Supplementary

S1 A description of the conventional model to obtain the partition function of RNA secondary structures

We show here an abstract form of computing the partition function.

Algorithm S1 An abstract form of calculating the partition function of RNA secondary structures

1: $Z[0] = 1$
2: for $k = 1$ to $N$ do
3: $Z[k] = \sum_{p=0}^{k-1} Z[p] t(k|p) + \sum_{p=0}^{k-2} \sum_{q=p+1}^{k-1} Z[p] Z[q] t(k|p, q) + c_k$
4: end for
5: $Z = Z[N]$

This procedure is quite similar to Algorithm 1 in our paper though we have additional terms in the recursion. As we can see in the McCaskill model, we need such calculations in the case of RNA secondary structures. We can construct Algorithm 3 from this by following the conversion of Algorithm 1 to Algorithm 2 in our paper.
A method to obtain $g_k(\cdot)$ in constant time calculation for distributions of hamming distance from a reference structure

Firstly, we show full description of $g_k(\cdot)$.

$$g_1(i, j) = \sum_{p=1}^{j-1} \sum_{q=p+1}^j S[p][q]$$  \hspace{1cm} (S1)

$$g_2(i, j, k) = \sum_{p=1}^k \sum_{q=k+1}^j S[p][q]$$  \hspace{1cm} (S2)

$$g_3(i, j, k) = \sum_{q=k+1}^j \sum_{p=i}^{q-1} S[p][q]$$  \hspace{1cm} (S3)

$$g_4(i, j) = \sum_{p=1}^{j-1} \sum_{q=p+1}^j S[p][q] + 1 - 2S[i][j]$$  \hspace{1cm} (S4)

$$g_5(i, j, k, l) = \sum_{p=1}^{k-1} \sum_{q=p+1}^j S[p][q] + \sum_{q=k+1}^{l+1} \sum_{p=q}^j S[p][q]$$

$$+ \sum_{p=l+1}^{k-1} \sum_{q=p+1}^j S[p][q] + 1 - 2S[i][j]$$  \hspace{1cm} (S5)

$$g_6(i, j, k) = \sum_{p=k}^{j-1} S[p][j] + \sum_{q=i+1}^j S[i][q]$$

$$+ \sum_{p=i+1}^{k-1} \sum_{q=k}^j S[p][q] + 1 - 2S[i][j]$$  \hspace{1cm} (S6)

$$g_7(i, j, k) = \sum_{p=1}^k \sum_{q=p+1}^j S[p][q]$$  \hspace{1cm} (S7)

$$g_8(i, j, k) = \sum_{p=i}^k S[p][k]$$  \hspace{1cm} (S8)

$$g_9(i, j, k) = g_3(i, j, k)$$  \hspace{1cm} (S9)

Equations (S1) - (S9) represent newly accumulated hamming distance from the reference structure by each transition.

They show that we need $O(n^2)$ calculation for $g_k(\cdot)$ which obtains hamming distance between two structures. This is one of the bottlenecks since these functions are embedded in recursive process. If we pre-calculate a vector $C$ before the recursive process, we obtain $g_k(\cdot)$ by $O(1)$ calculation. Definition of
vector $C$ corresponding to structure vector $S$ is as follows:

$$C[i][j] = \sum_{k=1}^{j-1} \sum_{l=k+1}^{j} S[k][l]$$  \hspace{1cm} (S10)

Let us call it a cumulative structure vector.

$C$ can be computed efficiently by dynamic programming technique:

**Initialization:**

$$C[i][i] = 0 \text{ (1 ≤ i ≤ n)} \hspace{1cm} (S11)$$

$$C[i][i+1] = S[i][i+1] \text{ (1 ≤ i ≤ n - 1)} \hspace{1cm} (S12)$$

**Recursion (1 ≤ i ≤ n - 1, i + 1 < j ≤ n):**

$$C[i][j] = S[i][j] + C[i+1][j] + C[i][j-1] - C[i+1][j-1]$$  \hspace{1cm} (S13)

This pre-calculation requires $O(n^2)$ time.

We wrote down $O(1)$ procedure for $g_k(\cdot)$ below:

$$g_1(i, j) = C[i][j]$$  \hspace{1cm} (S14)

$$g_2(i, j, k) = C[i][j] - C[i][k] - C[k+1][j]$$  \hspace{1cm} (S15)

$$g_3(i, j, k) = C[i][j] - C[i][k]$$  \hspace{1cm} (S16)

$$g_4(i, j) = C[i][j] + 1 - 2S[i][j]$$  \hspace{1cm} (S17)

$$g_5(i, j, k, l) = C[i][j] - C[k][l] + 1 - 2S[i][j]$$  \hspace{1cm} (S18)

$$g_6(i, j, k) = C[i][j] - C[i+1][k-1] - C[k][j-1] + 1 - 2S[i][j]$$  \hspace{1cm} (S19)

$$g_7(i, j, k) = C[i][j] - C[k][j]$$  \hspace{1cm} (S20)

$$g_8(i, j, k) = C[i][j] - C[i][k-1] - C[k][j]$$  \hspace{1cm} (S21)
S3 Pre-calculating the maximum of distance between structures

We can find exact maximum value of hamming distance $d_{\text{max}}$ though it never exceeds sequence length $n$:

$$d_{\text{max}} = \max_{S_c \in \zeta} \left[ \sum_{i=1}^{n-1} \sum_{j=i+1}^n S_r[i][j] \oplus S_c[i][j] \right], \quad (S22)$$

where $\zeta$ is a set of all possible candidate structure vectors and $S_r$ is a reference structure vector.

We construct a $O(n^3)$ dynamic programming procedure so as to obtain $d_{\text{max}}$:

Initialization $(1 \leq i \leq n)$:

$$D_{i,i} = D_{i,i}^1 = D_{i,i}^b = D_{i,i}^m = D_{i,i}^{m-1} = 0 \quad (S23)$$

Recursion $(1 \leq i \leq n-1, i+1 < j \leq n)$:

$$D_{i,j} = \max_k \left\{ D_{i,k} + D_{k+1,j}^1 - C[i][k] - C[k+1][j] \right\} + C[i][j] \quad (S24)$$

$$D_{1,j}^b = \max_k \left\{ D_{1,k}^b - C[i][k] \right\} + C[i][j] \quad (S25)$$

$$D_{i,j}^b = \max_{k,l} \left\{ D_{i+1,k-1}^m + D_{k,j-1}^1 - C[i+1][k-1] - C[k][j-1] \right\}$$

$$+ C[i][j] - 2S_r[i][j] + 1 \quad (S26)$$

$$D_{i,j}^m = \max_k \left\{ D_{i,k-1}^m + D_{1,j}^1 - C[i][k-1] - C[k][j] \right\} + C[i][j], \quad (S27)$$

where $C$ is a vector which is defined in Supplementary section 0.1. We finally have $d_{\text{max}}$ as $D_{1,n}$.
S4 A time-saving method for calculating $d_{5'-3'}$ exact distributions

We show a time-saving procedure for calculating $d_{5'-3'}$ exact distributions.

**Algorithm S2** Exact calculation of a $d_{5'-3'}$ distribution by DFT approach

1: $O(n^3)$ pre-calculation for $Z_{i;j}^b$, $(1 \leq i < j \leq n)$ by McCaskill model
2: for $k = 1$ to $n - 1$ do
3: $x = \exp\left(\frac{2\pi i k}{n+1}\right)$
4: $O(n^3)$ recursions for $Z_{1,n}[k]$ described on equations (S28) - (S31)
5: end for
6: for $k = 1$ to $n - 1$ do
7: $p(d_{5'-3'} = k) = \left\{\sum_{r=1}^{n-1} Z_{1,n}[r] \left(\cos\left(\frac{2\pi(r-1)}{n-1}\right) - i \sin\left(\frac{2\pi(r-1)}{n-1}\right)\right)\right\} / (n - 1)$
8: end for

Recursions implied above are as follows:

**Initialization** $(1 \leq i \leq n)$:

\[
\begin{align*}
Z_{i,i}[k] &= 1.0 & (S28) \\
Z_{i,i}^1[k] &= 0 & (S29)
\end{align*}
\]

**Recursion** $(1 \leq i < j \leq n)$:

\[
\begin{align*}
Z_{i,j}[k] &= x^{j-i} + \sum_{l=i}^{j-1} Z_{i,l}[k]Z_{l+1,j}^1[k]x \\
Z_{i,j}^1[k] &= \sum_{k=i+1}^{j} Z_{i,k}^b x^{j-k+1} & (S31)
\end{align*}
\]
S5  A framework of algorithm for the distribution of two-dimensional integer vector

In Algorithm S3, we show a method to expand our original algorithm to two dimensions.

Algorithm S3 2D expansion of the original model in Algorithm 4.

1: /* DP phase (distributed processing is available) */
2: for $S_1 = 0$ to $S_{1_{\text{max}}}$ do
3:   for $S_2 = 0$ to $S_{2_{\text{max}}}$ do
4:     $x = \left( \exp \left[ 2\pi i \frac{S_1}{S_{1_{\text{max}}+1}} \right], \exp \left[ 2\pi i \frac{S_2}{S_{2_{\text{max}}+1}} \right] \right)$
5:     $Z[S_1][S_2][0] = 1$
6: for $k = 1$ to $N$ do
7:     $Z[S_1][S_2][k] = \sum_{p=0}^{k-1} Z[S_1][S_2][p]t(k|p)x_1^{s_1(p,k)}x_2^{s_2(p,k)}$
8:     \[\sum_{p=0}^{k-2} \sum_{q=p+1}^{k-1} Z[S_1][S_2][p]Z[S_1][S_2][q]t(k|p,q)x_1^{s_1(p,q,k)}x_2^{s_2(p,q,k)}\]
9:     $\zeta_{S_1,S_2} = Z[S_1][S_2][N]$
10: end for
11: end for
12: /* DFT phase */
13: for $S_1 = 0$ to $S_{1_{\text{max}}}$ do
14:   for $S_2 = 0$ to $S_{2_{\text{max}}}$ do
15:     $z'_{S_1,S_2} = \left\{ \sum_{r=0}^{S_{2_{\text{max}}}} \zeta_{S_1,r} \exp \left[ -2\pi i \frac{rS_2}{S_{2_{\text{max}}+1}} \right] \right\} / (1 + S_{2_{\text{max}}})$
16:   end for
17: end for
18: for $S_2 = 0$ to $S_{2_{\text{max}}}$ do
19:   for $S_1 = 0$ to $S_{1_{\text{max}}}$ do
20:     $z_{S_1,S_2} = \left\{ \sum_{r=0}^{S_{1_{\text{max}}}} z'_{r,S_2} \exp \left[ -2\pi i \frac{rS_1}{S_{1_{\text{max}}+1}} \right] \right\} / (1 + S_{1_{\text{max}}})$
21:   end for
22: end for
23: $Z = \sum_{S_1=0}^{S_{1_{\text{max}}}} \sum_{S_2=0}^{S_{2_{\text{max}}}} z_{S_1,S_2}$

where $x = (x_1, x_2)$, subscripts of 1 or 2 represent that they are variables or functions for the first and second components of a two-dimensional score vector respectively, and $P_S$ means the probability that RNA sequence folds into a structure whose score is $(S_1, S_2)$.
We have $p_S$, the probability of obtaining score $S$, by the following equation:

$$p_S = \frac{z_S}{Z} \quad (S32)$$
A concrete description of recursions for the distribution of hamming distance from two reference structures

We show here concrete recursions to obtain the distribution of hamming distance from two reference structures. Naive expansion is quite simple; all we have to do is just exchanging $x^{g_k(i)}$ for $x_1^{g_k(i)} x_2^{g_k(i)}$ such as:

$$Z_{i,j} = x_1^{g_1(i,j)} x_2^{g_2(i,j)} + \sum_{k=1}^{j-1} Z_{i,k} Z_{k+1,j} x_1^{g_1(i,j,k)} x_2^{g_2(i,j,k)} \quad (S33)$$

However, there are many meaningless calculations which can be cut by utilizing sparseness of the distribution. From constraints such as triangle inequality, we must satisfy the following expressions:

$$\forall S \in \zeta, \quad |d(S_{R_1}, S) - d(S_{R_2}, S)| \leq d(S_{R_1}, S) \leq d(S_{R_1}, S_{R_2}) + d(S_{R_2}, S) \quad (S34)$$

$$\forall S \in \zeta, \quad d(S_{R_1}, S) \leq d_{1\max} \quad (S35)$$

$$\forall S \in \zeta, \quad d(S_{R_2}, S) \leq d_{2\max} \quad (S36)$$

$$\forall S \in \zeta, \quad \exists m \in \mathbb{N}, \quad d(S_{R_1}, S) + d(S_{R_2}, S) + d(S_{R_1}, S_{R_2}) = 2m \quad (S37)$$

where $\mathbb{N}$ is a set of natural numbers, $\zeta$ is a set of all possible secondary structure vectors, $S_{R_i} (i = 1, 2)$ is a structure vector of $i$-th reference, and $d(S_1, S_2)$ means hamming distance between $S_1$ and $S_2$. Equation (S34) is derived from triangle inequality, and equation (S35) and (S36) are originated in definitions of $d_{1\max}$ and $d_{2\max}$. A reason of equation (S37) is a little complicated. To put it simply, 1 bit transition of structure vector of $S$ invariably causes 1 hamming distance changing from any other structures, and every structure vector can visit each other by repetition of 1 bit transitions.

We modified our algorithm to reduce meaningless calculation by converting axes. Abstract form is shown in Algorithm S4.
Algorithm S4 Improved 2D algorithm by DFT approach

1: \( \delta = d(S_{R_1}, S_{R_2}) \)
2: \( d'_{1\max} = \delta \)
3: \( d'_{2\max} = \frac{d_{1\max} + d_{2\max} - \delta}{2} \)
4: for \( d_1 = 0 \) to \( d'_{1\max} \) do
5: for \( d_2 = 0 \) to \( d'_{2\max} \) do
6: \( x = \exp \left( 2\pi i \frac{d_1}{d'_{1\max} + 1} \right) \)
7: \( y = \exp \left( 2\pi i \frac{d_2}{d'_{2\max} + 1} \right) \)
8: \( Z[d_1][d_2][0] = 1 \)
9: for \( k = 1 \) to \( N \) do
10: \( Z[d_1][d_2][k] = c_k x_s(k) y_s(k) + \sum_{i=0}^{k-2} \sum_{j=i+1}^{k-1} Z[d_1][d_2][i] Z[d_1][d_2][j] t(k, j, i, j) x_s(i,j,k) y_s(i,j,k) \)
11: \( \quad + \sum_{i=0}^{k-1} Z[d_1][d_2][i] t(k, i) x_s(i,k) y_s(i,k) \)
12: end for
13: end for
14: for \( d_1 = 0 \) to \( d'_{1\max} \) do
15: for \( d_2 = 0 \) to \( d'_{2\max} \) do
16: \( Z'[d_1][d_2] = \left\{ \sum_{r=0}^{d'_{1\max}} Z[d_1][r][N] \exp \left[ -2\pi i \frac{r d_2}{d'_{2\max} + 1} \right] \right\} / (1 + d'_{2\max}) \)
17: end for
18: end for
19: for \( d_2 = 0 \) to \( d'_{2\max} \) do
20: for \( d_1 = 0 \) to \( d'_{1\max} \) do
21: \( Z''[d_1][d_2] = \left\{ \sum_{r=0}^{d'_{1\max}} Z'[r][d_2] \exp \left[ -2\pi i \frac{r d_1}{d'_{1\max} + 1} \right] \right\} / (1 + d'_{1\max}) \)
22: end for
23: end for
24: for \( d_1 = 0 \) to \( d'_{1\max} \) do
25: for \( d_2 = 0 \) to \( d'_{2\max} \) do
26: if \( (|d_1 - d_2| \leq \delta \leq d_1 + d_2) \) and \( (d_1 + d_2 + \delta \) is even) then
27: \( p(d_1, d_2) = Z''[d_1 - d_2 + \delta][d_1 + d_2 - \delta] \)
28: else
29: \( p(d_1, d_2) = 0 \)
30: end if
31: end for
32: end for
33: end for
In this case, we need a little modification to recursions as follows:

\[ Z_{i,j} = x \left( g_{i,j}^1 - g_{i,j}^2 + \Delta t_1(i,j) \right) \]

\[ + \sum_{k=1}^{j-1} Z_{i,k} Z_{k+1,j} x \left( g_{i,k}^1 - g_{i,k}^2 + \Delta t_2(i,k) \right) \]

\[ + \left( \sum_{k=1}^{j} \sum_{l=k+1}^{j} Z_{i,k} Z_{l,j} \left( g_{i,k}^1 - g_{i,k}^2 + \Delta t_2(i,k) \right) \right) \]

\[ (S38) \]

\[ Z_{i,j}^1 = \sum_{k=i+1}^{j} Z_{i,k} Z_{k+1,j} \left( g_{i,k}^1 - g_{i,k}^2 + \Delta t_2(i,k) \right) \]

\[ (S39) \]

\[ Z_{i,j}^h = \sum_{k=i+1}^{j-1} Z_{i,k} Z_{k+1,j} \left( e^{-f_3(i,j) / kbT} \right) x \left( g_{i,k}^1 - g_{i,k}^2 + \Delta t_2(i,k) \right) \]

\[ + \sum_{k=i+1}^{j-1} \sum_{l=k+1}^{j} Z_{i,k} Z_{k+1,l} \left( e^{-f_3(i,j) / kbT} \right) x \left( g_{i,k}^1 - g_{i,k}^2 + \Delta t_2(i,k) \right) \]

\[ + e^{-f_3(i,j) / kbT} x \left( g_{i,j}^1 - g_{i,j}^2 + \Delta t_2(i,j) \right) \]

\[ (S40) \]

\[ Z_{i,j}^m = \sum_{k=1}^{j} \left( e^{-f_3(k-1) / kbT} \right) x \left( g_{i,k}^1 - g_{i,k}^2 + \Delta t_2(i,k) \right) \]

\[ + Z_{i,k} Z_{1,k-1} \left( e^{-f_3(k) / kbT} \right) x \left( g_{i,k}^1 - g_{i,k}^2 + \Delta t_2(i,k) \right) \]

\[ (S41) \]

\[ Z_{i,j}^{m1} = \sum_{k=i+1}^{j} Z_{i,k} e^{-f_3(j-k) / kbT} x \left( g_{i,k}^1 - g_{i,k}^2 + \Delta t_2(i,k) \right) \]

\[ (S42) \]

where \( g_k^1() \) is \( g_k^2() \) whose reference structure vector is \( S_{R_k} \), and \( \Delta \delta_k() \) are:

\[ \Delta \delta_1(i,j) = E[i][j] \]

\[ \Delta \delta_2(i,j,k) = E[i][j] - E[i][k] - E[k+1][j] \]

\[ \Delta \delta_3(i,j,k) = E[i][j] - E[i][k] \]

\[ \Delta \delta_4(i,j) = \Delta \delta_3(i,j) \]

\[ \Delta \delta_5(i,j,k,l) = E[i][j] - E[k][l] \]

\[ \Delta \delta_6(i,j,k) = E[i][j] - E[i+1][k-1] - E[k][j-1] \]

\[ \Delta \delta_7(i,j,k) = E[i][j] - E[k][j] \]

\[ \Delta \delta_8(i,j,k) = E[i][j] - E[i][k-1] - E[k][j] \]

\[ \Delta \delta_9(i,j,k) = \Delta \delta_8(i,j,k) \]
\( E[i][j] \) is defined by the hamming distance of partial structure vector:

\[
E[i][j] = \sum_{p=i}^{j-1} \sum_{q=p+1}^{j} S_{R_1}[p][q] \oplus S_{R_2}[p][q] \tag{S52}
\]

We have \( E[i][j] \) effectively by the following recursions:

 Initialization:

\[
E[i][i] = 0 \quad (1 \leq i \leq n) \tag{S53}
\]

\[
E[i][i+1] = S_{R_1}[i][i+1] \oplus S_{R_2}[i][i+1] \quad (1 \leq i \leq n-1) \tag{S54}
\]

 Recursion \((1 \leq i \leq n-1, i+1 < j \leq n)\):

\[
E[i][j] = S_{R_1}[i][j] \oplus S_{R_2}[i][j] + E[i+1][j] + E[i][j-1] - E[i+1][j-1] \tag{S55}
\]

Meaning of \( \Delta \delta_k(\cdot) \) is newly accumulated hamming distance between two reference structures by each transition, and it indicates the lower limit of \( g^1_k(\cdot) + g^2_k(\cdot) \).

This formulation empirically contributes increase in speed several times though, of course, it depends on RNA sequence and reference structures. At least, we guarantee twice as fast as the original algorithm. Original and modified algorithms require \( d_{1\text{max}}d_{2\text{max}} \) and \( \frac{\delta(d_{1\text{max}}+d_{2\text{max}}-\delta)}{2} \)-time continuous calculations respectively. Thus, our modification contributes the following acceleration rate \( r \):

\[
r = \frac{d_{1\text{max}}d_{2\text{max}}}{\delta(d_{1\text{max}}+d_{2\text{max}}-\delta)} = \frac{2d_{1\text{max}}d_{2\text{max}}}{\delta(d_{1\text{max}}+d_{2\text{max}}-\delta)} \tag{S56}
\]

\( r \) is a monotonic decrease function of \( \delta \) from \( \delta = 0 \) to \( \delta = \frac{d_{1\text{max}}+d_{2\text{max}}}{2} \):

\[
\frac{\partial r}{\partial \delta} = \frac{2\delta - d_{1\text{max}} - d_{2\text{max}}}{(\delta(d_{1\text{max}}+d_{2\text{max}}-\delta))^2} \leq 0 \quad (0 < \delta \leq \frac{d_{1\text{max}}+d_{2\text{max}}}{2}) \tag{S57}
\]

On the other hand, we have the following inequality from equations (S35) - (S36) and the property of arithmetic mean:

\[
\delta \leq \min(d_{1\text{max}}, d_{2\text{max}}) \leq \frac{d_{1\text{max}}+d_{2\text{max}}}{2} \tag{S58}
\]

Accordingly, we obtain the lower limit of \( r \) as follows:

\[
r = \frac{2d_{1\text{max}}d_{2\text{max}}}{\delta(d_{1\text{max}}+d_{2\text{max}}-\delta)} \geq \frac{2d_{1\text{max}}d_{2\text{max}}}{\min(d_{1\text{max}}, d_{2\text{max}})(d_{1\text{max}}+d_{2\text{max}}-\min(d_{1\text{max}}, d_{2\text{max}}))} = 2 \tag{S59}
\]

11
S7 Explosion of the number of possible structures

We have a massive number of possible structures with the distances from the reference structure. It is due to the combinatorial explosion of possible base pairs. Fig. S1 shows the number of structures of each Hamming distance from the reference.

Figure S1: The number of possible structures