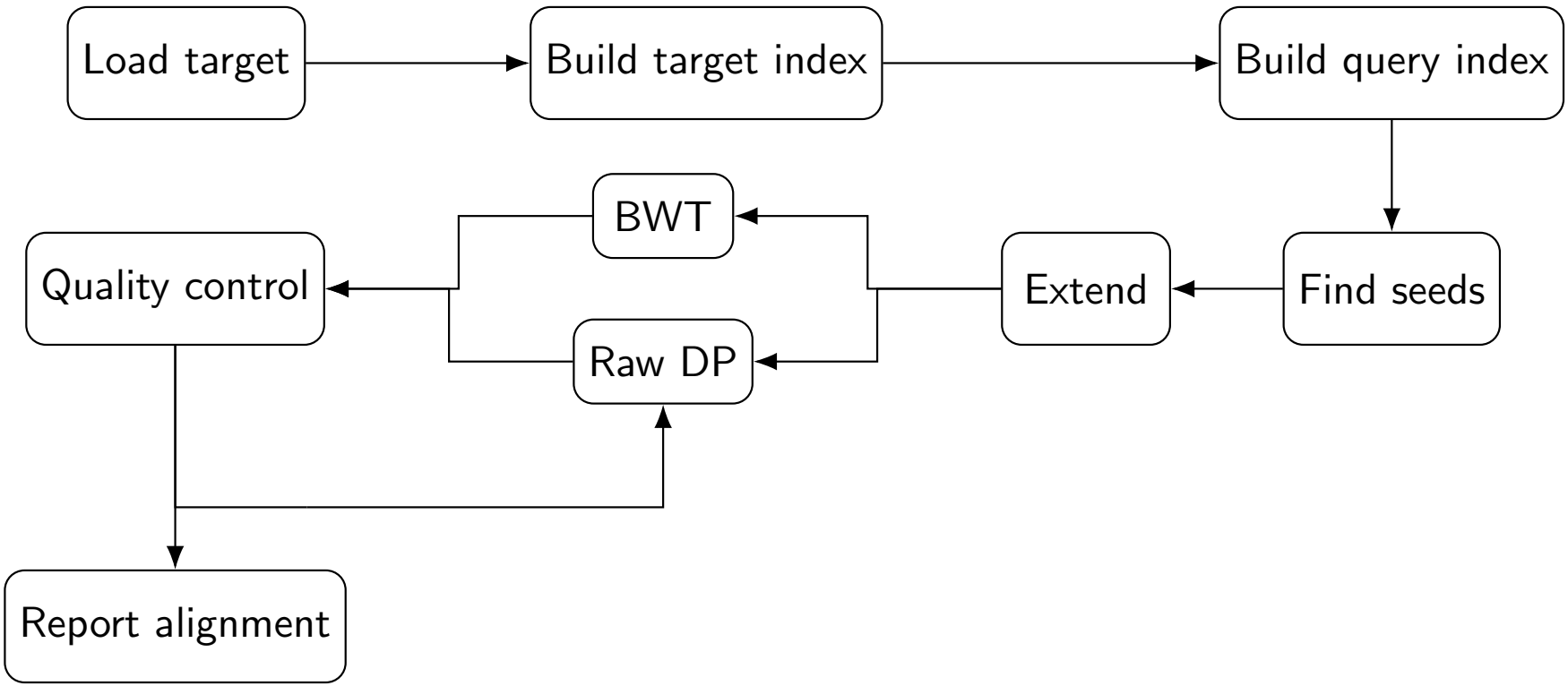
Fast Python sequence aligner

Piotr Styczyński

MIM UW

2024

Seed and Extend approach



Seed extension

Algorithm Standard LIS construction O(n log n)

Require: $n \geq 0$
  $lis\_len \leftarrow 0$ ▷*Length of LIS*
  $parent \leftarrow \{\infty, \infty, \infty, ..., \infty\}_{n+1}$ ▷*Mapping to reconstruct LIS*
  $sub \leftarrow \{\infty, \infty, \infty, ..., \infty\}_{n+1}$ ▷*Array with indices for matches that form LIS*
  $i \leftarrow 0$
  while $i < n$ **do** ▷*Iterate over all elements* $i = 0, 1, 2..., n-1$
    $start \leftarrow 1$
    $end \leftarrow lis\_len$
    while $start \leq end$ **do** ▷*Binary search over existing longest sequence*
      $middle \leftarrow \left\lfloor \frac{start+end}{2} \right\rfloor$
      if $matches_q[sub[middle]] < matches_q[i]$ **then**
        $start \leftarrow middle + 1$
      else
        $start \leftarrow middle - 1$
    $parent[i] \leftarrow sub[start - 1]$ ▷*We pin current value to the found parent*
    $sub[start] \leftarrow i$
    if $start > lis\_len$ **then**
      $lis\_len = start$
    $i \leftarrow i + 1$

Seed extension

Algorithm Reconstruct LIS by following parent array O(n)

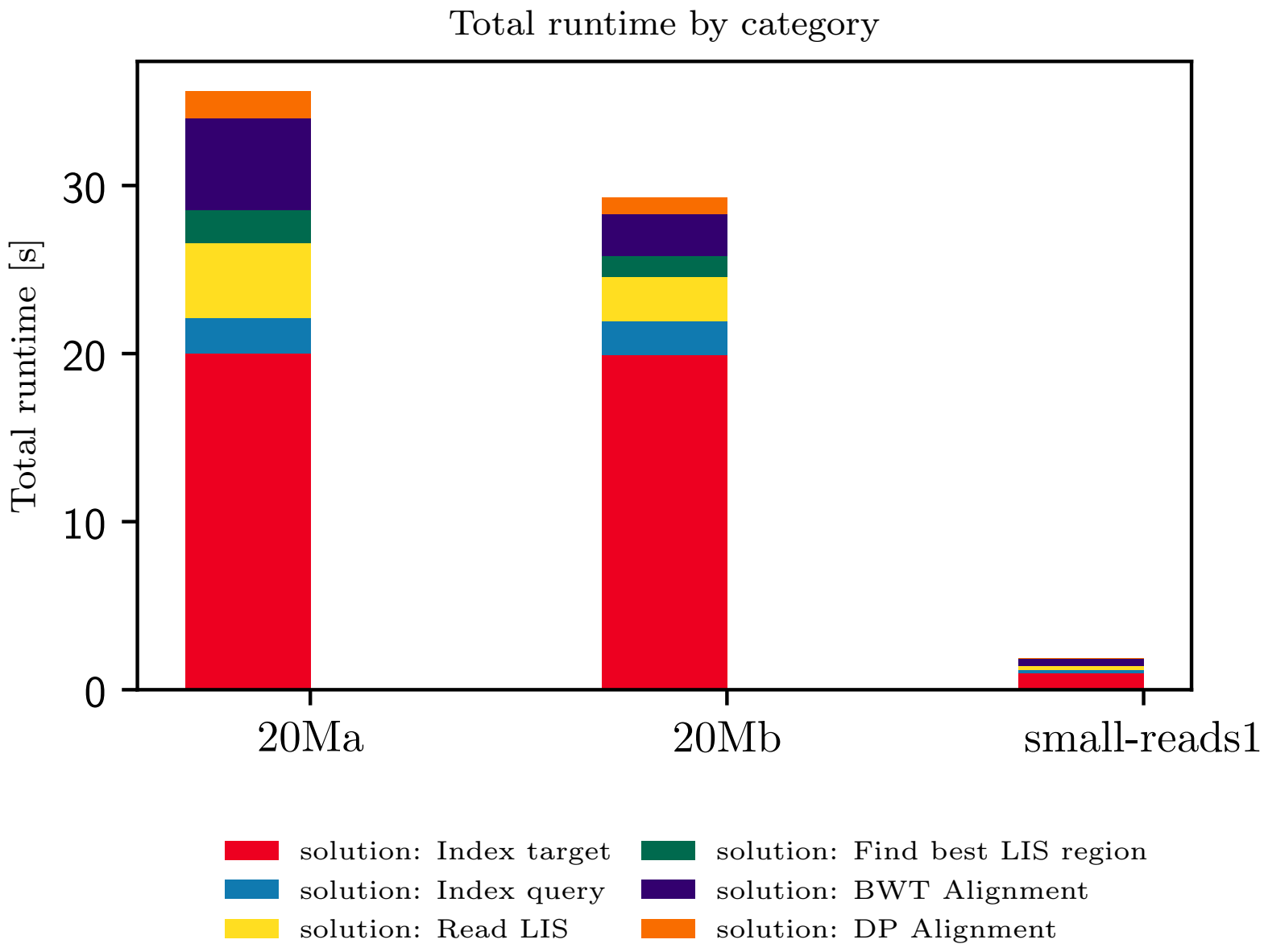
$current\_node \leftarrow sub[lis\_len]$
$result \leftarrow \{0, 0, 0, ..., 0\}_{lis\_len}$
$result[lis\_len - 1] \leftarrow current\_node$ ▷*Will contain all indices from matches describing the output subsequence*
$j \leftarrow lis\_len - 1$
while $1 \leq j$ **do**
    $current\_node \leftarrow parent[current\_node]$
    $result[j - 1] \leftarrow current\_node$
    $j \leftarrow j - 1$

Seed extension

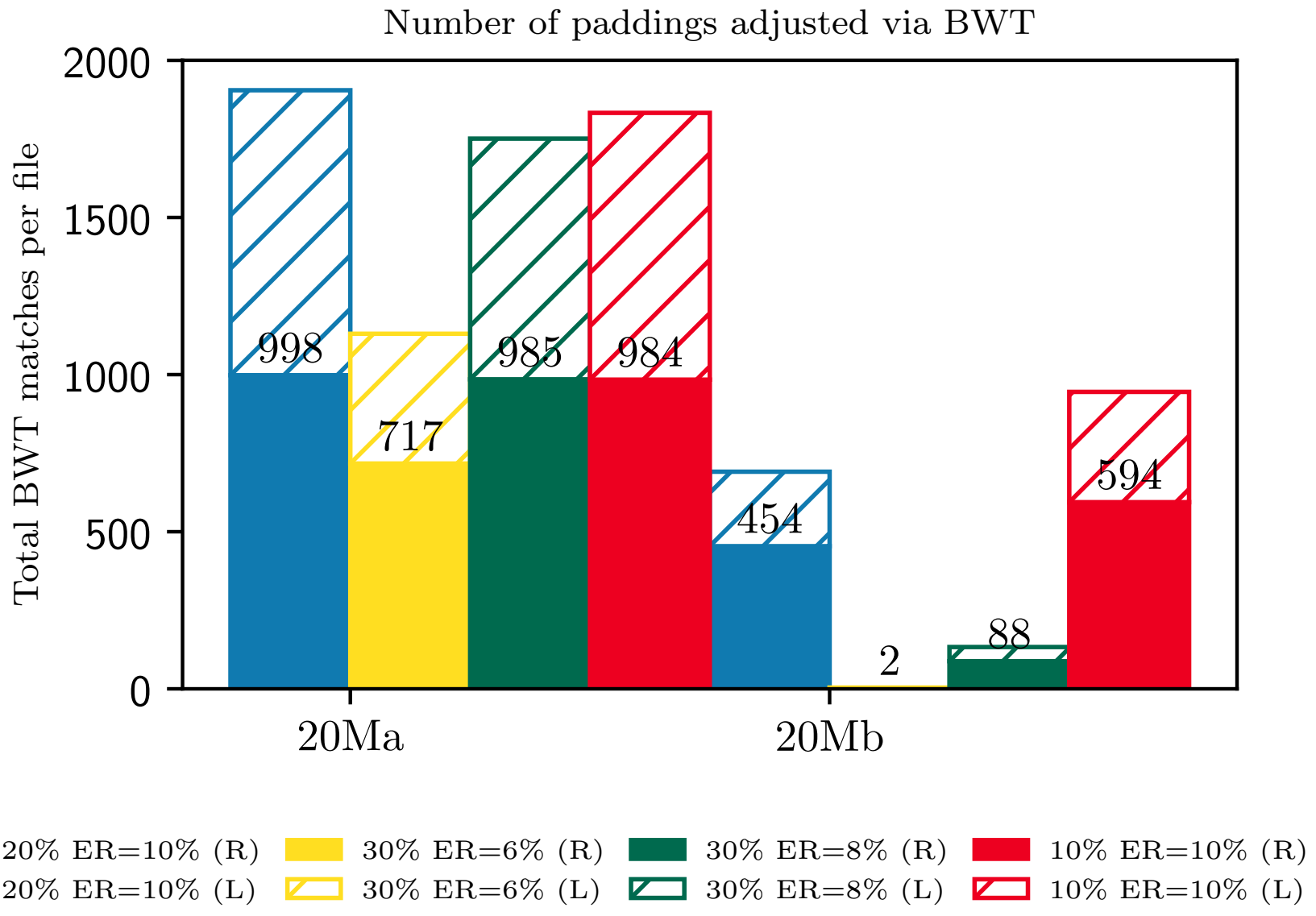
Algorithm Segmented-LIS heuristic O(n log n)

Require: $n \geq 0$
  $lis\_len \leftarrow 0$ ▷*Length of LIS*
  $parent \leftarrow \{\infty, \infty, \infty, \ldots, \infty\}_{n+1}$ ▷*Mapping to reconstruct LIS*
  $sub \leftarrow \{\infty, \infty, \infty, \ldots, \infty\}_{n+1}$ ▷*Array with indices for matches that form LIS*
  $i \leftarrow 0$
  while $i < n$ **do** ▷*Iterate over all elements $i = 0, 1, 2\ldots, n-1$*
    $start \leftarrow 1$
    $end \leftarrow lis\_len$
    while $start \leq end$ **do** ▷*Binary search-like*
      $middle \leftarrow \left\lfloor \frac{start+end}{2} \right\rfloor$
      if $matches_T[sub[middle]] > matches_T[i] - max\_diff$ **then** ▷*Encountered old entry*
        $end \leftarrow start - 1$ ▷*Breaks loop*
      else if $matches_Q[sub[middle]] < matches_Q[i]$ **then**
        $start \leftarrow middle + 1$
      else
        $start \leftarrow middle - 1$
    $parent[i] \leftarrow sub[start - 1]$ ▷*We pin current value to the found parent*
    $sub[start] \leftarrow i$
    if $start > lis\_len$ **then**
      $lis\_len = start$
    $i \leftarrow i + 1$

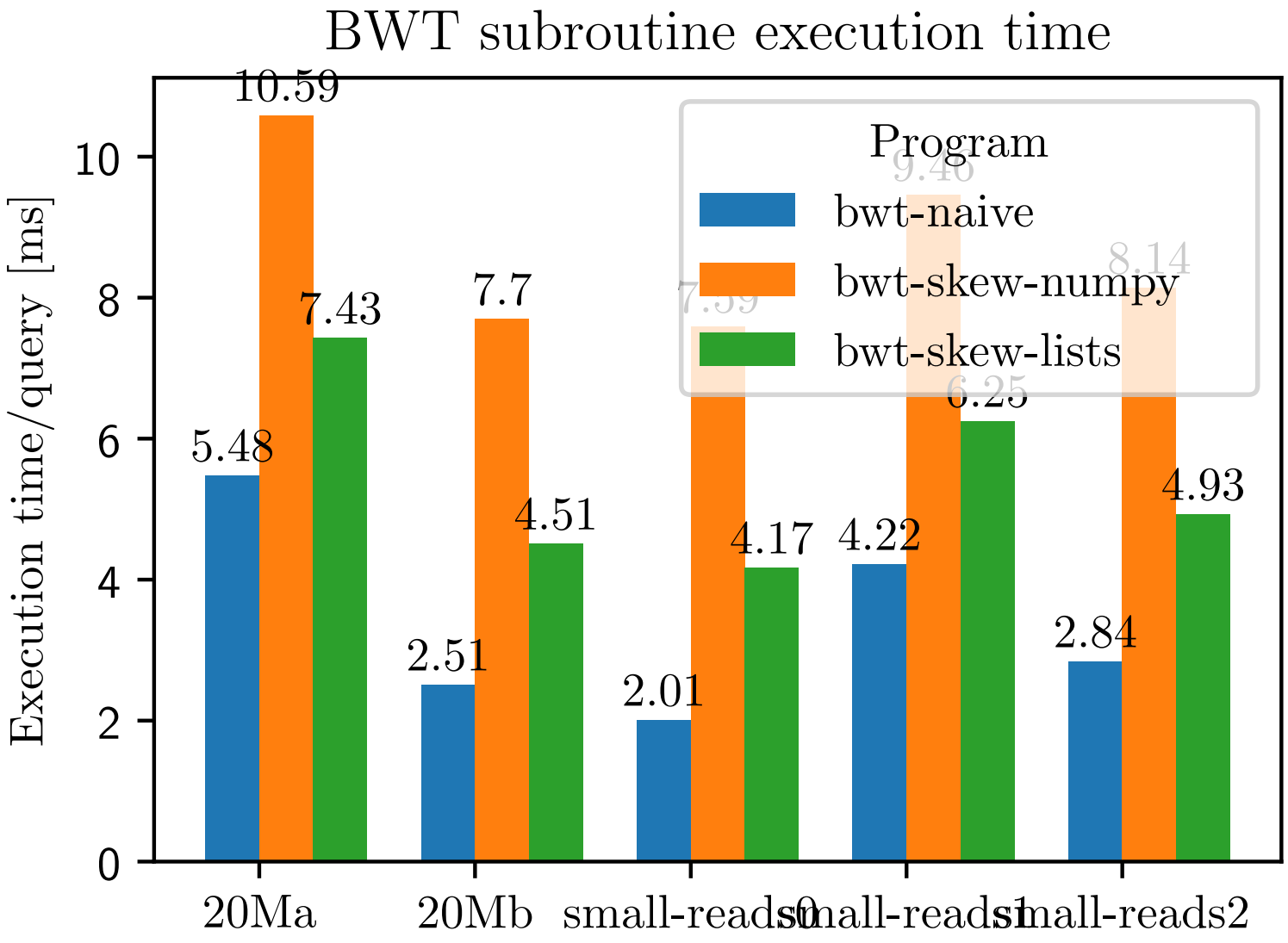
Execution times



Aligner routine effectiveness



BWT routine implementation



Raw DP routine implementation