Aho-corasick RNA Splicing Checker

자기주도연구 : RNA Splicing 현상 모델링

June 4, 2022

We first determine that whether $\mathbb{S}(x) = \emptyset$ for a given string x.

Table 1: introns pattern cases

ex1	n = 8	ex2	n = 14	i	n-(i×2)+1
elements	cases	elements	cases	elements	cases
1	7	1	13	1	n-1
2	5	2	11	2	n-3
3	3	3	9	3	n-5
4	1	4	7	4	n-7
...	...	...	...	...	...
...	...	7	1	n/2	1
result	16	result	39	result	$n^2/4$

Procedure IntronPatternMaker(X)

Input : A string X of length n; $x = x_1x_2...x_n$
Output: pattern string subset and their end points in $X = (M_I, P)$, where M_I denotes a set of Patterns and P denotes a set of the corresponding pattern's start points

`/* i indicates length of m pattern while j is number cases of pattern length i. */`

`/* make patterns starting from the back of the strings */`

for $i \leftarrow 1$ *to* $n/2$ **do**
  for $j \leftarrow n - i * 2 + 1$ *to* 1 **do**
    $b = 2i + j - 1$
    $m = x_b...x_{i+j}$
    if $M_I \not\subset m$ **then**
      $(M_I, P) \leftarrow (m, b)$
    `/* when there are patterns that are exactly same, save the closest pattern's start point to` $n/2$ `as index in` P `*/`
    if $M_I \subset m$ **then**
      $(m, k) = (M_I, P)$ where $M_I = m$
      $(M_I, P) \leftarrow min((m, k), (m, b))$

return (M_I, P)

Table 2: exons pattern cases

1	2	3	...	k-2	k-1	k
			...		-	
			...	-	-	
			...	-	-	
		-	...	-	-	
	-	-	...	-	-	
-	-	-	...	-	-	

Procedure ExonPatternMaker(X,C)

Input : A string X of length n and subset C containing k cut points; $X = x_1x_2...x_k...x_n$
Output: pattern string subset and their start points in $X = (M_e, P)$, where M_e denotes a set of Patterns and P denotes a set of the corresponding pattern's start points

```
/* make patterns backwards from where the intron pattern match starts (cut
   point C) to the first of the strings.                                  */
```

for $i \leftarrow 1$ *to* k **do**
  $b = C_i - 1$ **for** $j \leftarrow 1$ *to* b **do**
    $m = x_b...x_{b-j}$
    $(M_I, P) \leftarrow (m, b)$
return (M_E, P)

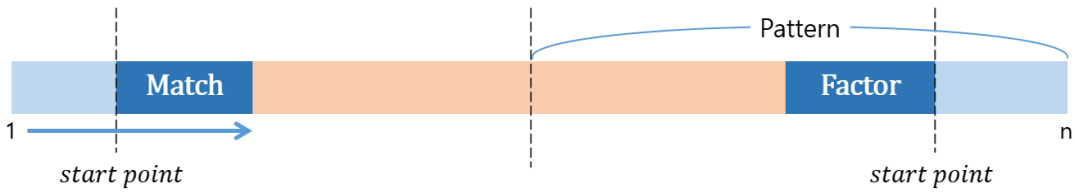

Figure 1: PatternMatchFinder

Procedure PatternMatchFinder((M,P),X,start)

Input : There are four inputs : $(M, P), X, andstart$. M and P each denotes the subset for pattern and its corresponding index point where the pattern starts. X is a string of length n and *start* contains the index where X is first read; $X = x_1x_2...x_{start}...x_n$

Output: There are three outputs : $(M_R, P_R), Q_R$. M_R denotes the succeed match string to guild M, P_R denotes the start point of corresponding M_R on X, and Q_R denotes matching M's start point on X

`/*` m `is a length` k `pattern in subset` M `and` X `is a length n text.` `*/`

Construct a DFA $A = ((Q, Z), \Sigma, \delta, (0, 0), Q\{(0, 0)\})$ for M, where $Q = 0, 1, ..., k$, denotes the length of pattern for $0 \leq i \leq k$ and $Z = P$ denotes the index of pattern start points.

`/* construct the goto function` $\mathbb{G}$ `*/`

$\mathbb{G}(0, \theta(a)) \neq (M_1, Z_1) \in \Sigma) \leftarrow 0$
for $i \leftarrow$ *1 to* k **do**
  $\mathbb{G}(i, M_{i+1}) \leftarrow (i + 1, Z_{i+1})$

`/* construct the failure function` $\mathbb{F}$`, the output function` $\mathbb{O}$`, and the pattern data function` $\mathbb{D}$ `*/`

$\mathbb{F} \leftarrow 0$
for $i \leftarrow$ *1 to* k **do**
  if $\mathbb{G}(i, \theta(a)) = (i + 1, Z_{i+1})$ **then**
    $(v, \emptyset) \leftarrow \mathbb{F}$(i)
    while $\mathbb{G}(v, \theta(a)) \neq \emptyset$ **do**
      $(v, \emptyset) \leftarrow \mathbb{F}(v)$
      $\mathbb{D}(v) \leftarrow null$
    $\mathbb{F}(i + 1) \leftarrow \mathbb{G}(v, \theta(a))$
    $\mathbb{O}(i + 1) \leftarrow min(\mathbb{O}(i + 1), \mathbb{O}(k))$
    $\mathbb{D}(i) \leftarrow \theta(a)$

`/* read` T `using` $\mathbb{G}, \mathbb{F}, \mathbb{O}$ `*/`

$q \leftarrow 0$ $z \leftarrow 0$ **for** $i \leftarrow$ *start to* n **do**
  while $G(q, T(i)) \neq \emptyset$ **do**
    $(q, z) \leftarrow \mathbb{F}(q)$
  $(q, z) = \mathbb{G}(q, T(i))$
  if $\mathbb{O}(q) \neq \emptyset$ **then**
    $(L_R, Q_R) \leftarrow \mathbb{O}(q)$ `// (Pattern node, Index of` M`)`
    $M_R[q] \leftarrow \mathbb{D}(q)$ `// Founded pattern subset` $\theta(M)$
    if $q = 0$ **then**
      $P_R \leftarrow i$ `// Index of` $\theta(M)$

return $(M_R, P_R), Q_R$

Algorithm 1: SplicingOperationChecker(X)

Input : A string X of length n
Output: Y/N
$X = x_1 x_2 ... x_n$

`/* From` $x_1 ... x_n$`, find intron match */`
$(W_I, Q_P) =$ IntronPatternMaker(X)
$(M_I, P_I), Q_I \leftarrow$ PatternMatchFinder$((W_I, Q_P), X, 1)$

`/* From` $x_{Q_I} ... x_n$`, find exon match */`
$(W_E, P_I) =$ ExonPatternMaker(X, P_I)
$(M_E, P_E), Q_E \leftarrow$ PatternMatchFinder$((W_E, P_I), X, Q_I)$

`/* Check if match did happened. */`
if $M_I \neq \emptyset$ & $M_E \neq \emptyset$ **then**
  return Y
return N

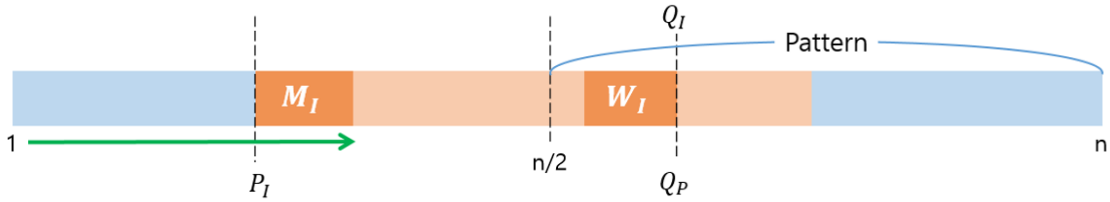

Figure 2: IntronPatternMaker & PatternMatchFinder for intron match

Figure 3: ExonPatternMaker & PatternMatchFinder for exon match

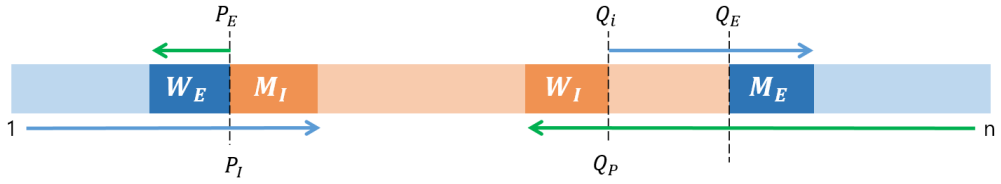

Figure 4: FullSplicingOperationChecker(X)