#187　メンバー / TakumiOtagaki / 研究進捗 /

WIP jax-capr

Updated at 2025-10-22 16:54

はじめに

実装力がないと言われてクッソーと思ったのでMax Wardらに任せるのではなく、彼らの `jax-rnafold` を利用してjax-caprを実装してやろうじゃないか。と考えています。

jax-rnafold の dp table

実装で利用されているのは

- $\xi(i)$: 実質 $Z(i,n)$
- $P(i,j)$: 実質 $Z^b(i,j)$
- $M(p,i,j)$: Z^m と Z^{m1} を併せ持っている（ここがちょっとややこしい）
  - i, j がマルチループの中にあって、その中に 少なくとも p 個の塩基対が存在するような全ての部分構造 $\sigma[i;j]$ のボルツマンファクターの和
  - i, j による塩基対は含めない。

また、outside algorithm が未実装であるという課題があるが、これは落ち着いてやればできそう。
forward の漸化式から半自動的に outside を導出できる.

forward 漸化式 in jax-rnafold

$$
\begin{aligned}
\xi(i+1) &= \xi(i)\cdot \exp(-\beta\cdot 0) \\
&\quad + \sum_{\substack{j \\ j<i}} \bar{\xi}(j) P(i,j) \\
P(i,j) &= B(f_1(i,j)) + \sum_{i<h<l<j} P(h,l) B(f_2(h,l)) \\
&\quad + M(2, i+1, j-1) B(M_c) B(M_p) \\
M(p,i,j) &= M(p, i+1, j) B(M_u) \\
&\quad + \sum_{i<k<j} P(i,k) B(M_p) M(\max(0,p-1), k+1, j)
\end{aligned}
$$

ここで $\bar{\xi}$ のようにバーをつけることでbackwardを表すことにすると

$$\bar{\xi}(i) = \bar{\xi}(i-1) + \sum_{\substack{j \\ j<i-1}} \bar{\xi}(j) P(i-1, j)$$

$$\begin{aligned}
\bar{P}(h,l) &= \bar{\xi}(h)\xi(l+1) \\
&\quad + \sum_{\substack{i,j \\ i<h,l<j}} \bar{P}(i,j) \left(\begin{array}{l} B(f_2(i,j,h,l)) \\ +B(M_c + M_p) \left\{ \begin{array}{l} +M(1,i+1,h-1) \cdot B(M_u(j-l-1) \\ +M(1,l+1,j-1) \cdot B(M_u(h-i-1) \\ +M(1,i+1,h-1) \cdot M(1,l+1,j-1) \end{array} \right\} \end{array} \right) \\
&= \bar{\xi}(h)\xi(l+1) \\
&\quad + \sum_{\substack{i,j \\ i<h,l<j}} \bar{P}(i,j) \left(\begin{array}{l} B(f_2(i,j,h,l)) \\ +B(M_c + M_p) \left[\begin{array}{l} M(1,i+1,h-1) \cdot \{B(M_u(j-l-1)) + M(1,l+1,j-1)\} \\ +M(1,l+1,j-1) \cdot B(M_u(h-i-1)) \end{array} \right] \end{array} \right)
\end{aligned}$$

のようにしてかける。
また、懸念として $B(f_2)$ の計算で internal outer mismatch を考慮する際に、少し散らかった式が出てくるだろうということが挙げられる。これは実装の際にもう少し詳しくみてみることにする。

以下のようにして計算量を下げることができる。

$$\bar{P}^m(i,l) := \sum_{\substack{j \\ l<j}} \bar{P}(i,j) M(1, l+1, j-1)$$

$$\bar{P}^{m1}(i,l) := \sum_{\substack{j \\ l<j}} \bar{P}(i,j) B(M_u(j-l-1))$$

これらはいずれも より長い領域の計算が終わっていればすぐに計算できるし、$O(n^3)$ で全て計算することができる。

これらを用いることで、元々の $\bar{P}(h,l)$ は以下のように書き直すことができる。すなわち、

$$
\begin{aligned}
\bar{P}(h,l) &= \bar{\xi}(h)\xi(l+1) \\
&\quad + \sum_{\substack{i,j \\ i<h,l<j}} \bar{P}(i,j) \left(\begin{matrix} B(f_2(i,j,h,l)) \\ +B(M_c+M_p) \end{matrix} \begin{bmatrix} M(1,i+1,h-1)\cdot\{B(M_u(j-l-1))+M(1,l+1,j-1)\} \\ +M(1,l+1,j-1)\cdot B(M_u(h-i-1)) \end{bmatrix} \right) \\
&= \bar{\xi}(h)\xi(l+1) + \sum_{\substack{i,j \\ i<h,l<j}} B(f_2(i,j,h,l)) \\
&\quad + B(M_c+M_p) \sum_{\substack{i \\ i<h}} \sum_{\substack{j \\ l<j}} \begin{bmatrix} \bar{P}(i,j)M(1,i+1,h-1)B(M_u(j-l-1)) \\ +\bar{P}(i,j)M(1,i+1,h-1)M(1,l+1,j-1) \\ +\bar{P}(i,j)M(1,l+1,j-1)B(M_u(h-i-1)) \end{bmatrix} \\
&= \bar{\xi}(h)\xi(l+1) + \sum_{\substack{i,j \\ i<h,l<j}} B(f_2(i,j,h,l)) \\
&\quad + B(M_c+M_p) \sum_{\substack{i \\ i<h}} \begin{bmatrix} M(1,i+1,h-1)\bar{P}^{m1}(i,l) \\ +M(1,i+1,h-1)\bar{P}^{m}(i,l) \\ +B(M_u(h-i-1))\bar{P}^{m}(i,l) \end{bmatrix} \\
&= \bar{\xi}(h)\xi(l+1) + \sum_{\substack{i,j \\ i<h,l<j \\ h-i-1+j-l-1\le 30}} B(f_2(i,j,h,l)) \\
&\quad + B(M_c+M_p) \sum_{\substack{i \\ i<h}} \begin{bmatrix} M(1,i+1,h-1)\bar{P}^{m1}(i,l) \\ +\{M(1,i+1,h-1)+B(M_u(h-i-1))\}\bar{P}^{m}(i,l) \end{bmatrix}
\end{aligned}
$$

となって、これは $O(n^3)$ で全て計算することができる。
この方向性で実装を進めれば、待望の outside partition function $P(i,j)$ を手に入れることができる...

これをベクトル化するためには 同じ配列長さ d となる任意の h,l つまり

$$
{}^{\forall}(h,l),(h',l'),\ \text{s.t. } l-h-1=l'-h'-1(=d),(h,l)\neq(h',l')
$$

に対して $\bar{P}(h,l)$ と $\bar{P}(h',l')$ の計算は互いに依存しないため、それぞれを vectorization することができる。
さらに、各 $\bar{P}(h,l)$ の計算の中で登場する $\sum_i$ に関しても vectorization することで効率的に和を計算することができる。
これらのベクトル化をかませば、対角線のサイズ $O(n)$ だけの計算時間で計算を終わらせることができる（メモリが無限にある限り）。

続いて、M についても outside algorithm を走らせる必要があって、

$$
\begin{aligned}
M(2,i,j) &= M(2,i+1,j)B(M_u) + \sum_{i<k<j} P(i,k)B(M_p)M(1,k+1,j) \\
M(1,i,j) &= M(1,i+1,j)B(M_u) + \sum_{i<k<j} P(i,k)B(M_p)M(0,k+1,j) \\
M(0,i,j) &= M(0,i+1,j)B(M_u) + \sum_{i<k<j} P(i,k)B(M_p)M(0,k+1,j)
\end{aligned}
$$

であって $M(2,i,j)$ は P の漸化式でも出てくることから、以下のように書き下すことができる。

$$\bar{M}(2,h,l) = \bar{M}(2,h-1,l)B(M_u) + \bar{P}(h,l)B(M_c + M_p)$$
$$\bar{M}(1,h,l) = \bar{M}(1,h-1,l)B(M_u) + \sum_{i<h-1} P(i,h-1)B(M_p)\bar{M}(2,i,l)$$
$$\bar{M}(0,h,l) = \bar{M}(0,h-1,l)B(M_u) + \sum_{i<h-1} P(i,h-1)B(M_p)\left(\bar{M}(0,i,l) + \bar{M}(1,i,l)\right)$$

のように書き下すことができる。

これにて、$\boldsymbol{\xi}, \boldsymbol{P}, \boldsymbol{M}$ それぞれの outside algorithm が仕上がった。あとは実装。

ここまで丁寧な情報整理ができていればすぐに実装できる気がする。