BIOINFORMATICS

Vol. 00 no. 00 2004
Pages 1–8

Exploiting Conserved Structure for Faster Annotation of Non-coding RNAs Without Loss of Accuracy

Zasha Weinberg[a*] and Walter L. Ruzzo[b]

[a]Dept. of Computer Science & Engineering, University of Washington, Seattle, WA, 98195, USA; [b]Depts. of Computer Science & Engineering and Genome Sciences, University of Washington, Seattle, WA, 98195, USA

ABSTRACT

Motivation: Non-coding RNAs (ncRNAs)—functional RNA molecules not coding for proteins—are grouped into hundreds of families of homologs. To find new members of an ncRNA gene family in a large genome database, Covariance Models (CMs) are a useful statistical tool, as they use both sequence and RNA secondary structure information. Unfortunately, CM searches are slow. Previously, we introduced “rigorous filters,” which provably sacrifice none of CMs’ accuracy, while often scanning much faster. A rigorous filter, using a profile hidden Markov model (HMM), is built based on the CM, and filters the genome database, eliminating sequences that provably could not be annotated as homologs. The CM is run only on the remainder. Some biologically important ncRNA families could not be scanned efficiently with this technique, largely due to the significance of conserved secondary structure relative to primary sequence in identifying these families. Current heuristic filters are also expected to perform poorly on such families.

Results: By augmenting profile HMMs with limited secondary structure information, we obtain rigorous filters that accelerate CM searches for virtually all known ncRNA families from the Rfam Database and tRNA models in tRNAscan-SE. These filters scan an 8-gigabase database in weeks instead of years, and uncover homologs missed by heuristic techniques to speed CM searches.

Availability: software in development; contact the authors.

Supplementary information: http://bio.cs.washington.edu/supplements/zasha-ISMB-2004 (Additional technical details on the method; predicted homologs.)

Keywords: non-coding RNA, covariance model, gene family, profile hidden Markov model, rigorous filter.

Contact: {zasha,ruzzo}@cs.washington.edu

1 INTRODUCTION

Non-coding RNAs (ncRNAs) are functional RNA molecules that do not code for proteins. Well-known examples are tRNAs and spliceosomal RNAs, but recent discoveries reveal ncRNAs to be much more numerous and significant than previously thought (Storz, 2002; Hüttenhofer *et al.*, 2002; Eddy, 2002), e.g., with microRNAs and analogous bacterial RNAs that regulate other genes (Kennedy, 2002; Gottesman, 2002; Wagner & Flardh, 2002), and regulatory mRNA structural elements (Grillo *et al.*, 2003; Lai, 2003) that are essentially ncRNAs.

To exploit prior work on the over 100 known ncRNA families, it is useful to annotate genomes with family homologs. Since secondary structure is often functionally important to RNAs, this task requires modeling both sequence and secondary structure. Techniques for finding ncRNA family members include searching for patterns that can include base pairing (Macke *et al.*, 2001; Dsouza *et al.*, 1997; Grillo *et al.*, 2003), and searching for specific types of ncRNA, e.g., tRNAs (Lowe & Eddy, 1997; Fichant & Burks, 1991; Pavesi *et al.*, 1994), microRNAs (Lim *et al.*, 2003) and small nucleolar RNAs (Lowe & Eddy, 1999; Edvardsson *et al.*, 2003). These methods require significant expert input, making them hard to extend to new ncRNA families.

Two methods requiring modest manual work per family are covariance models (CMs) (Eddy & Durbin, 1994; Durbin *et al.*, 1998) and ERPIN (Gautheret & Lambert, 2001). Both require only a multiple alignment of family members annotated with a secondary structure. From this a statistical model is built and used to search a genome database. In tests, both techniques exhibit high sensitivity and specificity on, e.g., tRNAs (Gautheret & Lambert, 2001; Lowe & Eddy, 1997). A limitation of ERPIN is that it cannot accommodate non-consensus bulges in helices (which CMs can). Additionally, to prune its search, ERPIN sometimes requires the user to specify score thresholds for each helix, thus requiring more expert input and/or compromising accuracy. A limitation of CMs is that they cannot represent pseudoknots (which ERPIN can). It is not clear which limitation is more significant, but studies suggest that pseudoknots contain little information (Eddy & Durbin, 1994), whereas indels are common in many contexts.

*To whom correspondence should be addressed.

In addition to these theoretical issues, one serious pragmatic issue affects CMs: scans are very slow.

This paper seeks to address the impractical speed of CMs without sacrificing their accuracy. CMs are used in the Rfam Database (Griffiths-Jones *et al.*, 2003) to annotate an 8-gigabase genome database called RFAMSEQ for over 100 ncRNA families. CMs are too slow to be used directly; e.g., searching RFAMSEQ to find tRNAs would take about 1 year on a 2.8 GHz Intel Pentium 4 PC. Obviously, this is improving as computers get faster, but both the number of ncRNA families (currently well over 100) and the quantity of sequence data are rapidly expanding. To improve speed, Rfam uses a BLAST-based heuristic (Altschul *et al.*, 1997). For each ncRNA family, the known members are BLASTed against RFAMSEQ; the full CM is run only on matches returned by BLAST. These searches are acceptably fast, but the BLAST heuristic may miss family members that would be found with a regular (slower) CM search (Griffiths-Jones *et al.*, 2003), a particular concern for families with low sequence conservation, where tuning BLAST to be as sensitive as CMs may be difficult or impossible.

CMs are also used by tRNAscan-SE (Lowe & Eddy, 1997), the leading tool for annotation of tRNAs. To improve speed, tRNAscan-SE uses two programs previously created specifically for tRNA searches; if either of these programs, using permissive settings, reports a possible tRNA, the CM is run. Again, the heuristics may miss tRNAs that CMs would find.

We develop a *rigorous filter*. Unlike heuristics, rigorous filters guarantee that all sequences classified as homologs by the CM will be found; a rigorous filter will never increase the false negative rate over that of the CM. Previously, we created rigorous filters based on profile HMMs (Weinberg & Ruzzo, 2004). A profile HMM is built from the CM, and run against the database. Based on the output, much of the database can be eliminated as provably not containing any family members that would be detected by the CM. The CM is run only on what remains.

Surprisingly, although based on sequence conservation alone, these filters enabled rigorous scans for 126 of the 139 ncRNA families in Rfam 5.0 (ignoring those families using the Rfam local alignment feature; see section 6). Rigorous filtering remained impractical for the other 13 families. These 13 families tend to exhibit relatively low sequence conservation, but strong conservation of secondary structure. Not only are these families difficult for our previous rigorous filters, but one would expect that the BLAST filtering heuristic would miss many homologs in these families, since compensatory mutations preserving base pairing in helical regions, for example, can easily destroy the short exact matches that are central to BLAST alignments. Thus, incorporation of secondary structure information is essential for successful filtering of increasingly diverged ncRNA families.

In this paper, we introduce three innovations aimed at extending practical rigorous filtering to biologically important situations such as these. First, we present two techniques that deviate from the usual profile HMM architecture to include limited structure information to improve filtering at the cost of increased CPU time, while still being rigorous. The *sub-CM* technique mixes CMs and profile HMMs, using CMs for key structural elements. The *store-pair* technique uses additional HMM states to store information to better model key base pairs.

By varying parameters, both techniques can generate many filters; some run very quickly, but may not filter selectively, while others are more selective but slower. Our third innovation, to minimize overall scan time, is to run several filters in series, starting with the quickest, and ending with the most selective. We solve the problem of selecting the optimal series of filters as a classic shortest path problem.

We applied these techniques to the 13 ncRNA families that were not practical with our previous technique, including biologically important ncRNAs such as the tRNAs, signal recognition particle (SRP), RNase P, two snRNAs and three riboswitches (Winkler & Breaker, 2003; Vitreschak *et al.*, 2004)—mRNA structural elements that regulate thiamin, lysine and vitamin B12 genes. It is practical to rigorously scan an 8-gigabase nucleotide database for all but 2 of these families. In all 11 successful families, these scans find new hits (putative homologs) missed by the BLAST heuristic; in many cases the new hits are supported by annotations or are otherwise biologically plausible.

When applied to tRNAscan-SE, our techniques permit a rigorous scan of three of its four CMs (archaeal, eubacterial and nuclear eukaryotic), finding several hits missed by tRNAscan-SE's heuristics. We are unable to improve speed for its fourth model (organellar), although tRNAscan-SE by default runs the raw CM for these small genomes.

The next section gives results. Section 3 reviews salient features of CMs, and section 4 summarizes profile HMM-based rigorous filters. Section 5 describes the sub-CM, store-pair and filter selection techniques. Section 6 concludes.

2 RESULTS

2.1 Summary

Table 1 lists results on 11 Rfam ncRNA families and 3 tRNAscan-SE models. Rigorous filtering is practical on these families, averaging about 100 times faster than a raw CM, and for all families uncovers family members missed by Rfam/tRNAscan-SE heuristics.

The rigorous filters run slower than tRNAscan-SE, by a factor of 10 for prokaryotes and 100 for eukaryotes, for which tRNAscan-SE was chiefly designed. However, our scans are rigorous, and, unlike tRNAscan-SE, not specific to tRNAs. It may be possible to tune tRNAscan-SE to find the new homologs, but for BLAST, this is not straightforward. For example, we ran a BLAST filter for RF00168. Finding even 9 of the 11 new hits required an E-value threshold of 10000, degrading

ncRNA family	avg len	% id	# known	# new	filters +CM time (days)	est. CM time (days)	data-base size (Mb)
RF00001 5S rRNA	115	61	5460	14	14.0	651	8295
RF00004 U2 snRNA	171	60	466	1	9.1	2110	8295
RF00005 tRNA	71	43	58609	5158	31.0	335	8295
RF00009 nuclear RNase P	290	41	69	3	14.2	6240	8295
RF00010 bacterial RNase P	317	62	413	1	31.9	9029	8295
RF00017 SRP	299	49	128	13	14.9	5493	8295
RF00023 tmRNA	345	46	226	21	18.7	9587	8295
RF00029 Group II intron	75	55	5708	331	6.9	666	8295
RF00059 thiamin element	104	52	276	7	10.7	2485	8295
RF00168 lysine riboswitch	181	49	60	11	21.2	1724	8295
RF00174 cobalamin riboswitch	202	47	170	7	38.1	5081	8295
tRNAscan-SE archaea	-	-	1016	15	0.1	5	47
tRNAscan-SE eubacteria	-	-	13624	87	1.8	80	640
tRNAscan-SE *Drosophila* nuclear	-	-	296	1	0.7	21	117
tRNAscan-SE *C. elegans* nuclear	-	-	822	16	2.7	18	100
tRNAscan-SE human nuclear	-	-	608	121	26.2	562	3070

Table 1. Results of rigorous filtering experiments. Each row in this table is one Rfam or tRNAscan-SE ncRNA family, described in the first column, with Rfam Id if applicable. Next is its average length in nucleotides and % identity (sequence conservation) as reported in Rfam (not available for tRNAscan-SE models). # known is the number of members in RFAMSEQ reported by Rfam, or by running tRNAscan-SE on the appropriate subset of RFAMSEQ. # new is the number of *additional* matches in RFAMSEQ that our technique found. The CPU time taken to scan RFAMSEQ or subset on a 2.8 GHz Pentium 4 is next, then the estimated time for a pure CM scan (extrapolated from a 10 Kbase scan). The RFAMSEQ or subset size is given in megabases. tRNAscan-SE was run with default parameters, with the domain of life specified with appropriate flags; for raw CM and rigorous scans, its default window length of 250 was used. Note: RF00005's scan covered more sequence data, including organellar DNA, so has many more hits than tRNAscan-SE model scans.

BLAST's selectivity so that searching takes 3 times longer than our rigorous scan.

2.2 Impractical ncRNAs

Our technique did not improve scan time significantly for two Rfam families. The SECIS element (RF00031) is a single long hairpin structure. This paper's techniques work best on RNAs with many hairpins, where spending more time on one hairpin does not inflate the overall ncRNA run time so significantly. We have no clear reason why RF00177, the 5' domain of the small subunit ribosomal RNA, proved impractical.

The technique was also unsuccessful on tRNAscan-SE's organellar tRNA model. The organellar model is based on training sequence from all three domains of life, which may dilute primary sequence features exploited by profile HMMs. Also, organellar tRNAs' score threshold is set lower than normal, so it is harder to prove that a sequence must score below the threshold.

2.3 Buried treasures

All families scanned in this paper revealed hits missed by Rfam/tRNAscan-SE. Many new hits found with rigorous filtering are supported by an annotation. 5S rRNA (RF00001) has 8 hits annotated as such, at least 4 based on studies specifically of 5S rRNA. Eukaryotic nuclear RNase P RNA (RF00009) has 3 hits in *Drosophila*, one of which is a partial RNase P sequence identified in a study of eukaryotic RNase P. Bacterial RNase P RNA (RF00010) has one archaeal hit in an annotated partial RNase P sequence. The thiamin element (RF00059) has three new hits upstream of predicted thiamin biosynthesis genes. The cobalamin riboswitch (RF00174) has new hits in the cbiA and cbiX genes, both required for cobalamin synthesis. Of 11 new hits for the lysine riboswitch (RF00168), all are upstream of genes: 6 to lysine-specific permeases, 3 to more general amino acid transporters, 1 to lysA (lysine biosynthesis), and 1 with no annotated function. Preliminary inspection of new group II intron (RF00029) and tRNA (RF00005 and tRNAscan-SE) hits, identified several with supporting annotation. Thus, rigorous filters have uncovered homologs missed by heuristics, and will potentially lead to a better understanding of ncRNA families.

3 SIMPLIFIED CMS

Covariance Models (CMs) are statistical models that can detect when positional sequence and secondary structure resembles a given multiple RNA alignment. To simplify the presentation, as in (Weinberg & Ruzzo, 2004), we simplify CMs, e.g., ignoring unusual inserted nucleotides, and describe them unconventionally in terms of stochastic context-free grammars (SCFGs). This paper's techniques extend to fully general CMs analogously to the profile HMM technique (Weinberg & Ruzzo, 2004). Readers unfamiliar with context-free grammars may find chapter 9 of (Durbin *et al.*, 1998) helpful.

3.1 CM grammar rules

Consider RNA molecules with sequence CAG or GAC with the C,G bases paired. A context-free grammar (CFG) for this is $\mathrm{S}_1 \to c\mathrm{S}_2 g | g\mathrm{S}_2 c$ and $\mathrm{S}_2 \to a$. (By convention nucleotides in the CFG are lowercase.) S_1 and S_2 are called *states*, and S_1 the *start state*. The first rule says that S_1 may be replaced by either $c\mathrm{S}_2 g$ or $g\mathrm{S}_2 c$. So, we can produce the string CAG by the following steps, beginning with the start state: $\mathrm{S}_1 \to c\mathrm{S}_2 g \to cag$. The series of steps from start state to RNA sequence is called a *parse*.

CMs have states $\mathrm{S}_1, \mathrm{S}_2, \ldots, \mathrm{S}_n$ for each of n (possibly base-paired) alignment positions. CFG rules of a restricted form codify sequence and structure characteristics, although this paper does not explain how to select these rules. All rules are of the form $\mathrm{S}_i \to x_L \mathrm{S}_{i+1} x_R$, where x_L (left nucleotide) and x_R (right) may either be a nucleotide (a,c,g,u) or the empty character ϵ, which produces no nucleotide. If x_L and x_R are both nucleotides, the rule emits paired nucleotides. If $x_L = \epsilon$ or $x_R = \epsilon$ or both, the rule emits an unpaired nucleotide or no nucleotide; such rules can accommodate missing consensus positions. Special *bifurcation states* have rules like $\mathrm{S}_i \to \mathrm{S}_j \mathrm{S}_k$ for $j, k > i$, which allow for ncRNAs with multi-loops. Figure 1 demonstrates these concepts.

3.2 CM genome annotation

Each rule has a probability. Rules more consistent with an ncRNA family have higher probabilities than less plausible rules. A parse's probability is the product of the probabilities of the rules used in that parse, e.g., parse probability $\Pr(\mathrm{S}_1 \to c\mathrm{S}_2 g \to cag) = \Pr(\mathrm{S}_1 \to c\mathrm{S}_2 g) \times \Pr(\mathrm{S}_2 \to a)$. Instead of probabilities, CMs usually employ odds ratios, relative to a simple background model. For computational convenience, the logarithm of the odds ratio is used; a parse's score is the sum of the logarithmic scores for the rules used in the parse.

For each genome database subsequence, the highest-scoring, or *Viterbi*, parse is computed by dynamic programming (Eddy & Durbin, 1994; Durbin *et al.*, 1998). If a subsequence's Viterbi score exceeds a user-supplied, family-specific threshold, the subsequence is considered a family member. The CM Viterbi algorithm requires a user-supplied *window length*, which is an upper bound on the family member's length, and a factor in CM scan time complexity.

4 PROFILE HMM FILTERS

This section recaps background from (Weinberg & Ruzzo, 2004) on rigorous profile HMM filters relevant to *augmented* rigorous HMM filters, the main contribution of this paper. Given a CM, we create a profile HMM whose Viterbi score for any sequence is always an upper bound on that of the CM. Although profile HMMs are less powerful than CMs, their Viterbi algorithm is much faster, which makes them an attractive filter. First, we describe how we will filter with a profile or augmented HMM, then explain how to convert CMs into profile HMMs, modeled as stochastic regular grammars. To guarantee rigorous filtering, the HMM rules' logarithmic scores are constrained such that the HMM's score for a database subsequence is an upper bound on the CM's score, a property maintained in augmented HMMs.

4.1 Filtering

Using the HMM, we compute CM score upper bounds for subsequences ending at each nucleotide position in the database sequence. When an upper bound exceeds the threshold, a CM scan is applied to a window of size *window length*. If the HMM-generated upper bound is below the threshold, the CM cannot report a homolog at that location, so it is safely filtered out.

4.2 Profile HMM grammar

Regular grammars are less powerful than SCFGs in that their rules cannot emit paired nucleotides. Their rules must be of the form $\mathrm{S}_i \to x_L \mathrm{S}_{i+1}$.

Consider a CM with two states, and rules $\mathrm{S}_1 \to a\mathrm{S}_2 u | c\mathrm{S}_2 g; \quad \mathrm{S}_2 \to \epsilon$. A profile HMM cannot represent the fact that the bases are paired, but can reflect the sequence information by breaking S_1 into two HMM states: $\overline{\mathrm{S}}_1^L$ handles the left nucleotide and $\overline{\mathrm{S}}_1^R$ the right. (HMM states will be written with a bar to differentiate them from CM states.) Here is a regular grammar: $\overline{\mathrm{S}}_1^L \to a\overline{\mathrm{S}}_2^L | c\overline{\mathrm{S}}_2^L; \quad \overline{\mathrm{S}}_2^L \to \overline{\mathrm{S}}_1^R; \quad \overline{\mathrm{S}}_1^R \to g|u$. This profile HMM grammar encodes the fact that the first nucleotide is A or C, and the second G or U, but it sacrifices the information that only A-U or C-G pairs are permitted; e.g., it allows AG. This sacrifice is a limitation of profile HMMs.

In general, a CM state S_i is expanded into a *left HMM state* $\overline{\mathrm{S}}_i^L$ and a *right HMM state* $\overline{\mathrm{S}}_i^R$. All CM rules $\mathrm{S}_i \to x_L \mathrm{S}_{i+1} x_R$ are converted into HMM rules $\overline{\mathrm{S}}_i^L \to x_L \overline{\mathrm{S}}_{i+1}^L$ and $\overline{\mathrm{S}}_i^R \to x_R \overline{\mathrm{S}}_{i-1}^R$. (The subscript on $\overline{\mathrm{S}}^R$ is decremented since right nucleotides are emitted in reverse order.) If $i = 1$, then we omit $\overline{\mathrm{S}}_{i-1}^R$. See Table 2 for an example.

4.3 Constraints on scores

To ensure rigorous filtering, we define constraints ensuring that the HMM's Viterbi parse score for any database subsequence upper bounds the CM's Viterbi score. Any CM parse consists of a sequence of rules, which can be mapped to HMM rules (according to our construction), yielding a corresponding HMM parse. A parse score is the sum of its rules' logarithmic scores. We enumerate all CM rules $\mathrm{S}_i \to x_L \mathrm{S}_{i+1} x_R$ for all i. For each rule, we require the sum

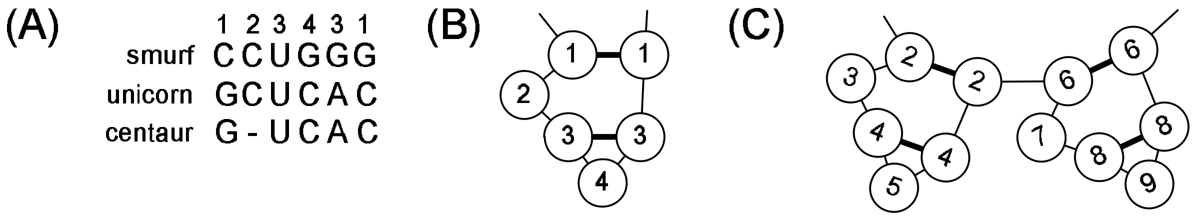

Fig. 1. RNA multiple alignment, structure and CM. (A) A hypothetical multiple RNA alignment. Dashes (-) indicate missing nucleotides. (B) The structure. Thick lines are conserved base-pairs. Numbers refer to alignment positions; positions 1 and 3 are base paired, so appear twice. A CM encoding these sequences and structures is $\mathrm{S}_1 \rightarrow c\mathrm{S}_2g|g\mathrm{S}_2c$; $\mathrm{S}_2 \rightarrow \epsilon\mathrm{S}_3\epsilon|c\mathrm{S}_3\epsilon$; $\mathrm{S}_3 \rightarrow u\mathrm{S}_4a|u\mathrm{S}_4g$; $\mathrm{S}_4 \rightarrow c\mathrm{S}_5\epsilon|g\mathrm{S}_5\epsilon$; $\mathrm{S}_5 \rightarrow \epsilon$. (Note: normally CMs use less rigid grammars, allowing anomalous nucleotides with lower probability.) A parse of the unicorn sequence is $\mathrm{S}_1 \rightarrow g\mathrm{S}_2c \rightarrow gc\mathrm{S}_3\epsilon c \rightarrow gcu\mathrm{S}_4a\epsilon c \rightarrow gcuc\mathrm{S}_5\epsilon a\epsilon c \rightarrow gcuc\epsilon\epsilon a\epsilon c = gcucac$. (C) This structure has two loops like in (B), implemented with bifurcation state $\mathrm{S}_1 \rightarrow \mathrm{S}_2\mathrm{S}_6$.

CM state rules	left HMM state rules	right HMM state rules
$\mathrm{S}_1 \rightarrow c\mathrm{S}_2g\|g\mathrm{S}_2c$	$\overline{\mathrm{S}}_1^L \rightarrow c\overline{\mathrm{S}}_2^L\|g\overline{\mathrm{S}}_2^L$	$\overline{\mathrm{S}}_1^R \rightarrow c\|g$
$\mathrm{S}_2 \rightarrow \epsilon\mathrm{S}_3\epsilon\|c\mathrm{S}_3\epsilon$	$\overline{\mathrm{S}}_2^L \rightarrow \epsilon\overline{\mathrm{S}}_3^L\|c\overline{\mathrm{S}}_3^L$	$\overline{\mathrm{S}}_2^R \rightarrow \epsilon\overline{\mathrm{S}}_1^R$
$\mathrm{S}_3 \rightarrow u\mathrm{S}_4a\|u\mathrm{S}_4g$	$\overline{\mathrm{S}}_3^L \rightarrow u\overline{\mathrm{S}}_4^L$	$\overline{\mathrm{S}}_3^R \rightarrow a\overline{\mathrm{S}}_2^R\|g\overline{\mathrm{S}}_2^R$
$\mathrm{S}_4 \rightarrow c\mathrm{S}_5\epsilon\|g\mathrm{S}_5\epsilon$	$\overline{\mathrm{S}}_4^L \rightarrow c\overline{\mathrm{S}}_5^L\|g\overline{\mathrm{S}}_5^L$	$\overline{\mathrm{S}}_4^R \rightarrow \epsilon\overline{\mathrm{S}}_3^R$
$\mathrm{S}_5 \rightarrow \epsilon$	$\overline{\mathrm{S}}_5^L \rightarrow \overline{\mathrm{S}}_4^R$	

Table 2. Example of converting a CM to a profile HMM. The CM grammar of Figure 1 is converted to a profile HMM grammar, rule by rule. The HMM can be read in sequential order by going down the middle column, then up the right column.

of the corresponding HMM rules' logarithmic scores to be greater or equal to the CM rule's score.

For example, CM rule $\mathrm{S}_3 \rightarrow u\mathrm{S}_4a$ corresponds to HMM rules $\overline{\mathrm{S}}_3^L \rightarrow u\overline{\mathrm{S}}_4^L$ and $\overline{\mathrm{S}}_3^R \rightarrow a\overline{\mathrm{S}}_2^R$ (see Table 2); $\mathrm{S}_3 \rightarrow u\mathrm{S}_4g$ corresponds to $\overline{\mathrm{S}}_3^L \rightarrow u\overline{\mathrm{S}}_4^L$ (again) and $\overline{\mathrm{S}}_3^R \rightarrow g\overline{\mathrm{S}}_2^R$. Let l_1 be the logarithmic score for $\overline{\mathrm{S}}_3^L \rightarrow u\overline{\mathrm{S}}_4^L$, l_2 for $\overline{\mathrm{S}}_3^R \rightarrow a\overline{\mathrm{S}}_2^R$ and l_3 for $\overline{\mathrm{S}}_3^R \rightarrow g\overline{\mathrm{S}}_2^R$. Let the score of CM rule $\mathrm{S}_3 \rightarrow u\mathrm{S}_4a$ be -1 and $\mathrm{S}_3 \rightarrow u\mathrm{S}_4g$ be -2. Then each CM rule yields one inequality: $l_1 + l_2 \geq -1$ and $l_1 + l_3 \geq -2$, e.g., $l_1 = -1, l_2 = l_3 = 0$.

Any assignment of HMM scores (like l_1, l_2, l_3) satisfying all inequalities ensures the upper bound. A method to optimize the scores for filtering is given in (Weinberg & Ruzzo, 2004); in this paper, we assume scores are given.

5 AUGMENTED FILTERS

We present two techniques that augment profile HMMs for more selective filtering. We first describe how to select an efficient series of filters from many potential filters. Then we describe how to create candidate filters with each technique. The complete process of filter creation and selection has taken from 1 to 50 CPU hours per family.

5.1 Selecting a series of filters

Given a set of filters, we wish to select an optimal series of filters to apply. First, each filter is run on a training sequence to estimate its *filtering fraction* and *run time*. The filtering fraction is the fraction of the original database remaining after filtering, which the next filter or CM must be run on. Fractions range from 0 (perfect) to 1 (worst). Run time is CPU time per nucleotide. As an example, Figure 2 shows input filters created with the store-pair and sub-CM techniques, and the series of filters selected.

We wish to select a series of filters, preceding the CM, to minimize expected total run time. For example, suppose filter f_1 has filtering fraction 0.25 and run time 1 second per kilobase, and f_2 has fraction 0.01 at 10 s/Kb. If the CM takes 200 s/Kb, running f_2 beforehand takes $10 + 0.01 \times 200 = 12$ s/Kb. Better is running f_1, then f_2, and then the CM, at $1 + 0.25 \times 10 + 0.01 \times 200 = 5.5$ s/Kb.

To formalize this, we make two assumptions: (1) the estimated fractions and run times are accurate and constant, even though in reality they vary by sequence scanned, and (2) a filter's fraction is unaffected by which filters were applied previously. In the above example, f_1 followed by f_2 may filter better than f_2's fraction of 0.01, but we have never observed this effect to be significant. If these assumptions are significantly violated, scanning time may be unnecessarily increased, but rigorous filtering is still guaranteed. Our supplemental paper, at http://bio.cs.washington.edu/supplements/zasha-ISMB-2004, discusses these assumptions in more detail. (Briefly, test sequences must be sufficient large to obtain acceptably robust estimates, particularly for low filtering fractions. To reduce test scan time for filters with low fractions,

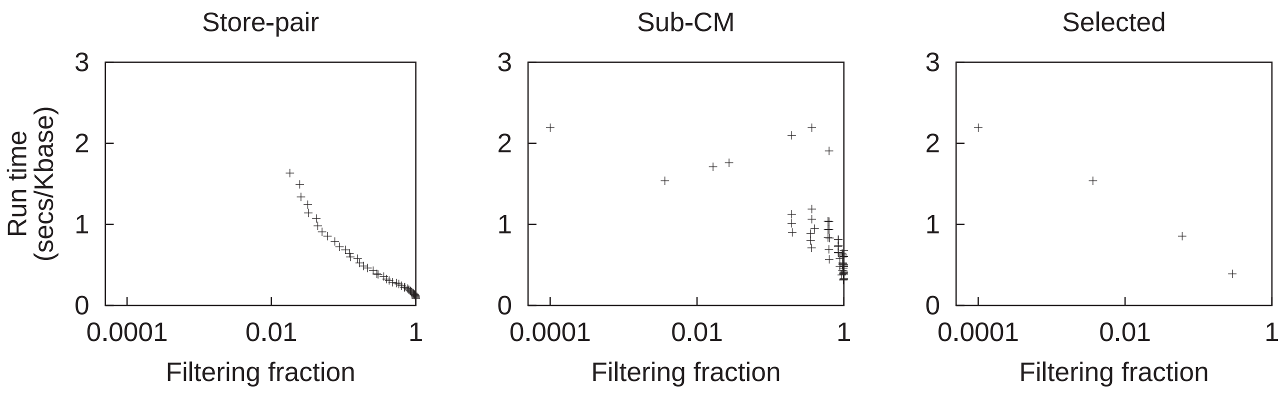

Fig. 2. Filter creation & selection. Filters for Rfam tRNA (RF00005) generated by the store-pair and sub-CM techniques and those selected for actual filtering are plotted by filtering fraction and run time. The CM runs at 3.5 secs/Kbase. The four selected filters are run one after another, from highest to lowest fraction.

large test sequences are prefiltered with less selective filters. The G+C content of the test sequence is also an important consideration.)

Given the above assumptions, selection of the best filter sequence can be cast as a shortest path graph problem (Cormen *et al.*, 1999). Nodes represent both filters and the CM, and a special INPUT node has fraction 1, taking time 0. The weight of the edge from filter f_1 to f_2 is the time it would take to run filter f_2 immediately after f_1: f_2's time multiplied by f_1's filtering fraction. The shortest path from INPUT to CM yields the optimal series of filters.

5.2 Sub-CM technique

Sub-CMs exploit information in hairpins, which are base-paired helices (including bulges and internal loops) that end in a single-stranded loop. In many ncRNA families, much secondary structure information is carried in short hairpins that use only a fraction of the CM's states. This motivates representing these hairpins using the appropriate part of the original CM (sub-CM), while using a profile HMM for the remainder of the ncRNA. Although the resulting hybrid grammar will take longer to scan than an HMM, it will often filter much more selectively, and still be faster than a full CM.

For a sub-CM rooted at state i, the CM Viterbi algorithm computes the maximum score from $\overline{\mathrm{S}}_i^L$ to $\overline{\mathrm{S}}_i^R$. In the augmented HMM algorithm, the Viterbi score to $\overline{\mathrm{S}}_i^R$ is the highest sum of sub-CM score ($\overline{\mathrm{S}}_i^L$ to $\overline{\mathrm{S}}_i^R$) plus HMM score for state $\overline{\mathrm{S}}_i^L$.

Typical hairpins are much smaller than the window length parameter W of the overall family. Sub-CM scan time could be saved by using a hairpin-specific window length $W' < W$. The lengths $W' \dots W$ must still be considered, or ncRNAs with unusually long hairpins but otherwise strong homology may be missed. To consider lengths $> W'$ efficiently, the maximum possible score for any sequence of that length is precomputed once and used. If W' is high enough, these maximum scores will be low, and sub-CM filtering will still be selective.

To create candidate sub-CM filters, for each CM hairpin, a sub-CM is created at its first paired position, with $W' = W$. A test sequence is then scanned, and binary search used to find the smallest W' for the hairpin such that the filtering fraction is the same as when $W' = W$. Then, a sub-CM is created at each subsequent paired position. After this, augmented HMMs are created manually that combine promising sub-CMs for yet more selective (if slower) filters. We plan to automate this process in a manner similar to the creation of store-pair filters (see below), estimating the run time and reduction in average Viterbi score of combined sub-CMs using the properties of the constituent sub-CMs, allowing fast prediction of promising combined sub-CMs.

5.3 Store-pair technique

Earlier we noted that profile HMMs cannot reflect which base pair was used, but only which nucleotides were used in the left and right positions. For example, the profile HMM of Table 2 could remember that each side of position 1 is C or G, but not that only C-G or G-C pairs are allowed.

However, an HMM with extra states can allow only C-G or G-C pairs. For each state $\overline{\mathrm{S}}_{1\dots5}^L$ and $\overline{\mathrm{S}}_{5\dots1}^R$, we create four states, one for each possible nucleotide that could be emitted by $\overline{\mathrm{S}}_1^L$. If $\overline{\mathrm{S}}_1^L(C)$ is the state for the emission of a C, then rule $\overline{\mathrm{S}}_1^L(C) \to g\overline{\mathrm{S}}_2^L(C)$ has score $-\infty$, since the emission of G is an inconsistency. The rule $\overline{\mathrm{S}}_1^L(C) \to c\overline{\mathrm{S}}_2^L(C)$ has the same score as in the original profile HMM. Using this extra information on the right side, the rule $\overline{\mathrm{S}}_1^R(C) \to c$ has score $-\infty$, since the rule is specific to the non-canonical C-C pair.

By contrast, $\overline{\mathrm{S}}_1^R(C) \rightarrow g$ corresponds to a C-G pair, and so maintains a high score.

5.3.1 Degrees of freedom The store-pair strategy can be generalized in three ways. First, any combination of states that represent base pairs in the CM can be multiplied, at the cost of a larger HMM. Note that applying the strategy to base-pair states nearer a loop will result in fewer total states.

To get maximal information, it is desirable to multiply the number of states by 5, which covers the four nucleotides plus the absent nucleotide case, i.e. the 5 symbols $\{a, c, g, u, \epsilon\}$. However, multiplying by 5 is not necessary. For example, in the above example, it suffices to store only 3 possible events: $\{c\}$, $\{g\}$ or $\{a, u, \epsilon\}$ — in the last case (not C or G), the scores are the same. More generally, we can choose any partition of $\{a, c, g, u, \epsilon\}$ (i.e., place each of the 5 events into exactly one subset), thus making a trade-off between the improved filtering achieved by having more information versus the increased scanning time of extra states.

Finally, the right nucleotide can be "stored" (i.e. $\overline{\mathrm{S}}_i^R$ instead of $\overline{\mathrm{S}}_i^L$), and used for the left HMM state's scores. Mathematically, this is simply the reverse of what was done above. When fewer than 5 events are stored, storing the right nucleotide may yield more information than the left.

5.3.2 Creating filters Exploiting these degrees of freedom in the store-pair technique, we now show how to create a useful set of filters, obtaining a time versus filtering curve as in Figure 2 from which to select a series. Noting that the run time of a store-pair filter is roughly proportional to the number of states in the resulting HMM, we propose to find, for each possible number of states, the filter with the lowest filtering fraction.

To efficiently solve this problem, we make three simplifying assumptions. First, instead of trying to minimize the filtering fraction, we attempt to minimize the average Viterbi score; if the Viterbi score is reduced, then fewer scores should be above the threshold, and the filtering fraction reduced. The second assumption is one of independence: the reduction in Viterbi scores versus the original profile HMM caused by applying store-pair to a set of base pairs is the sum of the Viterbi score reductions for each individual base pair. Finally, the user selects a constant c; if the profile HMM has n states, the store-pair HMMs should have fewer than cn states. $c = 250$ is a generous bound, since empirically the HMM is faster than the CM by a factor of approximately W (the window length); if $c > W$, the filter will be slower than the CM.

These assumptions permit a dynamic programming algorithm. For the ith CM state, while restricting store-pair to states $i \ldots n$, we recursively compute the optimal store-pair HMM of each possible number of states from 1 to cn. The optimal HMM and its estimated score reduction are stored by number of HMM states.

The base case, state n, has just the rule $\mathrm{S}_n \rightarrow \epsilon$. There is no room to apply store-pair, so the table has only one entry, containing the original profile HMM. For the recursion, we first enumerate all store-pair modifications of state i. For each modification, we run the resulting HMM on a short training sequence, to calculate its average Viterbi score reduction. We then consider each entry in state $i + 1$'s table. The combined score reduction is the sum of the score reduction just estimated on the training example plus the score reduction in the $i + 1$ table entry. The resulting number of states can be computed with simple arithmetic. The table for state i is then updated with the new HMM, unless there is already a better HMM with the same number of states. Bifurcation child states' tables are combined analogously.

Finally, we obtain a table for the first state giving, for each number of states up to cn, the (heuristically) optimal store-pair HMM. To prune away the large number of similar-performing HMMs, we run through the table, beginning with the fewest states, looking for the first HMM that predicts a Viterbi score reduction of at least 0.5 (a user-supplied parameter). We then store this HMM in a file, and look for the first HMM with predicted score reduction of an additional 0.5, until the table is exhausted, and a set of HMMs is saved.

We note that, as an alternative to calculating the average Viterbi score on a test sequence, we have been using the logarithm of the infinite-length forward algorithm score (Weinberg & Ruzzo, 2004), which was previously proposed as an approximation to the expected Viterbi score. On Rfam 5.0 tRNA (RF00005), we found that the two statistics correlated (correlation coefficient 0.999), and produced substantially the same filters, but the infinite-length forward algorithm score can be computed more quickly.

6 CONCLUSIONS

In terms of future work, it would be desirable to have more broadly applicable rigorous filters, particularly for the SECIS element. Our work also provides a benchmark against which to test the sensitivity of heuristic filters that may run faster.

We also expect that CMs will be extended to improve their accuracy and versatility. One recent extension to CMs is the *local alignment* feature (Eddy, 2003), which allows a match to a part of the ncRNA, and is intended to detect homologs of ncRNA domains. Our techniques could, in principle, be applied to local alignments, although the sub-CM technique will require a larger window length in hairpins, which may degrade its speed.

In conclusion, covariance models are useful in annotating genomes with homologs of known ncRNA gene families, but their slow speed is a practical problem. We have designed a methodology that significantly speeds up scanning for virtually all ncRNA families in Rfam 5.0 and three tRNAscan-SE models, with guarantees that no additional homologs will be missed. By speeding CMs without loss of accuracy, our technique improves our ability to refine CM-based models to better characterize each ncRNA family, and reveals biologically plausible homologs missed by previous techniques.

7 ACKNOWLEDGEMENTS

This material is based upon work supported in part by National Institutes of Health grants R01 HG02602 and NIH HG-00035.

REFERENCES

Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W. & Lipman, D. J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.,* **25**, 3389–3402.

Cormen, T. H., Leiserson, C. E. & Rivest, R. L. (1999) *Introduction to algorithms*. MIT Press, Cambridge, USA.

Dsouza, M., Larsen, N. & Overbeek, R. (1997) Searching for patterns in genomic data. *Trends Genet.,* **13**, 497–498.

Durbin, R., Eddy, S., Krogh, A. & Mitchison, G. (1998) *Biological sequence analysis*. Cambridge U. Press, Cambridge, UK.

Eddy, S. R. (2002) Computational genomics of noncoding RNA genes. *Cell,* **109**, 137–140.

Eddy, S. R. (2003) *Infernal User's Guide*. ftp://ftp.genetics.wustl.edu/pub/eddy/software/infernal/Userguide.pdf.

Eddy, S. R. & Durbin, R. (1994) RNA sequence analysis using covariance models. *Nucleic Acids Res.,* **22**, 2079–2088.

Edvardsson, S., Gardner, P. P., Poole, A. M., Hendy, M. D., Penny, D. & Moulton, V. (2003) A search for H/ACA snoRNAs in yeast using MFE secondary structure prediction. *Bioinformatics,* **19**, 865–873.

Fichant, G. & Burks, C. (1991) Identifying potential tRNA genes in genomic DNA sequences. *J. Mol. Biol.,* **220**, 659–671.

Gautheret, D. & Lambert, A. (2001) Direct RNA motif definition and identification from multiple sequence alignments using secondary structure profiles. *J. Mol. Biol.,* **313**, 1003–1011.

Gottesman, S. (2002) Stealth regulation: biological circuits with small RNA switches. *Genes Dev.,* **16**, 2829–2842.

Griffiths-Jones, S., Bateman, A., Marshall, M., Khanna, A. & Eddy, S. R. (2003) Rfam: an RNA family database. *Nucleic Acids Res.,* **31**, 439–441. http://rfam.wustl.edu.

Grillo, G., Licciulli, F., Liuni, S., Sbisà, E. & Pesole, G. (2003) PatSearch: a program for the detection of patterns and structural motifs in nucleotide sequences. *Nucleic Acids Res.,* **31**, 3608–3612.

Hüttenhofer, A., Brosius, J. & Bachellerie, J.-P. (2002) RNomics: identification and function of small, non-messenger RNAs. *Curr. Opin. Chem. Biol.,* **6**, 835—-843.

Kennedy, D. (2002) Breakthrough of the year. *Science,* **298**, 2283.

Lai, E. C. (2003) RNA sensors and riboswitches: self-regulating messages. *Curr. Biol.,* **13**, R285–R291.

Lim, L. P., Lau, N. C., Weinstein, E. G., Abdelhakim, A., Yekta, S., Rhoades, M. W., Burge, C. B. & Bartel, D. P. (2003) The microRNAs of *Caenorhabditis elegans*. *Genes Dev.,* **17**, 991–1008.

Lowe, T. & Eddy, S. R. (1997) tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence. *Nucleic Acids Res.,* **25**, 955–64.

Lowe, T. M. & Eddy, S. R. (1999) A computational screen for methylation guide snoRNAs in yeast. *Science,* **283**, 1168–1171.

Macke, T. J., Ecker, D. J., Gutell, R. R., Gautheret, D., Case, D. A. & Sampath, R. (2001) RNAMotif, an RNA secondary structure definition and search algorithm. *Nucleic Acids Res.,* **29**, 4724–4735.

Pavesi, A., Conterio, F., Bolchi, A., Dieci, G. & Ottonello, S. (1994) Identification of new eukaryotic tRNA genes in genomic DNA databases by a multistep weight matrix analysis of transcriptional control regions. *Nucleic Acids Res.,* **22**, 1247–1256.

Storz, G. (2002) An expanding universe of noncoding RNAs. *Science,* **296**, 1260–1263.

Vitreschak, A. G., Rodionov, D. A., Mironov, A. A. & Gelfand, M. S. (2004) Riboswitches: the oldest mechanism for the regulation of gene expression? *Trends Genet.,* **20**, 44–50.

Wagner, E. & Flardh, K. (2002) Antisense RNAs everywhere? *Trends Genet.,* **18**, 223–226.

Weinberg, Z. & Ruzzo, W. L. (2004) Faster genome annotation of non-coding RNA families without loss of accuracy. In *Proc. Eighth Annual Inter. Conf. on Computational Molecular Biology (RECOMB)*. Sheridan Printing p. ? To appear.

Winkler, W. C. & Breaker, R. R. (2003) Genetic control by metabolite-binding riboswitches. *Chembiochem.,* **4**, 1024–1032.