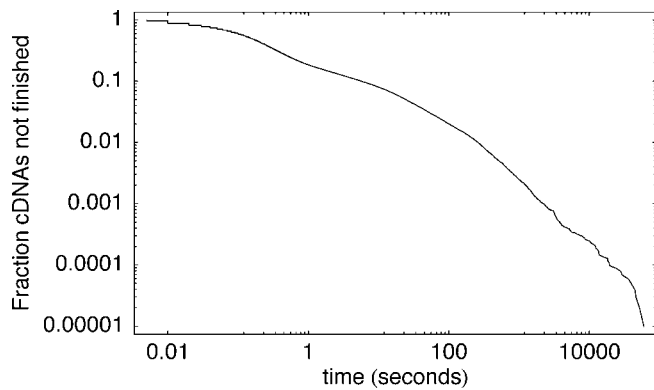


Figure 3. Distribution of Running Times of SPA on All 102,143 cDNAs of the FANTOM3 Dataset

The horizontal axis shows the time in seconds, and the vertical axis shows the fraction of all cDNAs that took longer than the corresponding time to finish. Both axes are shown on a logarithmic scale.
DOI: 10.1371/journal.pgen.0020024.g003

more than ten), and SPA attempts an alignment at each locus. At many of these loci only part of the cDNA can be mapped, and this will lead to unmapped pieces at the 5' or 3' end in the initial alignment. SPA will then extend the loci at the corresponding ends and re-tile these regions at ever finer resolution in an attempt to map the unmapped 5' and 3' ends. At every step of this re-tiling a significant number of defined positions are added, and SPA has to perform the dynamic programming to identify the best alignment through these positions at every step. Another reason for long running times is that some cDNAs contain a significant amount of repetitive sequence, which produces a very large number of defined positions in the dynamic programming matrix.

To give a specific example, the cDNA with RIKEN identifier F830048L18 took almost 4 h to run. SPA considered 20 different loci for this cDNA, one of which was on Chromosome 1 and the other 19 of which were on pieces of unassembled chromosome. The initial alignments of 15 of these loci all had unmapped 3' ends of about 75 nucleotides. For all 15, SPA extended the locus at the 3' end (by 50 kilobases), and then re-tiled this area with finer and finer tile size in an attempt to map the 3' end piece. This failed in all 15 cases. For the other five loci the initial alignment had an unmapped piece at the 5' end. For these loci SPA extended the loci at the 5' end and re-tiled with finer tile size. For one of the loci (locus 4) this led to a successful mapping of the piece at the 5' end. This final alignment contains three exons in a genomic locus of 2,013 nucleotides.

Results for Human Full-Length cDNAs

As an initial test of SPA, and to extensively compare its performance with other algorithms, we used a recent dataset [6] of 20,207 human full-length cDNAs. (The paper mentions over 21,000 cDNAs, but we only found GenBank records for 20,207.) We mapped the cDNAs of this dataset using SPA, Sim4 [2], Spidey [5], BLAT [3], and the recently published algorithm GMAP [4] (see Materials and Methods for details). Sim4, BLAT, and Spidey were chosen because they are the most commonly used spliced alignment algorithms, and GMAP was chosen because it is a recent fast algorithm that

is reported [4] to produce mappings that are in quality comparable to or better than those that BLAT produces.

Evaluating the quality of the mappings produced by the different algorithms is complicated by the fact that we obviously do not know what the "correct" mapping is for each cDNA. In fact, deciding which of several mappings for a single cDNA is the best is equivalent to defining a scoring model for cDNA mappings. Since we are of the opinion that SPA's scoring model is by far the most realistic and sophisticated model available, we believe that a higher scoring under this model is a good proxy for higher quality of the mappings. Thus, as a first test we scored each algorithm's mappings using SPA's scoring model, and compared the scores with the scores of SPA's mappings. The results are shown as the solid lines in Figure 4. In each panel the solid green line shows the distribution of the score differences for cDNAs for which SPA has a higher scoring mapping, and the solid red line shows the distribution of score differences for cDNAs for which the other algorithm scores higher. For example, the top left panel shows that there are over 6,000 cDNAs for which SPA's mapping scored better than Sim4's mapping, versus only 210 cDNAs for which Sim4 scored better. There are almost 4,000 cDNAs for which the score difference in SPA's favor is at least ten, and a little over 700 cDNAs for which the score difference is at least 100. In contrast, there are only about 200 cDNAs for which Sim4's mapping is better by at least ten, and less than 40 for which the difference is at least 100 in Sim4's favor. To give an indication of the scale of these score differences, a difference of 100 corresponds to between 30 and 60 more matched nucleotides. We see that, for all four algorithms, the number of cDNAs for which SPA scores better is at least ten times as high as the number of cDNAs for which the other algorithm scores better.

Although we believe these results provide strong evidence that SPA produces higher quality mappings, one could always argue that the apparent higher quality of SPA's mappings under this scoring model is simply a result of the fact that SPA used the same model to produce the mappings and does not necessarily indicate higher quality mapping. To address this we performed the same comparisons using scoring models that are adapted to the mappings of each of the other algorithms. That is, for each other algorithm, we ran our parameter estimation procedure to infer the mismatch probability p_{mm} , the splice boundary probabilities $P_{sb}(s_1s_2s_3s_4)$, and the probability distributions of insertions $P_i(k)$, deletions $P_d(k)$, and intron lengths $P_{int}(k)$ that maximize the overall probability of this algorithm's mappings. We then compared the mappings of SPA and each other algorithm under the optimal scoring model for that algorithm. The results are shown as the dashed lines in Figure 4. One can see that the distributions of score differences between SPA and the other algorithms are highly robust under the change of the scoring function.

The dashed lines substantially deviate from the solid lines only for very high score differences in SPA's favor and for small score differences in the other algorithm's favor. These observations can be understood as follows. Almost all mappings that have a much higher score under SPA's scoring model contain one or more very short introns of length less than 30 in the other algorithm's mapping. SPA's scoring highly penalizes these short introns (see Materials and Methods),

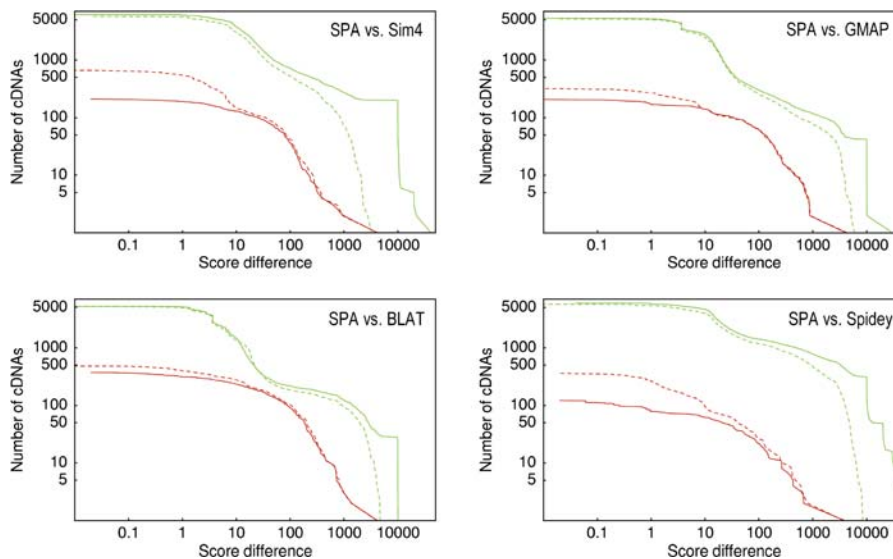


Figure 4. Distributions of Differences in the Log-Posterior Probability of the Mappings of the Human cDNAs

Shown are the differences between SPA and Sim4 (top left), GMAP (top right), BLAT (bottom left), and Spidey (bottom right). The green curves show the distributions of score differences for the cases in which SPA had a higher scoring mapping, and the red curves show the distributions of score differences for mappings where the other algorithm had a higher scoring alignment. The solid lines were obtained using SPA's scoring function and the dashed lines using for each algorithm the scoring function that maximizes the posterior probability of the entire set of mappings of that algorithm. All axes are shown on logarithmic scales.

DOI: 10.1371/journal.pgen.0020024.g004

but under the algorithm's own scoring model this penalty is significantly reduced. This causes the difference in the green lines at very high score differences.

Mappings with very small score differences generally only differ in their choice of splice sites or by subtle differences in the assignment of mismatches and gaps. For example, one mapping may have a single insertion of length six and the other mapping two insertions of length three. For such cases SPA's scoring slightly prefers SPA's mapping and the other algorithm's scoring slightly prefers its own mapping. Because the large majority of cases for which the other algorithm scored better involved only small differences, changing from SPA's scoring model to the other algorithm's scoring model causes the number of such cases to increase significantly. In contrast, because the large majority of mappings for which SPA scored better involved larger differences, the total number of cDNAs for which SPA scored better is affected relatively little by changing the scoring function. Mappings with differences of ten or more differ substantially, e.g., by the mapping of an additional exon or a 5' or 3' end that is unmapped in the other mapping, and changes to the scoring hardly affect the score difference between such mappings.

In summary, for each algorithm, and independent of the scoring function's parameters, SPA produces a better alignment on about 25% (over 5,000) of the cDNAs, a substantially better alignment on at least 5% (1,000 or more) of the cDNAs, and a very large improvement for a few percent of the cDNAs (200 or more).

Given that, for the solid lines in Figure 4, we use our own scoring function for assessing the quality of the mappings, and given that SPA of course attempts to find the mapping with maximal score, one may wonder why there are any cDNAs at all for which the other algorithms find mappings that are considered better under SPA's scoring model (the solid red lines). The reason is that, because of time

constraints, we used a heuristic algorithm to find the best alignment. The red curves thus indicate cases for which our heuristic algorithm failed to find the alignment with optimal score. Fortunately, these cases are rare, amounting to about 1% of the cDNAs. Cases where SPA's mapping has a lower score by a large amount are rarer still.

All these comparisons still depend on a scoring function for evaluating the quality of the mappings. We additionally compared the mappings in a number of ways that are completely independent of a scoring model. First, in Figure 5 we compare a number of global statistics across the mappings of the different algorithms (a much more detailed table of global statistics is provided in Table S1). In the bar chart in the upper left of Figure 5 we show the relative numbers of errors of different types in the mappings of the five algorithms. We have rescaled the bars such that, for each type of error, the total number of errors in the SPA mappings is set to one.

In an ideal situation every cDNA nucleotide would be mapped to a matching nucleotide in the genome. SPA has a total of 472,392 nucleotides that are not mapped to matching nucleotides in the genome, which corresponds to just under 1% of all nucleotides in the data. The top five bars in the bar chart in Figure 5 show the relative numbers of unmatched bases in the tested algorithms. We see that all other algorithms have over 20% more unmatched bases than SPA. The pie charts in Figure 5 show how these unmatched nucleotides are distributed over different types of errors for each of the algorithms. All algorithms failed to produce mappings for some of the cDNAs (see Materials and Methods). As the pie chart for Spidey shows, unmapped cDNAs account for almost half of Spidey's unmatched nucleotides. Unmapped cDNAs also account for a significant fraction of GMAP's unmatched nucleotides, and for a moderate fraction of BLAT's unmatched nucleotides. For the large majority of

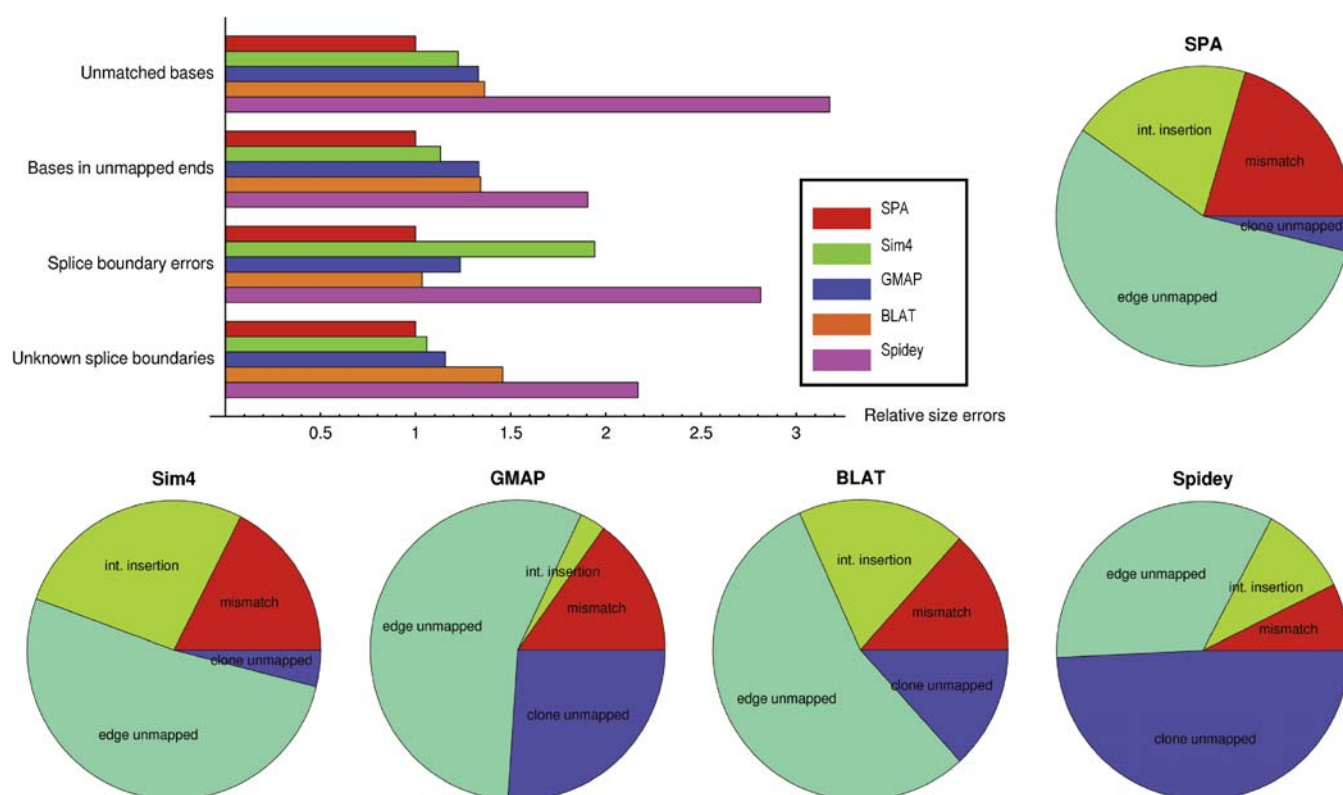


Figure 5. Comparison of Global Mapping Statistics for the Mappings of the Human cDNAs for SPA, Sim4, GMAP, BLAT, and Spidey

The bar chart shows the relative numbers of nucleotides in different types of errors between SPA and the other algorithms. The number of splice boundary errors is defined as the number of nucleotides in mismatches, insertions, and deletions that occur within ten alignment positions of a splice boundary. The fraction of unknown splice boundaries is the fraction of all splice boundaries that do not contain a GT-AG, GC-AG, or AT-AC splice boundary. The numbers of errors in SPA are scaled to one. The pie charts show the percentages of the unmatched nucleotides in each of the algorithms' mappings that are the result of unmapped cDNAs (dark blue), unmapped 5' and 3' ends (light blue), internal insertions (green), and mismatches (red).

DOI: 10.1371/journal.pgen.0020024.g005

these cDNAs, BLAT and GMAP simply did not report a significant alignment. SPA and Sim4 did produce alignments for most of these cDNAs, but manual inspection revealed that these were typically very low quality alignments with only a small fraction of the cDNA mapped. We suspect that many of these cDNAs are experimental artifacts, including chimeric cDNAs.

Unmapped nucleotides at the 5' and 3' ends of cDNAs account for more than half of the unmatched nucleotides for all algorithms but Spidey. As shown in the second set of bars in the upper left in Figure 5, all other algorithms have substantially more nucleotides in these unmapped ends than SPA. As shown in Table S1, unmapped poly-A tails account for only a small proportion of these unmapped ends (see also Materials and Methods). The difference between SPA and the other algorithms is caused mostly by SPA's ability to identify additional 5' or 3' exons that the other algorithms miss. For these cases we often found that one or more initial/terminal exons were separated from the other exons by a region in which the quality of the mapping was poor, containing an internal piece of the cDNA that could not be mapped at all. Frequently there are assembly gaps in these regions as well, and we suspect that misassemblies occur in some of these regions. The other algorithms, but in particular GMAP, tend not to extend their mappings beyond problematic areas, which causes them to miss the initial/terminal exons that lie

beyond this region and that can be accurately mapped. As a result of this, the number of internal insertions is much smaller for GMAP than for the other algorithms.

The bottom two sets of bars in Figure 5 compare the quality of the mappings around the splice boundaries (see Materials and Methods for details). We see that SPA had fewer errors (mismatches, insertions, and deletions occurring within ten nucleotides of splice boundaries) and that the fraction of splice boundaries that do not match any of the known boundaries, i.e., the canonical GT-AG, the non-canonical GC-AG, or the U12 spliceosome boundary AT-AC, was lower for SPA than for any of the other algorithms. Sim4's mappings contained almost twice as many errors around splice boundaries as SPA's mappings. The main reason for this is that Sim4 attempts to find canonical GT-AG splice sites and that it is willing to introduce multiple mismatches and insertions to accomplish this. In contrast, BLAT's mappings contained only a few more errors around splice boundaries, but BLAT's mappings contained a much higher fraction of boundaries that do not match any known splice site.

Table 1 shows the ten most frequent splice sites in the SPA, Sim4, and Spidey alignments with their associated frequencies. We found that the canonical GT-AG signal and non-canonical GC-AG signal used by the U2 spliceosome [7] were the two most abundant boundaries for all five algorithms. Interestingly, in spite of Sim4's strong preference for canon-

Table 1. The Ten Most Frequently Occurring Splice Sites in the Mappings of the Human Dataset by SPA, Sim4, GMAP, BLAT, and Spidey

SPA		Sim4		GMAP		BLAT		Spidey	
Splice Site	Percent	Splice Site	Percent	Splice Site	Percent	Splice Site	Percent	Splice Site	Percent
GT-AG	96.0%	GT-AG	94.1%	GT-AG	96.6%	GT-AG	94.9%	GT-AG	92.5%
GC-AG	1.06%	GC-AG	1.36%	GC-AG	1.06%	GC-AG	1.01%	GC-AG	1.05%
AG-CC	0.27%	CT-AC	0.55%	AT-AC	0.17%	CT-AC	0.20%	GC-CA	0.20%
TG-CC	0.18%	GT-AA	0.38%	CT-GG	0.04%	AT-AC	0.14%	GT-TG	0.17%
AT-AC	0.17%	GT-TG	0.22%	CA-CC	0.03%	GA-AG	0.13%	GT-CT	0.15%
AG-GC	0.15%	CT-GC	0.21%	CC-GG	0.03%	CT-AG	0.11%	GT-GG	0.15%
TG-GC	0.14%	GA-AG	0.20%	CA-GC	0.03%	GT-TG	0.10%	GC-CT	0.15%
AC-CC	0.13%	AT-AG	0.16%	GC-GT	0.03%	AT-AG	0.09%	GG-CA	0.14%
TC-CC	0.11%	CT-AA	0.14%	CC-TG	0.03%	GT-GG	0.09%	GC-GG	0.12%
AA-AA	0.11%	CT-AG	0.13%	CA-GG	0.03%	CC-AG	0.09%	GT-CA	0.11%

The frequency of occurrence is shown to the right of each splice boundary sequence.
DOI: 10.1371/journal.pgen.0020024.t001

ical boundaries, and its willingness to allow errors around the splice boundaries to accommodate them (see Figure 5), it still ended up with a lower frequency of GT-AG boundaries than the mappings of SPA, GMAP, and BLAT. Spidey had an even lower frequency of canonical GT-AG. The overall frequency of GT-AG boundaries was a bit higher in GMAP's mappings than in SPA's mappings. This is the result of the relatively high frequency of non-canonical boundaries in areas of the cDNAs that remain unmapped in GMAP's mappings. When we restricted the analysis to splice boundaries that lay in areas of the cDNAs that were mapped by both SPA and GMAP, SPA had a higher frequency of GT-AG boundaries than GMAP.

Both BLAT and Sim4 showed a disproportionately high number of CT-AC boundaries. In the case of BLAT this is a result of the fact that it does not consider the possibility that the cDNA has been misoriented, whereas SPA, GMAP, and Spidey do consider this possibility. Sim4 also does not explicitly consider the possibility that the cDNA has been misoriented but instead attempts to find either canonical GT-AG or reverse-complemented CT-AC boundaries independently for each splice boundary. This frequently leads to mappings for which, in a single cDNA, some boundaries are GT-AG and others are CT-AC. Other frequent alternative boundaries of Sim4 and BLAT are mostly single-point mutant versions of the canonical GT-AG boundary. Spidey finds splice boundaries by first looking for a donor site and then choosing the acceptor site independently. This is reflected in the fact that all of its top ten boundaries use a donor site that matches either GT or GC. The AT-AC signal used by the U12 spliceosome [8] occurs near the top of the list of splice sites for SPA (0.17%), GMAP (0.17%), and BLAT (0.14%), whereas Sim4 and Spidey seem to miss this experimentally verified alternative boundary.

Finally, the splice boundary inference procedure employed by SPA leads to the emergence of a set of related splice sites, i.e., AG-CC, TG-CC, AG-GC, TG-GC, AC-CC, and TC-CC, that are even more abundant in SPA's mappings than the AT-AC boundaries and that together account for almost 1% of all splice boundaries. In SPA's initial parameter set, only the known boundaries GT-AG, GC-AG, and AT-AC were given high probabilities (see Materials and Methods). Thus, in the first round of mappings, whenever ambiguous boundaries

occurred that did not allow for any of the known splice sites, SPA placed the boundary essentially at random. This seems to be the behavior of GMAP as well. However, the splice boundary inference revealed that many of these unknown boundaries can be explained by the novel set of splice sites just mentioned. These splice sites were thus given higher probability in the parameter set, and in the second round of mappings SPA now chose these boundaries with much higher frequency, as shown in Table 1. Since none of the other algorithms employ such splice boundary inference, none of them recover these novel alternative boundaries. However, when we applied our splice boundary inference procedure to GMAP's or BLAT's mappings, these alternative boundaries also appeared with relatively high frequency in their mappings. We are currently investigating these new splice sites in more detail (T. M. Chern, E. van Nimwegen, and M. Zavolan, unpublished data).

Another way of evaluating the quality of the mappings of the different algorithms is to compare the mappings with a reference set of human gene structures. Unfortunately, obtaining a large collection of human gene structures will always involve computational methods for mapping transcripts to the genome, which could introduce a systematic bias that favors the type of algorithm that was used in producing the reference set. However, the Consensus CDS project [9] combines a number of different lines of evidence with manual curation to produce a set of highly trusted protein coding exons in the human genome that are hopefully not significantly affected by systematic biases. As a further comparison of the mappings we intersected the mappings of all five algorithms with the full set of CCDS exons. The results are shown in Table 2.

We see that all mappings intersect roughly half of the nucleotides in CCDS exons. In this test as well SPA outperformed all other algorithms, although the difference is less than 1% in all cases. As in all other tests so far, Spidey performed worst on this test. It is noteworthy that Sim4 outperformed BLAT significantly, and that BLAT in turn outperformed GMAP on this test.

As a final test of the mappings we compared how well the sequences within the mappings of each algorithm are conserved across vertebrates. To this end we used the phastcon profiles [10], which assign a conservation score (a

Table 2. The Intersection between the Mappings of the Different Algorithms and the CCDS Exons

Statistic	Algorithm				
	SPA	Sim4	GMAP	BLAT	Spidey
Intersection CCDS	2,097,375	2,094,077	2,079,646	2,085,062	2,059,784
Difference	0	3,298	17,729	12,313	37,591

The first statistic is the total number of nucleotides in the mappings that intersect the set of 4,164,450 nucleotides of the CCDS exons. The second statistic is the difference between the number of intersecting nucleotides in the SPA mappings and the mappings of each of the other algorithms.

DOI: 10.1371/journal.pgen.0020024.t002

number between zero and one) to each nucleotide in the human genome based on a comparison with the genomes of chimpanzee, dog, mouse, rat, chicken, zebrafish, and fugu (see Materials and Methods for details). The results are shown in Figure 6. The left panel shows that, at any level of conservation, the number of nucleotides with at least that level of conservation is larger in SPA's mappings than in the mappings of any other algorithm. Thus, the higher number of mapped nucleotides in SPA's mappings are not concentrated in badly conserved regions but include highly conserved regions. In fact, as the right panel of Figure 6 shows, the nucleotides that are unique to each of the algorithms' mappings do not differ substantially in their distribution of conservation level. The right panel of Figure 6 shows the ratio of conservation distributions of nucleotides that are unique to SPA's mappings and nucleotides that are unique to each of the other algorithms' mappings. We see that, with the exception of Spidey, which has a significantly underrepresented fraction of highly conserved nucleotides, the ratio stays within the range 0.8–1.2 over the entire range of conservation level. These results show that the relatively large number of nucleotides that are mapped only by SPA have roughly the same distribution of conservation level as the relatively small number of nucleotides that are mapped only by the other algorithms.

Results on FANTOM3 Mouse Full-Length cDNAs

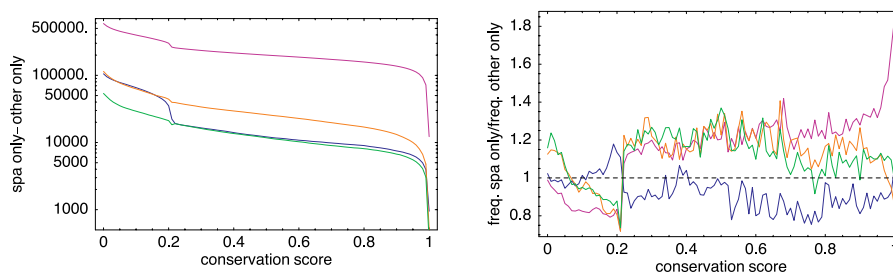
We ran SPA on all 102,143 cDNAs of the FANTOM3 dataset that had been mapped for the FANTOM3 project using BLAT

with the `-fine` option, followed by post-processing to distinguish deletions from introns. As in the mappings of the human dataset, we compared the log-posterior probabilities of all SPA mappings with the FANTOM3 mappings using both SPA's scoring function as well as a scoring function that maximizes the overall posterior probability of the FANTOM3 mappings. The results are shown in Figure 7. As in the mappings of the human cDNAs, the change in scoring function affected these distributions relatively little. At almost any score difference there are more than ten times as many mappings with at least such a score difference in SPA's favor as there are in BLAT's favor. Overall, SPA had a better mapping on more than 70% of the cDNAs, a substantially better mapping (difference of ten or more) on more than 10% of the cDNAs, and a much better mapping (difference of 100 or more) on almost 5% of the cDNAs. In contrast, BLAT's mapping was much better than SPA's mapping on only 0.3% of the cDNAs.

To compare the quality of the mappings independently of a scoring function we again calculated a number of global mapping statistics. The relative numbers of nucleotides in errors of various kinds are shown in Figure 8. A more detailed set of statistics is shown in Table S2. One can see that all types of errors are significantly overrepresented in BLAT's mappings compared to SPA's mappings. The largest differences are in the number of unmapped cDNAs, the number of nucleotides in unmapped 5' and 3' ends, and the fraction of splice boundaries that do not match any of the known splice sites. However, the number of internal mismatches (internal insertions and mismatches) is also larger by about 20% in BLAT's mappings, and the number of nucleotides in errors around splice boundaries is larger by more than 50% in BLAT's mappings. Thus, the global statistics confirm the overall higher quality of SPA's mapping especially in the mapping of 5' and 3' ends and in the correct mapping of the splice boundaries.

The ten most common splice boundaries in the SPA and BLAT mappings of the FANTOM3 cDNAs are shown in Table 3.

The canonical GT-AG boundary was the most frequent in both SPA's and BLAT's mappings, but its frequency was 1.2% higher in SPA's mappings. The known non-canonical boundary GC-AG was about equally common in the SPA and the BLAT mappings. Both SPA and BLAT found the boundary AT-AC of the U12 spliceosome. Also very abundant in the

**Figure 6.** Comparison of Conservation Statistics for the Mappings of the Human Dataset

Shown are the data for SPA versus Sim4 (green), GMAP (blue), BLAT (orange), and Spidey (pink). The left panel shows, as a function of conservation score c , the difference in the number of nucleotides with conservation score at least c between the SPA mappings and the mappings of each other algorithm. The right panel shows the relative distribution of conservation scores for the nucleotides that were unique to SPA's mappings and the nucleotides that were unique to the other algorithm's mappings. The panel shows, for each conservation score c , the ratio between the fraction of all nucleotides unique to SPA's mappings that have conservation score c and the fraction of all nucleotides unique to the other algorithm's mappings that have conservation score c .

DOI: 10.1371/journal.pgen.0020024.g006

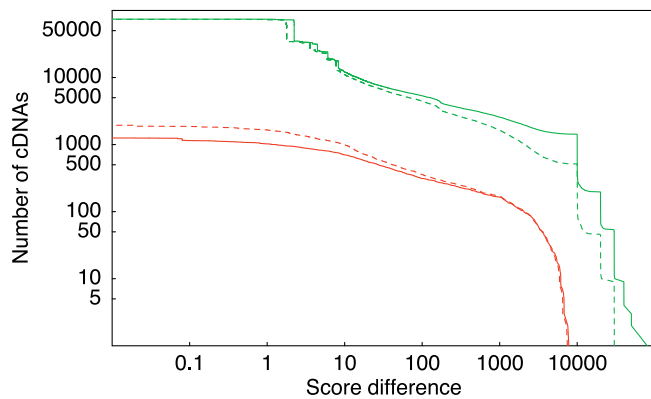


Figure 7. Distributions of Differences in the Log-Posterior Probability of SPA and Post-Processed BLAT Alignments on the FANTOM3 Mouse cDNAs

The green curve shows the distribution of score differences for the cases in which SPA had a higher mapping score. The red curve shows the distribution of score differences for mappings where post-processed BLAT had a higher alignment score. Solid lines were obtained using SPA's scoring function and dashed lines using the scoring function that maximizes the posterior probability of BLAT's mappings. Both axes are shown on logarithmic scales.

DOI: 10.1371/journal.pgen.0020024.g007

SPA mappings were boundaries where one or both of the ends of the intron were in an assembly gap (where the wildcard nucleotide N occurs in the genome). As we describe below, BLAT often has problems with mappings in these areas. The fourth most abundant boundary in the BLAT mappings was CT-AC, i.e., the reverse-complement of the canonical boundary. As with the mappings of the human cDNAs, this boundary's high abundance is a result of the fact that BLAT does not explicitly recognize reverse-complemented cDNAs. It is interesting that one of the set of novel related splice boundaries that we identified in the human cDNA mappings also occurs with relatively high frequency in the mappings of the mouse cDNA, i.e., the TG-CC boundary. It is also interesting that many of the other high frequency boundaries, e.g., GT-CA, GG-AG, and GT-TG, occur in the top ten of both algorithms.

Figure 9 shows examples of the most common differences

between the SPA and BLAT mappings that we have identified through manual inspection. Figure 9A and 9F show examples with large differences in score between the SPA and the BLAT mapping. Figure 9A illustrates BLAT's inability to deal with repeat elements. Even though, as the SPA mapping illustrates, the cDNA can be mapped without any errors, BLAT introduces a spurious insertion of one repeat unit, which then leads to more than 1,100 nucleotides at the 5' end of the cDNA remaining unmapped, i.e., being interpreted as an insertion. Similarly, Figure 9F shows that even a single nucleotide insertion can lead BLAT to miss a large part (more than 100 nucleotides) at the 5' end of the mapping.

Figure 9B and 9E show examples of moderate score differences. Figure 9B illustrates BLAT's inability to appropriately deal with assembly gaps. As can be seen in the SPA mapping, the last 13 nucleotides of the exon on the left fall in an assembly gap (N's in the genome). SPA correctly assumes that the most likely interpretation is that the N's will match the cDNA nucleotides and that the splice boundary will be the canonical GT-AG boundary. Consistent with this model, the right end of the intron has the canonical acceptor site AG. BLAT on the other hand treats ten of the last 13 nucleotides of the exon as an insertion, introduces a single nucleotide deletion, and uses a non-canonical splice boundary. Such errors in the BLAT mappings are common whenever the alignment passes through an assembly gap. Figure 9E shows another example of a moderate difference in score that also involves a different choice of splice boundary. SPA uses the relatively common GC-AG boundary and has only a single mismatch in this area of the alignment. In contrast, BLAT uses the uncommon AG-AT boundary and introduces eight mismatches in this area.

Figure 9C shows a small score difference involving an ambiguous splice boundary. SPA chooses an AT-TC boundary whereas BLAT prefers the CT-AT boundary. With the probabilities of splice signals that we inferred, the boundary AT-TC has a probability of 8.4×10^{-5} whereas the boundary CT-AT does not occur at all. We suspect that BLAT prefers the CT-AT boundary because its underlying model for scoring splice boundaries is the Hamming distance to the canonical GT-AG boundary. Finally, Figure 9D shows one of

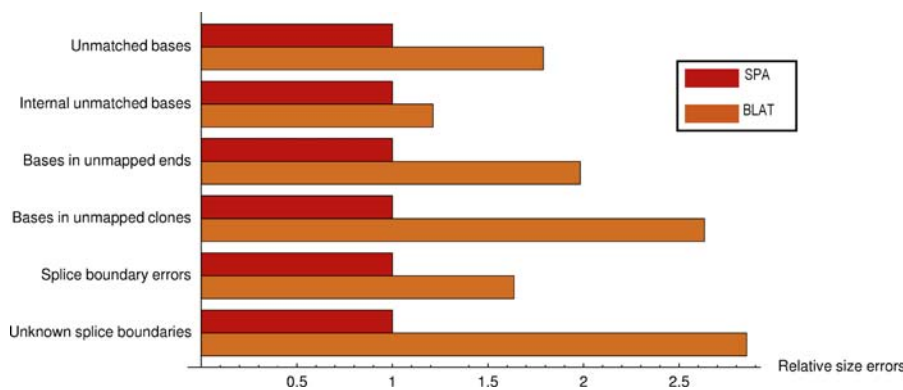


Figure 8. Comparison of Global Statistics of the SPA Mappings and Post-Processed BLAT Mappings of the FANTOM3 Mouse cDNAs

The bars show, from top to bottom, the total number of unmatched nucleotides, the number of internally unmatched nucleotides (mismatches plus internal insertions), the number of nucleotides in unmapped 5' and 3' ends, the number of nucleotides in unmapped cDNAs, the number of errors (mismatches, insertions, and deletions) within ten alignment positions of the splice boundaries, and the fraction of splice boundaries not matching any known splice site (GT-AG, GC-AG, or AT-AC). The bars are scaled such that SPA's total numbers of errors are set to one for each type.

DOI: 10.1371/journal.pgen.0020024.g008

Table 3. The Ten Most Frequently Occurring Splice Sites in the Mappings of the FANTOM3 Dataset by SPA and BLAT

SPA		BLAT (Post-Processed)	
Splice Site	Percent	Splice Site	Percent
GT-AG	97.8%	GT-AG	96.6%
GC-AG	0.82%	GC-AG	0.79%
GT-NN	0.18%	NN-NN	0.18%
NN-AG	0.18%	CT-AC	0.16%
AT-AC	0.12%	AT-AC	0.1%
NN-NN	0.12%	GT-NN	0.05%
GT-CA	0.05%	GG-AG	0.05%
GG-AG	0.05%	GT-TG	0.05%
TG-CC	0.04%	GT-CA	0.04%
GT-TG	0.03%	GT-GG	0.04%

The splice sites are ordered with the most frequent at the top, and the frequency of occurrence is shown next to the sequence.

DOI: 10.1371/journal.pgen.0020024.t003

the most common small differences between the SPA and BLAT mappings. A single-nucleotide insertion in the cDNA close to the 5' end of the cDNA leads BLAT to leave the first four nucleotides unmapped, even though the first three map perfectly to the genome.

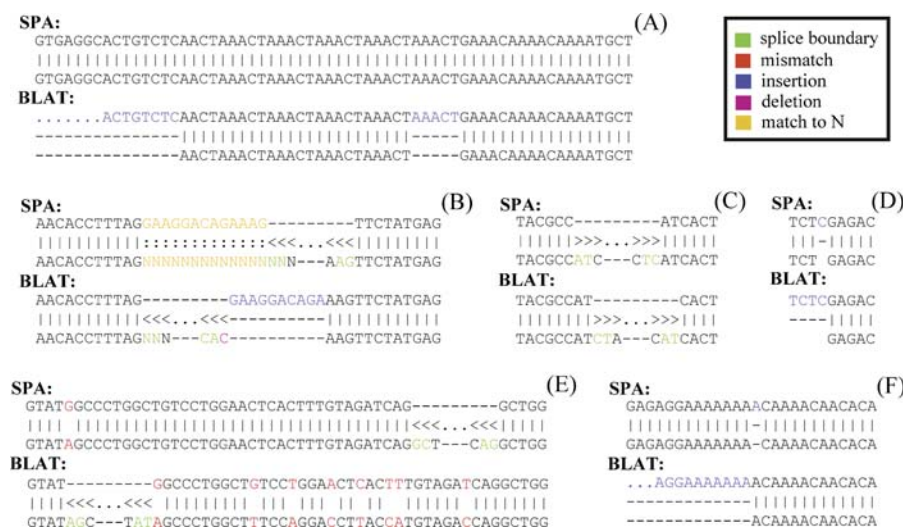
We also observed a large number of cases where there were mismatches within the first two nucleotides of the cDNA. SPA often interprets these as mismatches whereas BLAT treats them as insertions. We suspect the experimental methodology that was used to obtain the FANTOM3 dataset leads to an overrepresentation of errors at the first two cDNA nucleotides.

Since the FANTOM3 dataset is the largest collection of mammalian cDNAs, covering a large proportion of mouse genes, it provided us the opportunity to examine SPA's and BLAT's ability to deal with cDNAs deriving from families of paralogous genes. This is of interest especially since sets of paralogous genes that occur close to each other on the

chromosome are particularly challenging for mapping algorithms. We extracted the genomic loci of two such gene families and compared the mappings of SPA and BLAT that intersect these loci. For the family of intercellular adhesion molecules (ICAMs) we found that there were essentially no differences in SPA's mappings and BLAT's mappings. We found ten cDNAs corresponding to *ICAM* genes, eight mapping to *ICAM-1* on Chromosome 9 and two mapping to *ICAM-2* on Chromosome 11. Both algorithms map these cDNAs correctly.

The *beta defensin* gene family provides a more interesting example. Figure 10 shows a screen shot of the University of California Santa Cruz (UCSC) genome browser [11] displaying the SPA and BLAT mappings of two cDNAs that both map to the *defensin* locus on Chromosome 8. SPA's and BLAT's mappings of the cDNA 9230103N16 are essentially identical and perfectly overlap the known *Defb12* gene. However, there is a substantial difference in the mapping of cDNA 4931406G05. This cDNA maps to the negative strand. SPA's mapping ends with 44 nucleotides that fall in a genomic gap and are mapped to N's. BLAT, in contrast, introduces two extra short exons separated by very long introns in an attempt to map these 44 nucleotides. The resulting small exons have several mismatches and insertions, and both long introns have non-canonical splice sites. As a result, BLAT's mapping spans multiple *defensin* genes whereas SPA's mapping does not overlap any known *defensin* gene.

Figure 11 shows two screen shots from UCSC's genome browser [11] displaying SPA's mapping of cDNA 4933424L15 on Chromosome 11 and BLAT's mapping of the same cDNA on Chromosome 8. BLAT's mapping intersects two separate *defensin* genes (*Defb11* and *Defb15*) whereas SPA's mapping does not overlap any known *defensin* gene. Both these mappings fail to map the whole cDNA. SPA's mapping has an unmapped 3' end of about 350 nucleotides, whereas BLAT's mapping has an unmapped 5' end of about 560 nucleotides. It appears that this cDNA may be a chimera formed by combining the region from Chromosome 8 with the region of Chromosome 11.

**Figure 9.** Examples of Differences in the SPA and BLAT Mappings

DOI: 10.1371/journal.pgen.0020024.g009

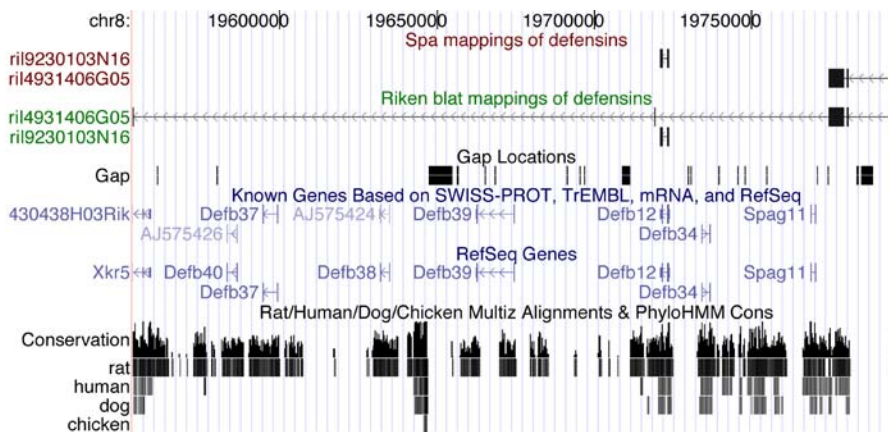


Figure 10. Part of the *defensin* Locus on Chromosome 8 Together with the Mappings of SPA and BLAT of Two cDNAs That Map to This Locus
The figure was made using UCSC's genome browser [11].
DOI: 10.1371/journal.pgen.0020024.g010

Discussion

To realize the full potential of large collections of full-length cDNAs, such as the FANTOM3 dataset that we investigate in this paper, the mapping of the cDNAs to the genome has to be very accurate. For instance, if one wants to use the data to study proximal promoters, the 5' end of the cDNA has to be correctly mapped, as a missing first exon can move the apparent transcription start site tens of kilobases downstream from the real start site. If one wants to use the data to study splice variation, it is necessary that all splice boundaries are correctly identified. This includes correctly identifying rare splice variations that may give important

information about the underlying splicing mechanisms. Finally, accurate mapping of 3' UTRs is an important prerequisite for the algorithms that predict microRNA targets, as these algorithms generally require multi-species alignments of 3' UTRs [12,13].

The main structural problem with existing algorithms for fast cDNA-to-genome alignment is their ad hoc scoring schemes that cannot appropriately trade off the likelihoods of different gene structures against the likelihoods of different sequencing errors. Moreover, their scoring functions cannot easily be adapted to the particulars of the dataset under study. The speed of mapping is another challenge.

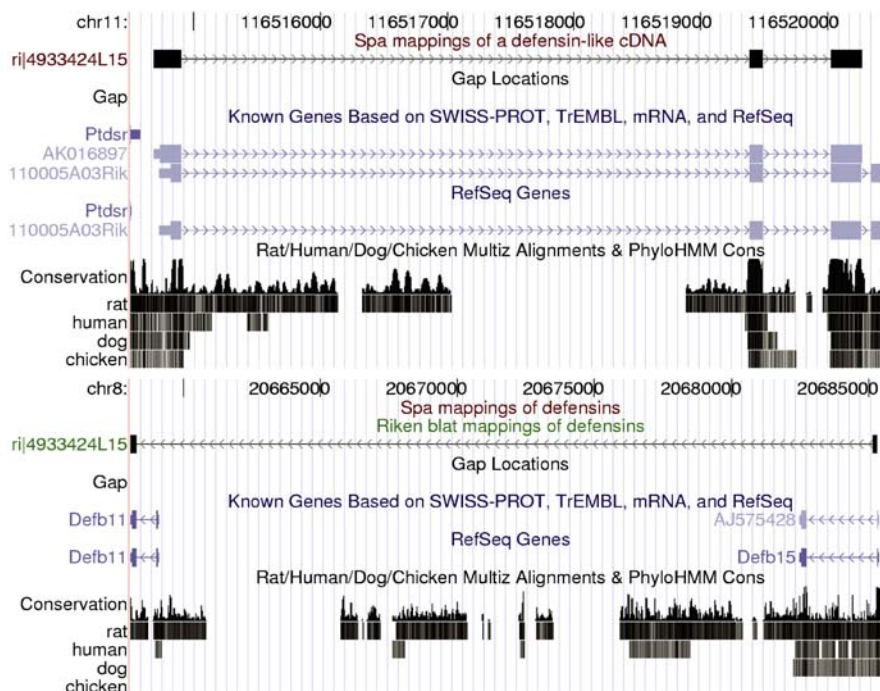


Figure 11. BLAT Mapping of a cDNA to the *defensin* Locus on Chromosome 8 Together with the SPA Mapping of the Same cDNA to a Locus on Chromosome 11

The figure was made using UCSC's genome browser [11].
DOI: 10.1371/journal.pgen.0020024.g011

Mammalian genomes contain several billion nucleotides, and current full-length mRNA and expressed sequence tag libraries comprise several million transcripts. To be able to keep the mappings up-to-date with the genome assembly one needs to be able to map all transcripts in a matter of days on a moderately sized computing cluster, i.e., 100 CPUs or fewer. This requires average running times per cDNA on the order of seconds, which can only be achieved using heuristic algorithms.

The approach to the problem of mapping cDNAs to the genome that we introduce here is novel in several aspects. First, it is the first rigorous fully probabilistic approach to this problem that evaluates the quality of possible mappings by taking both the likelihood of the implied gene structures and the likelihood of the implied sequencing errors explicitly into account. This involves assigning prior probabilities to individual gene structures and constructing a likelihood model for the errors that are introduced during cDNA sequencing. Both of these components can be completely, and automatically, adapted to the organism under study, and to the experimental procedure that was used to obtain the cDNA sequences. In contrast to other algorithms that use simple gap penalties our model allows for arbitrary distributions of intron, insertion, and deletion lengths.

Another novel aspect of our approach is our model for the splice boundaries. In contrast to the simpler models used by other algorithms, we allow a general distribution $P(s_1s_2s_3s_4)$ for the probabilities that an intron starts with the dinucleotide s_1s_2 and ends with s_3s_4 . We developed a Bayesian inference procedure that uses a set of mappings in which the precise locations of the splice boundaries are left ambiguous to infer the probabilities $P(s_1s_2s_3s_4)$.

The parameters in our probabilistic models need not be specified externally but can be estimated by the algorithm itself. This is done by starting from a default set of parameters and iteratively mapping the data and re-estimating the parameters from the resulting mappings to fine-tune both the parameters of the probabilistic model and the mappings that are produced using it. The model also explicitly accounts for the possibility of the cDNA being mistakenly reverse-complemented, and it reports information about such cases.

We implemented our probabilistic model in a fast mapping algorithm called SPA. The algorithm is sufficiently fast to enable the mapping of the 102,143 cDNAs of the FANTOM3 dataset to the mouse genome in one night on a 50-CPU Linux cluster. This speed allows us to map all available mouse expressed sequence tag and cDNA sequences in a matter of days and update our mappings whenever a new genome assembly version becomes available. We intend to provide such a database of mappings in the near future.

We compared the performance of SPA with that of four other mapping programs—Sim4, GMAP, BLAT, and Spidey—using a dataset [6] of over 20,000 human full-length cDNAs. Both under SPA's scoring model and under scoring models that are optimally adapted to each of the other algorithms we showed that there are more than ten times as many cDNAs that are better mapped by SPA than by any of the other algorithms. Comparing mapping statistics globally we found that SPA mapped significantly more nucleotides to matching nucleotides in the genome than any of the other algorithms. The largest improvements are due to better mappings of the 5' and 3' ends of the cDNAs and due to the better quality of

the alignments around the splice boundaries. These improvements have important consequences for downstream studies since these are precisely the important areas for studying promoters, regulatory sites in 3' UTRs, and the mechanisms of alternative splicing. We also compared each set of mappings with a reference set of trusted protein coding exons [9] and found that SPA's mappings matched more nucleotides of these trusted exons than any of the other algorithms. Finally, we compared the conservation statistics [10] of all mappings. We found that, at any level of conservation, the number of nucleotides with at least that level of conservation was higher in SPA's mappings than in the mappings of any of the other algorithms.

An interesting result to emerge from our splice boundary inference procedure is the novel set of putative splice boundaries AG-CC, TG-CC, AG-GC, TG-GC, AC-CC, and TC-CC that together account for almost 1% of the splice boundaries in the human dataset. One of these boundaries, TG-CC, also occurs within the top ten splice boundaries of the mappings of the FANTOM3 cDNAs.

We mapped the 102,143 full-length mouse cDNAs of the FANTOM3 dataset with SPA and compared the mappings with the FANTOM3 mappings obtained using BLAT. Again we found that SPA matched significantly more nucleotides, and that SPA had an overall lower number of errors. The most significant differences are again in the improved mappings of 5' and 3' ends, and the improved mappings around splice boundaries. We also compared the mappings of SPA and BLAT for cDNAs that intersect the genomic loci of two families of paralogous genes. For the family of *defensin* genes we found two cDNAs that BLAT erroneously mapped to large genomic loci that span multiple *defensin* genes. The SPA mappings, in contrast, were much more compact and did not intersect any known *defensin* genes.

Even though SPA significantly improves on current mapping algorithms and we believe that it provides the current state of the art in mapping cDNAs to the genome, there are still several limitations to our approach. The first limitation is in our relatively simple probabilistic model of gene structures. Within our Bayesian framework it would have been straightforward to implement more sophisticated probabilistic models for gene structures, e.g., such as those used in gene-prediction programs like Genscan [14]. We decided, however, to keep our gene structure model simple for the following reasons. First, we want our model to apply equally to coding and noncoding transcripts, and more sophisticated models are generally suited for protein coding genes only. Second, we want to let the cDNA data “speak for themselves” as much as possible and minimize the influence of the gene structure prior. Models such as the one used by Genscan [14] have many parameters, most of which are not well-known for organisms that are less studied than human or mouse, and some of which are uncertain even for well-studied organisms. By keeping our model simple we avoid biasing our mappings with a prior whose parameters have not been correctly estimated, and we make it easier to apply our algorithm to cDNAs from less studied organisms. One thing that could be improved is the distribution over exon lengths, which our model effectively assumes is exponential, whereas the observed distribution of exon lengths is more complex [15].

Probably more than 50% of the remaining “errors” in the SPA mappings are due to unmapped bases at the 5' and 3'

ends of the cDNAs. These are often caused by initial and terminal exons that are more than 500 kilobases away from the other exons in the gene, or by ends of the cDNA that cannot be mapped at all in the neighborhood of the locus where the rest of the cDNA maps. We believe that some of these are the result of errors in the assembly, and SPA could be further improved by making it take the possibility of assembly errors explicitly into account, and letting it report areas of suspected genome misassembly. Another fraction of the unmapped 5' and 3' ends could be the result of chimeras, i.e., transcripts that are a fusion of a 5' end from one locus and a 3' end from another locus [16]. Currently SPA is not taking the possibility of chimeric cDNAs into account, but it would be straightforward to adapt the algorithm to do so. That is, since SPA already provides mappings for all genomic loci to which the cDNA can be (partly) mapped, it would be easy to have the algorithm check, in cases of unmapped 5' or 3' ends, whether a full mapping can be obtained by combining the 5' end mapping of one locus with the 3' end mapping of another locus. Finally, SPA currently does not take into account the possibility of genomic polymorphisms, but rather assumes that all differences between the mapped cDNA and the genome derive from sequencing errors. For organisms such as human and mouse this is unlikely to affect the quality of the mappings since the polymorphism rate is generally significantly lower than the rate of sequencing errors. However, for some organisms the polymorphism rate can exceed the sequencing error rate and may be highly heterogeneous across the chromosomes. Such cases could be dealt with by allowing the mismatch probability to vary across the cDNA.

The most promising yet much more challenging direction for future work is not to map a single cDNA at a time but to take the information from the entire set of cDNAs into account at the same time in the mapping procedure. For instance, by recognizing that the same small "insertion" between two exons occurs in multiple cDNAs, one would realize that this "insertion" is really a small exon that failed to map to the genome. While it is clear that considering multiple cDNAs at the same time can significantly improve the quality of the mappings, the development of a rigorous probabilistic model for such an approach requires that we specify priors not merely over single gene structures, but rather over all possible splice variations of a gene. That is, we will have to move from probabilistic models for single transcript "genes" to probabilistic models for transcript clusters that derive from a common genomic region. We imagine that SPA's mappings of the FANTOM3 cDNAs, together with the splice analysis pipeline that we developed previously [15,17], will provide a first step in this direction.

Materials and Methods

As described in the Results the posterior probability $P(m|c,g)$ that our model assigns to a mapping m given a cDNA c and a genome g is given in terms of the prior probability $P(m|g)$ of the gene structure of m and the probability $P(c|m)$ of the sequencing errors that the mapping m implies. In the following sections we derive $P(c|m)$ and $P(m|g)$ in detail and give explicit expressions for the log-posterior probability $s(m|c,g) = \log[P(m|c,g)]$ of any mapping m . Below we show how the problem of finding a mapping m with optimal score can be reduced to the problem of finding an optimal alignment under an appropriately adapted scoring scheme. Then we derive in detail how an optimal alignment can be found by dynamic programming.

Following that, we detail the heuristic algorithm for finding the best mapping and, in particular, explain how a set of defined positions is obtained for each cDNA and how this set is expanded recursively whenever the quality of the mapping requires this. Finally, we explain the details of the procedure for inferring the parameters that maximize the posterior probability of a given set of alignments.

Transcript-to-cDNA error model. To calculate the probability $P(c|m)$ of the sequencing errors we assume the following model of the sequencing process: as the cDNA is "copied" from its mRNA template from beginning to end there is a probability $P_\delta(k)$ at each position to skip the next k nucleotides of the transcript and a probability $P_i(n)$ to insert n random nucleotides into the cDNA. Every nucleotide that is incorporated into the cDNA has a constant probability p_{mm} to be mutated. We assume that all misincorporations are equally likely, so that the probability of obtaining a particular mismatching nucleotide is $p_{mm}/3$. Whenever an insertion of n nucleotides occurs we assume that the inserted nucleotides are randomly chosen, such that the probability of obtaining an insertion with n particular nucleotides is $P_i(n)4^{-n}$. The quantities $P_\delta(0)$ and $P_i(0)$ give the probabilities of not deleting and not inserting any nucleotides respectively between consecutive nucleotides of the transcript. Under this model we have for the probability $P(c|m)$ of the observed cDNA c given the mapping m :

$$P(c|m) = (p_m)^{n_m} \left(\frac{p_{mm}}{3}\right)^{n_{mm}} \prod_{k=0}^{\infty} \left[P_\delta(k)^{n_\delta(k)} \left(\frac{P_i(k)}{4^k}\right)^{n_i(k)} \right], \quad (3)$$

where n_m is the number of matching nucleotides in the alignment, $p_m = 1 - p_{mm}$ is the probability of a match, n_{mm} is the number of mismatching aligned nucleotides, $n_\delta(k)$ is the number of deletions of length k , and $n_i(k)$ is the number of insertions of length k into the cDNA. For future use, we define the following scores

$$s_m = \log(p_m) \quad (4)$$

$$s_{mm} = \log(p_{mm}) - \log(3) \quad (5)$$

$$s_\delta(k) = \log[P_\delta(k)] \quad (6)$$

$$s_i(k) = \log[P_i(k)] - k\log(4). \quad (7)$$

Using this notation we can write for the logarithm of the probability $s(c|m) = \log[P(c|m)]$:

$$s(c|m) = n_m s_m + n_{mm} s_{mm} + \sum_{k=0}^{\infty} n_\delta(k) s_\delta(k) + n_i(k) s_i(k). \quad (8)$$

Gene structure prior. In our model the prior probability $P(m|g)$ of the mapping m given the genome g depends on the lengths of the introns and exons of m and on the sequences at the splice boundaries. Note that the lengths of the exons and introns are independent of the genome sequence g and that g enters into the probabilities of only the splice boundaries of m . We thus use the general identity

$$P(m|g) = \frac{P(g|m)P(m)}{P(g)}, \quad (9)$$

with $P(m)$ the probability of the intron and exon lengths of m , $P(g|m)$ the probability of observing the genome sequence g given the gene structure m , and $P(g)$ the probability of observing the genome.

We assume the following simple model for the distribution of intron and exon lengths of a transcript: scanning the transcript from its starting position in the genome there is a probability $P_{\text{int}}(k)$ at each position to insert an intron of length k between the current nucleotide of the transcript and the next. Note that the probability to extend the exon by another nucleotide is $P_{\text{int}}(0)$. Thus, generally, if the gene structure has n_e exons that contain B_e nucleotides in total, and $n_{\text{int}}(k)$ introns of length k we have

$$P(m) = [P_{\text{int}}(0)]^{B_e - n_e} \prod_{k=1}^{\infty} [P_{\text{int}}(k)]^{n_{\text{int}}(k)} \quad (10)$$

for the probability of obtaining the intron and exon lengths of m . Note that $n_{\text{int}}(0) = B_e - n_e$ is the number of times that no intron is introduced between neighboring exonic nucleotides.

We further assume that the gene structure depends on the genome sequence g only through the nucleotides that occur at the splice boundaries. That is, we assume there are probabilities $P_{sb}(s_1 s_2 s_3 s_4)$ for an intron to start with nucleotides $s_1 s_2$ and end with nucleotides $s_3 s_4$.

Under this model the probability $P(g|m)$ of observing the genome sequence g , given the locations of the splice boundaries, is given by

$$P(g|m) = 4^{-|g|+4n_{\text{int}}} \prod_{s_1 s_2 s_3 s_4} [P_{\text{sb}}(s_1 s_2 s_3 s_4)]^{n_{\text{sb}}(s_1 s_2 s_3 s_4)}, \quad (11)$$

where n_{int} is the number of introns, $n_{\text{sb}}(s_1 s_2 s_3 s_4)$ is the number of splice boundaries $s_1 s_2 s_3 s_4$, and $|g|$ is the length of the genome. Note that we assume that every nucleotide of the genome outside of the splice boundaries has a probability 1/4. Finally, the probability $P(g)$ of observing all nucleotides in the genome is simply given by

$$P(g) = 4^{-|g|}. \quad (12)$$

For future reference, we define the following notation:

$$s_{\text{int}}(k) = \log[P_{\text{int}}(k)] \quad (13)$$

$$s_{\text{sb}}(s_1 s_2 s_3 s_4) = \log[4^4 P_{\text{sb}}(s_1 s_2 s_3 s_4)]. \quad (14)$$

Using this notation and Equations 9, 11, and 12, we obtain the following expression for the logarithm of the probability $s(m|g) = \log[P(m|g)]$ of the gene structure given the genome

$$s(m|g) = \sum_{s_1 s_2 s_3 s_4} n_{\text{sb}}(s_1 s_2 s_3 s_4) s_{\text{sb}}(s_1 s_2 s_3 s_4) + \sum_{k=0}^{\infty} n_{\text{int}}(k) s_{\text{int}}(k). \quad (15)$$

Total score. Putting together $s(c|m)$ and $s(m|g)$, and defining the total log probability $s(m|c,g) = \log[P(m|c,g)]$ we have

$$s(m|c,g) = s(c|m) + s(m|g) + \text{constant}, \quad (16)$$

where the constant equals $-\log[P(c|g)]$ and does not depend on the mapping m . Since we are interested only in finding the mapping m that maximizes the posterior probability $P(m|c,g)$ we ignore this constant from now on. Written out more explicitly, our “score” $s(m|c,g)$ for a mapping m given a cDNA c and genome g becomes

$$s(m|c,g) = n_{\text{m}} s_{\text{m}} + n_{\text{mm}} s_{\text{mm}} + n_{\text{sb}}(s_1 s_2 s_3 s_4) s_{\text{sb}}(s_1 s_2 s_3 s_4) + \sum_{k=0}^{\infty} n_i(k) s_i(k) + n_{\delta}(k) s_{\delta}(k) + n_{\text{int}}(k) s_{\text{int}}(k). \quad (17)$$

Thus, our final score for a mapping and gene structure m depends on the number of matched and mismatched nucleotides, the splice boundaries, and the number of insertions, deletions, and introns of each length k . In the next section we describe how the transcript with maximal score $s(m|c,g)$ can be identified through dynamic programming. For cDNAs that contain no or a negligible number of sequencing errors the score $s(m|c,g)$ will be dominated by the number of matched nucleotides n_{m} and the optimal mapping will be simply the mapping with the maximum number of matching nucleotides. For exon boundaries that can be mapped in multiple ways without introducing mismatches the splice boundary scores s_{sb} will ensure that the mapping with the most likely splice boundary is chosen. If no canonical splice boundary is present at an exon junction the score will trade off the likelihood of different sequencing errors around the splice boundary against the likelihood of different non-canonical boundaries. If a gap occurs in the cDNA the score will trade off the probability of an intron of that size with its splice boundaries, against the probability of a deletion of the same size. Much more complex trade offs of course occur in practice. For example, one mapping may have a small exon of length four flanked by certain splice boundaries and introns of length k_1 and k_2 , while an alternative mapping has a single intron of length $k_1 + k_2$ in combination with four inserted nucleotides, and the score will use the distributions of intron lengths, splice boundaries, and cDNA insertions to determine which of these is most likely. Since all these distributions can be specified by the user and can be estimated from reference mappings, this gives SPA the capability of correctly assessing the likelihood of very complicated cases of ambiguous mappings that other algorithms cannot properly deal with.

Obtaining the MAP transcript through cDNA-to-genome alignment. A mapping m , in our sense of the word, can be specified by first specifying an alignment of the cDNA to the genome, and then specifying for each cDNA gap which part of that gap should be considered intron (including the possibilities “none” or “all”). Note that, given an alignment between cDNA c and genome g , we can determine the set of introns that maximize the likelihood of the resulting mapping m by calculating, for each gap in the cDNA, the combination of intron/deletion that maximizes the likelihood for that gap. For example, assume that in the alignment nucleotides $i + 1$ through $j - 1$ in the genome correspond to a gap in the cDNA. If we interpret this gap entirely as a deletion it will contribute a factor

$s_{\text{int}}(0) + s_{\delta}(j - i - 1)$ to the score of the mapping. If we interpret it entirely as an intron it will contribute $s_{\text{int}}(j - i - 1) + s_{\text{sb}}(s_{i+1} s_{i+2} s_{j-2} s_{j-1}) + s_{\delta}(0)$. If we assume the intron starts at $i + 1 + k$ and ends at $j - 1 - l$ then the contribution to the mapping score is $s_{\text{int}}(j - i - k - l - 1) + s_{\delta}(k + l) + s_{\text{sb}}(s_{i+k+1} s_{i+k+2} s_{j-l-2} s_{j-l-1})$. To find the optimal mapping given the cDNA gap we thus have to find the combination of k and l that maximizes the score. We thus define the following “cDNA gap” scoring function:

$$s_{\text{cg}}(i, j) = \max[s_{\text{int}}(0) + s_{\delta}(j - i - 1), \max_{k,l} s_{\text{int}}(j - i - k - l - 1) + s_{\delta}(k + l) + s_{\text{sb}}(s_{i+k+1} s_{i+k+2} s_{j-l-2} s_{j-l-1})]. \quad (18)$$

This is the maximum possible contribution to the mapping score from any cDNA-to-genome alignment that contains a cDNA gap from nucleotides $i + 1$ through $j - 1$ in the genome. The overall maximum probability cDNA-to-genome alignment, using the scoring function $s_{\text{cg}}(i, j)$ to score gaps from i to j in the cDNA. From now on, when we refer to an alignment it should be understood that we mean the maximum probability mapping m that is consistent with the alignment.

Any alignment of the cDNA to the genome can be specified by giving all pairs of cDNA and genome nucleotides that are mapped to each other. That is, if a total of n nucleotides are mapped between the cDNA and genome, then an alignment can be given as a set of n pairs (i_k, j_k) corresponding to the positions of the k th mapped pair in the cDNA and genome respectively. Between every two consecutive pairs of mapped nucleotides (i_k, j_k) and (i_{k+1}, j_{k+1}) the nucleotides at positions $i_k + 1$ through $i_{k+1} - 1$ in the cDNA correspond to a gap in the genome, and the nucleotides $j_k + 1$ through $j_{k+1} - 1$ in the genome correspond to a gap in the cDNA (both of these gaps may be length zero).

We now show that the score $s(m|c,g)$ of an alignment can be written as a sum of contributions $s[(i_{k+1}, j_{k+1})|(i_k, j_k)]$. Every alignment is initialized with an empty alignment $(i_0, j_0) = (0, 0)$ that considers the whole cDNA to be an insertion and has a score $s_i(l)$, with l the length of the cDNA. We also define $s_{\text{cg}}(0, j) = 0$ (mapping the first cDNA nucleotide to position j of the genome has a score that does not depend on j). We now define the “jump score” $s[(i, j)|(i', j')]$ as the change in score when an alignment that ended at (i', j') is extended by adding the pair (i, j) to it. This score is given by

$$s[(i, j)|(i', j')] = s_{\text{m}}(i, j) + s_i(i - i' - 1) + s_{\text{cg}}(j', j) + s_i(l - i) - s_i(l - i'), \quad (19)$$

where $s_{\text{m}}(i, j) = s_{\text{m}}$ if nucleotides i and j match, and $s_{\text{m}}(i, j) = s_{\text{mm}}$ if they do not. The terms contributing to the jump score are illustrated in Figure 12.

It is now easy to verify that, using the definition of the jump score (Equation 19), the total score of an alignment, as given by Equation 17, can be written as

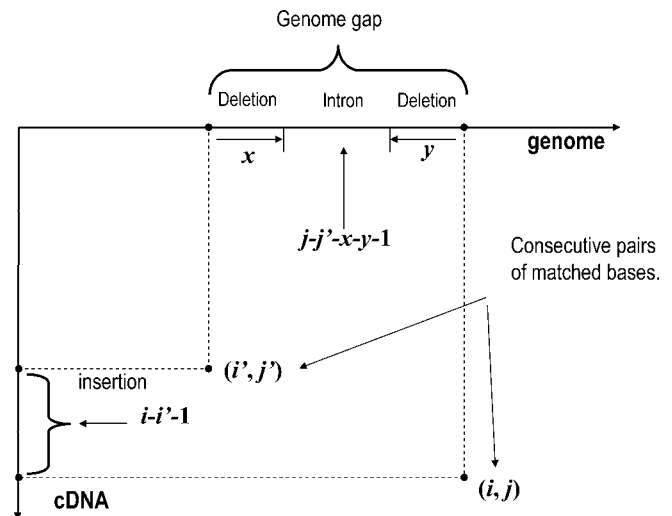


Figure 12. Illustration of the Contributions to the “Jump Score” When Extending an Alignment Ending at (i', j') by Adding a Pair of Mapped Nucleotides at (i, j)

DOI: 10.1371/journal.pgen.0020024.g012

$$s(m|c, g) = \sum_{k=1}^n s[(i_{k+1}, j_{k+1})|(i_k, j_k)]. \quad (20)$$

Dynamic programming solution. Let $s(i, j)$ be the score of the best alignment that ends with aligning position i in the cDNA to position j in the genome. We can obtain $s(i, j)$ by considering all previous “parent” positions (i', j') with $i' < i$ and $j' < j$:

$$s(i, j) = \max_{i' < i, j' < j} [s(i, j)|(i', j')] + s(i', j'). \quad (21)$$

This method is guaranteed to find the optimal alignment of the cDNA to the genome, but it is clear that it is generally computationally infeasible. For a chromosome of length C and a cDNA of length l the number of scores that need to be checked scales as $C^2 l^2$, e.g., 10^{22} for a cDNA of length $l = 1,000$ and a chromosome of length $C = 10^8$ nucleotides.

However, we generally do not need to check all parent positions (i', j') to find the one that maximizes the score at (i, j) . We can calculate an upper bound $ds_{\max}$ on the jump scores $s[(i, j)|(i', j')]$ and proceed as follows. Calculate the scores $s[(i, j)|(i', j')]$ for each parent pair (i', j') starting with the pair (i', j') that has the highest score $s(i', j')$ and proceeding through the parent pairs in order of decreasing score. Keep track of the best score $s_*(i, j) = s(i', j') + s[(i, j)|(i', j')]$ that has been obtained from all parent pairs considered thus far. If, for the current parent pair under consideration, the maximal obtainable score $s(i', j') + ds_{\max}$ is smaller than $s_*(i, j)$, then we know that $s_*(i, j)$ is the optimal score at (i, j) .

We can improve speed further by treating the case $(i', j') = (i - 1, j - 1)$ separately. In the overwhelming majority of cases the maximal score at (i, j) is obtained by extending the mapping from the immediately preceding pair of positions $(i - 1, j - 1)$. Since insertions, deletions, and introns are relatively rare, the maximal possible jump score $ds_{\max}$ associated with jumps that include a gap of size at least one in cDNA or genome is much lower than the score for jumping from the immediately preceding pair. We thus first calculate the score $s(i, j) = s(i - 1, j - 1) + s[(i, j)|(i - 1, j - 1)]$ separately and then proceed to check all other parent pairs using an upper bound $ds_{\max}$ over all possible “jump scores” that include a gap of at least length one.

With this procedure the search for the best parent will typically end within a small number of steps, and in return we pay a relatively small computational cost for keeping the parent pairs in a list ordered by their scores $s(i', j')$. The overall time complexity using this procedure scales as $C \log(C)$. This is already small enough that, in principle, the full dynamic programming could be run for a cDNA and genomic locus of moderate size (e.g., less than 1,000 and 10^5 nucleotides, respectively). However, it is still orders of magnitude too slow to map large numbers of cDNAs.

Heuristic alignment strategy. Note that our dynamic programming scheme does not require that a score $s(i, j)$ is calculated for all possible positions (i, j) in the dynamic programming matrix. We can preselect any subset of positions (i, j) and apply the dynamic programming only to these “defined positions.” That is, for every defined position (i, j) we only consider parent positions (i', j') that are also defined.

Our heuristic mapping strategy is to first locate areas of homology between cDNA and genome, and to then restrict the dynamic programming to positions that are in or near the areas of homology between cDNA and genome. Specifically, we proceed as follows. (1) We use the BLAT gServer [3] to identify genomic loci from which the cDNA may derive. BLAT generally reports multiple loci and we retain all loci that are within 5% of the locus with the highest score (using BLAT’s score). When the length of the best locus is less than $3/2$ the length of the cDNA (indicating a potential pseudogene), we also retain all loci that are longer than $3/2$ the length of the cDNA and that have scores within 20% of the top scoring locus. We consider at most 20 different loci. We extend each locus by 2,000 nucleotides to the left and right and run the following steps separately for each locus. (2) We perform a “tiling step” with “tile size” k : we identify all perfect k -mer matches between the locus and cDNA, merge overlapping matches, and extend them diagonally until the first mismatches on both sides. All positions in these diagonals are added to the list of defined positions. In addition, we add a “fuzz” (a diagonally oriented rectangle) of defined positions of length $r + 3k/2$ and width $2r$ at both ends of each diagonal. By default we set the radius $r = 4$ and the initial tile size is $k = 16$. (3) We perform an initial identification of the optimal alignment through the defined positions so obtained. (4) We check for insertions (unmapped nucleotides) at the start or end of the cDNA. Insertions of bases at the 3' end that consist entirely or almost entirely of adenosines are likely to correspond to poly-A tails that were not removed, and these are not considered insertions but are

flagged as probable poly-A tails. If insertions exist we add 50,000 nucleotides to the locus at the corresponding end or ends and perform a tiling of this region in the genome with the inserted portion of the cDNA. All defined positions obtained in this tiling are added to the global list of defined positions, and a new optimal alignment is constructed. (5) If insertions (at the start, end, or internally) of length four or more exist, the tile size is reduced from k to $k - 4$ and a re-tiling of the unmapped portion of the cDNA and the corresponding region in the genome is performed. All defined positions so obtained are added to the global list of defined positions. In addition, a fuzz of defined positions of length $r + 3k/2$ and width $2r$ is added at all splice boundaries that are not GT-AG and at insertions of length less than four. With this expanded list of defined positions a new optimal alignment is produced. (6) Step 5 is repeated until all insertions have disappeared or tile size $k = 4$ has been reached.

To check for misoriented cDNAs the above procedure is performed both with the original cDNA, and with its reverse-complement. We assign a prior probability π (which is also estimated from a set of mappings) to the probability that the cDNA is misoriented. SPA reports the reverse-complemented alignment and flags the cDNA as misoriented if $s_{\text{mo}} + \log(\pi) > s_{\text{co}} + \log(1 - \pi)$, with s_{mo} the score of the optimal alignment of the reverse-complemented cDNA, and s_{co} the score of the optimal alignment of the cDNA in correct orientation.

To guarantee that the processing time for each cDNA stays within certain bounds there are internal upper bounds on the number of defined positions that are allowed. If the original tiling of locus and cDNA produced more than $d_{\max}$ defined positions then the tile size is increased by four and the entire procedure is repeated starting with this larger tile size. This process is repeated until the initial number of defined positions becomes less than $d_{\max}$.

When extending the locus by 50,000 nucleotides at the 5' end or 3' end we allow at most $e_{\max}$ new defined positions. Finally, at each re-tiling the number of defined positions is allowed to grow by at most $r_{\max}$. If this upper bound is exceeded at some point during re-tiling, the program exits re-tiling and reports the optimal alignment obtained thus far.

All the parameters that control our heuristics and upper bounds can be changed by the user. This gives one the flexibility to perform a more thorough search and alignment for “problem cDNAs” that give bad alignments while ensuring processing speed when running in batch on large numbers of cDNAs. All the results that we report here were obtained with the default settings $d_{\max} = 120,000$, $e_{\max} = 50,000$, and $r_{\max} = 25,000$.

Estimating gene structure and sequencing error parameters. Given a set of mappings we infer the set of parameters that maximize the overall posterior probability of the mappings as follows. Starting from the first matched nucleotide, each mapping is scanned from left to right, and stepping from the last mapped pair of nucleotides to the next mapped pair of nucleotides the following statistics are recorded: (1) the numbers of matching and mismatched pairs of aligned nucleotides, (2) the number of times n_k^i that k nucleotides are inserted in the cDNA, with n_0^i the number of times no insertion is observed between consecutive mapped nucleotides, and (3) the number of times n_k^d that k nucleotides are deleted from the transcript, and (4) the number of times n_k^{int} an intron of length k occurs. If the mappings do not explicitly distinguish deletions from introns we record simply the number n_k^e of cDNA gaps of length k .

The parameters p_m and $p_{\text{mm}} = 1 - p_m$ are set to the fractions of all mapped pairs that are matches and mismatches respectively.

To illustrate the procedure for inferring the distribution of insertion, deletion, and intron lengths, Figure 13 shows the length distribution of alignment gaps in the SPA mappings of the FANTOM3 dataset. An essentially identical figure is obtained if one uses the BLAT mappings instead of the SPA mappings. The blue curve shows the length distribution of genome gaps, i.e., insertions, and the red curve shows the length distribution of cDNA gaps, i.e., introns and/or deletions. The distributions of genome gaps and cDNA gaps are almost identical at small lengths. Beyond lengths of about 50 there is a steep rise in the number of cDNA gaps, with a peak around 85, and a very long tail stretching beyond 100,000. It is intuitively clear that essentially all cDNA gaps of lengths less than 40 correspond to deletions, and that almost all cDNA gaps of length more than 50 correspond to introns. This is also confirmed by explicit studies of intron length distributions [18]. Thus, when no distinction is made between introns and deletions in the reference mappings, we estimate the distributions $P_g(n)$ and $P_{\text{int}}(n)$ by assuming that all cDNA gaps of length less than 30 are deletions, and that all others are introns.

In principle we could estimate the distribution $P_i(k)$ directly from

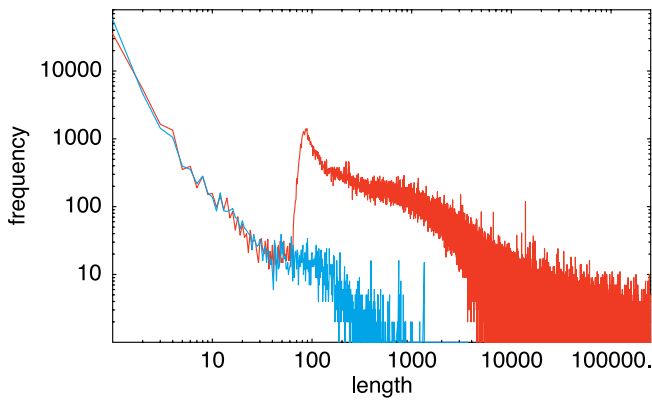


Figure 13. Distributions of Gap Lengths in the SPA Alignments of the FANTOM3 Dataset

The blue curve shows the total number of occurrences of genome gaps, i.e., cDNA insertions, of different lengths. The red curve shows the total number of occurrences of any gaps, i.e., deletions or introns, in the cDNA as a function of gap length. Both axes are shown on a logarithmic scale. DOI: 10.1371/journal.pgen.0020024.g013

the observed counts, i.e., $P_i(k) = n_k^i/n$ with $n = \sum_{i=0}^{\infty} n_k^i$. However, this is not satisfactory for large k since, as Figure 13 shows, we do not have enough observations to estimate $P_i(k)$ from the data for each k . Therefore, we chose to set $P_i(k)$ directly from the data for $k \leq 3$, and estimate $P_i(k)$ for $k > 3$ by a smoothly decreasing function of k . Based on the distributions n_k^i that we have observed in various datasets we chose to model $P_i(k)$ with an exponential $\exp(-\alpha k)$ for large k . We thus set $P_i(k) = n_k^i/n$ for $k \leq 3$, and for $k > 3$ we assume

$$P_i(k) = [1 - P_i(0) - P_i(1) - P_i(2) - P_i(3)](1 - e^{-\alpha})e^{-\alpha(k-4)}. \quad (22)$$

We estimate α by maximizing the likelihood of the counts n_k^i given Equation 22. The maximum likelihood value of α is given by

$$\alpha = \log \left[\frac{\langle k \rangle - 3}{\langle k \rangle - 4} \right], \quad (23)$$

with $\langle k \rangle$ the average length of the insertions, averaged over all insertions of length four or more.

If the reference mappings distinguish the introns from the deletions we estimate the distributions $P_\delta(k)$ and $P_{\text{int}}(k)$ in a way similar to the estimation of $P_i(k)$. For deletions we again set $P_\delta(k) = n_k^\delta/n$ directly for $k \leq 3$, and for $k > 3$ we again assume an exponentially decaying distribution

$$P_\delta(k) = [1 - P_\delta(0) - P_\delta(1) - P_\delta(2) - P_\delta(3)](1 - e^{-\beta})e^{-\beta(k-4)} \quad (24)$$

and for the estimate of β we have

$$\beta = \log \left[\frac{\langle k \rangle - 3}{\langle k \rangle - 4} \right], \quad (25)$$

where $\langle k \rangle$ is the average length of deletions of length four or more.

The distributions of intron lengths that we have observed for various mappings do not seem to fit a simple functional form and we thus decided to estimate $P_{\text{int}}(k)$ empirically over the entire range of lengths. We record the lengths l_s and l_l of the shortest and longest intron in the reference dataset and divide the range $l_l - l_s$ into 100 equally spaced bins on a logarithmic scale. For all k in a bin we set the probability $P_{\text{int}}(k)$ equal to the frequency of introns with lengths falling in that bin divided by the length of the bin. For introns larger than l_l we simply set the probability equal to the probability of the largest intron, i.e., $P_{\text{int}}(k) = P_{\text{int}}(l_l)$ for $k > l_l$. As Figure 13 shows, the frequency of introns drops very sharply as one approaches length l_s from above and we thus set $P_{\text{int}}(k)$ to a very small number for $k < l_s$, thereby effectively excluding introns that are smaller than the smallest observed in the reference set.

Note that when one wants to map a large set of cDNAs from an organism for which no reference mappings are available, one may estimate appropriate parameters for this dataset by first running SPA with a set of default parameters, using the resulting mappings as the set of reference mappings, and estimating the parameters from them. If necessary this procedure may be repeated. Once the parameters are inferred, a final run of SPA with these parameters yields the final

mappings. This is the procedure that we applied to obtain the SPA mappings.

Remapping of high mismatch-rate cDNAs. We observed that, although the rate of mismatches in almost all cDNAs matched the rate p_{mm} estimated from the mappings, a small fraction of cDNAs (on the order of 1%) showed a much higher mismatch rate. To optimize the mappings of these high mismatch-rate cDNAs we decided to classify all mappings into a low mismatch-rate class and a high mismatch-rate class, to estimate the mismatch rates in these classes separately, and to remap all cDNAs in the high mismatch-rate class with the high mismatch rate.

For each cDNA c we count the number of matching nucleotides m_c and the number of mismatching nucleotides n_c in the mapping. We assume that the mappings are drawn from a mixture in which a fraction ρ has a high mismatch rate h , and a fraction $1 - \rho$ has a low mismatch rate l . Under this mismatch model the probability P of the data is given by

$$P = \prod_c [h^{n_c}(1-h)^{m_c}\rho + l^{n_c}(1-l)^{m_c}(1-\rho)]. \quad (26)$$

We then determine the parameters ρ , h , and l that maximize P for the given mappings. Using these parameters we calculate for each cDNA c the posterior probability

$$H_c = \frac{h^{n_c}(1-h)^{m_c}\rho}{h^{n_c}(1-h)^{m_c}\rho + l^{n_c}(1-l)^{m_c}(1-\rho)} \quad (27)$$

that the cDNA stems from the high mismatch-rate class. Finally, we remap all cDNAs with $H_c \geq 0.5$ using the high mismatch rate h .

Inferring splice boundary probabilities. To infer the splice boundary probabilities $P_{sb}(s_1s_2s_3s_4)$ we first identify all the splice boundaries in the mappings that map without any errors in the ten positions to the left and right of the boundary. We refer to these here as trusted boundaries. As illustrated in Figure 1, a splice boundary can be degenerate in that it can be moved to the left or right without introducing any errors in the mappings. For each trusted boundary we determine how much the splice boundary can be moved to the left and right without introducing errors.

For each trusted boundary b we denote the number of different possible error-free splice boundaries as k_b and denote the quadruplet of nucleotides formed by the first two nucleotides and last two nucleotides of the intron that is obtained when the splice boundary is put at the a th of the k_b positions as s_a . Imagine that we have chosen a splice boundary position a_b for each boundary b . We call such a collection of positions an “assignment” A of splice boundaries to the data D . Given an assignment A and a set of splice boundary probabilities $p_{\vec{s}}$ the probability of the data D is given by

$$P(D|p, A) = \prod_{\vec{s}} (p_{\vec{s}})^{n_{\vec{s}}(A)}, \quad (28)$$

with $n_{\vec{s}}(A)$ the number of times boundary $\vec{s}$ occurs in the assignment A . To obtain the posterior probability $P(p|D)$ of splice boundary probabilities given the data D we need a prior probability $P(p)$ over splice boundary probabilities and we need to sum over the “nuisance parameters” A . Formally this gives us

$$P(p|D) = \frac{\sum_A P(D|p, A)P(p)}{\int d\vec{p} \sum_A P(D|\vec{p}, A)P(\vec{p})}, \quad (29)$$

where $P(p)$ is the prior probability for the set of splice boundary probabilities $p_{\vec{s}}$ and the integral $d\vec{p}$ is over all splice boundary probability distributions $\vec{p}_{\vec{s}} \geq 0$ with $\sum_{\vec{s}} \vec{p}_{\vec{s}} = 1$. It can be argued [19] that an ignorance prior on the probabilities $p_{\vec{s}}$ is given by a uniform distribution over their logs

$$P(p) \propto \prod_{\vec{s}} (p_{\vec{s}})^{-1}. \quad (30)$$

Putting all this together we have

$$P(p|D) = \frac{\sum_A \prod_{\vec{s}} (p_{\vec{s}})^{n_{\vec{s}}(A)-1}}{\sum_A \int \prod_{\vec{s}} (\vec{p}_{\vec{s}})^{n_{\vec{s}}(A)-1} d\vec{p}}. \quad (31)$$

Finally, the expectation value $\langle p_{\vec{s}} \rangle$ for a given component $p_{\vec{s}}$ is given by

$$\langle p_{\vec{s}} \rangle = \frac{\sum_A \int p_{\vec{s}} \prod_{\vec{\sigma}} (\vec{p}_{\vec{\sigma}})^{n_{\vec{\sigma}}(A)-1} d\vec{p}}{\sum_A \int \prod_{\vec{\sigma}} (\vec{p}_{\vec{\sigma}})^{n_{\vec{\sigma}}(A)-1} d\vec{p}}. \quad (32)$$

The integrals can be performed analytically to yield

$$\langle p_{\vec{s}} \rangle = \frac{\sum_A \frac{n_{\vec{s}}(A)}{n} \prod_{\vec{\sigma}} \Gamma[n_{\vec{\sigma}}(A)]}{\sum_A \prod_{\vec{\sigma}} \Gamma[n_{\vec{\sigma}}(A)]}, \quad (33)$$

with n the total number of splice boundaries in the data. Note that we can interpret this expression as follows. We consider all assignments A and give each assignment A a weight $W(A) = \prod_{\vec{s}} \Gamma[n_{\vec{s}}(A)]$. We then calculate the proportions $n_{\vec{s}}(A)/n$ of each splice boundary $\vec{s}$ in assignment A and average them over the assignments using the weights $W(A)$. Note that, by applying Stirling's approximation, the weights $W(A)$ can be shown to approximately equal $e^{-nH(A)}$, with $H(A)$ the entropy of the frequency distribution $n_{\vec{s}}(A)/n$ of splice boundaries in assignment A . That is, the smaller the entropy of the distribution of observed frequencies in assignment A , the larger the weight $W(A)$ of this assignment in the average.

In practice there are too many possible assignments to explicitly sum over all of them, and we use Monte-Carlo Markov chain sampling to estimate the sum as follows. (1) Start with a random assignment A of the splice junctions. (2) Pick a boundary b at random. (3) For all a , calculate the assignment weights W_a that are obtained when assigning the splice junction to the a th position. (4) Assign the splice boundary to position a with probability

$$p_a = W_a / \left(\sum_{k=1}^{h_b} W_k \right). \quad \text{Note that, because } \Gamma(n+1)/\Gamma(n) = n, p_a \text{ is simply}$$

proportional to the total number of occurrences $n_{\vec{s}_a}$ of $\vec{s}_a$ at all other splice boundaries in the current assignment. (5) Return to Step 2.

During this sampling, we measure the average frequencies $n_{\vec{s}}/n$ of each of the possible splice junctions $\vec{s}$. Note that once a particular splice junction $\vec{s}$ no longer occurs in the assignment, it can never reappear. That is, the sampling automatically minimizes the number of different splice boundary sequences $\vec{s}$.

For the mappings of the mouse and human datasets we ran this sampling algorithm several times and verified that the averaged frequencies $n_{\vec{s}}/n$ average to the same values in all runs for each organism. These average frequencies then give the splice boundary probabilities $P_{sb}(s_1s_2s_3s_4)$. We have implemented the Monte-Carlo Markov chain sampling algorithm in a separate C program that will be distributed with the SPA code so that users may estimate their own splice boundary probabilities from their own sets of trusted boundaries.

Performance comparison on human full-length cDNAs. We performed a first mapping of the dataset of human full-length cDNAs using SPA with a “default” set of parameters for the distributions $P(m|g)$ and $P(c|m)$. These initial parameters were estimated from BLAT alignments of the set of human RefSeq [20] mRNAs and contain increased splice boundary probabilities for the known splice sites GT-AG, GC-AG, and AT-AC. We estimated a new set of parameters from these mappings and remapped the high mismatch-rate cDNAs as described above. We then performed a second round of mappings with these new parameters. Finally, we produced a final set of parameters from this second round of mappings and confirmed that the parameters estimated after the first and second round of mappings are almost identical.

Since Sim4 and Spidey are much slower than BLAT, GMAP, and SPA, they have to be provided with relatively small genomic loci for each cDNA in order to run the whole dataset in a reasonable amount of time. For fairness of comparison, for each cDNA we ran Sim4 and Spidey on the same set of loci from the BLAT gtfServer as SPA was run on. We produced BLAT alignments using the `-fine` option. BLAT does not explicitly distinguish deletions from introns, and we chose to consider every deletion of more than 30 nucleotides an intron.

For GMAP and Spidey we found that a number of output files gave inconsistent alignments. In GMAP's case these were very rare. In a few cDNAs the cDNA sequence in GMAP's output did not exactly match the cDNA in the input. In Spidey's case there was a significant number of cDNAs where the mapping contained exons whose genomic coordinates overlapped. There were also a few cases where the reported exon lengths did not match the number of bases in the reported alignment. We considered all cDNAs with such corrupted mappings as unmapped. For Spidey there were also a significant number cases in which the reported genomic loci were incorrect by one or a few nucleotides. We fixed these errors using a post-processing script and still considered these mappings.

We applied our parameter estimation procedure to the mappings of Sim4, GMAP, BLAT, and Spidey, including the splice boundary inference, to obtain parameter sets adapted to the mappings of each of these algorithms. For each algorithm X we scored all its mappings and all SPA's mappings using the parameter set estimated from X 's mappings. The distributions of the score differences for each

algorithm are shown as the dashed lines in Figure 4. Note that SPA does not consider introns of length less than 30. To score the alignments of the other algorithms using SPA's scoring we assigned a log probability of $-10,000$ to each intron of length less than 30.

For each 3' end insertion in each of the mappings we calculated the fraction of A nucleotides in this end segment of the cDNA. Whenever this fraction was 0.8 or larger we considered the end insertion a poly-A tail. The number of nucleotides in poly-A tails is shown in Table S1.

We found that the initial and terminal exons that were mapped by only some of the algorithms contained a higher fraction of non-canonical boundaries, boundaries lying in genome gaps, et cetera. Therefore, to compare the quality of the alignments around splice boundaries we restricted the analysis to those boundaries that occurred at positions in the cDNA that were mapped by both algorithms being compared. That is, for each combination of SPA and another algorithm X , we compared the mapping of each cDNA and extracted the segment of the cDNA that was mapped by both algorithms. More specifically, we determined the first and last cDNA nucleotide mapped by SPA, and the first and last nucleotide mapped by X , and intersected the two cDNA segments bounded by these starts and ends. We then extracted only the splice boundaries that lay within this intersection. For each splice boundary we counted the number of mismatches, insertions, and deletions that lay within ten alignment positions of the splice boundary, and summed these over all cDNAs. We also determined the fraction of the boundaries that did not match a known splice site, i.e., GT-AG, GC-AC, or AT-AC.

For the conservation statistics we downloaded the phastcon profiles [10] from the UCSC genome database [21]. For each combination of SPA and one of the other algorithms we determined the set of genomic nucleotides S that occurred only in SPA's mappings, and the set of genomic nucleotides O that occurred only in the mappings of the other algorithm. For each conservation score c we then calculated the difference in the number of nucleotides with at least conservation score c that were unique to SPA's mappings and the number of nucleotides with conservation score at least c that were unique to the other algorithm's mappings. These distributions are shown in the left panel of Figure 6.

For the right panel of Figure 6 we compared the distribution of conservation scores of the nucleotides S that were unique to SPA's mappings with the distribution of conservation scores of the nucleotides O that were unique to the mappings of each of the other algorithms. Formally, we divided the interval from zero to one into 100 equally sized bins and determined the percentages $p_S(c) = S(c)/S$ and $p_O(c) = O(c)/O$ of nucleotides at each conservation score that were unique to the respective mappings. The right panel of Figure 6 shows the ratio $p_S(c)/p_O(c)$ of these distributions. Thus, whenever the curve in the right panel is above one, it means that the nucleotides with that conservation score were relatively more common in the set S than they were in the set O .

Supporting Information

Table S1. Statistics of the Mappings of the Set of Human cDNAs
Found at DOI: 10.1371/journal.pgen.0020024.st001 (37 KB PDF).

Table S2. Statistics of the Mappings of the FANTOM3 Mouse Full-Length cDNAs
Found at DOI: 10.1371/journal.pgen.0020024.st002 (36 KB PDF).

Acknowledgments

Development of the algorithm presented in this paper was initiated by MZ and EvN when they were, respectively, at the Laboratory of Computational Genomics and the Center for Studies in Physics and Biology of Rockefeller University. During that time a draft version of the program was programmed by RS. The bulk of SPA's development was done by EvN and MZ at the Biozentrum in Basel, and it was programmed by NP.

Author contributions. EvN and MZ conceived and designed the experiments. EvN, NP, and MZ performed the experiments. EvN and MZ analyzed the data. NP and RS contributed reagents/materials/analysis tools. EvN and MZ wrote the paper.

Funding. The authors received no specific funding for this study.

Competing interests. The authors have declared that no competing interests exist. ■

References

1. Carninci P, Kasukawa T, Katayama S, Gough J, Frith MC, et al. (2005) The transcriptional landscape of the mammalian genome. *Science* 309: 1559–1563.
2. Florea L, Hartzell G, Zhang Z, Rubin GM, Miller W (1998) A computer program for aligning a cDNA sequence with a genomic DNA sequence. *Genome Res* 8: 967–974.
3. Kent WJ (2002) BLAT—The BLAST-like alignment tool. *Genome Res* 12: 656–664.
4. Wu TD, Watanabe CK (2005) GMAP: A genomic mapping and alignment program for mRNA and EST sequences. *Bioinformatics* 21: 1859–1875.
5. Wheelan SJ, Church DM, Ostell JM (2001) Spidey: A tool for mRNA-to-genomic alignments. *Genome Res* 11: 1952–1957.
6. Ota T, Suzuki Y, Nishikawa T, Otsuki T, Sugiyama T, et al. (2004) Complete sequencing and characterization of 21,243 full-length human cDNAs. *Nat Genet* 36: 40–45.
7. Jackson IJ (1991) A reappraisal of non-consensus mRNA splice sites. *Nucleic Acids Res* 19: 3795–3798.
8. Hall SL, Padgett RA (1994) Conserved sequences in a class of rare eukaryotic nuclear introns with non-consensus splice sites. *J Mol Biol* 239: 357–365.
9. National Center for Biotechnology Information (2005) Consensus CDS [database]. Bethesda (Maryland): National Center for Biotechnology Information. Available: <http://www.ncbi.nlm.nih.gov/projects/CCDS>. Accessed 24 February 2006.
10. Siepel A, Bejerano G, Pedersen JS, Hinrichs A, Hou M, et al. (2005) Evolutionarily conserved elements in vertebrate, insect, worm, and yeast genomes. *Genome Res* 15: 1034–1050.
11. Kent WJ, Sugnet CW, Furey TS, Roskin KM, Pringle TH, et al. (2002) The human genome browser at UCSC. *Genome Res* 12: 996–1006.
12. Lewis BP, Burge CB, Bartel DP (2005) Conserved seed pairing, often flanked by adenosines, indicates that thousands of human genes are microRNA targets. *Cell* 120: 15–20.
13. Krek A, Grun D, Poy MN, Wolf R, Rosenberg L, et al. (2005) Combinatorial microRNA target predictions. *Nat Genet* 37: 495–500.
14. Burge C, Karlin S (1997) Prediction of complete gene structures in human genomic DNA. *J Mol Biol* 268: 78–94.
15. Zavolan M, Kondo S, Schonbach C, Adachi J, Hume D, et al. (2003) Impact of alternative initiation, splicing, and termination on the diversity of the mRNA transcripts encoded by the mouse transcriptome. *Genome Res* 13: 1290–1300.
16. Romani A, Guerra F, Trerotola M, Alberti S (2003) Detection and analysis of spliced chimeric mRNAs in sequence databanks. *Nucleic Acids Res* 31: e17.
17. Zavolan M, van Nimwegen E, Gaasterland T (2002) Splice variation in mouse full-length cDNAs identified by mapping to the mouse genome. *Genome Res* 12: 1377–1385.
18. Lim LP, Burge CB (2001) A computational analysis of sequence features involved in recognition of short introns. *Proc Natl Acad Sci U S A* 98: 11193–11198.
19. Jaynes ET (2003) Probability theory: The logic of science. Cambridge: Cambridge University Press. 727 p.
20. Pruitt KD, Tatusova T, Maglott DR (2005) NCBI Reference Sequence (RefSeq): A curated non-redundant sequence database of genomes, transcripts, and proteins. *Nucleic Acids Res* 33: D501–D504.
21. Karolchik D, Baertsch R, Diekhans M, Furey TS, Hinrichs A, et al. (2003) The UCSC genome browser database. *Nucleic Acids Res* 31: 51–54.