Automatic RNA Secondary Structure Determination with Stochastic Context-Free Grammars

Leslie Grate
Department of Computer Engineering
University of California, Santa Cruz, CA 95064, USA
Email: leslie@cse.ucsc.edu

Keywords: RNA secondary structure, multiple alignment, stochastic context-free grammars, 16S and 23S rRNA, large RNA modeling, mutual information, minimum length encoding.

Abstract

We have developed a method for predicting the common secondary structure of large RNA multiple alignments using only the information in the alignment. It uses a series of progressively more sensitive searches of the data in an iterative manner to discover regions of base pairing; the first pass examines the entire multiple alignment. The searching uses two methods to find base pairings. Mutual information is used to measure covariation between pairs of columns in the multiple alignment and a minimum length encoding method is used to detect column pairs with high potential to base pair. Dynamic programming is used to recover the optimal tree made up of the best potential base pairs and to create a stochastic context-free grammar. The information in the tree guides the next iteration of searching. The method is similar to the traditional comparative sequence analysis technique. The method correctly identifies most of the common secondary structure in 16S and 23S rRNA.

Introduction

Multiple alignment of structural RNA is a more difficult problem than multiple alignment of protein. It requires a different type of search that is computationally more expensive than the standard Hidden Markov Model method (Krogh *et al.* 1994; Baldi *et al.* 1994; Krogh & Hughey 1995) used for aligning proteins or DNA because the base pairing that forms the secondary structure of RNA can't be modeled by an HMM.

Humans align RNA using an iterative technique known as Comparative Sequence Analysis, described in detail in (James, Olsen, & Pace 1989). This technique involves iterating 2 phases, a search of the multiple alignment for new base pairing and re-aligning the data in light of the newly found base pairing.

Many different methods of searching for and predicting secondary structure in a given multiple alignment have been developed. Methods that minimize the free energy of the structure of a single RNA molecule (Tinoco Jr., Uhlenbeck, & Levine 1971; Turner, Sugimoto, & Freier 1988; Gouy 1987; Zuker 1989) have not been as successful as methods that use phylogenetic analysis of similar RNA molecules (Fox & Woese 1975; Woese *et al.* 1983; James, Olsen, & Pace 1989). Combinations of the these approaches tends to improve the results (Waterman 1989; Le & Zuker 1991; Han & Kim 1993; Chiu & Kolodziejczak 1991; Sankoff 1985; Winker *et al.* 1990; Lapedes 1992; Klinger & Brutlag 1993; Gutell *et al.* 1992). Our own previous work employed minimum length encoding with a Dirichlet mixture prior and a Gibbs sampling methodology (Grate *et al.* 1994) to predict secondary structure.

The best methods for multiple alignment of RNA make use of stochastic context-free grammars (SCFGs) (Sakakibara *et al.* 1994; Eddy & Durbin 1994). SCFGs are a generalization of HMMs and can model the base pairing interactions of RNA. Their use has been limited to short sequences due to the large amount of computation involved.

The entire Comparative Sequence Analysis procedure has been implemented by Eddy and Durbin (Eddy & Durbin 1994) with good results on short sequences. They use a statistical method to measure covariation between pairs of columns in the multiple alignment and use this information to guide a process of iterative realignment of the sequences. The complexity of their algorithm places an upper limit on its use of some 140 bases (Durbin 1994).

We are developing a similar system that will be more general and able to work on long RNA sequences (such as 16S or 23S ribosomal RNA). We employ mutual information to measure covariation coupled with a probability based minimum length encoding method of helix detection to sense base pairs in conserved regions. To function on long sequences we have developed fast serial and Maspar parallel computer algorithms. The system will produce multiple alignments and SCFGs that can be used for searching databases. This system will form part of a software Ribosomal Workbench.

We briefly describe here how stochastic context-free grammars relate to traditional multiple alignments.

We then discuss the complexity of the algorithms and why searching for base paired regions in RNA is more difficult than aligning proteins. We then describe our methods of detecting possible base pairs and the secondary structure searching methodology that will form the core of the iterative multiple alignment system under development. At present only the search of a fixed multiple alignment and prediction of secondary structure via a stochastic context-free grammar (Sakakibara *et al.* 1994) is functional. Results on 16S and 23S rRNA are presented.

Multiple Alignments and SCFGs

The standard methods of describing RNA secondary structure are a multiple alignment, Figure 1A, and a secondary structure picture, Figure 1B . A third method uses a tree (Shapiro & Zhang 1990; Searls 1993), and this forms the basis of a SCFG representation, Figure 1C. These 3 methods are easily translated into one another with minimal loss of information.

SCFGs

A SCFG is a more abstract type of representation because it really is a probabilistic model of all possible sequences. However, the tree structure and the parameters are chosen such that only sequences that match it well have high probability, non-matching sequences have an extremely small probability. It is this flexibility that gives the power to handle variant RNA structures (see (Sakakibara *et al.* 1994) for complete discussion).

SCFGs are inherently hierarchical. A full, low-level SCFG will specify details down to the individual base or base pair level. A higher level view abstracts the individual bases into the logical structures of helix and loop. In this paper, we adopt this higher level view.

The method of reading a SCFG is to begin at the root, labeled **Start** in Figure 1C, and proceed by following the arrows around the tree reading each item as you encounter it. The majority of RNA helixes can be fit into these tree structures but pseudo-knots (Figure 2) can't because the arms of the helixes intertwine. Thus SCFGs can't model pseudo-knots, and we make the assumption that all the helixes can be properly nested into a tree structure. We term this the proper nesting of helixes assumption.

The three types of objects in our SCFGs are loops, helixes and branches. Loops represent single stranded regions, helixes represent base paired regions, while branches do not represent any bases, rather they direct the structure of the tree.

The main parameter of high level SCFG loops and helixes is the length. Other information can indicate bounds on the length, probability distributions on the length, and how the length is interpreted (if it is an average, or minimum, etc). Additionally, a low-level SCFG loop will contain a set of nucleotide distributions per base location in the loop. A low-level SCFG helix

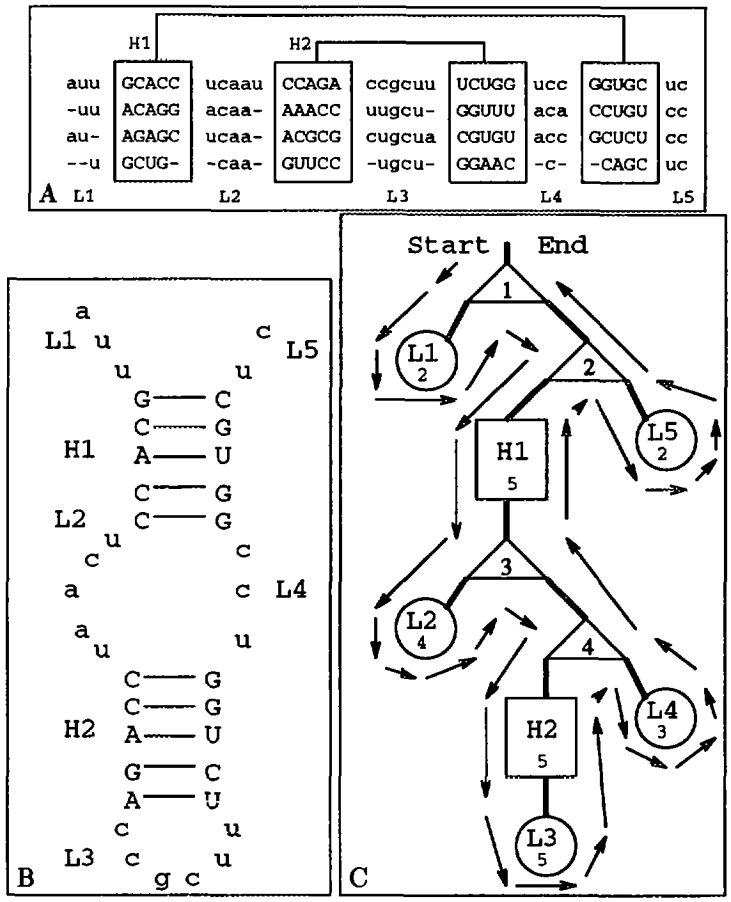

Figure 1: A: Multiple alignment of an RNA double stem loop with the base paired regions indicated. B: Secondary structure picture of the first sequence in the multiple alignment. C: An SCFG describing the same structure. Helixes are rectangles, loops are circles, and triangles are branching points that do not represent bases. Read it by following the arrows from start to end. Loops are read only once, helixes are read twice: 5' when entering, and 3' when leaving. It will read: L1, 5'H1, L2, 5'H2, L3, 3'H2, L4, 3'H1, L5. Branches can be labeled with the probability that a given side of the branch is missing. For example, branch 3 can have a reasonable probability of deleting the right side, removing H2, L3 and L4.

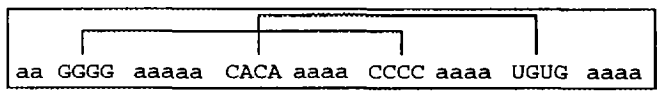

Figure 2: Structure of a pseudo knot. The arms of the helix intertwine each other, and can't fit into the tree structure of an SCFG.

will contain pairwise distributions for each base pair location.

The branch parameters can specify the probability that a given side of the branch is present in the structure. This allows an entire part of the SCFG to be optional.

Conversion between representations A multiple alignment with base paired regions labeled can be easily translated into a SCFG. Translation into a SCFG entails ignoring any pseudo-knots or tertiary interactions and adopting some method of establishing loop and helix lengths. The example in Figure 1C uses average length, which are the small numbers in the loops and helixes.

Helix regions are always aligned, whereas depending on the level of detail, loop regions might not be. An extremely detailed SCFG can specify alignment in loop regions at the expense of a much larger and possibly computationally prohibitive grammar.

Complexity of Models

Hidden Markov Models use very local information from the sequences to determine conserved regions and local best alignments. The base content of RNA helixes is often not conserved at all, as a result HMM's do poorly.

HMM methods take time linear in the number of bases in the sequence. However, searching a sequence of length n for potential base paired regions (which we call the helix finding algorithm) requires an n^2 algorithm. For each location, all locations 3' to it potentially could base pair. This describes a lower triangular matrix, giving about $n^2/2$ pairs to examine.

Discovering an optimal tree structure for a SCFG from a set of h potential base paired regions is an h^3 algorithm using dynamic programming. We call this the tree construction algorithm.

Using a fixed, pre-determined SCFG with m elements[1] to align one sequence of length n to the grammar (called parsing the sequence) is an n^3m algorithm in time and uses n^2m space (memory) (Sakakibara *et al.* 1994; Eddy & Durbin 1994). We call this the parsing algorithm.

You can see that as we progress to more complicated algorithms, the number of items the algorithm operates on must be controlled in order to get results.

The method of Eddy and Durbin iterates the tree construction (to find base pairs) and Parsing (to perform multiple alignment) algorithms at the level of individual bases. Thus, their method has an overall time complexity of n^3m, where n is the number of bases in the sequence. They state this places a useful upper limit of around 140 bases on their system (Durbin 1994).

[1] In a low-level SCFG, the elements would be individual bases. In a high level grammar, they could be whole helix or loop objects

Our ultimate goal is to create a system that is able to create multiple alignments of similar quality but on much larger sequences. The algorithms are inherently complex, so we must look for ways to reduce the number of items they examine.

Reducing n

The length of the E.coli 16S sequence is 1542 bases and for 23S is 2904 bases. The length of the multiple alignments for these two families are 2688 and 7977 bases respectively (Larsen *et al.* 1993). Clearly these sequences are much too large to directly attack by the parsing algorithm at the individual base level. Some methods must be employed to reduce the number of items to parse on into the few hundreds before parsing will be feasible.

Our first idea is to use a "best first" method to find the best helixes first. The second idea is to make use of hierarchy inherent in SCFGs and the assumption of proper nesting of helixes. The third idea is to note that most base pairs occur in long stretches that form helical regions.

Best First The first idea allows use of algorithms faster than tree construction to identify possible helical regions. The helix finding algorithm we developed is faster than tree construction (being only an n^2 algorithm). It employs an adjustable minimum length encoding function to quickly identify potential base paired regions. This provides an adjustable strength filter that can be tuned to reject all but the most promising helix forming candidates.

Divide and Conquer The second idea allows us to use a divide and conquer approach to limit the amount of searching. The assumption of proper nesting of helixes allows us to partition the sequence into smaller independent areas once a particular helix is decided to exist (Figure 3).

Figure 3: When two parts of a sequence are determined to be base paired (shaded boxes), the inside region I becomes independent from the outside regions L and R. Searching for base pair can then focus on region I, and the concatenation of L and R.

High Level Grouping The third idea allows us to get away from having our algorithms always examine individual base pairs by grouping them into higher level objects such as helixes. In many cases we can treat a helix as just two items (the 5' and 3' sides) rather than as a set of base pairs. With this model, any number of individual base pairs in a base paired region could be replaced with only 2 "points", one for

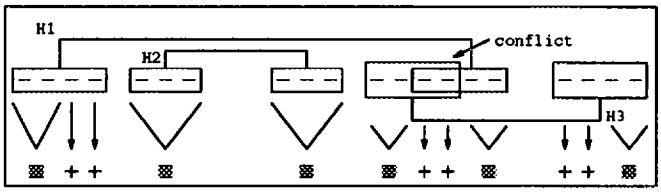

Figure 4: How individual bases map to "points". The bases making up helix H2 are only used in H2, so are uncontested and both 5' and 3' sides are mapped to single points, shaded boxes. Potential helixes H1 and H3 are both trying to use some of the same bases. The bases that are contested are each mapped to individual points, plus signs. In this small example, the 20 base locations can be represented as only 12 points. Greater savings occur in practice.

the 5′ side, and one for the 3′ side.

If an individual base has the possibility of being in more than one helix (it is "contested" by 2 or more helixes) it can't be uniquely assigned to a single helix. In this case, the parts of the helixes that conflict are not reduced using the above model, and the bases that are contested must be mapped into individual points (Figure 4). Thus an uncontested helix of length 6 (12 bases) could be represented as only 2 points, a six-fold reduction in the number of items an algorithm must examine.

Making use of these three ideas is an effective way of reducing the number of items that a parsing or tree construction algorithm must examine.

Detecting Possible Base Pairs

How do you tell if two given regions of an RNA sequence are base paired? If there is only one sequence, the most information is in the number of Watson-Crick (WC) base pairs (or the fairly frequent wobble pairs GU, UG) that *could* form between the given regions. If there are two or more similar sequences there is a wealth of information available. Random mutations will make the sequences slightly different, but if a region must be base paired for the RNA to function, then it is extremely unlikely that one base of a WC pair will change while the other will not also change in a complementary manner. These compensatory mutations (covariations) are strong evidence that the two bases interact. This forms the basis of the comparative sequence analysis technique (see (James, Olsen, & Pace 1989) for complete discussion).

However, the character of helixes spans the entire range from complete conservation (no mutations discovered so far in nature) to "it doesn't matter as long as it is WC".

Because of this, it is advantageous to use at least two methods of scoring a region. One method should identify covariation, while the other should examine the base content.

Mutual Information

The mutual information formulation as a measure of covariation has been used by many researchers (Eddy & Durbin 1994; Gutell *et al.* 1992; Korber *et al.* 1993; Gulko 1995) with good success. We use the it to measure covariation between pairs of columns.

Given a set of N aligned sequences, and two columns, i and j, in the alignment, the counts of the number of occurrences of the pairs and the individual bases on the 5' and 3' sides are tabulated. The ***mutual information*** of one pair of bases x (from column i) and y (from column j) (where x, y $\in \{A, C, G, U\}$) is given by:

$$\mathrm{MI[xy]} = \frac{\mathrm{count[xy]}}{N} * \log_2 \frac{\mathrm{count[xy]} * N}{\mathrm{count5'[x]} * \mathrm{count3'[y]}} \quad (1)$$

This is summed over all pairs with non-zero count to give the total mutual information score:

$$\mathrm{MIcolPair(i,j)} = \sum_{\text{all pairs xy} \neq 0} \mathrm{MI[xy]} \quad (2)$$

This function is zero if there is no covariation and otherwise is positive. The mutual information score for a set of column pairs is the sum of the scores for the individual column pairs.

Base Content Scoring

Our formulation of the base content score is based on the comparison of two probabilistic models: one for the occurrence of base pairs, and the other for the occurrence of individual bases, and is the same as used in our earlier work (Grate *et al.* 1994). For a pair of columns, i and j, with base x from column i, and y from column j:

$$\mathrm{LR[xy]} = \frac{\mathrm{PrPair[xy]}}{\mathrm{PrSingle[x]} * \mathrm{PrSingle[y]}} \quad (3)$$

is the likelihood odds ratio of having x and y be a base pair, where PrPair and PrSingle are the parameters of the models (see Figures 5 and 6). A value greater than 1 indicates that pairing is more likely than not.

When $\log_2$ is taken the formula becomes:

$$\mathrm{BCscore[xy]} = \log_2 \mathrm{PrPair[xy]} - \log_2 \mathrm{PrSingle[x]} - \log_2 \mathrm{PrSingle[y]} \quad (4)$$

and is our base content score. When viewed in this framework the base content score has a direct minimum length encoding interpretation as the number of bits saved by encoding the two bases as a pair rather than encoding them each as individual bases. A positive value indicates that the pairing is a better model, and the larger the value the better.

pair		3' A	C	G	U
5'	A	.0128	.0108	.0154	.1275
	C	.0142	.0108	.2728	.0131
	G	.0176	.1378	.0198	.0427
	U	.2097	.0122	.0628	.0200
Individual background		.3059	.2471	.2118	.2352

Figure 5: Probabilities for pairs and individual bases. From (Sakakibara *et al.* 1994).

		3' A	C	G	U	N	-
	A	-2.866	-2.597	-1.973	1.228	-0.5	-0.5
	C	-2.597	-2.497	1.972	-2.203	-0.5	-0.5
5'	G	-1.973	1.972	-1.118	0.083	-0.5	-0.5
	U	1.228	-2.203	0.083	-1.471	-0.5	-0.5
	N	-0.5	-0.5	-0.5	-0.5	-0.5	-0.5
	-	-0.5	-0.5	-0.5	-0.5	-0.5	0.0

Figure 6: Precomputed 6 by 6 table of scores made from the data in Figure 5.

With this formulation, the base content score for the entire column pair is the sum of the base content scores for the individual pairs in the columns:

$$\text{BCcolPair(i,j)} = \sum_{\text{all pairs xy} \neq 0} \text{count[xy]} * \text{BCscore[xy]} \quad (5)$$

and the base content score for a series of column pairs is the sum of the individual column pair base content scores.

Score Data Our data for the probabilities, PrPair and PrSingle, is based on a previous study of 16S rRNA (Sakakibara *et al.* 1994) and our own studies, and is shown in Figure 5. The derived scores used are given in Figure 6. Here we also add in the letter N for any base that is not exactly known, and -, used for spacing in the multiple alignment.

We derive this 6 by 6 table of scoring values as follows. We want a general set of scores, not just those reflecting the biases in the original 16S data. Accordingly, a symmetric probability matrix is computed by averaging the sum of each pairs 5'3' and 3'5' values. Then equation 4 is applied using the symmetric probabilities and the individual base probabilities to compute the score entries for the normal bases. The non-base entries are arbitrarily chosen to have negative scores, but not so large that would totally remove a region from consideration. It is important for the -,- entry to be zero, as this allows alignment insertions inside of helixes without destroying the base content score. For runtime speed these real numbers are scaled up and rounded to integers so that all computations use integer arithmetic.

Filtering using the Helix Finding Algorithm

The helix finding algorithm examines all possible placements of a given length helix in a range of columns. Given a range of n columns, a minimum helix length L and a minimum spacing between the 5' and 3' sides S, there are

$$\frac{(n-2L-S)^2}{2}$$

possible configurations of the 5' and 3' sides that could form helixes. The value of S is set to 4, the minimum practical length of an RNA hairpin loop.

A large value of L (such as 5 or 6) results in few potential helixes being found, whereas a small value of L (such as 2 or 3) will find many. This is the main filter parameter used in searching. When searching a large number of columns, L is set to a large value (actually, 4 is plenty large). We use the term 4-filter to indicate L has a value of 4, likewise for 3-filter.

For each possible helix position the base pair content score is computed for the L columns involved. If the base pair content score for the columns is > 0 then the covariation score is computed. If the covariation score is larger than a user adjustable threshold (the default is a value that corresponds to one bit per base pair), the helix is accepted as a potential helix candidate.

Thus, the only potential helix candidates are those that have a high probability of being base paired, and have enough covariation to indicate that they likely to be base paired.

Bulges

Bulges (unpaired bases with in a helix) have to be taken into account if one hopes to find many helixes. In theory, a helix with a bulge can be viewed as two smaller helixes with a short loop in between. In practice, this sometimes works reasonably well. However there are some shortcomings with this method when applied to normal human readable multiple alignments. The problem is that the columns of a multiple alignment classify the bases as either base paired or not. It is then difficult to indicate that in some sequences a given column is bulged, whereas in others the same column is base paired. The usual solution is to have only one column. Humans use the base pairings for the family to note these columns, but a program whose job it is to *discover* the base pairing information can see these columns as non-base paired.

Searching for bulges is quite easy, but it significantly increases the computation time. For a helix of N base pairs and B bulged bases, there are $\binom{2(N-1)}{B}$ configurations of the bulged bases. For example, given

a helix of length 6 there are 10 potential locations for a 1 long bulge and 45 locations for 2 bulged bases[2]. These 55 extra checks would have to be applied at each potential location for a 6 long helix in order to determine if a 6 long helix with one or two base bulges existed at that location.

The "best first" hypothesis says we are looking for the best and most obvious helixes first, and these probably do not have obscuring bulges. Accordingly, bulge searching would only be used in later stages of searching or refining in smaller areas, where the extra expense is justified. We have not yet fully integrated searching for bulges into the current system.

The Tree Construction Algorithm

Once a set of potential helixes are determined, the tree construction algorithm will find the optimal score tree that best fits them. It is assumed that the real secondary structure is the one that fits the best helical regions into a properly nested tree structure.

The method outlined previously is used to establish the number and type of "points" the algorithm examines. This allows the algorithm to choose between helixes that are competing for the same bases. The respective scores for the columns are mapped along into the system of "points". The cost function used by the algorithm employs both the base pair content and covariation scores in a simple weighted linear combination. The score for a particular pair of points is

$$\text{pointPairScore(i,j)} = W_{cov} * \text{MIofPoints(i,j)} + W_{bp} * \text{BPscore(i,j)}. \quad (6)$$

The weights are user adjustable parameters that allow choosing which type of information you want to believe more. For these experiments, both types of information had equal weights. However, the structures were stable even as the relative weighting was varied by a factor of 10.

The above raw score can be further adjusted based on the context of the particular points. Specifically, we can give a benefit for the elongation of a helix over that of adding an isolated base pair. We call this the *stacking benefit*, and is also user adjustable. The cost function per point pair the tree construction uses is then:

$$\text{score(i,j)} = \text{pointsPairScore(i,j)} + W_{stack} * \text{F(i,j)}. \quad (7)$$

The function of the context of the pair of points, F(i,j), is arbitrary, but for these experiments is simply 1 if the context is such that the point pair is elongating an existing series of adjacent points (thus building up a helix) and 0 if the context is an isolated base pair.

The optimal tree structure is calculated using standard dynamic programming applied to the above score function. Dynamic programming efficiently examines all possible legal tree structures of the potential helixes. Conceptually, the method progresses from the 5' end to the 3' end computing and saving the score of each possible sub-tree in a lower triangular matrix using the following method.

Each matrix entry, $V_{i,j}$, is the score of the optimal tree structure for the points from i to j. The matrix columns are indexed with j (the more 3' side), and the rows by i (the more 5' side). The matrix diagonal, $i = j$ is initialized to zero. Then j is stepped from 2 to the total number of points n and each location $V_{i,j}$ with $i < j$ is filled in according to:

$$V_{i,j} = \max[V_{i+1,j}, V_{i,j-1}, V_{i+1,j-1} + \text{score(i,j)}, \max_{i<y<j}[V_{y+1,j} + V_{i,y}]\ ] \quad (8)$$

When the algorithm finishes at the 3' end, the score of the best tree will have been found and stored in $V_{1,n}$.

Tracing back through this matrix of scores recovers the optimal tree structure. This tree is then output as an SCFG.

Finding the Secondary Structure

The following procedure to find helixes. The helix finding algorithm, with a particular value of the length parameter, is run on the current search region(s), creating a set of potential helix candidates. Then the tree construction algorithm is used on the set of potential helix candidates to determine the helixes in the best tree structure. These helixes define smaller areas for further search using the divide and conquer methodology and we iterate this process with smaller values of the filter parameter until the areas are too small, or no new helixes are found.

The experiments on 16S and 23S have indicated that only 3 passes are necessary for even these large alignments.

Parallel Computation

All these algorithms can be parallelized. We have implemented many of them on our Maspar MP-2204 (Nickolls 1990). It is a SIMD array processor with 4096 individual processing elements. Based on the serial verses parallel implementation of the HMM system, the Maspar can provide a speed up by a factor of 40 over a Sun Sparc 2 (Hughey 1993).

A parallel implementation and timing analysis of the helix finding algorithm shows that the entire 16S size first pass search takes less than 2 seconds on a limited number of sequences with no saving of search results. Analysis indicates that generalizing the algorithm to match the level of the serial implementation will result in an execution time of about 35 seconds.

[2]These are $\binom{10}{1}$ and $\binom{10}{2}$.

Experiments

All results were achieved with the serial implementation of the algorithms written in standard C++, running on an HP/Apollo 68040 based Unix workstation (rated about half a Sun Sparc 2).

16S Results

The data set is the 60 representative sequences in the 2866 base pair long 16S multiple alignment obtained from the Ribosomal Database Project (Larsen *et al.* 1993). The number of potential base paired regions is about 3.59 million. The columns of this multiple alignment were examined with the helix finding algorithm employing the 4-filter resulting in 1661 (possibly overlapping) regions. These 1661 regions were examined for covariation resulting in 35 regions. These 35 regions are processed into 200 points.

The resulting tree contains 20 helixes, all of which are known in the E.coli secondary structure, are indicated in the XRNA (Weiser, Gutell, & Noller 1994) drawing, Figure 7. Even at this early stage, the SCFG output contains 20 helixes, 41 loops, and 40 branches, and is too large to include in this paper. The total execution time is 10 minutes of real time.

Using these helixes to bound further searching, smaller areas are chosen for a second pass using the 3-filter and the 2-filter. The result is that 55 out of the 77 helixes shown in E.coli are found. Lowering the threshold for detection of covariation allows finding 5 more helixes. Searching some of the areas using a prototype bulge searching filter identifies more helixes, including the central pseudo-knot around location 17-918. The pseudo-knot cannot be included in the resulting tree by the tree construction algorithm. However, the searching methodology does find that it is a highly likely helix, and could be included in a model that allows pseudo-knots.

23S Results

The data are the 30 bacterial sequences extracted from the version 3 of the 23S alignment obtained from the Ribosomal Database Project (Larsen *et al.* 1993). The same steps as used in 16S were applied to this very large, 7977 column alignment.

The first pass search using the 4-filter results in 8205 potentially base paired regions, of which only 72 pass the covariation analysis. These 72 (possibly overlapping) regions are processed into 116 points, and the resulting optimal tree contains 44 helixes, all of which are known (data not shown). The execution time is 17 minutes of real time.

Second and third passes find a few more helixes but not too many. Using the prototype bulge filter identifies more helixes. The alignment of 23S is not as clean as that for 16S. The fact that the method is able to pick out many of the helix regions in the presence of inexact alignments indicates its robustness.

Further Work

We are continuing to work toward the goal of the full multiple alignment system. The methods described here seem to be sensitive enough and fast enough that an iterative process is feasible. The information discovered must be used to adjust the multiple alignment, thus closing the loop. Adjusting large multiple alignments using a HMM on a serial computer is very expensive, and using a parallel computer is probably necessary. There are 23S sequences that have not been aligned that we wish to add to the multiple alignment.

The system now only chooses the most optimal solution. It is highly desirable to be able to report suboptimal solutions and how suboptimal they are. We are working on incorporating the methods of (Zuker 1989; Stormo & Haussler 1994) to provide this function.

Pseudo-knots must be supported. The search function can identify pseudo-knots, but the SCFG formalism can't handle them. We are working on extensions to SCFGs that will allow limited but biologically useful pseudo-knot structures without damaging the ability to parse in n^3 time.

We are developing more sensitive tests on column pairs to determine if they are base paired. This would improve detection of lone base pairs and provide better sets of potential helixes for further analysis. We are experimenting with a neural net approach (Reklaw, Hanna, & Haussler 1995) to combine the mutual information and base pair composition of column pairs to discover and rank potential helixes. The 37 inputs are the counts of each of the 36 possible pairs (in our 6 letter alphabet) and the mutual information covariation.

The results on 16S and 23S rRNA indicate that this neural net method is more sensitive than either raw mutual information covariation or base content scoring alone to potential base pairs. The first pass of searching identifies a few more helixes in 16S and many more in 23S. However some of these helixes are incorrect: they are false positives. We intend to overcome the problem of false positives by performing multiple runs on random data sets and comparing the resulting tree structures. The stable core structure would then be chosen as the best representative structure. Another possibility is to prune back certain low scoring helixes at the edge of the tree.

Finally, we note that the method can be used to detect normal base pairing interactions between different RNA molecules.

Acknowledgments

The author thanks Michael Brown, Saira Mian, Bryn Weiser, Jessie Reklaw and the entire UCSC Computational Biology group and our collaborators in the Biology department. Our papers are online on our web site, http://www.cse.ucsc.edu/research/compbio, and by anonymous FTP, ftp.cse.ucsc.edu.

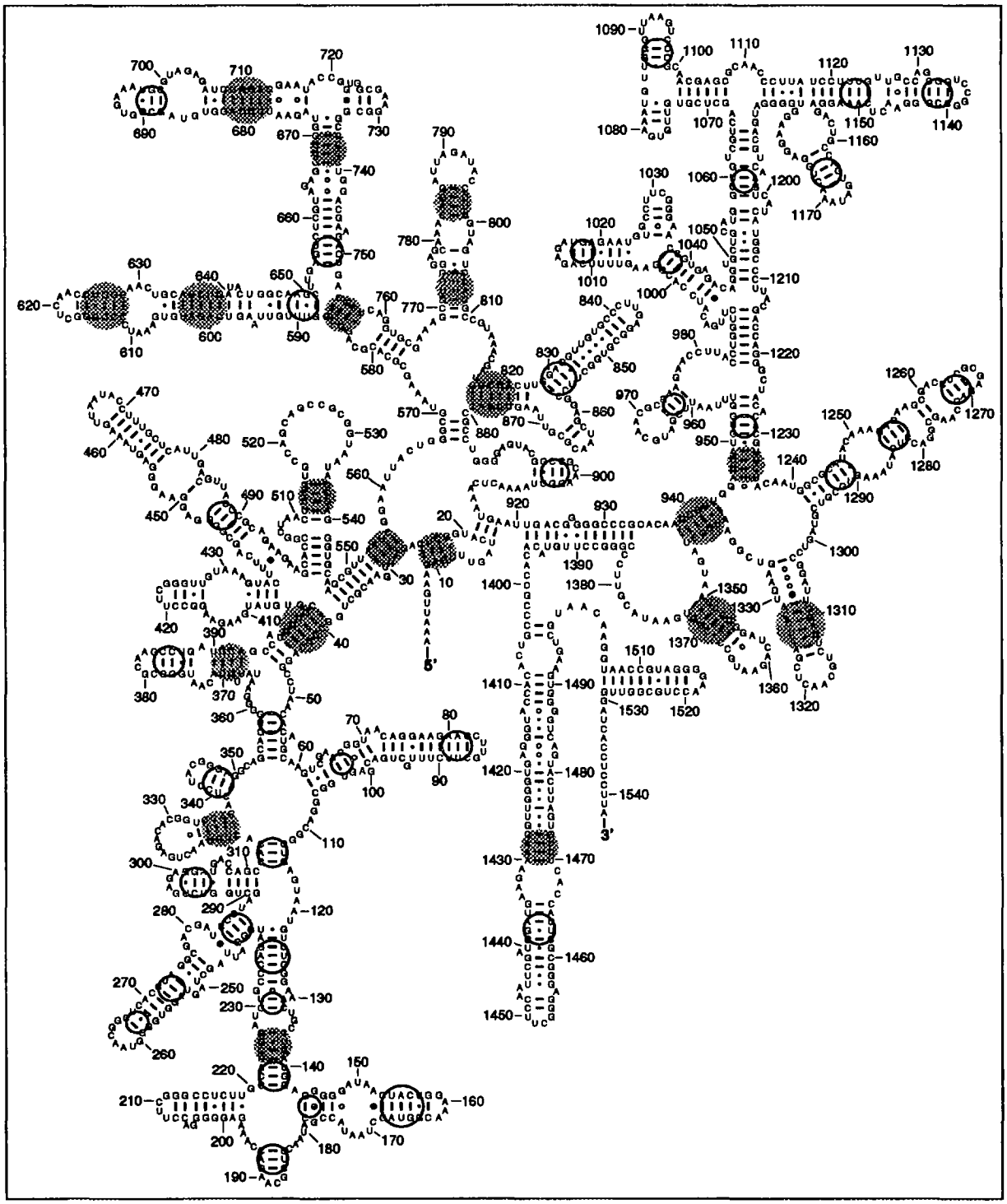

Figure 8: Standard E.coli secondary structure (courtesy of H. Noller) with helixes found in the first pass shaded, second pass larger open circles, third pass smaller open circles. The helixes that are not found fall into 3 categories: A) The bases are too conserved, so that there is no covariation signal. B) Only a few of the 60 sequences have that particular structure, so there is not enough information to decide it is a common feature of all the sequences. C) The alignment contains many bulges or large gaps or inserts in that area.

References

Baldi, P.; Chauvin, Y.; Hunkapillar, T.; and McClure, M. 1994. Hidden Markov models of biological primary sequence information. *PNAS* 91:1059–1063.

Chiu, D. K., and Kolodziejczak, T. 1991. Inferring consensus structure from nucleic acid sequences. *CABIOS* 7:347–352.

Durbin, R. 1994. Personal communication.

Eddy, S. R., and Durbin, R. 1994. RNA sequence analysis using covariance models. *NAR* 22:2079–2088.

Fox, G. E., and Woese, C. R. 1975. 5S RNA secondary structure. *Nature* 256:505–507.

Gouy, M. 1987. Secondary structure prediction of RNA. In Bishop, M. J., and Rawlings, C. R., eds., *Nucleic acid and protein sequence analysis, a practical approach.* Oxford, England: IRL Press. 259–284.

Grate, L.; Herbster, M.; Hughey, R.; Mian, I.; Noller, H.; and Haussler, D. 1994. RNA modeling using Gibbs sampling and stochastic context free grammars. In *ISMB-94.* Menlo Park, CA: AAAI/MIT Press.

Gulko, B. 1995. To appear in Masters Thesis, UCSC.

Gutell, R. R.; Power, A.; Hertz, G. Z.; Putz, E. J.; and Stormo, G. D. 1992. Identifying constraints on the higher-order structure of RNA: continued development and application of comparative sequence analysis methods. *NAR* 20:5785–5795.

Han, K., and Kim, H.-J. 1993. Prediction of common folding structures of homologous RNAs. *NAR* 21:1251–1257.

Hughey, R. 1993. Massively parallel biosequence analysis. Technical Report UCSC-CRL-93-14, University of California, Santa Cruz, CA.

James, B. D.; Olsen, G. J.; and Pace, N. R. 1989. Phylogenetic comparative analysis of RNA secondary structure. *Methods in Enzymology* 180:227–239.

Klinger, T., and Brutlag, D. 1993. Detection of correlations in tRNA sequences with structural implications. In Hunter, L.; Searls, D.; and Shavlik, J., eds., *ISMB-93.* Menlo Park: AAAI Press.

Korber, T. M. B.; Farber, R. M.; Wolpert, D. H.; and Lapedes, A. S. 1993. Covariation of mutations in the v3 loop of human immunodeficiency virus type 1 envelope protein: An information theoretic analysis. *PNAS* 90:7176–7180.

Krogh, A., and Hughey, R. 1995. SAM: Sequence alignment and modeling system software. Technical Report UCSC-CRL-95-7, University of California, Santa Cruz, Computer and Information Sciences Dept., Santa Cruz, CA 95064.

Krogh, A.; Brown, M.; Mian, I. S.; Sjölander, K.; and Haussler, D. 1994. Hidden Markov models in computational biology: Applications to protein modeling. *JMB* 235:1501–1531.

Lapedes, A. 1992. Private communication.

Larsen, N.; Olsen, G. J.; Maidak, B. L.; McCaughey, M. J.; Overbeek, R.; Macke, T. J.; Marsh, T. L.; and Woese, C. R. 1993. The ribosomal database project. *NAR* 21:3021–3023.

Le, S. Y., and Zuker, M. 1991. Predicting common foldings of homologous RNAs. *J. Biomolecular Structure and Dynamics* 8:1027–1044.

Nickolls, J. R. 1990. The design of the Maspar MP-1: A cost effective massively parallel computer. In *Proc. COMPCON Spring 1990*, 25–28. IEEE Computer Society.

Reklaw, J.; Hanna, R.; and Haussler, D. 1995. Prediction of ribosomal RNA base-pairing by neural networks. Manuscript in progress.

Sakakibara, Y.; Brown, M.; Hughey, R.; Mian, I. S.; Sjölander, K.; Underwood, R. C.; and Haussler, D. 1994. Stochastic context-free grammars for tRNA modeling. *NAR* 22:5112–5120.

Sankoff, D. 1985. Simultaneous solution of the RNA folding, alignment and protosequence problems. *SIAM J. Appl. Math.* 45:810–825.

Searls, D. B. 1993. The computational linguistics of biological sequences. In Hunter, L., ed., *Artificial Intelligence and Molecular Biology.* AAAI Press. chapter 2, 47–120.

Shapiro, B. A., and Zhang, K. 1990. Comparing multiple RNA secondary structures using tree comparisons. *CABIOS* 6(4):309–318.

Stormo, G. D., and Haussler, D. 1994. Optimally parsing a sequence into different classes based on multiple types of evidence. In *ISMB-94*, 369–375.

Tinoco Jr., I.; Uhlenbeck, O. C.; and Levine, M. D. 1971. Estimation of secondary structure in ribonucleic acids. *Nature* 230:363–367.

Turner, D. H.; Sugimoto, N.; and Freier, S. M. 1988. RNA structure prediction. *Annual Review of Biophysics and Biophysical Chemistry* 17:167–192.

Waterman, M. S. 1989. Consensus methods for folding single-stranded nucleic acids. In Waterman, M. S., ed., *Mathematical Methods for DNA Sequences.* CRC Press. chapter 8.

Weiser, B.; Gutell, R.; and Noller, H. F. 1994. XRNA: An X windows environment RNA editing/display package. In preparation.

Winker, S.; Overbeek, R.; Woese, C. R.; Olsen, G. J.; and Pfluger, N. 1990. Structure detection through automated convariance search. *CABIOS* 6(4):365–371.

Woese, C. R.; Gutell, R. R.; Gupta, R.; and Noller, H. F. 1983. Detailed analysis of the higher-order structure of 16S-like ribosomal ribonucleic acids. *Microbiology Reviews* 47(4):621–669.

Zuker, M. 1989. On finding all suboptimal foldings of an RNA molecule. *Science* 244:48–52.