**1.**

Najděte v textu *T* všechny výskyty řetězců, které mají od vzorku *P* Hammingovu vzdálenost rovnou nejvýše *k*. Použijte metodu dynamického programování ( [TSA] str. 199 – 202).

a) *T* = ccacbaabccaccbcabccc

*P* = abcba

*k* = 2

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | c | c | a | c | b | a | a | b | c | c | a | c | c | b | c | a | b | c | c | c |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| a | 3 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| b | 3 | 4 | 2 | 2 | 1 | 1 | 2 | 1 | 0 | 2 | 2 | 2 | 1 | 2 | 1 | 2 | 2 | 0 | 2 | 2 | 2 |
| c | 3 | 3 | 4 | 3 | 2 | 2 | 2 | 3 | 2 | 0 | 2 | 3 | 2 | 1 | 3 | 1 | 3 | 3 | 0 | 2 | 2 |
| b | 3 | 4 | 4 | 5 | 4 | 2 | 3 | 3 | 3 | 3 | 1 | 3 | 4 | 3 | 1 | 4 | 2 | 3 | 4 | 1 | 3 |
| a | 3 | 4 | 5 | 4 | 6 | 5 | 2 | 3 | 4 | 4 | 4 | 1 | 4 | 5 | 4 | 2 | 4 | 3 | 4 | 5 | 2 |
| i | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |

Hledané výskyty končí na pozicích 6, 11, 15, 20.

b) *T* = 000111011000101010111110

*P* = 110010

*k* = 3

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 4 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 4 | 5 | 2 | 2 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 4 | 4 | 5 | 2 | 3 | 2 | 1 | 0 | 2 | 2 | 0 | 1 | 2 | 3 | 1 | 2 | 1 | 2 | 1 | 2 | 2 | 1 | 1 | 1 | 0 |
| 0 | 4 | 4 | 4 | 5 | 3 | 4 | 3 | 1 | 1 | 3 | 2 | 0 | 1 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 2 | 2 | 1 |
| 1 | 4 | 5 | 5 | 5 | 5 | 3 | 4 | 4 | 1 | 1 | 4 | 3 | 1 | 1 | 4 | 3 | 3 | 2 | 3 | 2 | 2 | 3 | 3 | 2 | 3 |
| 0 | 4 | 4 | 5 | 5 | 6 | 6 | 4 | 4 | 5 | 2 | 1 | 4 | 3 | 2 | 1 | 5 | 3 | 4 | 2 | 4 | 3 | 3 | 4 | 4 | 2 |
| i | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |

Hledané výskyty končí na pozicích 9,10, 12, 13, 14, 16, 18, 20, 21, 24

**2.**

Ukažte, že Levenshteinova vzdálenost splňuje axiomy metriky.

Uvažujme dvě libovolná slova *u, v, w* nad abecedou A {*a*1, *a*2, ..., *an*}. Levenshteinovu vzdálenost slov u, v, označme d(*u,v*). Axiomy metriky jsou:

1. d(*u,v*) = d(*v,u*),

2. d(*u,v*) ≥ 0,

3. d(*u,v*) = 0 ⇔ *u* = *v,*

4. d(*u,w*) ≤ d(*u,v*) + d(*v,w*).

1. Sporem. Nechť BÚNO d1 = d(*u,v*) < d(*v,u*). Nechť dále *v* vzniká z *u* pomocí *k* operací vypuštění symbolu, *l* operací substituce symbolu a *m* operací vložení symbolu, *k + l + m* = d1. Potom je též možné vytvořit *u* z *v* pomocí odpovídajících *m* operací vypuštění příslušných symbolů, *l* operací substituce symbolu a *k* operací vložení symbolu. (Každá operace má svůj odpovídající protějšek – vložení symbolu *ai* do *u* má protějšek vypuštění symbolu *ai* z *v*a obráceně, substituce symbolu *ai* symbolem *aj* v *u* má protějšek substituci symbolu *aj* symbolem *ai* ve *v*.) To znamená, že *u* lze vytvořit z v rovněž pomocí d1 operací. Vzdálenost d(*v,u*) je ale rovna minimálnímu počtu operací nezbztných pro tuto akci, tedy d(*v,u*) ≤ d1. To je spor s předpokladem d1 < d(*v*,*u*).

2. Zřejmé záporný počet operací vypuštění, vložení nebo substituce symbolu ve slově není definován.

3. Pokud d(*u,v*) = 0, nebyla použita žádná operace transformující *u* ve *v*, tedy *u* = *v*.

Pokud *u = v*, je minimální počet operací potřebný na změnu *u* ve *v* roven 0.

4. Změníme *u* ve *v* pomocí d*(u,v*) operací, a dále změníme *v* ve *w* pomocí d(*v,w*) operací. Tím jsme změnili *u* ve *w* pomocí d(*u,v*) + d(*v,w*) operací. Protože vzdálenost d(*u,w*) je rovna minimálnímu počtu operací nutných ke změně *u* ve *w*, musí platit d(*u,w*) ≤ d(*u,v*) + d(*v,w*).

(Dokonce, položíme-li *u = w ≠ v*, vidíme pomocí 3., že 0 = d(*u,w*), d(*u,v*) > 0, d(*v,w*) > 0,

tedy v tomto případě platí d(*u,w*) < d(*u,v*) + d(*v,w*), takže někdy nastává i ostrá nerovnost v 4.

To bychom podotkli, jen abychom ukázali netrivialitu takto definované vzdálenosti.)

**3.**

Napište všechna slova, která mají od vzorku aba nad abecedou {a, b, c} Levenshteinovu vzdálenost rovnu

a) 1

b) 2

a) Vložením: aaba, abaa; substitucí: aaa, bba, abb, cba, aca, abc; smazáním: ba, aa, ab.

b) Délka 1 (dvě smazání: a, b).

Délka 2 (jedno smazání a jedna substituce): ac, bb, bc, ca, cb.

Délka 3 (dvě substituce): aab, aac, aca, acb, acc, baa, bbb, bbc, bca, caa, cbb, cbc, cca.

(jedno smazání i vložení, jen výsledky různé od předchozích): bab, bac, cab.

Délka 4 (jedno vložení a jedna substituce): aaaa, aaab, aaac, aabb, aabc, aaca, abbb, abbc, abca, abcb, abcc, acaa, acab, acac, acba, acbb, acbc, acca, baaa, babb, babc; baca, bbaa, bbab, bbac, bbba, bbca, bcba, caaa, cabb, cabc, caca, cbaa, cbab, cbac, cbba, cbca, ccba.

Délka 5 (dvě vložení): aaaba, aaaba, aabaa, aabab, aabac, aabba, aabca, aacba, abaab, abaac, ababa, ababa, ababa, ababa, ababb, ababb, ababc, abaca, abaca, abacb, abacc, abbaa, abbac, abbba, abbca, abcaa, abcab, abcac, abcba, abcba, abcca, acaba, acaba, acbaa, acbab, acbac, acbba, acbca, accba, baaba, babaa, babab, babac, babba, babca, bacba, bbaba, bcaba, caaba, cabaa, cabab, cabac, cabba, cabca, cacba, cbaba, ccaba.

**4.**

Abeceda *A* obsahuje *n* symbolů, slovo *w* má délku *m*. Odhadněte shora počet slov nad abecedou *A*, která mají od *w* Hammingovu vzdálenost rovnu *k* (0 ≤ *k* ≤  *m*).

Můžeme provést nejvýše C(*m, k*) × n*k*  substitucí, kde C(*m, k*) je kombinační číslo „*m* nad *k*“.

**5.**

Abeceda *A* obsahuje *n* symbolů, slovo *w* má délku *m*. Odhadněte shora počet slov nad abecedou *A*, která mají od *w* Levenshteinovu vzdálenost rovnu *k* (0 ≤ *k* ≤  *m*).

Nejvíce nových slov vygenerujeme jednou operací vložení, tak může vzniknout ze slova délky *m* až *n* × (*m*+1) nových slov. protože se slova vkládáním budou prodlužovat, vytvoříme nejvýše (*n* × (*m*+*k*))*k* nových slov*.* Pokuste se o přesnější odhad.

**6.**

Najděte v textu *T* všechny výskyty řetězců, které mají od vzorku *P* Levenshteinovu vzdálenost rovnou nejvýše *k*. Použijte metodu dynamického programování ( [TSA] str. 202 – 205).

a) *T* = aacacacbaabbbcbbcacc

*P* = abbcba

*k* = 2

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | a | a | c | a | c | a | c | b | a | a | b | b | b | c | b | b | c | a | c | c |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| a | 3 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| b | 3 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 2 | 1 | 1 | 2 | 1 | 1 | 2 |
| b | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 2 | 2 | 1 | 0 | 1 | 2 | 2 | 1 | 2 | 2 | 2 | 2 |
| b | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 3 | 2 | 1 | 0 | 1 | 2 | 2 | 2 | 3 | 3 | 3 |
| a | 3 | 3 | 3 | 4 | 3 | 4 | 3 | 4 | 3 | 2 | 2 | 3 | 2 | 1 | 1 | 2 | 3 | 3 | 2 | 4 | 4 |
| i | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |

Hledané výskyty končí na pozicích 9 (acba),10 (acbaa) , 12 (abb, aabb) , 13 (bbb, abbb), 14 (bbbc, abbbc), 15 (abbbcb), 18(bbca, cbbca).

b) *T* = 010011101000010101011100

*P* = 11100

*k* = 1

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 2 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 2 | 2 | 1 | 1 | 2 | 1 | 0 | 0 | 1 | 1 | 1 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 2 |
| 1 | 2 | 3 | 2 | 2 | 2 | 2 | 1 | 0 | 1 | 1 | 2 | 2 | 3 | 3 | 2 | 2 | 1 | 2 | 1 | 2 | 1 | 1 | 0 | 1 | 2 |
| 0 | 2 | 2 | 3 | 2 | 2 | 3 | 2 | 1 | 0 | 1 | 1 | 2 | 2 | 3 | 3 | 2 | 2 | 1 | 2 | 1 | 2 | 2 | 1 | 0 | 1 |
| 0 | 2 | 2 | 3 | 3 | 2 | 3 | 3 | 2 | 1 | 1 | 1 | 1 | 2 | 2 | 4 | 3 | 3 | 2 | 2 | 2 | 2 | 3 | 2 | 1 | 0 |
| i | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |

Hledané výskyty končí na pozicích 8 (1110), 9 (11101), 10 (111010), 11 (10100, 110100), 23 (1110), 24 (1100, 11100).

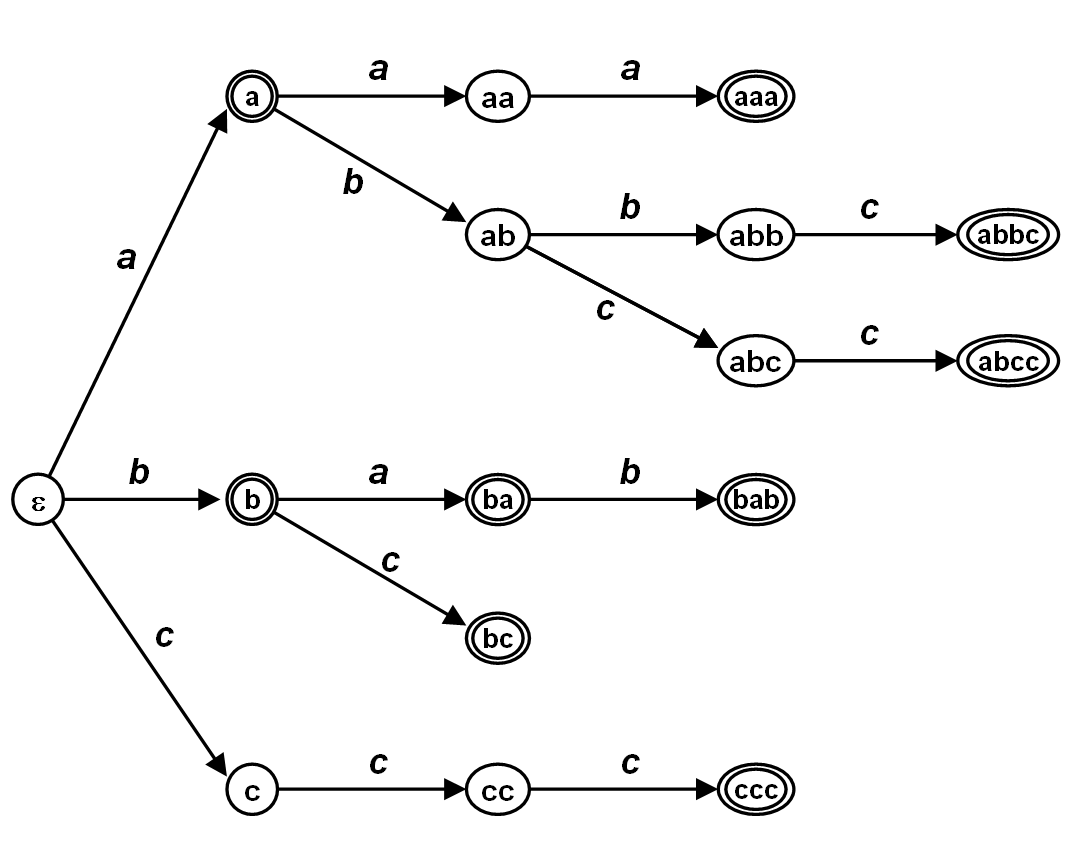
**7.**

Sestrojte deterministický automat nad abecedou A, který přijímá právě množinu M slov nad touto abecedou.

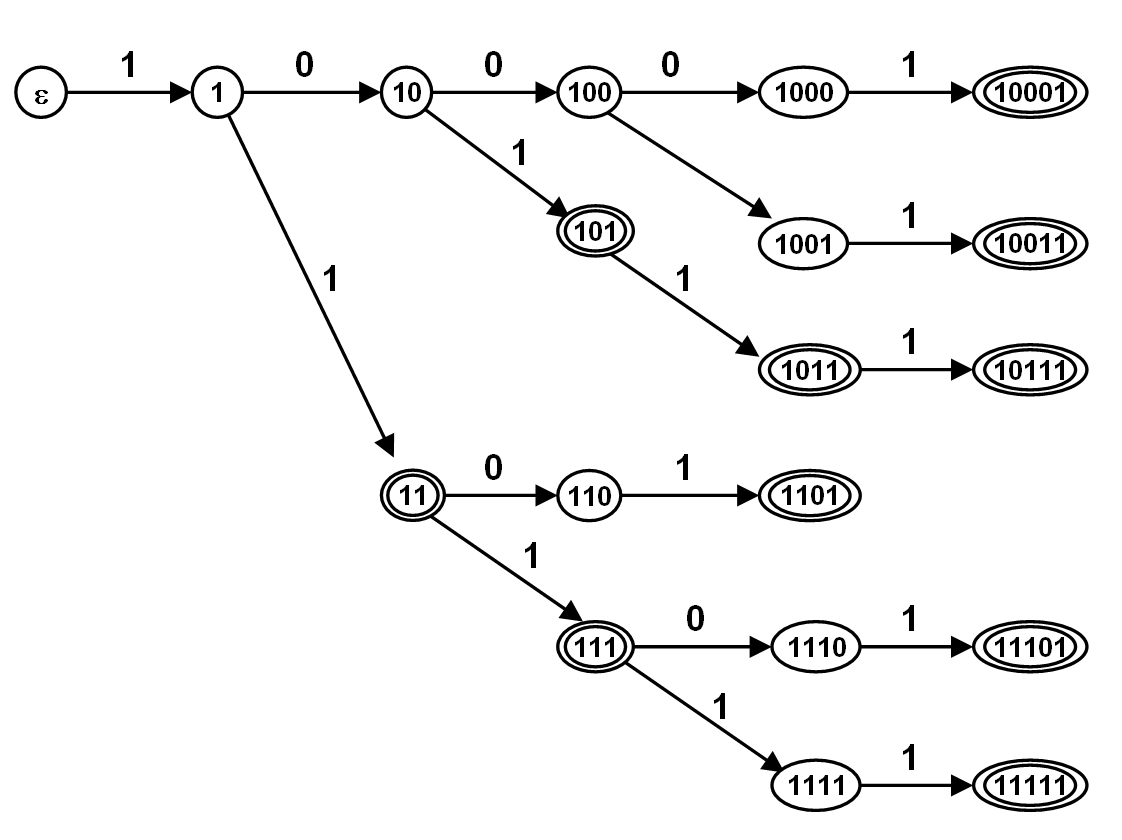
a) A = {a, b, c}, M = {a, b, ba, bc, aaa, bab, ccc, abbc, abcc } .

b) A = {0, 1}, M = {10, 11, 101, 111, 1011, 1101, 10001, 10011, 10111, 11101, 11111 }.

a)



b)



**8.**

Sestrojte deterministický automat, který v textu nad abecedou A vyhledá právě každé slovo množiny M z předchozí úlohy.

(Řešení explicitně neuvádím, přidá se smyčka ohodnocená celou abecedou do prvního stavu ε a provede se standardní determinizace.)

**9.**

Sestavte automat, který v textu nad abecedou A vyhledává všechna slova popsaná regulárním výrazem R.

a) A = {a,b,c}, R = c\*(ac + bb)\*

b) A = {0, 1}, R = 0\*(101 + 11)\*0

Návod: Zopakujte si postup vytvoření NKA přijímajícího jazyk popsaný daným regulárním výrazem (je i v [TSA], Algoritmus 1.47., str. 21), . Tento NKA opatříme v počátečním stavu smyčkou pro všechny znaky abecedy a provedeme standardní algoritmus pro převod NKA na DKA.

**10.**

Demonstrujte fakt, že paměťová složitost deterministického automatu pro vyhledávání slov odpovídajících regulárnímu výrazu může růst exponenciálně s počtem přijímaných slov.

Použijte regulární výraz R = *a*(*a+b*)*m*−1 (*m* ≥ 1).

Návod: Vytvořte přechodovou tabulku příslušného automatu pro *m* ≥ 5 a sledujte, jak v tomto případě postupuje algoritmus pro převod NKA na DKA.

**11.**

Sestavte tabulky pro simulaci činnosti vyhledávacího automatu metodou bitového paralelizmu pro daný text *t*, vzorek *p* a Hammingovu vzdálenost *k*,

a) *t* = abcbcaaccbbaa

*p* = bbac

*k* = 2

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **R0** |  | a | b | c | b | c | a | a | c | c | b | b | a | a |
| b | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| b | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| a | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| c | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **R1** |  | a | b | c | b | c | a | a | c | c | b | b | a | a |
| b | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| b | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| a | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| c | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **R2** |  | a | b | c | b | c | a | a | c | c | b | b | a | a |
| b | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| b | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| a | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| c | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |

Stínované pozice odpovídají konci nalezeného řetězce, jehož Hammingova vzdálenost od vzoru je rovna *k.*

b) *t* = accbbaaabcba

*p* = acbb

*k* = 2

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **R0** |  | a | c | c | b | b | a | a | a | b | c | b | a |
| a | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| c | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| b | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| b | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **R1** |  | a | c | c | b | b | a | a | a | b | c | b | a |
| a | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| c | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| b | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| b | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **R2** |  | a | c | c | b | b | a | a | a | b | c | b | a |
| a | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| c | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| b | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| b | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

Stínované pozice odpovídají konci nalezeného řetězce, jehož Hammingova vzdálenost od vzoru je rovna *k.*