Advanced Algorithms - Exercise 1

Richter, Bachmann, Kadikowski, Pett, Genov, Mehnert

April 17, 2012

1 Exact DNA matching with Horspool

text t = ACACTCCCCCGACAACC
pattern p = ACAACC

SHIFT	A	C	*
	3	1	6

```
ACACTCCCCCGACAACC
ACAACC
SHIFT[T] = 6.

ACACTCCCCCGACAACC
      ACAACC
SHIFT[A] = 3.

ACACTCCCCCGACAACC
         ACAACC
SHIFT[A] = 3.

ACACTCCCCCGACAACC
           ACAACC
BING > MATCHING
```

In this example the algorithm shows no poor performance, even though the alphabet is small and the pattern consists of only two different letters.

The safe shifts are computed by using the pattern.

Assume there exists a text t $t_1, \dots t_n$ and a pattern p $p_1 \dots p_n$. If there exists the letter β in the pattern and in the text followed by a letter which is not in the pattern, then the algorithms shifts the pattern along the text via a safe shift. If there is again a β in the pattern and the pattern is shifted such that the second occurence of β can not match, then it may result in a missing occurence of the pattern. However, this would contradict the computation of the safe shifts of our preprocessing. Therefore, safe shifts, if computed correctly, cannot be unsafe.

2 Exact multiple DNA matching with Wu-Manber

- Find all occurrences of the patterns ATATATA, TATAT, TAGACG in the string AGATAGACGATATATACG using the Wu-Manber Algorithm. Use a block size of 2. You may use the identity as the hash function, so you have no collisions
  text t = AGATAGACGATATATACG

block size $B = 2$
hash function = id
shortest pattern length $(lmin) = 5$

pattern	
p^1	ATATATA
p^2	TATAT
p^3	TAGACG

	AT	TA	AG	GA	AC	CG
SHIFT(BL)	0	0	3	2	1	0
	AT	TA	CG			
HASH(BL)	2	1	3			

AGA$\underline{\texttt{TA}}$GACGATATATACG
SHIFT[TA] = 0. List = HASH[TA] = 1.
Compare p^1 against text. No match. Shift by one.

AGAT$\underline{\texttt{AG}}$ACGATATATACG
SHIFT[AG] = 3.

AGATAGA$\underline{\texttt{CG}}$ATATATACG
SHIFT[CG] = 0. List = HASH[CG] = 3.
Compare p^3 against text. Pattern matches. Shift by one.

AGATAGAC$\underline{\texttt{GA}}$TATATACG
SHIFT[GA] = 2.

AGATAGACGA$\underline{\texttt{TA}}$TATACG
SHIFT[TA] = 0. List = HASH[TA] = 1.
Compare p^1 against text. No match. Shift by one.

AGATAGACGAT$\underline{\texttt{AT}}$ATACG
SHIFT[AT] = 0. List = HASH[AT] = 2.
Compare p^2 against text. No match. Shift by one.

AGATAGACGATA$\underline{\texttt{TA}}$TACG
SHIFT[TA] = 0. List = HASH[TA] = 1.
Compare p^1 against text. No match. Shift by one.

AGATAGACGATAT$\underline{\texttt{AT}}$ACG
SHIFT[AT] = 0. List = HASH[AT] = 2.
Compare p^2 against text. Pattern matches. Shift by one.

AGATAGACGATATA$\underline{\texttt{TA}}$CG
SHIFT[TA] = 0. List = Hash[TA] = 1.
Compare p^3 against text. Pattern matches. Shift by one.

AGATAGACGATATAT$\underline{\texttt{AC}}$G
SHIFT[AC] = 1.

AGATAGACGATATATA$\underline{\texttt{CG}}$
SHIFT[CG] = 0. List = HASH[CG] = 3.
Compare p^3 against test. No match. Shift by one.

```
Text ends. Finished
```

Patterns occur at position: 4-9, 10-16, 11-15.

- Assume the DNA alphabet and a block size B = 2. For which patterns would we observe the shift worst-case independent of the text?
  We would observe the worst-case shift for 16 patterns if every pattern ends with a different block. Therefore, every possible block in a text would result in a comparison of a pattern and thus we could only shift by 1 letter per iteration step (worst case!).
  It would be possible to use only eight patterns if all possible blocks are the end or the next-to-last block in a pattern. That would lead to the same worst-case shift but we could skip the second hash table if the current block is the next-to-last block of a pattern.

- Does this worst-case occur if the block size is chosen as proposed in the algorithm pseudocode? Why?
  No, because the pseudocode states that:
  $B = log_{|\Sigma|(2*lmin*r)}$ with r = number of patterns.
  If $r = 16$, $\Sigma = 4$ and $lmin = 2$ we would get $B = 3$ which is greater than B of the previous assumption.

3 Trie-Horspool

- Trie construction

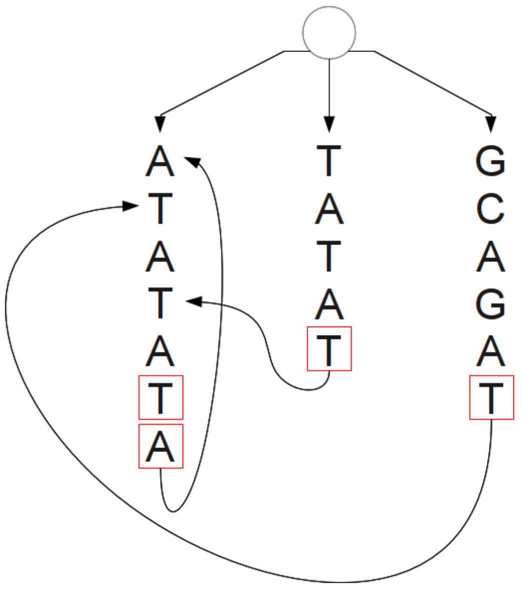

Figure 1: Red: Accepting states, connectors indicate search path after match

- Preprocessing

  Shift table:
  All safe shifts correspond to the minimal possible safe shift of the three patterns.

- Search procedure
  We start at the root and we know that our patterns have to either start with A, T or G. Since the minimal pattern length is 5, we start at the fifth position. According to the character we read, we

d	A	C	G	T
p^1	2	-	-	1
p^2	1	-	-	2
p^3	2	1	3	5
min	1	1	3	1

compare the previous characters with the according pattern. When we reach a node (red squares), the match is counted. After finding the pattern, we proceed according to the connections in Figure 1. If a mismatch occurs, the respective safe shift is used and the procedure starts again.

```
AGATAGACGATATATACG
.x..A > mismatch G, safe shift d[G] = 3
      xC > mismatch (no pattern), safe shift d[C] = 1
 xATAGACG > MATCH! p3 found, change to p1, mismatch, safe shift d[G] = 3
        xATAT > mismatch, safe shift d[G] = 3
        xATATATA > Match! p2, p1, safe shift d[G] = 3
              xACG > mismatch, safe shift d[T] --> reached end of text
```