Supplementary Material

Sm-I  Computing Inside and Outside Probabilities

In order to determine all inside and outside variables for a given sequence \( r \in L_r \), we decided to use the SCFG \( G_r \) as the basis for a special version of Earley’s algorithm. In particular, we chose to rely on the formalism presented in [Goo98, Goo99] for describing parsers, which is called semiring parsing.

The advantage of using an Earley-style parser description is that the corresponding semiring parser can handle general grammars, which means we do not have to transform the grammar \( G_r \) into Chomsky normal form (CNF). This is especially useful, since the number of productions of the CNF of grammar \( G_r \) would be huge. For this reason, computing the needed inside and outside values by the usual inside outside algorithm for grammars in CNF would be less efficient.

Sm-I.1 Notations

In the sequel, we number the nucleotides in a given RNA sequence \( r \) of length \( n \) in the usual 5’ → 3’ direction (i.e., in the usual reading order from left to right), such that the corresponding RNA sequence can be written as \( r_1 \ldots r_n \). Equivalently, for a secondary structure \( s \) (in dot-bracket representation) of size \( n \), we can write \( s_1 \ldots s_n \).

Moreover, for \( A \) an intermediate (or non-terminal) symbol of the considered grammar \( G_r \), let \( \alpha_A(i, j) \) denote the inside variables (computed by the usual inside algorithm) and \( \beta_A(i, j) \) denote the outside variables (computed by the usual outside algorithm) for a given word \( r \in L_r \) of size \( n \), \( 1 \leq i, j \leq n \). Consequently, \( \alpha_A(i, j) \) is the probability of a leftmost derivation that generates the subword \( r_i \ldots r_j \) (of a word \( r \in L_r = L(G_r) \)) from the intermediate symbol \( A \) and \( \beta_A(i, j) \) is the probability of a derivation which, starting with the intermediate symbol \( S \) (the axiom of grammar \( G_r \)), generates the sentential form \( r_1 \ldots r_{i-1} A r_{j+1} \ldots r_n \).

Furthermore, we need to define a new set of productions that has to be used by our semiring parser in order to compute the desired inside and outside probabilities. This production set contains the so-called dotted rules that are considered by Earley’s algorithm. It can easily be obtained by modifying the rule set \( R_{G_r} \) of the grammar \( G_r \) in the following way: Introduce a new symbol \( \bullet \notin \Sigma_{G_r} \cup I_{G_r} \) that is used to mark the current position up to which the parsing has proceeded; according to the fact that Earley’s algorithm parses input words from left to right, this symbols must thus be “shifted” from the leftmost position to the rightmost one in each production rule of the grammar used for parsing. For this reason, we replace each production \( \text{rule} \in R_{G_r} \) of the form \( \text{rule} = A \rightarrow \alpha_1 \ldots \alpha_k \) with \( \alpha_i \in I_{G_r} \cup \Sigma_{G_r}, 1 \leq i \leq k \), by \( k+1 \) new productions \( \text{rule}_0 = A \rightarrow \bullet \alpha_1 \ldots \alpha_k, \text{rule}_1 = A \rightarrow \alpha_1 \bullet \ldots \alpha_k, \ldots, \text{rule}_{k-1} = A \rightarrow \alpha_1 \ldots \bullet \alpha_k \) and \( \text{rule}_k = A \rightarrow \alpha_1 \ldots \alpha_k \bullet \), if \( \text{rule} = A \rightarrow \epsilon \), it is replaced by the new production \( \text{rule}_0 = A \rightarrow \epsilon \bullet \).

The resulting dotted production set will be denoted by \( R_{G_r, \bullet} \) in the sequel. Moreover, each set of \( k+1 \) productions that were derived from an original production \( \text{rule} = A \rightarrow \alpha_1 \ldots \alpha_k \in R_{G_r} \) will be denoted \[\text{Note that for the computation of this probability, one always summarizes over all corresponding derivation trees.}\]
by $\mathcal{R}_{G_r}(\text{rule})$, such that $\bigcup_{\text{rule} \in \mathcal{R}_{G_r}} \mathcal{R}_{G_r}(\text{rule}) = \mathcal{R}_{G_r}$. Obviously, $\mathcal{R}_{G_r}$ contains exactly the rules that have to be considered by Earley’s algorithm for grammar $G_r$.

Last but not least, note that for defining the desired Earley-based semiring parser, we use an item-based parser description. Therefore, in contrast to the usual inside outside algorithm for the computing the inside values $\alpha_{A(i,j)}$ and outside values $\beta_{A(i,j)}$, $1 \leq i, j \leq n$, for $A$ an intermediate symbol of the considered grammar and $n$ the length of the input word, the corresponding semiring parser used in this work computes inside and outside values for so-called items. Here, items are defined by three components, having the form $[i, \text{ind}(\text{rule}), j]$, where for a given input word $r \in \mathcal{L}_{G_r}$ of length $n$, $i$ and $j$, $1 \leq i, j \leq n + 1$, define positions in $r$ (i.e., in front of the first character, in between two characters or after the last character). Additionally, $\text{ind}(\text{rule})$ denotes the index of production $\text{rule} \in \mathcal{R}_{G_r}$ in an appropriate ordering (details will follow later) of production set $\mathcal{R}_{G_r}$. In fact, an item of the form $[i, \text{ind}(A \rightarrow \alpha \bullet \beta), j]$ asserts that $A \Rightarrow \alpha \beta \Rightarrow r_{i} \ldots r_{j-1} \beta$. Consequently, by semiring parsing, the inside and outside values are computed for each production $\text{rule} \in \mathcal{R}_{G_r}$ and not as needed for each non-terminal symbol $A \in \mathcal{I}_{G_r}$. However, the needed inside and outside values $\alpha_{A(i,j)}$ and $\beta_{A(i,j)}$ can easily be derived from the corresponding inside and outside results for items $[i, \text{ind}(A \rightarrow \gamma), j]$, as we will see later.

Sm-I.2 Deriving the Inside and Outside Values of Items

First, we want to describe how to compute the inside and outside values of all items by semiring parsing, using a corresponding item-based description of an Earley-style parser.

Sm-I.2.1 Inside Computation

To obtain the inside values of all items $[i, \text{ind}(\text{rule}), j]$, $1 \leq i, j \leq n + 1$ (for an RNA sequence $r$ of size $n$) and $\text{rule} \in \mathcal{R}_{G_r}$, by semiring parsing based on Earley’s algorithm, we can use the following formulae, which we derived according to [Goo98, Goo99]:

- Scanning:
\[
\text{IN}[i, \text{ind}(A \rightarrow \alpha w_j \bullet \beta), j + 1] = \delta_{w_j, r_j} \cdot \text{IN}[i, \text{ind}(A \rightarrow \alpha \bullet w_j \beta), j]
\]

where for $w_j$ an arbitrary terminal symbol of the underlying grammar $G_r$ and $r_j$ the (terminal) symbol read at position $j$ of the input string $r$,

\[
\delta_{w_j, r_j} = \begin{cases} 
1, & \text{if } w_j = r_j, \\
0, & \text{if } w_j \neq r_j,
\end{cases}
\]

according to the definition of Kronecker’s delta.

- Prediction:
\[
\text{IN}[j, \text{ind}(B \rightarrow \bullet \gamma), j] = \begin{cases} 
\Pr(B \rightarrow \gamma), & \text{if } S \Rightarrow r_1 \ldots r_{j-1} B \delta \text{ for some } \delta, \\
0, & \text{else},
\end{cases}
\]
where \( \Pr(\text{rule}) \) denotes the probability of production \( \text{rule} \in \mathcal{R}_{G_r} \) as given by the SCFG \( G_r \).

Note that this top down filtering is usually made by Earley’s algorithm to ensure that only such items can be predicted that might later be used by the completion rule. However, this is not needed here, since for any superfluously predicted item, the resulting probability will later be set to 0 by a scan. Thus, we can simply predict all items by

\[
\text{IN}[j, \text{ind}(B \to \gamma), j] = \Pr(B \to \gamma).
\]

- Completion:

\[
\text{IN}[i, \text{ind}(A \to \alpha B \bullet \beta), j] = \sum_{i \leq k \leq j} \text{IN}[i, \text{ind}(A \to \alpha \bullet B \beta), k] \cdot \sum_{\text{rule}_B \in \mathcal{R}_B} \text{IN}[k, \text{ind}(\text{rule}_B), j],
\]

where \( \mathcal{R}_B = \{ \text{rule} \in \mathcal{R}_{G_r} \bullet | \text{rule} = B \to \gamma \bullet \} \).

Moreover, the desired semiring parser algorithm for the correct computation of all inside values additionally requires the definition of a convenient ordering of the considered items \([i, \text{ind}(\text{rule}), j], 1 \leq i, j \leq n+1\) for \( n \) the length of the input word and \( \text{rule} \in \mathcal{R}_{G_r} \bullet \), such that no item precedes any other item on which it depends. Details on how we derived the corresponding ordering used in this work will follow. In principle, we can define an ordering by first and last parameters \( i, j \in \{1, \ldots, n+1\} \) that matches the order of consideration of items induced by Earley’s algorithm and especially an appropriate ordering of the considered rule set \( \mathcal{R}_{G_r} \bullet \) by indices \((p, q)\), for \( p \in \{1, \ldots, \text{card}(\mathcal{R}_{G_r})\} \) and \( q \in \{0, \ldots, k(p)\} \), where \( k(p) \) denotes the conclusion length of the production \( \text{rule} \in \mathcal{R}_{G_r} \) indexed by \( p \).

Based on the previously introduced formulae and the appropriate ordering that will be formally defined hereafter, we finally obtain Algorithm 1 that shows how to perform the complete inside computation.

### Sm-I.2.2 Ordering of Items

According to [Goo98, Goo99], we initially need to define an ordering on the items \([i, \text{ind}(\text{rule}), j], 1 \leq i, j \leq n+1\) for \( n \) the length of the input word and \( \text{rule} \in \mathcal{R}_{G_r} \bullet \). In fact, we have to take care that when successively computing the values of all items, no item precedes any item on which it depends. For this reason, in [Goo98, Goo99], each item \( x \) is associated with a “bucket” \( B \); they write \( \text{bucket}(x) = B \). The buckets have to be ordered as follows: If item \( y \) depends on item \( x \), then \( \text{bucket}(x) \leq \text{bucket}(y) \). There are two types of buckets: looping buckets and non-looping buckets. In fact, if items \( x \) and \( y \) depend (directly or indirectly) on each other, then they are both associated with a special looping bucket \( B \), such that \( \text{bucket}(x) = B = \text{bucket}(y) \). A bucket is also called looping bucket if an item in it depends on itself. Otherwise, the bucket is called non-looping. If item \( x \) is associated with a non-looping bucket, then its value can easily be computed, as this value depends only on the values of items in earlier buckets. However, in the case of item \( x \) being associated with a looping bucket, the computation is much more complex, which is due to the fact that the value of \( x \) then depends potentially on the values of other items in the same bucket. In fact, this means that infinite loops may occur, for two different reasons: First, if the values of two items in the same bucket are mutually dependent, or second if an item depends on its own value. Although such infinite loops may require computation of infinite sums, there exists a way to efficiently compute or approximate them, as shown in [Goo98, Goo99].
Algorithm 1 Computation of Inside Values

Require: RNA sequence \( r \in L_r \) of length \( n \geq 1 \),

set \( R_{G_r} \) of production rules used by Earley’s algorithm for parsing \( r \) with \( G_r \), and
probabilities \( \Pr(rule) \) of the productions \( rule \in R_{G_r} \), trained on RNA structure data.

for \( j = 1 \ldots n + 1 \) do
  for \( i = j \ldots 1 \) do
    for \( p = 1 \ldots \text{card}(R_{G_r}) \) do
      for \( q = 0 \ldots k(p) \) do
        \( \text{rule} = \text{ind}^{-1}(p, q) \) /* \( \text{rule} \in R_{G_r} \) is the rule having index \( (p, q) \) in our ordering. */
        if \( \text{rule} = A \rightarrow \omega w_{j-1} \bullet \beta \) then
          /* Scanning: */
          \( \text{IN}[i, (p, q), j] = \delta w_{j-1, r_{j-1}} \cdot \text{IN}[i, (p, q - 1), j - 1] \)
        else if \( \text{rule} = B \rightarrow \bullet \gamma \) then
          /* Prediction: */
          \( \text{IN}[j, (p, q), j] = \Pr(B \rightarrow \gamma) \)
        else if \( \text{rule} = A \rightarrow \alpha B \bullet \beta \) then
          /* Completion: */
          \( \text{IN}[i, (p, q), j] = \sum_{i \leq k \leq j} (\text{IN}[i, (p, q - 1), k] \cdot (\sum_{\text{rule} \in R_B} \text{IN}[k, \text{ind}(\text{rule} B), j])) \)
        end if
      end for
    end for
  end for
end for

Fortunately, as the SCFG \( G_r \), considered in this work is loop-free, each item \([i, \text{ind}(\text{rule} \in R_{G_r} \bullet), j]\) can be associated with a non-looping bucket \( B \) (of size one). Thus, considering the restriction that no item precedes any item on which it depends, an ordering on the items \([i, \text{ind}(\text{rule}), j]\) can be defined by appropriately iterating over positions \( i \) and \( j \), respectively, as well as by using a suitable ordering (indexing) of the elements in \( R_{G_r} \bullet \). Since we use an Earley-style parser, it is obvious that in order to calculate all values of items \([i, \text{ind}(\text{rule} \in R_{G_r} \bullet), j]\), \( 1 \leq i, j \leq n + 1 \) and \( rule \in R_{G_r} \bullet \), we first have to iterate over all values \( j \) from 1 to \( n + 1 \). This means we “shift” the symbol \( \bullet \) from left to right. For each value of \( j \in \{1, \ldots, n + 1\} \), we then have to iterate over all values \( i \) from \( j \) down to 1. Thus, we can first make a prediction for \( i = j \) and then scanning or completion steps for \( i < j \). However, the problem of finding an appropriate ordering of \( R_{G_r} \bullet \) that has to be applied for every pair of fixed positions \( i \) and \( j \) in order to derive the values for items \([i, \text{ind}(\text{rule}), j], \text{rule} \in R_{G_r} \bullet \), is more complicated.

In this work, the ordering of the rules in \( R_{G_r} \bullet \) is defined by index values \((p, q)\), given as follows:

- The first index \( p \in \{1, \ldots, \text{card}(R_{G_r})\} \) corresponds to a set of productions \( R_{G_r \bullet}(\text{rule}) \subset R_{G_r} \bullet \) (the
17Recall that symbol \( \bullet \) is used to mark the current position \( j \), \( 1 \leq j \leq n + 1 \), in the input word up to which the parsing has proceeded.
one that was derived from production \( r \in \mathcal{R}_G \) and

- the corresponding second index \( q \in \{0, \ldots, \text{card}(\mathcal{R}_G, \cdot (r)) \} \) corresponds to a single production \( r \in \mathcal{R}_G, \cdot (r) \) (the one in which symbol \( \bullet \) occurs after the \( q \)th symbol in the conclusion, see above).

Obviously, this ordering within the sets \( \mathcal{R}_G, \cdot (r) \) is appropriate, since if \( r \in A \rightarrow \alpha B \beta \) is indexed by \( p \in \{1, \ldots, \text{card}(\mathcal{R}_G)\} \), then item \([i, (p, q)] = \text{ind}(A \rightarrow \alpha B \beta, j)\) depends on item \([i, (p, q - 1)] = \text{ind}(A \rightarrow \alpha B \beta, j')\) for \( j' \leq j \). Consequently, it remains to find a suitable distinct index \( p \in \{1, \ldots, \text{card}(\mathcal{R}_G)\} \) for any set \( \mathcal{R}_G, \cdot (r) \) corresponding to the original production \( r \in \mathcal{R}_G \), such that the resulting ordering ensures that no item precedes any item on which it depends.

It is easy to see that for predictions and scanning steps, no problems can occur due to our ordering (implied by index \( q \)) within any set \( \mathcal{R}_G, \cdot (r) \). Thus, the center of attention has to be laid on the completion steps. In fact, suppose the value of an item \([i, (p, q)] = \text{ind}(A \rightarrow \alpha B \beta, j)\] has to be computed by completion. Then, this value depends on the values of items \([i, (p, q - 1)] = \text{ind}(A \rightarrow \alpha B \beta, k)\] and \([k, \text{ind}(r \in \mathcal{R}_B, j)]\), for \( i \leq k \leq j \). Whereas in all cases, the value of \([i, (p, q - 1), k]\] has been computed at this point (due to our ordering of \( \mathcal{R}_G, \cdot (A \rightarrow \alpha B \beta) \) and since \( k \leq j \)), problems may arise for \([k, \text{ind}(r \in \mathcal{R}_B, j)]\). Particularly, if \( \alpha \) can not be the empty word, i.e., if \( |\alpha| \geq 1 \) holds, then we only have to consider \( i + 1 \leq k \leq j \), in which cases values of items \([k, \text{ind}(r \in \mathcal{R}_B, j)]\] have already been determined in previous iterations, since \( k > i \). However, if \( \alpha \) can be empty, then \([i, (p, q)] = \text{ind}(A \rightarrow \alpha B \beta, j)\] also depends on \([i, (p', q') = \text{ind}(r \in \mathcal{R}_B, j)]\]. Thus, \( B \rightarrow \gamma \) has to be considered before \( A \rightarrow \alpha B \beta \), which implies \( p' < p \) must hold. In fact, as this holds for any \( r \in \mathcal{R}_B \), we can conclude that if \( \alpha \) can be empty, then in an appropriate ordering, \( A \rightarrow \alpha B \beta \) has to be placed after all productions \( B \rightarrow \gamma \in \mathcal{R}_G \) that have premise \( B \).

According to these observations, the desired ordering can easily be constructed in the following way: Start by assigning the smallest indices \( p \in \{1, \ldots, \text{card}(\mathcal{R}_G)\} \) to productions of the form \( r = I \rightarrow \alpha \), where the first symbol \( t \) of the conclusion is any terminal symbol from \( \Sigma_G \). Then, assign the remaining indices to the other sets \( \mathcal{R}_G, \cdot (r) \), for \( r \in \mathcal{R}_G \), taking into account the previously discussed restrictions. For the sake of simplicity, let us first consider the grammar \( G \) that models the language \( L \) of all secondary structures. For this grammar, we could for example use the following ordering of the corresponding rule set \( \mathcal{R}_G \), i.e., the following ordering by first indices \( p \in \{1, \ldots, \text{card}(\mathcal{R}_G)\} \):

<table>
<thead>
<tr>
<th>Index ( p )</th>
<th>Rule ( r )</th>
<th>Index ( p )</th>
<th>Rule ( r )</th>
<th>Index ( p )</th>
<th>Rule ( r )</th>
<th>Index ( p )</th>
<th>Rule ( r )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( Z \rightarrow \epsilon )</td>
<td>2</td>
<td>( A \rightarrow (m^mL)^{m^m} )</td>
<td>3</td>
<td>( P \rightarrow (L) )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>( C \rightarrow ZC )</td>
<td>5</td>
<td>( C \rightarrow Z )</td>
<td>6</td>
<td>( H \rightarrow ZH )</td>
<td>7</td>
<td>( H \rightarrow Z )</td>
</tr>
<tr>
<td>8</td>
<td>( B \rightarrow ZB )</td>
<td>9</td>
<td>( B \rightarrow Z )</td>
<td>10</td>
<td>( U \rightarrow ZU )</td>
<td>11</td>
<td>( U \rightarrow \epsilon )</td>
</tr>
<tr>
<td>12</td>
<td>( T \rightarrow C )</td>
<td>13</td>
<td>( T \rightarrow A )</td>
<td>14</td>
<td>( T \rightarrow CA )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>( T \rightarrow AT )</td>
<td>16</td>
<td>( T \rightarrow CAT )</td>
<td>17</td>
<td>( F \rightarrow Z^{m_{m-1}H} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>( G \rightarrow BA )</td>
<td>19</td>
<td>( G \rightarrow AB )</td>
<td>20</td>
<td>( G \rightarrow BAB )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>( M \rightarrow UAO )</td>
<td>22</td>
<td>( O \rightarrow UAN )</td>
<td>23</td>
<td>( N \rightarrow UAN )</td>
<td>24</td>
<td>( N \rightarrow U )</td>
</tr>
<tr>
<td>25</td>
<td>( L \rightarrow F )</td>
<td>26</td>
<td>( L \rightarrow P )</td>
<td>27</td>
<td>( L \rightarrow G )</td>
<td>28</td>
<td>( L \rightarrow M )</td>
</tr>
</tbody>
</table>
The derivation of a corresponding ordering for the considered SCFG $G_r$ generating all RNA sequences is straightforward. Thus, we have defined an appropriate ordering of $R_{G_r}$ by indices $(p, q)$, for $p \in \{1, \ldots, \text{card}(R_{G_r})\}$ and $q \in \{0, \ldots, k(p)\}$, where $k(p) = \text{card}(R_{G_r}(\text{rule}))$ if $R_{G_r}(\text{rule})$ can be found under index $p$.

Sm-I.2.3 Outside Computation

Algorithm 2 Computation of Outside Values

**Require:** RNA sequence $r \in L_r$ of length $n \geq 1$,
- set $R_{G_r}$ of production rules used by Earley’s algorithm for parsing $r$ with $G_r$, and the corresponding inside values (computed by Algorithm 1).

1. $\text{OUT}[1, \text{ind}(S \rightarrow T\star), n + 1] = 1$

2. for $j = n + 1 \ldots 1$ do

3. for $i = 1 \ldots j$ do

4. for $p = \text{card}(R_{G_r}) \ldots 1$ do

5. for $q = k(p) \ldots 0$ do

   - rule = ind$^{-1}(p, q)$ /*rule $\in R_{G_r}$ is the rule having index $(p, q)$ in our ordering. */

   - if rule $= A \rightarrow \alpha w_j \bullet \beta$ then

     /* Scanning (reverse): */

     $\text{OUT}[i, (p, q - 1), j] = \delta_{w_j, r_j} \cdot \text{OUT}[i, (p, q), j + 1]$

   - else if rule $= B \rightarrow \bullet \gamma$ then

     /* Prediction (reverse): */

     do nothing

   - else if rule $= A \rightarrow \alpha B \bullet \beta$ then

     /* Completion (reverse): */

     for $k = i \ldots j$ do

     $\text{OUT}[i, (p, q - 1), k] =$

     $\text{OUT}[i, (p, q - 1), k] + \text{OUT}[i, (p, q), j] \cdot \left(\sum_{\text{rule}_B \in R_B} \text{IN}[k, \text{ind}((\text{rule}_B), j)]\right)$

     for rule$_B \in R_B$ do

     $\text{OUT}[k, \text{ind}((\text{rule}_B), j)] =$

     $\text{OUT}[k, \text{ind}((\text{rule}_B), j)] + \text{OUT}[i, (p, q), j] \cdot \text{IN}[i, (p, q - 1), k]$

     end for

     end for

   - end if

   end for

end for

end for

end for

end for

Once the inside values have been computed (with Algorithm 1), the corresponding outside values of all
items \([i, \text{ind}(\text{rule}), j], 1 \leq i, j \leq n + 1\) (for an RNA sequence \(r\) of size \(n\)) and \(\text{rule} \in \mathcal{R}_{G_r} \cdot \) can be calculated with Algorithm 2. This Earley-based semiring parser algorithm uses the reversed previously introduced ordering of items and makes use of the following formulae for the outside computations (for details, we refer to [Goo98, Goo99]):

- Scanning (reverse):
  \[
  \text{OUT}[i, \text{ind}(A \rightarrow \alpha \bullet w_j \beta), j] = \delta_{w_j, r_j} \cdot \text{OUT}[i, \text{ind}(A \rightarrow \alpha w_j \bullet \beta), j + 1].
  \]

- Prediction (reverse):
  There is nothing to do, since this value is obtained while performing a (reverse) completion computation.

- Completion (reverse):
  \[
  \text{OUT}[i, \text{ind}(A \rightarrow \alpha \bullet B \beta), k] = \text{OUT}[i, \text{ind}(A \rightarrow \alpha \bullet B \beta), k] + \sum_{\text{rule}_B \in \mathcal{R}_B} \text{IN}[k, \text{ind}(\text{rule}_B), j] \cdot \text{OUT}[i, \text{ind}(A \rightarrow \alpha \bullet B \beta), k].
  \]
  and
  \[
  \text{OUT}[k, \text{ind}(\text{rule}_B), j] = \text{OUT}[k, \text{ind}(\text{rule}_B), j] + \text{OUT}[i, \text{ind}(A \rightarrow \alpha \bullet B \beta), j] \cdot \text{IN}[i, \text{ind}(A \rightarrow \alpha \bullet B \beta), k],
  \]
  for \(i \leq k \leq j\) and \(\text{rule}_B \in \mathcal{R}_B\).

Since the number of production rules considered for the inside and outside computations is given by \(\text{card}(\mathcal{R}_{G_r} \cdot)\) and is thus not dependent on the input size, Algorithms 1 and 2 need cubic time and quadratic space in the worst-case.

**Sm-I.3 Deriving the Needed Inside and Outside Probabilities**

Finally, since for a given sequence \(r \in \mathcal{L}_r\) of length \(n\), an item of the form \([i, \text{ind}(A \rightarrow \alpha \bullet), j + 1]\), \(1 \leq i, j \leq n + 1\), asserts that \(A \Rightarrow r_i \ldots r_j\), it is easy to see that

\[
\sum_{\text{rule} = A \rightarrow \alpha \bullet} \text{IN}[i, \text{ind}(\text{rule}), j + 1] = \sum_{\text{rule} \in \mathcal{R}_A} \text{IN}[i, \text{ind}(\text{rule}), j + 1] = \alpha_A(i, j)
\]

is the probability of a leftmost derivation that generates the subword \(r_i \ldots r_j\) of \(r\) from the intermediate symbol \(A\). Furthermore, recall that \(\beta_A(i, j)\) is defined as the probability of a derivation which, starting with the intermediate symbol \(S\) (the axiom of the grammar \(G_r\)), generates the expression \(r_1 \ldots r_{i-1} A r_{j+1} \ldots r_n\). For this outside probability, it obviously does not matter what subword \(r_i \ldots r_j\) of \(r\) is derived from intermediate symbol \(A\), i.e., it is independent on which rule \(A \rightarrow \alpha \bullet \in \mathcal{R}_A\) generates subword \(r_i \ldots r_j\). Consequently, for \(\text{rule} = A \rightarrow \alpha \bullet \in \mathcal{R}_A\), the outside value for item \([i, \text{ind}(\text{rule}), j + 1]\) is either equal to zero (if \(r_i \ldots r_j\) can not be derived from non-terminal \(A\) using production \(\text{rule}\)), or it is equal to the outside value for any items \([i, \text{ind}(\text{rule}'), j + 1]\), where \(\text{rule}' \in \mathcal{R}_A\) and \(\text{OUT}[i, \text{ind}(\text{rule}'), j + 1] \neq 0\)
(which means that production rule can be used to generate subword \( r_i \ldots r_j \) of \( r \) from \( A \)). Accordingly, the needed outside probability for symbol \( A \) is equal to one of the non-zero values (if any) of the corresponding production rules with premise \( A \), which can be written as:

\[
\max_{\text{rule} \in A \rightarrow \alpha} \text{OUT}[i, \text{ind(rule)}, j + 1] = \max_{\text{rule} \in \mathcal{R}_A} \text{OUT}[i, \text{ind(rule)}, j + 1] = \beta_A(i, j).
\]

Thus, for any given RNA sequence \( r \in \mathcal{L}_r \) of size \( n \), we can derive the desired inside and outside probabilities \( \alpha_A(i, j) \) and \( \beta_A(i, j) \), for each \( A \in \mathcal{I}_G \) and \( 1 \leq i, j \leq n \), by computing the inside and outside values of all items by semiring parsing based on an Earley-style parser for the SCFG \( G_r \) and afterwards using the results for each \( \text{rule} \in \mathcal{R}_G \), of the form \( \text{rule} = A \rightarrow \alpha \) to obtain the corresponding probabilities for each \( A \in \mathcal{I}_G \) (in the previously described way). Consequently, for sequence \( r \) of size \( n \), there result cubic time complexity and quadratic memory requirements for the computation of all probabilities \( \alpha_A(i, j) \) and \( \beta_A(i, j) \), \( A \in \mathcal{I}_G \) and \( 1 \leq i, j \leq n \).

**Sm-II Details of the Sampling Algorithm**

In this section, we first present equations for computing the needed sampling probabilities for all considered cases (except for exterior loops, since they have already been presented in Section 3.2.1). Afterwards, we give a detailed description of the corresponding sampling algorithm, including detailed information on how to use the respective sampling probabilities. Note that these parts are written in a similar way as the corresponding section in [DL03], in order to illustrate the similarities and differences that arise when computing the sampling probabilities according to either approach.

**Sm-II.1 Equations for Computation of Sampling Probabilities**

Basically, the definitions of the needed sampling probabilities for all regular loop types can be derived in the same way as those already presented in Section 3.2.1 for exterior loops – by using only the corresponding inside outside values and the probabilities of the production rules of the considered SCFG.

**Sm-II.1.1 Sampling Probabilities for Substructures Between a Given Base Pair**

Given a base pair \( r_i, r_j \), then this pair can either be the closing base pair of a hairpin loop, the exterior pair of a base pair stack, the closing pair of a bulge or an interior loop, or close a multibranched loop. For all of these cases, the corresponding probabilities are given as follows:

\[
Q_{ij}^{HL}(i, j) = \frac{1}{q_{ij}(i, j)} \cdot \beta_L(i + 1, j - 1) \cdot (\alpha_F(i + 1, j - 1) \cdot \Pr(L \rightarrow F)),
\]

\[
Q_{ij}^{SP}(i, j) = \frac{1}{q_{ij}(i, j)} \cdot \beta_L(i + 1, j - 1) \cdot (\alpha_P(i + 1, j - 1) \cdot \Pr(L \rightarrow P)),
\]

\[
Q_{ij}^{BI}(i, j) = \frac{1}{q_{ij}(i, j)} \cdot \beta_L(i + 1, j - 1) \cdot (\alpha_G(i + 1, j - 1) \cdot \Pr(L \rightarrow G)),
\]

\[
Q_{ij}^{ML}(i, j) = \frac{1}{q_{ij}(i, j)} \cdot \beta_L(i + 1, j - 1) \cdot (\alpha_M(i + 1, j - 1) \cdot \Pr(L \rightarrow M)).
\]

Here, we have to use the normalizing factor

\[
q_{ij}(i, j) = \beta_L(i + 1, j - 1) \cdot \alpha_L(i + 1, j - 1).
\]
Thus, $Q_H^{ML}(i,j)$, $Q_S^{ML}(i,j)$, $Q_B^{ML}(i,j)$ and $Q_{ij}^{ML}(i,j)$ is the sampling probability for a hairpin loop, base pair stack, bulge or interior loop and multibranched loop, respectively, where for mutually exclusive and exhaustive cases, $Q_H^{ML}(i,j) + Q_S^{ML}(i,j) + Q_B^{ML}(i,j) + Q_{ij}^{ML}(i,j) = 1$ holds.

**Sm-II.1.2 Sampling Probabilities for Bulge and Interior Loops**

For sampling bulge and interior loops corresponding to the PF approach, we would have to use the following probabilities:

$$P_{hl}^{Bl}(i,j,h,l) = \begin{cases} P_{hl}^{B1}(i,j,h), & \text{if } h > i + 1 \text{ and } l = j - 1, \\ P_{hl}^{B2}(i,j,l), & \text{if } h = i + 1 \text{ and } l < j - 1, \\ P_{hl}^{IL}(i,j,h), & \text{if } h > i + 1 \text{ and } l < j - 1, \\ 0, & \text{else,} \end{cases}$$

where

$$P_{hl}^{B1}(i,j,h) = \frac{1}{Q_{ij}^{Bl}(i,j)} \cdot \beta_L(i+1,j-1) \cdot \Pr(L \to G) \times (\alpha_B(i+1,h-1) \cdot \alpha_A(h,j-1) \cdot \Pr(G \to BA)), $$

$$P_{hl}^{B2}(i,j,l) = \frac{1}{Q_{ij}^{Bl}(i,j)} \cdot \beta_L(i+1,j-1) \cdot \Pr(L \to G) \times (\alpha_A(i+1,l) \cdot \alpha_B(l+1,j-1) \cdot \Pr(G \to AB)), $$

$$P_{hl}^{IL}(i,j,h,l) = \frac{1}{Q_{ij}^{Bl}(i,j)} \cdot \beta_L(i+1,j-1) \cdot \Pr(L \to G) \times (\alpha_B(i+1,h-1) \cdot \alpha_A(h,l) \cdot \alpha_B(l+1,j-1) \cdot \Pr(G \to BAB)). $$

After the case of bulge or interior loop was sampled, $\{P_{hl}^{Bl}(i,j,h,l)\}$ would then be used for sampling $h$ and $l$ (together in one single sampling step) and for mutually exclusive and exhaustive cases, $\sum_{h=(i+1)}^{j-min_{ps}} \sum_{l=(h-1)+min_{ps}}^{(j-1)} P_{hl}^{Bl}(i,j,h,l) = 1$ (under the condition that $Q_{ij}^{Bl}(i,j) > 0$).

However, to ensure that the sampling algorithm runs in cubic time, we would then have to disregard long bulge and interior loops by using a constant max$_{BL}$ - just like with PFs$^{18}$. Nevertheless, we do not need to apply this restriction if we sample $h$ and $l$ one after the other with the following probabilities:

$$p_{hj}^{B1}(i,j,h) = \frac{1}{p_{ij}^{Bl}(i,j)} \cdot \beta_G(i+1,j-1) \cdot (\alpha_B(i+1,h-1) \cdot \alpha_A(h,j-1) \cdot \Pr(G \to BA)), $$

$$p_{il}^{B1}(i,j,l) = \frac{1}{p_{ij}^{Bl}(i,j)} \cdot \beta_G(i+1,j-1) \cdot (\alpha_A(i+1,l) \cdot \alpha_B(l+1,j-1) \cdot \Pr(G \to AB)), $$

$$p_{hl}^{B1}(i,j,h) = \frac{1}{p_{ij}^{Bl}(i,j)} \cdot \beta_G(i+1,j-1) \cdot (\alpha_B(i+1,h-1) \cdot \alpha_A(h,l) \cdot \alpha_B(l+1,j-1) \cdot \Pr(G \to BAB)), $$

$$p_{hl}^{IL}(i,j,h,l) = \frac{1}{\alpha_{AB}(h-1,j)} \cdot (\alpha_A(h,l) \cdot \alpha_B(l+1,j-1)), $$

where

$$\alpha_{AB}(i,j) = \sum_{l=i+min_{ps}}^{(j-2)} (\alpha_A(i+1,l) \cdot \alpha_B(l+1,j-1))$$

and

$$p_{hj}^{Bl}(i,j) = \beta_G(i+1,j-1) \cdot \alpha_G(i+1,j-1).$$

$^{18}$Note that when using the PF approach based on thermodynamics, $h$ and $l$ have to be sampled at once, since the free energy of a bulge or interior loops strongly depends on both the closing pair $r_{r_{h}}$ and the accessible pair $r_{r_{j}}$. 

46
In the case of a multibranched loop, the probabilities for sampling the first accessible base pair within this loop can be obtained by considering the intermediate symbols of multiloops. More specifically, we first sample \( \sum_{h=1}^{j-\min_{ps}} P_{hl}^B(i, j, h) + \sum_{l=i+\min_{ps}}^{j-\min_{ps}-1} P_{hl}^B(i, j, l) + \sum_{h=(i+2)}^{j-\min_{ps}-1} P_{hl}^B(i, j, h) = 1 \), and, under the condition that \( P_{hl}^B(i, j, h) > 0 \), also \( \sum_{l=(h-1)+\min_{ps}}^{j-\min_{ps}-1} \tilde{P}_{hl}^B(j, h, l) = 1 \).

### Sm-II.1.3 Sampling Probabilities for Multiloops

In the case of a multibranched loop, the probabilities for sampling the first accessible base pair \( r_{h_1}, r_{l_1} \) within this loop can be obtained by considering the intermediate symbols of \( G_r \) that generate (parts of) multiloops. More specifically, we first sample \( h \) and \( l \) according to the following conditional probabilities:

\[
\begin{align*}
P_{hl}^M(i, j, h) &= \frac{1}{p_{hl}^M(i, j)} \cdot \beta_M(i+1, j-1) \cdot (\alpha_U(i+1, h-1) \cdot \alpha_{AO}(h, j) \cdot \Pr(M \rightarrow UAO)), \\
\tilde{P}_{hl}^M(j, h, l) &= \frac{1}{\alpha_{AO}(h, j)} \cdot (\alpha_A(h, l) \cdot \alpha_O(l+1, j-1)),
\end{align*}
\]

where

\[
\alpha_{AO}(h, j) = \sum_{l=(h-1)+\min_{ps}}^{j-\min_{ps}-1} (\alpha_A(h, l) \cdot \alpha_O(l+1, j-1))
\]

and

\[
p_{hl}^M(i, j) = \beta_M(i+1, j-1) \cdot \alpha_M(i+1, j-1).
\]

Note that we have to take care of the \( \epsilon \)-rule \( U \rightarrow \epsilon \), which implies that symbol \( U \) may generate words of size zero. For this reason, \( h = i + 1 \) could be chosen, implying \( h - 1 < i + 1 \). However, \( \alpha_U(i+1, h-1) \) is only defined for \( i+1 \leq h-1 \). To fix this problem, we have used the term \( \alpha_U^*(i+1, h-1) \) instead of \( \alpha_U(i+1, h-1) \) in the previous two definitions, which is given as follows:

\[
\alpha_U^*(i+1, h-1) = \begin{cases} 
\alpha_U(i+1, h-1), & \text{if } i+1 \leq h-1, \\
\Pr(U \rightarrow \epsilon), & \text{if } i+1 > h-1.
\end{cases}
\]

Thus, \( \{\tilde{P}_{hl}^M(j, h, l)\} \) are probabilities for sampling \( l \) after \( h \geq i + 1 \) is sampled with probabilities \( \{P_{hl}^M(i, j, h)\} \). For mutually exclusive and exhaustive cases, we have \( \sum_{h=(i+1)}^{j-\min_{ps}} P_{hl}^M(i, j, h) = 1 \), and accordingly, \( \sum_{l=(h-1)+\min_{ps}}^{j-\min_{ps}-1} \tilde{P}_{hl}^M(j, h, l) = 1 \). Sampling both \( h \) and \( l \) yields the first accessible base pair \( r_{h_1}, r_{l_1} \) (which closes the first helix radiating out from this multiloop).

In order to sample the second accessible base pair \( r_{h_2}, r_{l_2} \), we consider the remaining structure fragment \( R_{(l_1+1)(j-1)} \) (between the 3' base \( r_{l_1} \) of the first accessible base pair \( r_{h_1}, r_{l_1} \) and the 3' base \( r_j \) of the closing base pair \( r_i, r_j \) of the considered multiloop). In fact, for any \( k \geq 1 \), the probabilities for sampling the \( (k+1) \)th accessible base pair \( r_{h_{k+1}}, r_{l_{k+1}} \) within this multibranched loop are computed by considering the structure fragment \( R_{(l_{k+1})(j-1)} \) and using the corresponding inside and outside variables for some specific multiloop generating intermediate symbols of the grammar \( G_r \). More specifically, we first sample \( h \) and \( l \) according to conditional probabilities, which are defined as follows:

\[
P_{hl}^{M_{k+1}}(l_{k+1}, j, h) = \frac{1}{p_{hl}^{M_{k+1}}(l_{k+1}, j)} \cdot \beta_X(l_{k+1}, j-1) \cdot (\alpha_U^*(l_{k+1}, h-1) \cdot \alpha_{AN}(h, j) \cdot \Pr(X \rightarrow UAN)),
\]

\[
\tilde{P}_{hl}^{M_{k+1}}(l_{k+1}, h_{k+1}, j, l_{k+1}) \quad \text{and} \quad \tilde{P}_{hl}^{M_{k+1}}(j, h_{k+1}, l_{k+1}).
\]

Note that \( \tilde{P}_{hl}^{M_{k+1}}(j, h_{k+1}, l_{k+1}) \) is only defined for \( l_{k+1} - h_{k+1} \leq \min_{ps} \).

47
\[ \hat{P}_{hl}^{M_{k+1}}(j, h, l) = \frac{1}{\alpha_{AN}(h, j)} \cdot (\alpha_A(h, l) \cdot \alpha_N^*(l + 1, j - 1)), \]

where

\[ \alpha_{AN}(h, j) = \sum_{l=(h-1)+\text{min}_p}^{(j-1)} (\alpha_A(h, l) \cdot \alpha_N^*(l + 1, j - 1)) \]

and

\[ p_{hl}^{M_{k+1}}(l, k) = \begin{cases} 
\beta_O(l_k + 1, j - 1) \cdot \alpha_O(l_k + 1, j - 1), & \text{if } (k + 1) = 2, \\
\beta_N(l_k + 1, j - 1) \cdot \alpha_N(l_k + 1, j - 1) - \beta_N(l_k + 1, j - 1) \cdot (\alpha_U(l + 1, j - 1) \cdot \Pr(N \rightarrow U)), & \text{if } (k + 1) \geq 3,
\end{cases} \]

as well as

\[ X = \begin{cases} 
O, & \text{if } (k + 1) = 2, \\
N, & \text{if } (k + 1) \geq 3.
\end{cases} \]

Again, we have used \( \alpha^*_U \) instead of \( \alpha_U \) and \( \alpha^*_N \) instead of \( \alpha_N \), which is defined as

\[ \alpha^*_N(l + 1, j - 1) = \begin{cases} 
\alpha_N(l + 1, j - 1), & \text{if } l + 1 \leq j - 1, \\
\Pr(N \rightarrow U) \cdot \Pr(U \rightarrow \epsilon), & \text{if } l + 1 > j - 1,
\end{cases} \]

in order to take care of possible cases where \( U \) and/or \( N \) generate words of size zero. According to these definitions, \{\hat{P}_{hl}^{M_{k+1}}(j, h, l)\} are probabilities for sampling \( l \) after \( h \geq l_k + 1 \) is sampled with probabilities \{\hat{P}_{hl}^{M_{k+1}}(l, k)\} and again, for mutually exclusive and exhaustive cases, we have \( \sum_{h=(k+1)}^{j-\text{min}_p} \hat{P}_{hl}^{M_{k+1}}(l, k, h) = 1 \), and \( \sum_{l=(h-1)+\text{min}_p}^{(j-1)} \hat{P}_{hl}^{M_{k+1}}(j, h, l) = 1 \). By sampling both \( h \) and \( l \), we obtain the desired \((k + 1)\)th accessible base pair \( r_{h_{k+1}}.r_{l_{k+1}} := r_h.r_l \) (which closes the \((k+1)\)th helix radiating out from this multiloop).

According to the definition of multibranched loops, we now have to address two different cases: either the considered multiloop contains no additional accessible base pair, or there is at least one more base pair accessible from the closing pair \( r_i.r_j \). These two mutually exclusive cases are addressed by the following two probabilities: Conditional on the sampled values for \( h_k \) and \( l_k \) (for the \( k \)th accessible base pair \( r_{h_k}.r_{l_k} \) in the considered multiloop), \( k \geq 2 \), we consider the following “decision” probability for no additional accessible base pairs on the structure fragment \( R_{(l_{k+1})(j-1)} \) (i.e., between the 3′ base \( r_{l_k} \) of the \( k \)th accessible base pair \( r_{h_k}.r_{l_k} \) and the 3′ base \( r_j \) of the closing base pair \( r_i.r_j \)):

\[ \hat{P}_{01}^{M_{k+1}}(l, j) = \frac{1}{p_{01}(l, j)} \cdot \beta_N(l_k + 1, j - 1) \cdot (\alpha_U(l_k + 1, j - 1) \cdot \Pr(N \rightarrow U)), \]

where

\[ p_{01}(l, j) = \begin{cases} 
\beta_N(l_k + 1, j - 1) \cdot (\alpha_U(l_k + 1, j - 1) \cdot \Pr(N \rightarrow U)), & \text{if } (j - l_k - 1) < \text{min}_p, \\
\beta_N(l_k + 1, j - 1) \cdot \alpha_N(l_k + 1, j - 1), & \text{if } (j - l_k - 1) \geq \text{min}_p.
\end{cases} \]

Accordingly, the probability that there is at least one more accessible base pair in the considered multiloop (i.e., on the structure fragment \( R_{(l_{k+1})(j-1)} \)) is given by \( 1 - \hat{P}_{01}^{M_{k+1}}(l, j) \).

If no additional accessible base pair is sampled, the sampling process for the considered multibranched loop (closed by pair \( r_i.r_j \)) is terminated; the resulting loop is thus a \((k+1)\)-loop, with \( k \) internal helices closed by the \( k \) sampled base pairs \( r_{h_p}.r_{l_p}, 1 \leq p \leq k \), accessible from the closing pair \( r_i.r_j \). Otherwise,
the next accessible base pair $r_{hk+1}, r_{lk+1}$ is sampled and afterwards, it has yet again to be decided whether the loop contains additional accessible base pairs or not (by another “decision” sampling). This process is then repeated until no additional base pair is sampled.

Sm-II.2 Formal Description of the Sampling Process

According to the previous discussion, it should be clear that a secondary structure for a given RNA sequence $r \in L_r$ of length $n$ can be sampled in the following recursive way: Start with the entire RNA sequence $R_{1n}$ and consecutively compute the adjacent substructures (single-stranded regions and paired substructures) of the exterior loop (from left to right). Any paired substructure, say the $k$th substructure of the exterior loops, has to be completed by successively folding other loops (hairpins, stacked pairs, bulges, interior and multibranched loops) before the $(k + 1)$th adjacent substructure is computed. This means that the folding process performed by the sampling algorithm corresponds to the native folding procedures of RNA molecules (from left to right, due to the aspects of co-transcriptional folding).

For a formal description on how the sampling algorithm works and explicit information on where each of the previously defined sampling probabilities has to be considered in order to perform the needed random choices, see Algorithms 3 to 6.

It remains to mention that when the probabilities $\alpha_x(i, j)$, $x \in \{AT, AB, AO, AN\}$, $1 \leq i, j \leq n$, are also precomputed, each of the needed sampling probabilities can be derived in constant time. Thus, after a preprocessing of the given RNA sequence (which includes the complete inside outside computation and takes cubic time and requires quadratic storage), corresponding secondary structures can be quickly generated. In fact, the time complexity of the sampling algorithm is bounded by $O(n^2)$, since any structure of size $n$ can have at most $\lfloor \frac{n - \min_{x} n_x}{2} \rfloor$ base pairs and any base pair can be sampled in linear time.
Algorithm 3 Sampling an entire secondary structure

Require: RNA sequence $r \in \mathcal{L}$, of length $n \geq 1$, and all previously defined sampling probabilities computed for $r$ (as global variables).

procedure computeRandomExteriorLoop ($n$)
    $sec = \emptyset$
    Set $i = 1$, $j = n$ and $k = 0$
    while $(j - i + 1) \neq 0$ do
        /*Create $(k+1)$th helix, starting with free base pair $h.l$, $i < h < l < j$, or leave $R_{ij}$ unpaired:*/
        $extLoopType = $ Sample exterior loop substructure type for $R_{ij}$
        if $extLoopType = P_E^0(i, j)$ /*case (a): $R_{ij}$ is single-stranded:*/ then
            return $sec$
        else if $extLoopType = P_{ij}^E(i, j)$ /*case (b): $i = h$ and $j = l$:*/
            Set $h = i$ and $l = j$
        else if $extLoopType = \sum_{h=(i+1)}^{j-1} P_{hi}^E(i, j, h)$ /*case (c): $i < h < l = j$:*/ then
            Sample $h \in [(i+1), (j+1) - \min_{ps}]$ according to probabilities $\{P_{hi}^E(i, j, h)\}$
            Set $l = j$
        else if $extLoopType = \sum_{l=(i-1)+\min_{ps}}^{j-1} P_{il}^E(i, j, l)$ /*case (d): $i = h < l < j$:*/ then
            Sample $l \in [(i-1) + \min_{ps}, (j-1)]$ according to probabilities $\{P_{il}^E(i, j, l)\}$
        else if $extLoopType = \sum_{h=(i+1)}^{j-1} P_{hl}^E(i, j, h)$ /*case (e): $i < h < l < j$:*/ then
            Sample $h \in [(i+1), j - \min_{ps}]$ according to probabilities $\{P_{hl}^E(i, j, h)\}$
            Sample $l \in [(h-1) + \min_{ps}, (j-1)]$ according to probabilities $\{P_{hl}^E(j, h, l)\}$
        end if
        /*Collect base pairs for $(k+1)$th substructure and add them to the entire structure:*/
        $sub = \{h.l, (h+1),(l-1), \ldots, (h+(\min_{hel}-1)),(l-(\min_{hel}-1))\}$
        $sub = sub \cup$ computeRandomLoop $(h+(\min_{hel}-1), l-(\min_{hel}-1))$
        $sec = sec \cup sub$
        /*Consider the remaining fragment $R_{(i+1)j}$:*/
        Set $i = l + 1$ /*=next unpaired base after free base pair $h.l$*/ and $k = k + 1$
    end while
    return $sec$
end procedure
Algorithm 4 Sampling any substructure of an entire secondary structure

procedure computeRandomLoop \( (i, j) \)

Set \( \text{sec} = \emptyset \)

\( \text{randLoopType} = \) Sample loop type closed by \( i, j \)

if \( \text{randLoopType} \equiv Q_{ij}^{HL}(i, j) \) /*i, j closes hairpin loop:/ then

return \( \text{sec} \)

else if \( \text{randLoopType} \equiv Q_{ij}^{SP}(i, j) \) /*i, j closes stacked pair:/ then

\( \text{sec} = \text{sec} \cup \{ (i + 1), (j - 1) \} \)

\( \text{sec} = \text{sec} \cup \text{computeRandomLoop} (i + 1, j - 1) \)

else if \( \text{randLoopType} \equiv Q_{ij}^{BI}(i, j) \) /*i, j closes bulge or interior loop:/ then

\( \text{sec} = \text{sec} \cup \text{computeRandomBulgeInteriorLoop} (i, j) \)

else if \( \text{randLoopType} \equiv Q_{ij}^{ML}(i, j) \) /*i, j closes multiloop:/ then

\( \text{sec} = \text{sec} \cup \text{computeRandomMultiLoop} (i, j) \)

end if

return \( \text{sec} \)

end procedure
Algorithm 5 Sampling a bulge or interior loop within a secondary structure

procedure computeRandomBulgeInteriorLoop (i, j)
if Sample strictly corresponding to PF approach then
	/*This requires to use a constant maxBL:*/
	Sample h and l according to probabilities \{P_{hl}^B(i, j, h, l)\}
else
	/*This allows maxBL = \infty (then no restrictions are applied):*/
	loopType = Sample bulge or interior loop type for R_{ij}
if loopType \equiv \sum_{h=(i+2)}^{j-\text{min}_p} P_{hl}^B(i, j, h) /*bulge on the left:*/
then
	Sample h \in [(i + 2), j - \text{min}_p] according to probabilities \{P_{hl}^B(i, j, h)\}
	Set l = j
else if loopType \equiv \sum_{l=i+\text{min}_p}^{j-2} P_{il}^B(i, j, l) /*bulge on the right:*/
then
	Set h = i
	Sample l \in [i + \text{min}_p, (j - 2)] according to probabilities \{P_{il}^B(i, j, l)\}
else if loopType \equiv \sum_{h=(i+2)}^{j-\text{min}_p-1} P_{hl}^B(i, j, h) /*interior loop:*/
then
	Sample h \in [(i + 2), j - \text{min}_p - 1] according to probabilities \{P_{hl}^B(i, j, h)\}
	Sample l \in [(h - 1) + \text{min}_p, (j - 2)] according to probabilities \{\hat{P}_{hl}^B(j, h, l)\}
end if
end if
sec = \{h, l, (h + 1), (l - 1), \ldots, (h + (\text{min}_h - 1)), (l - (\text{min}_l - 1))\}
sec = sec \cup computeRandomLoop (h + (\text{min}_h - 1), l - (\text{min}_l - 1))
return sec
end procedure
Algorithm 6 Sampling a multiloop within a secondary structure

procedure computeRandomMultiLoop (i, j)
    Set sec = \emptyset, k = 0 and l_k = i
    while \((j - l_k - 1) \geq \min_{ps}\) do
        /*Create \((k+1)\)th helix, starting with accessible pair \(h_{k+1}, l_{k+1}\), \(l_k < h_{k+1} < l_{k+1} < j\):*/
        if \((k + 1) = 1\) then
            Sample \(h \in [(i + 1), j - 2 \cdot \min_{ps}]\) according to probabilities \(\{P_{hl}^M(i, j, h)\}\)
            Sample \(l \in [(h + 1) + \min_{ps}, (j - 1) - \min_{ps}]\) according to probabilities \(\{\hat{P}_{hl}^M(j, h, l)\}\)
            Set \(h_1 = h\) and \(l_1 = l\)
        else
            Sample \(h \in [(i + 1), j - \min_{ps}]\) according to probabilities \(\{P_{hl}^{M_k+1}(i, j, h)\}\)
            Sample \(l \in [(h + 1) + \min_{ps}, (j - 1)]\) according to probabilities \(\{\hat{P}_{hl}^{M_k+1}(j, h, l)\}\)
            Set \(h_{k+1} = h\) and \(l_{k+1} = l\)
        end if
        /*Collect base pairs for \((k+1)\)th substructure and add them to the entire structure:*/
        sub = \{\(h.l, (h + 1).l - 1), \ldots, (h + (\min_{hel} - 1)).(l - (\min_{hel} - 1))\}\)
        sub = sub \cup \text{computeRandomLoop} \((h + (\min_{hel} - 1), l - (\min_{hel} - 1))\)
        sec = sec \cup sub
        /*Decide whether to leave the remaining fragment \(R_{(l_{k+1} + 1)(j-1)}\) unpaired or not:*/
        if \((k + 1) \geq 2\) then
            Sample "decision" according to \(P_{01}^{M_k+1}(l_{k+1}, j)\) and \(1 - P_{01}^{M_k+1}(l_{k+1}, j)\)
            if \(P_{01}^{M_k+1}(l_{k+1}, j)\) /*no additional base pairs on \(R_{(l_{k+1} + 1)(j-1)}\):*/ then
                return sec
            else
                Set \(k = k + 1\)
            end if
        end if
    end while
    return sec
end procedure
### Table 13: Results related to the shapes of selected predictions, obtained from our tRNA database (by 10-fold cross-validation procedures, using sample size 1000).

#### CSP<sub>freq</sub> (selection principle MF struct.):

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>PF</td>
<td>max&lt;sub&gt;BL&lt;/sub&gt; = 30</td>
<td></td>
<td>0.0633</td>
<td>0.1216</td>
<td>0.2071</td>
<td>0.2117</td>
<td>0.2639</td>
<td>0.3694</td>
</tr>
<tr>
<td>SCFG</td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 1, min&lt;sub&gt;hel&lt;/sub&gt; = 1</td>
<td>0.2099</td>
<td>0.3699</td>
<td>0.5594</td>
<td>0.5594</td>
<td>0.5599</td>
<td>0.6302</td>
<td></td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 1, min&lt;sub&gt;hel&lt;/sub&gt; = 2</td>
<td>0.2187</td>
<td>0.3833</td>
<td>0.5830</td>
<td>0.5830</td>
<td>0.5835</td>
<td>0.6607</td>
<td></td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 3, min&lt;sub&gt;hel&lt;/sub&gt; = 1</td>
<td>0.2450</td>
<td>0.4448</td>
<td>0.6417</td>
<td>0.6417</td>
<td>0.6422</td>
<td>0.7356</td>
<td></td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 3, min&lt;sub&gt;hel&lt;/sub&gt; = 2</td>
<td>0.2409</td>
<td>0.4364</td>
<td>0.6399</td>
<td>0.6399</td>
<td>0.6403</td>
<td>0.7379</td>
<td></td>
</tr>
</tbody>
</table>

#### CSP<sub>freq</sub> (selection principle MEA struct.):

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>PF</td>
<td>max&lt;sub&gt;BL&lt;/sub&gt; = 30</td>
<td></td>
<td>0.0416</td>
<td>0.1049</td>
<td>0.1923</td>
<td>0.1960</td>
<td>0.2496</td>
<td>0.3559</td>
</tr>
<tr>
<td>SCFG</td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 1, min&lt;sub&gt;hel&lt;/sub&gt; = 1</td>
<td>0.0555</td>
<td>0.2094</td>
<td>0.4193</td>
<td>0.4193</td>
<td>0.4207</td>
<td>0.4679</td>
<td></td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 1, min&lt;sub&gt;hel&lt;/sub&gt; = 2</td>
<td>0.0656</td>
<td>0.2446</td>
<td>0.4961</td>
<td>0.4961</td>
<td>0.4984</td>
<td>0.5613</td>
<td></td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 3, min&lt;sub&gt;hel&lt;/sub&gt; = 1</td>
<td>0.0772</td>
<td>0.2510</td>
<td>0.4928</td>
<td>0.4928</td>
<td>0.4942</td>
<td>0.5497</td>
<td></td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 3, min&lt;sub&gt;hel&lt;/sub&gt; = 2</td>
<td>0.1008</td>
<td>0.2917</td>
<td>0.5525</td>
<td>0.5525</td>
<td>0.5543</td>
<td>0.6241</td>
<td></td>
</tr>
</tbody>
</table>

#### CSP<sub>freq</sub> (selection principle Centroid):

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>PF</td>
<td>max&lt;sub&gt;BL&lt;/sub&gt; = 30</td>
<td></td>
<td>0.0264</td>
<td>0.0800</td>
<td>0.1595</td>
<td>0.1627</td>
<td>0.1932</td>
<td>0.2677</td>
</tr>
<tr>
<td>SCFG</td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 1, min&lt;sub&gt;hel&lt;/sub&gt; = 1</td>
<td>0.0374</td>
<td>0.1276</td>
<td>0.2973</td>
<td>0.2973</td>
<td>0.2978</td>
<td>0.3130</td>
<td></td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 1, min&lt;sub&gt;hel&lt;/sub&gt; = 2</td>
<td>0.0485</td>
<td>0.1623</td>
<td>0.3791</td>
<td>0.3791</td>
<td>0.3800</td>
<td>0.4097</td>
<td></td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 3, min&lt;sub&gt;hel&lt;/sub&gt; = 1</td>
<td>0.0536</td>
<td>0.1665</td>
<td>0.3773</td>
<td>0.3773</td>
<td>0.3778</td>
<td>0.4060</td>
<td></td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 3, min&lt;sub&gt;hel&lt;/sub&gt; = 2</td>
<td>0.0758</td>
<td>0.2150</td>
<td>0.4563</td>
<td>0.4563</td>
<td>0.4568</td>
<td>0.5003</td>
<td></td>
</tr>
</tbody>
</table>
### Table 14: Results related to the shapes of sampled structures, obtained from our tRNA database (by 10-fold cross-validation procedures, using sample size 1000).

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0 1 2 3 4 5</td>
</tr>
<tr>
<td>PF</td>
<td>max(_{BL} = 30)</td>
<td>0.5196 0.6740 0.8160 0.8239 0.8798 0.9556</td>
</tr>
<tr>
<td>SCFG</td>
<td>(\min_{HL} = 1, \min_{hel} = 1)</td>
<td>0.6838 0.9459 0.9903 0.9908 0.9995</td>
</tr>
<tr>
<td></td>
<td>(\min_{HL} = 1, \min_{hel} = 2)</td>
<td>0.6806 0.9006 0.9630 0.9635 0.9640 0.9991</td>
</tr>
<tr>
<td></td>
<td>(\min_{HL} = 3, \min_{hel} = 1)</td>
<td>0.7148 0.9459 0.9875 0.9880 0.9885 0.9991</td>
</tr>
<tr>
<td></td>
<td>(\min_{HL} = 3, \min_{hel} = 2)</td>
<td>0.7111 0.8997 0.9677 0.9681 0.9686 0.9995</td>
</tr>
<tr>
<td>PF</td>
<td>max(_{BL} = 30)</td>
<td>21.073 58.200 136.67 140.63 205.54 328.56</td>
</tr>
<tr>
<td>SCFG</td>
<td>(\min_{HL} = 1, \min_{hel} = 1)</td>
<td>16.202 98.357 327.26 327.27 327.51 418.80</td>
</tr>
<tr>
<td></td>
<td>(\min_{HL} = 1, \min_{hel} = 2)</td>
<td>25.205 142.50 453.03 453.03 453.10 527.04</td>
</tr>
<tr>
<td></td>
<td>(\min_{HL} = 3, \min_{hel} = 1)</td>
<td>24.883 130.04 392.78 392.79 393.05 494.79</td>
</tr>
<tr>
<td></td>
<td>(\min_{HL} = 3, \min_{hel} = 2)</td>
<td>34.898 173.73 513.05 513.06 513.08 595.26</td>
</tr>
<tr>
<td>PF</td>
<td>max(_{BL} = 30)</td>
<td>355.32 130.22 81.796 33.125 22.585 4.8848</td>
</tr>
<tr>
<td>SCFG</td>
<td>(\min_{HL} = 1, \min_{hel} = 1)</td>
<td>802.27 244.52 60.504 60.030 59.916 28.764</td>
</tr>
<tr>
<td></td>
<td>(\min_{HL} = 1, \min_{hel} = 2)</td>
<td>652.75 125.69 24.687 24.687 24.687 16.019</td>
</tr>
<tr>
<td></td>
<td>(\min_{HL} = 3, \min_{hel} = 1)</td>
<td>752.71 208.65 48.257 47.797 47.691 21.838</td>
</tr>
<tr>
<td></td>
<td>(\min_{HL} = 3, \min_{hel} = 2)</td>
<td>592.84 103.04 18.921 18.921 18.921 12.053</td>
</tr>
</tbody>
</table>
CSP<sub>freq</sub> (selection principle MF struct.):

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0  1  2  3  4  5</td>
</tr>
<tr>
<td>PF</td>
<td>max&lt;sub&gt;BL&lt;/sub&gt; = 30</td>
<td>0.0000 0.0009 0.0078 0.0513 0.0261 0.6353</td>
</tr>
<tr>
<td>SCFG</td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 1, min&lt;sub&gt;hel&lt;/sub&gt; = 1</td>
<td>0.0000 0.0026 0.0052 0.0131 0.0357 0.7128</td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 1, min&lt;sub&gt;hel&lt;/sub&gt; = 2</td>
<td>0.0000 0.0052 0.0139 0.0331 0.0522 0.7502</td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 3, min&lt;sub&gt;hel&lt;/sub&gt; = 1</td>
<td>0.0000 0.0044 0.0113 0.0314 0.0766 0.7781</td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 3, min&lt;sub&gt;hel&lt;/sub&gt; = 2</td>
<td>0.0009 0.0096 0.0244 0.0609 0.1027 0.8207</td>
</tr>
</tbody>
</table>

CSP<sub>freq</sub> (selection principle MEA struct.):

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0  1  2  3  4  5</td>
</tr>
<tr>
<td>PF</td>
<td>max&lt;sub&gt;BL&lt;/sub&gt; = 30</td>
<td>0.0000 0.0052 0.0139 0.0835 0.0696 0.6640</td>
</tr>
<tr>
<td>SCFG</td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 1, min&lt;sub&gt;hel&lt;/sub&gt; = 1</td>
<td>0.0000 0.0000 0.0000 0.0000 0.0261 0.3820</td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 1, min&lt;sub&gt;hel&lt;/sub&gt; = 2</td>
<td>0.0000 0.0009 0.0009 0.0035 0.0566 0.4769</td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 3, min&lt;sub&gt;hel&lt;/sub&gt; = 1</td>
<td>0.0000 0.0000 0.0000 0.0000 0.0261 0.3977</td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 3, min&lt;sub&gt;hel&lt;/sub&gt; = 2</td>
<td>0.0009 0.0009 0.0009 0.0035 0.0557 0.5387</td>
</tr>
</tbody>
</table>

CSP<sub>freq</sub> (selection principle Centroid):

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0  1  2  3  4  5</td>
</tr>
<tr>
<td>PF</td>
<td>max&lt;sub&gt;BL&lt;/sub&gt; = 30</td>
<td>0.0000 0.0026 0.0104 0.0775 0.0731 0.7214</td>
</tr>
<tr>
<td>SCFG</td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 1, min&lt;sub&gt;hel&lt;/sub&gt; = 1</td>
<td>0.0000 0.0000 0.0000 0.0000 0.0104 0.1097</td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 1, min&lt;sub&gt;hel&lt;/sub&gt; = 2</td>
<td>0.0000 0.0000 0.0000 0.0000 0.0148 0.1279</td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 3, min&lt;sub&gt;hel&lt;/sub&gt; = 1</td>
<td>0.0000 0.0000 0.0000 0.0000 0.0078 0.1236</td>
</tr>
<tr>
<td></td>
<td>min&lt;sub&gt;HL&lt;/sub&gt; = 3, min&lt;sub&gt;hel&lt;/sub&gt; = 2</td>
<td>0.0000 0.0000 0.0000 0.0009 0.0139 0.1549</td>
</tr>
</tbody>
</table>

Table 15: Results related to the shapes of selected predictions, obtained from our 5S rRNA database (by 10-fold cross-validation procedures, using sample size 1000).
<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0 1 2 3 4 5</td>
</tr>
<tr>
<td>PF</td>
<td>( \text{max}_{BL} = 30 )</td>
<td>0.0009 0.1662 0.3063 0.7580 0.6883 0.9817</td>
</tr>
<tr>
<td>SCFG</td>
<td>( \min_{HL} = 1, \min_{hel} = 1 )</td>
<td>0.0000 0.2855 0.4526 0.9852 0.9974 1.0000</td>
</tr>
<tr>
<td></td>
<td>( \min_{HL} = 1, \min_{hel} = 2 )</td>
<td>0.0017 0.4135 0.5754 0.9861 0.9983 0.9991</td>
</tr>
<tr>
<td></td>
<td>( \min_{HL} = 3, \min_{hel} = 1 )</td>
<td>0.0000 0.3308 0.4883 0.9904 0.9974 1.0000</td>
</tr>
<tr>
<td></td>
<td>( \min_{HL} = 3, \min_{hel} = 2 )</td>
<td>0.0026 0.4509 0.6372 0.9904 0.9974 0.9991</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0 1 2 3 4 5</td>
</tr>
<tr>
<td>PF</td>
<td>( \text{max}_{BL} = 30 )</td>
<td>0.0009 0.7571 3.4207 36.641 30.288 600.35</td>
</tr>
<tr>
<td>SCFG</td>
<td>( \min_{HL} = 1, \min_{hel} = 1 )</td>
<td>0.0000 0.5432 1.1811 20.640 51.834 573.72</td>
</tr>
<tr>
<td></td>
<td>( \min_{HL} = 1, \min_{hel} = 2 )</td>
<td>0.0017 1.1428 2.6615 32.051 64.332 608.06</td>
</tr>
<tr>
<td></td>
<td>( \min_{HL} = 3, \min_{hel} = 1 )</td>
<td>0.0000 0.6651 1.4309 22.983 54.635 569.80</td>
</tr>
<tr>
<td></td>
<td>( \min_{HL} = 3, \min_{hel} = 2 )</td>
<td>0.0026 1.3795 3.1949 36.673 71.080 609.58</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0 1 2 3 4 5</td>
</tr>
<tr>
<td>PF</td>
<td>( \text{max}_{BL} = 30 )</td>
<td>710.75 333.72 237.71 93.335 63.661 7.0951</td>
</tr>
<tr>
<td>SCFG</td>
<td>( \min_{HL} = 1, \min_{hel} = 1 )</td>
<td>999.67 941.77 866.98 336.69 167.10 16.476</td>
</tr>
<tr>
<td></td>
<td>( \min_{HL} = 1, \min_{hel} = 2 )</td>
<td>999.18 884.49 764.79 249.02 129.35 14.198</td>
</tr>
<tr>
<td></td>
<td>( \min_{HL} = 3, \min_{hel} = 1 )</td>
<td>999.93 947.19 874.03 331.75 163.09 15.620</td>
</tr>
<tr>
<td></td>
<td>( \min_{HL} = 3, \min_{hel} = 2 )</td>
<td>999.68 885.81 762.67 239.28 123.91 13.558</td>
</tr>
</tbody>
</table>

Table 16: Results related to the shapes of sampled structures, obtained from our 5S rRNA database (by 10-fold cross-validation procedures, using sample size 1000).
CSP\textsubscript{freq} (selection principle MF struct.):

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PF</td>
<td>0 1 2 3 4 5</td>
</tr>
<tr>
<td></td>
<td>max\textsubscript{BL} = 30</td>
<td>0.0661 0.1255 0.1586 0.2050 0.2183 0.4834</td>
</tr>
<tr>
<td></td>
<td>SCFG</td>
<td>min\textsubscript{HL} = 1, min\textsubscript{hel} = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min\textsubscript{HL} = 1, min\textsubscript{hel} = 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min\textsubscript{HL} = 3, min\textsubscript{hel} = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min\textsubscript{HL} = 3, min\textsubscript{hel} = 2</td>
</tr>
</tbody>
</table>

CSP\textsubscript{freq} (selection principle MEA struct.):

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PF</td>
<td>0 1 2 3 4 5</td>
</tr>
<tr>
<td></td>
<td>max\textsubscript{BL} = 30</td>
<td>0.0660 0.1123 0.1453 0.1984 0.2051 0.4902</td>
</tr>
<tr>
<td></td>
<td>SCFG</td>
<td>min\textsubscript{HL} = 1, min\textsubscript{hel} = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min\textsubscript{HL} = 1, min\textsubscript{hel} = 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min\textsubscript{HL} = 3, min\textsubscript{hel} = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min\textsubscript{HL} = 3, min\textsubscript{hel} = 2</td>
</tr>
</tbody>
</table>

CSP\textsubscript{freq} (selection principle Centroid):

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PF</td>
<td>0 1 2 3 4 5</td>
</tr>
<tr>
<td></td>
<td>max\textsubscript{BL} = 30</td>
<td>0.0793 0.1321 0.1653 0.1917 0.2449 0.5100</td>
</tr>
<tr>
<td></td>
<td>SCFG</td>
<td>min\textsubscript{HL} = 1, min\textsubscript{hel} = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min\textsubscript{HL} = 1, min\textsubscript{hel} = 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min\textsubscript{HL} = 3, min\textsubscript{hel} = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min\textsubscript{HL} = 3, min\textsubscript{hel} = 2</td>
</tr>
</tbody>
</table>

Table 17: Results related to the shapes of selected predictions, obtained from the S-151Rfam database (by 2-fold cross-validation procedures, using sample size 1000).
### CSOfreq:

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>PF</td>
<td>$\max_{BL} = 30$</td>
<td>0.3638</td>
</tr>
<tr>
<td>SCFG</td>
<td>$\min_{HL} = 1, \min_{hel} = 1$</td>
<td>0.2520</td>
</tr>
<tr>
<td></td>
<td>$\min_{HL} = 1, \min_{hel} = 2$</td>
<td>0.2717</td>
</tr>
<tr>
<td></td>
<td>$\min_{HL} = 3, \min_{hel} = 1$</td>
<td>0.2518</td>
</tr>
<tr>
<td></td>
<td>$\min_{HL} = 3, \min_{hel} = 2$</td>
<td>0.2715</td>
</tr>
</tbody>
</table>

### CS_num:

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>PF</td>
<td>$\max_{BL} = 30$</td>
<td>40.390</td>
</tr>
<tr>
<td>SCFG</td>
<td>$\min_{HL} = 1, \min_{hel} = 1$</td>
<td>10.743</td>
</tr>
<tr>
<td></td>
<td>$\min_{HL} = 1, \min_{hel} = 2$</td>
<td>12.968</td>
</tr>
<tr>
<td></td>
<td>$\min_{HL} = 3, \min_{hel} = 1$</td>
<td>12.468</td>
</tr>
<tr>
<td></td>
<td>$\min_{HL} = 3, \min_{hel} = 2$</td>
<td>15.059</td>
</tr>
</tbody>
</table>

### DS_num:

<table>
<thead>
<tr>
<th>Approach</th>
<th>Parameters</th>
<th>Shape Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>PF</td>
<td>$\max_{BL} = 30$</td>
<td>540.74</td>
</tr>
<tr>
<td>SCFG</td>
<td>$\min_{HL} = 1, \min_{hel} = 1$</td>
<td>892.14</td>
</tr>
<tr>
<td></td>
<td>$\min_{HL} = 1, \min_{hel} = 2$</td>
<td>849.32</td>
</tr>
<tr>
<td></td>
<td>$\min_{HL} = 3, \min_{hel} = 1$</td>
<td>888.89</td>
</tr>
<tr>
<td></td>
<td>$\min_{HL} = 3, \min_{hel} = 2$</td>
<td>840.03</td>
</tr>
</tbody>
</table>

Table 18: Results related to the shapes of sampled structures, obtained from the S-151Rfam database (by 2-fold cross-validation procedures, using sample size 1000).
Figure 2:
Figure 2: Comparison of loop profiles for *E. coli* tRNA\(^{\text{Ala}}\). Hplot, Bplot, Iplot, Mplot and Extplot display the probability that an unpaired base lies in a hairpin, bulge, interior, multibranched and exterior loop, respectively. For each considered variant, these five probabilities are computed by a sample of 1000 structures. Results for the PF approach (for \(\text{max}_{BL} = 30\)) are displayed by the thin black lines. For the SCFG approach, we chose \(\text{min}_{hel} = 1\) (thick gray lines) and \(\text{min}_{hel} = 2\) (thick dashed darker gray lines), combined with \(\text{min}_{HL} = 1\) (figures shown on the left) and \(\text{min}_{HL} = 3\) (figures on the right), respectively. The corresponding probabilities for the correct structure of *E. coli* tRNA\(^{\text{Ala}}\) are also displayed (by black points).
Figure 3: Comparison of the (areas under) ROC curves obtained for our tRNA database (computed by 10-fold cross-validation procedures, using sample size 1000). For each considered sampling variant, the corresponding ROC curves are shown for prediction principle MEA structure (figure on the left) and centroid (figure on the right), respectively.
Figure 4: Comparison of the (areas under) ROC curves obtained for our 5SrRNA database (computed by 10-fold cross-validation procedures, using sample size 1000). For each considered sampling variant, the corresponding ROC curves are shown for prediction principle MEA structure (figure on the left) and centroid (figure on the right), respectively.

(a) PF approach (with parameter max_{BL} = 30).

(b) SCFG approach (with the most realistic parameter combination min_{HL} = 3 and min_{hel} = 2).
Figure 5: Comparison of the (areas under) ROC curves obtained for the mixed S-151Rfam database (computed by two-fold cross-validation procedures, using the same folds as in [DWB06] and sample size 1000). For each considered sampling variant, the corresponding ROC curves are shown for prediction principle MEA structure (figure on the left) and centroid (figure on the right), respectively.