MONI: A Pangenomics Index for Finding MEMs Supplementary Material

Massimiliano Rossi[1], Marco Oliva[1], Ben Langmead[2], Travis Gagie[3⋆], and Christina Boucher[1⋆]

[1] Department of Computer and Information Science and Engineering,
University of Florida, Gainesville, FL,
rossi.m@ufl.edu, marco.oliva@ufl.edu, christinaboucher@ufl.edu,

[2] Department of Computer Science,
Johns Hopkins University, Baltimore, MD,
langmea@cs.jhu.edu,

[3] Faculty of Computer Science,
Dalhousie University, Halifax, Canada
travis.gagie@dal.ca,

1 Computation of the BWT given the prefix-free parsing

We provide the pseudocode for the construction algorithm of Kuhnle et al. (2020) which is shown in Algorithm 1. We first provide some additional notation. Given the prefix-free parsing P of S with dictionary D, we let $\mathcal{S}$ be the set of the distinct proper phrase suffixes of S of length at least w. Given a proper phrase suffix $\alpha \in \mathcal{S}$, we denote by $\mathcal{L}_\alpha$, the set of characters that precede all occurrences of α in S. We let $\mathcal{I}_\alpha$ be the set of positions of the phrases whose suffix is α in the BWT_P, and let α_{First} and α_{Last} the first and the last position in $\mathcal{I}_\alpha$. In addition, we denote the number of occurrences of α in BWT_P as $occs(\alpha)$, i.e. $occs(\alpha) = |\mathcal{I}_\alpha|$. Lastly, given a proper phrase suffix α and an index $i \in \mathcal{I}_\alpha$, we denote by $\alpha_i.\ell = \alpha.\ell$ the length $|\alpha|$ of α and by $\alpha_i.bwt$ as the character preceding α in the phrase. We refer to the character preceding all occurrences of α as $\alpha_i.bwt$ when $|\mathcal{L}_\alpha| = 1$.

Algorithm 1 Building RL BWT

```
 1: procedure BUILD RL BWT($P, D$)
 2:     $i \leftarrow 0$
 3:     for all $\alpha \in \mathcal{S}$ do
 4:         if $|\mathcal{L}_\alpha| = 1$ then
 5:             $i \leftarrow$ UPDATE RL BWT($i, \alpha.bwt, occs(\alpha)$)
 6:         else
 7:             for all $k \in \mathcal{I}_\alpha$ do
 8:                 $i \leftarrow$ UPDATE RL BWT($i, \alpha_k.bwt, 1$)
 9:     return RL BWT[$1..r$]

10: procedure UPDATE RL BWT($i, a, \ell$)
11:     if $a \neq$ RL BWT[$i$].$head$ then
12:         $i \leftarrow i + 1$
13:         RL BWT[$i$].$head \leftarrow a$
14:         RL BWT[$i$].$\ell \leftarrow 0$
15:     RL BWT[$i$].$\ell \leftarrow$ RL BWT[$i$].$\ell + \ell$
16:     return $i$
```

⋆ Both authors should be considered senior authors of the project.

2 Computation of the BWT and the thresholds given the prefix-free parsing

In this section we provide the pseudocode for the construction algorithm of Kuhnle et al. (2020) modified to compute in addition the threshold values. The pseudocode is shown in Algorithm 2.

Algorithm 2 Building RL BWT and *Thresholds*

```
 1: procedure BUILD THRESHOLDS($P, D$)
 2:     $i \leftarrow 0, j \leftarrow 1, \beta \leftarrow \varepsilon$
 3:     for all $\alpha \in \mathcal{S}$ do
 4:         $val \leftarrow \texttt{lcp}(\alpha, \beta)$
 5:         UPDATE LCP($val, j$)
 6:         if $|\mathcal{L}_\alpha| = 1$ then
 7:             $i \leftarrow$UPDATE RL BWT($i, \alpha.bwt, occs(\alpha)$)
 8:             UPDATE LCP($val, j$)
 9:             $j \leftarrow j + occs(\alpha)$
10:         else
11:             $prev \leftarrow -1$
12:             for all $k \in \mathcal{I}_\alpha$ do
13:                 if $prev \geq 0$ then
14:                     $val \leftarrow$MIN SLCP($prev, k$)
15:                     $val \leftarrow val + \alpha.\ell - w$
16:                 UPDATE LCP($val, j$)
17:                 $i \leftarrow$UPDATE RL BWT($i, \alpha_k.bwt, 1$)
18:                 UPDATE LCP($val, j$)
19:                 $j \leftarrow j + 1$
20:                 $prev \leftarrow k$
21:         $\beta \leftarrow \alpha$
22:     return RL BWT[1..$r$], THR[1..$r$]

23: procedure UPDATE LCP($val, pos$)
24:     if $val <$ LCP.$val$ then
25:         LCP.$val \leftarrow val$
26:         LCP.$pos \leftarrow pos$

27: procedure UPDATE RL BWT($i, c, \ell$)
28:     if $c \neq$ RL BWT[$i$].$head$ then
29:         UPDATE THRESHOLDS($i, c$)
30:         $i \leftarrow i + 1$
31:         RL BWT[$i$].$head \leftarrow c$
32:         RL BWT[$i$].$\ell \leftarrow 0$
33:         LCP $\leftarrow$INIT LCP
34:     RL BWT[$i$].$\ell \leftarrow$ RL BWT[$i$].$\ell + \ell$
35:     return $i$

36: procedure MIN SLCP($a, b$)
37:     if $b > a$ then
38:         SWAP($a, b$)
39:     return min{SLCP[$i$] | $a < i \leq b$}

40: procedure UPDATE THRESHOLDS($i, c$)
41:     for all $a \in \Sigma \setminus$ {RL BWT[$i$].$head$} do
42:         if LCP.$val <$ M[$a$].$val$ then
43:             M[$a$].$val \leftarrow$ LCP.$val$
44:             M[$a$].$pos \leftarrow$ LCP.$pos$
45:     if $c$ is the first $c$ in RL BWT then
46:         THR[$i$].$val \leftarrow 0$
47:         THR[$i$].$pos \leftarrow 0$
48:     else
49:         THR[$i$].$val \leftarrow$ M[$c$].$val$
50:         THR[$i$].$pos \leftarrow$ M[$c$].$pos$
51:     M[$c$] $\leftarrow$INIT M
```

Bibliography

Alan Kuhnle, Taher Mun, Christina Boucher, Travis Gagie, Ben Langmead, and Giovanni Manzini. Efficient construction of a complete index for pan-genomics read alignment. *Journal of Computational Biology*, 27(4):500–513, 2020.