אלגוריתם ב׳

באלגוריתם ב׳ נמצא פתרון לבעיית התאמת מחרוזות באמצעות אוטומטיים סופיים.

בסעיף זה נלמד מהו אוטומט סופי, כיצד הוא פועל וכיצד ניתן לבנות אותו. אחרי כן נשתמש בו למציאת מופעים של התבנית P בטקסט T.

<u>אלגוריתם להתאמת מחרוזות בעזרת אוטומט סופי</u>

לאחר שהכרנו אוטומט סופי דטרמיניסטי, עתה נחזור לבעיה המקורית שהיא התאמת מחרוזות באמצעות אוטומטים סופיים.

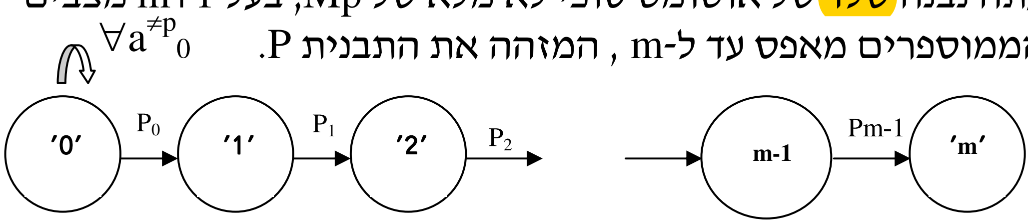
כאמור הנתונים הם מחרוזות של תווים: טקסט $T=[T_0,T_1,\ldots T_{n-1}]$
ותבנית $P=[P_0,P_1,\ldots,P_{m-1}]$

כאשר $m<=n$

ברור כי עבור $0<=k<=n-1$, $T_k\in\Sigma$

ועבור $P_j \in\Sigma,0<=j<=m-1$

עתה נבנה שלד של אוטומט סופי לא מלא של Mp, בעל m+1 מצבים הממוספרים מאפס עד ל-m , המזהה את התבנית P.



המעבר ממצב i -1 למצב i יתבצע על ידי קריאת התו P_{i-1} בקלט, כמתואר באיור מעלה.
במצב התחלתי, מצב 0, ישנה לולאה עצמית עבור כל תו ששונה מהתו P_0.

כללית המצב i של האוטומט Mp מתאים לרישא (Prefix) $P_0,P_1,\ldots,P_{i-1}$ של התבנית P.

בהתחלה, בהינתן שתי מחרוזות:

$T=[T_0,T_1,T_2,\ldots,T_{n-1}]$

$P=[P_0,P_1,P_2,\ldots,P_{m-1}]$

האוטומט הסופי Mp נמצא במצב 0, ומצביע קלט (input pointer) מצביע
ל- T_0

אם $T_0=P_0$ אז:

- האוטומט Mp יעבור למצב 1
- מצביע קלט יצביע ל- T_1

אחרת $P_0\neq T_0$

- האוטומט הסופי (Mp) ישאר במצב 0
- מצביע קלט יצביע ל- T_1

עתה **נניח** כי אחרי קריאת K התווים הראשונים של הטקסט $T_0,T_1,\ldots,T_{K-1}$ האוטומט הסופי Mp ימצא במצב j.
כלומר j התווים האחרונים של $T_0,T_1,\ldots,T_{K-1}$ הם $P_0,P_1,\ldots,P_{j-1}$. כלומר

$$T_0,T_1,\ldots T_{k-1}=T_0,T_1,\ldots,T_{k-j-1}\ P_0P_1\ldots P_{j-1}$$

ו- u התווים האחרונים של $T_0,T_1,\ldots,T_{k-1}$ הם לא רישא (Prefix) של התבנית $P=P_0,\ldots,P_{m-1}$, עבור u>j.

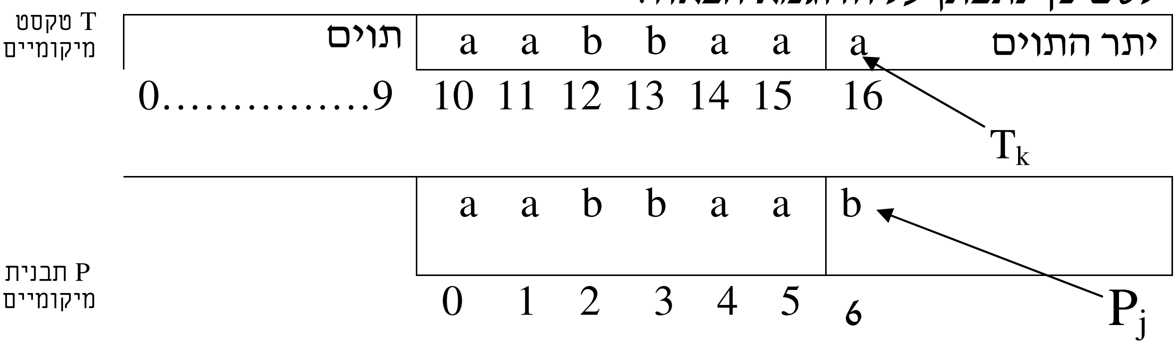
להלן האיור המתאר את תמונת המצב שהתקבלה עד כה.

T	$T_0\ T_1\ T_2\ldots T_{k-j-2}T_{k-j-1}$	$T_{k-j}\ldots\ldots T_{k-1}$	$T_k\ \ T_{k+1}\ldots T_{n-1}$
		$P_0P_1\ldots\ldots P_{j-1}$	$P_jP_{j+1}\ldots\ldots P_{m-1}$

כאמור $T_{k-j}\ T_{k-j+1}\ldots T_{k-1}=P_0P_1\ldots P_{j-1}$.
עתה נותר לבדוק האם $T_k=P_j$ או לאו?

אם $P_j=T_k$ האוטומט הסופי Mp יעבור למצב j+1 ומצביע הקלט (input pointer) יתקדם לתו הבא בקלט שהינו T_{k+1} .

אם $P_j \neq T_k$ מה עושים?
לשם כך נתבונן על הדוגמא הבאה:



רואים כי $T[10\ldots15]=P[0\ldots5]$
עתה נותר לבדוק האם $P[6]=T[16]$?

בדוגמא רואים שהתשובה היא שלילית, כלומר $P[6] \neq T[16]$.

באלגוריתם הנאיבי היינו ממשיכים את השוואת תווי התבנית $P_0\ P_1\ ..P_6$ מול תווי הטקסט (T) החל ממיקום 11.

זו הסיבה שהאלגוריתם הנאיבי להתאמת מחרוזות אינו יעיל משום שהוא מתעלם מהמידע (חלק מהטקסט T שראינו באיטרציה האחרונה של השוואת התווים) , אבל מידע כזה יכול להיות בעל ערך רב.

למשל באיטרציה האחרונה של השוואת התווים ראינו כי
$P[0\ldots5]=T[10\ldots15]$ והסיפא הארוך ביותר שמופיע ב- $P[0\ldots5]$ וב-
$T[10\ldots15]$ הוא aa שהוא גם הרישא של P.
כלומר בתום האיטרציה הזו ניתן לראות כי :
$P[0,1]=P[4,5]=T[14..15]$, לכן אין טעם להשוות שוב בין $P[0,1]$ ו-
$T[14..15]$.
עתה ברור שבאיטרציה הבאה תתבצע רק ההשוואה הבאה : האם
$P[2..6]=T[16..20]$.

לכן בדוגמא זו לאחר ההשוואה של $T[16]$ ו- $P[6]$, כיוון ש-
$P[6]=b\neq a=T[16]$, האוטומט הסופי Mp יעבור למצב 2 ולא למצב 0, כי
תת מחרוזת $p_0p_1=aa$ מובילה את החישוב למצב 2 (בדוק באוטומט!).
לכן נסיק כי :

סופית – אם $P_j \neq T_k$ האוטומט הסופי Mp נכנס למצב i המקיים :
i הוא המספר הגדול ביותר כך ש-
$P_0 P_1 \ldots P_{i-1}$ הוא סיפא (suffix) של $T_0 T_1 \ldots T_k$.

- למציאת המצב ה- i שהזכרנו קודם נעזר בפונקצית עזר f הנקראת **פונקצית כישלון** (failure function) או
- **פונקצית הרישא** (prefix function) אשר מוגדרת כך :
עבור כל מצב $j\geq1$

$$f(j)=\max\{s / s<j \text{ וגם } P_0 P_1\ldots P_{s-1} =\text{suffix}(P_0P_1 \ldots P_{j-1})\}.$$

כלומר f(j) הוא המספר הגדול ביותר s, כאשר s קטן מ- j, וגם

$$P_0P_1\ldots P_{s-1}=P_{j-s} P_{j-s+1}\ldots P_{j-1}$$

כלומר f(j) הוא אורכה של הרישא (prefix) הארוכה ביותר של P שהוא סיפא (suffix) של $P_0\ldots P_{j-1}$.

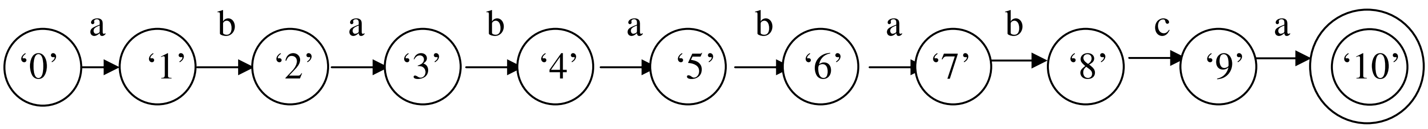
- אם לא קיים s כזה אזי ערכו של f(j) יהיה שווה לאפס.

דוגמא

נתונה התבנית הבאה

P= ababababca
0123456789

והאוטומט הסופי Mp הינו



עבור		
j=1	f(1)=0	כיוון שלא קיים s<1 שיקיים את התנאי הדרוש.
j=2	f(2)=0	כיוון שאין אף רישא של P שהוא סיפא של ab וגם באורך 1 לכל היותר.
j=3	f(3)=1	כיוון שהרישא הארוכה ביותר של P הוא a שהוא גם סיפא ממש של aba וגם באורך 2 לכל היותר. האורך של lal שווה ל- 1.
j=4	f(4)=2	כיוון שהרישא הארוכה ביותר של P הוא ab שהוא גם סיפא ממש של abab ו- lab l=2.
j=5	f(5)=3	כיוון שהרישא הארוכה ביותר של P הוא aba שהוא גם סיפא ממש של ababa ו- labal=3.
j=6	f(6)=4	כיוון שהרישא הארוכה ביותר של P הוא abab שהוא גם סיפא ממש של ababab ו- lababl=4.
j=7	f(7)=5	כיוון שהרישא הארוכה ביותר של P הוא ababa שהוא גם סיפא ממש של abababa ו- lababal=5.
j=8	f(8)=6	כיוון שהרישא הארוכה ביותר של P הוא ababab שהוא גם סיפא ממש של abababab ו- lababab l=6
j=9	f(9)=0	כיוון שאין שום רישא של P שהוא גם סיפא ממש של ababababc. לכן לפי ההגדרה f(9)=0.
j=10	f(10)=1	כיוון שהרישא הארוכה ביותר של P הוא a שהוא גם הסיפא ממש של ababababca ו- lal=1.

ולסיכום פונקצית הרישא מוגדרת כך עבור התבנית הבאה : abababababca

J (מצב)	1	2	3	4	5	6	7	8	9	10
f(j)	0	0	1	2	3	4	5	6	0	1

<u>דוגמא נוספת</u> : עבור התבנית הבאה P=aabbaab
ניתן לבדוק כי (בדוק!) פונקצית הכישלון/פונקצית הרישא תראה כך :

J (מצב)	1	2	3	4	5	6	7
f(j)	0	1	0	0	1	2	3

סימון : $f^{(r)}(j)$ פירושו f מופעל על j   r פעמים כלומר :

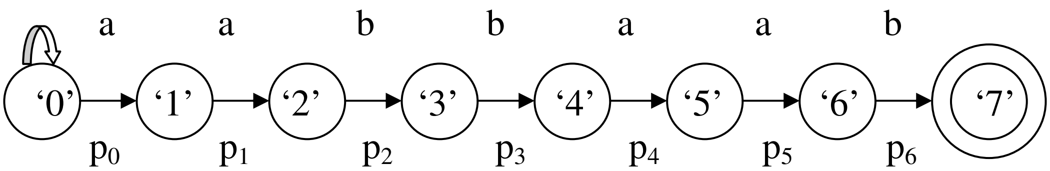
$$\begin{aligned} f^{(r)}(j) &= \begin{cases} f(f^{(r-1)}(j)) & m>1 \quad \text{לכל} \\ f(j) & m=1 \quad \text{ועבור} \end{cases} \\ f^{(1)}(j) &= \end{aligned}$$

בהמשך לדוגמא הנוספת

$$f^{(2)}(6) = f(f^{(1)}(6) = f(f(6)) = f(2) = 1$$

כי f(6)=2

בהמשך לדוגמא הנוספת החצים המקווקו ים מתארים את פונקצית הכישלון או פונקצית הרישא.



עתה נתבונן על המקרה הבא:

תוים	a	a	b	b	a	a	a	יתר התוים
0……………9	10	11	12	13	14	15	16 $\leftarrow T_k$	

P (תבנית)	a	a	b	b	a	a	b $\leftarrow P_j$	
מיקומיים	0	1	2	3	4	5	6	

מחד $T[10\ldots15] = P[0\ldots5]$ ולכן האוטומט הסופי נמצא במצב 6,
ומאידך $T[16] \neq P[6]$.
לאור ההתאמה החלקית האוטומט הסופי Mp יעבור למצב 2, כיוון ש- 2
היא אורכה של הרישא aa שהיא הארוכה ביותר מבין הרישות של
התבנית P שהיא הסיפא ממש של $T_0\ldots T_{15}$.
על סמך פונקצית הכישלון שראינו קודם, עבור התבנית הנתונה
P=aabbaab, מצאנו כי: $f^{(1)}(6) = f(6) = 2$ כצפוי.
כיוון שעתה האוטומט Mp נמצא במצב 2 וכי $aa=P_0P_1=P_0P_1\ldots P_{f(6)-1}$
והוא הרישא הארוך ביותר של P שהוא גם סיפא של $T_0\ldots T_k$
$(T_0\ldots T_{15})$.

עתה תמונת המצב הינה:

T טקסט	a	a	b	b	a	a	a	b	b	a	a	b
	10	11	12	13	15	14	16	17	18	19	20	21
P תבנית					a	a	b	b	a	a	b	
					0	1	2	3	4	5	6	

עתה עלינו לבדוק האם $T[16] = P[2]$ כלומר האם a=b?
ברור כי ש- $a \neq b$ ולכן לאור ההתאמה החלקית האוטומט הסופי Mp
יעבור למצב 1 (ולא למצב 0) כי על סמך פונקצית הכישלון מצאנו ש- 1
הוא אורכה של הרישא a שהוא הארוך ביותר מבין הרישות של התבנית P
שהוא הסיפא ממש של $T_0\ldots T_{15}$.
ואכן $f^{(1)}(2) = f(2) = 1$ כצפוי.
עתה האוטומט Mp נמצא במצב 1 וכי

$$\mathbf{a=P_0=P_0\ldots P_{f^{(2)}-1}=P_0\ldots P_{f(f(6))-1}=P_0\ldots P_{f^{(2)}(6)-1}}$$

<u>לסיכום</u>

נניח כי אחרי שהאוטומט הסופי Mp קרא K את התוים הראשונים $(T_0 \ldots T_{k-1})$ של הטקסט T והוא נמצא במצב j ו- $T_k \neq P_j$.

ברור שבנקודה זו האוטומט הסופי Mp יפעיל את פונקצית הכישלון על j עד שימצא את הערך הקטן ביותר של r שעבורו יתקיים :

1. $T_k = P_u$ ו- $f^{(r)}(j) = u$

או

2. $T_k \neq P_0$ ו- $f^{(r)}(j) = 0$

אם תנאי (1) מתקיים אזי האוטומט Mp יכנס למצב u+1

ואם תנאי (2) מתקיים אזי האוטומט Mp יכנס למצב 0

בכל מקרה מצביע קלט (input pointer) יתקדם לתו T_{k+1}.

אם תנאי (1) מתקיים אזי קל לראות כי אם $P_0 \ldots P_{j-1}$ הוא הרישא (Prefix) הארוך ביותר של P, שהוא גם סיפא (Suffix) של $T_0 \ldots T_{k-1}$, אזי $P_O P_1 \ldots P_{f^{(m)}(j)+1}$ הוא הרישא (Prefix) הארוך ביותר של התבנית P שהוא גם סיפא (Suffix) של $T_0 \ldots T_k$.

אם תנאי (2) מתקיים אז אין אף רישא של התבנית P שהוא סיפא של $T_0 \ldots T_k$.

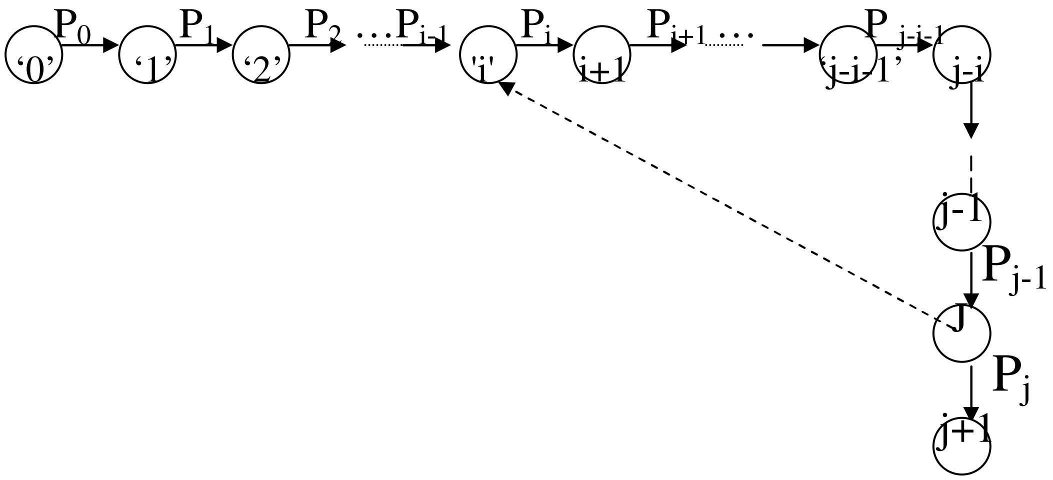
<u>אלגוריתם המחשב את פונקצית הכישלון - f</u>

נחשב את פונקצית הכישלון f בצורה איטרטיבית כמפורט מטה :

ברור כי לפי ההגדרה $f(1)=0$,

ונניח כי $f(1),f(2),\ldots,f(j)$ כבר חושבו ו- $f(j)=i$

כלומר $P_0P_1\ldots P_{i-1}=P_{j-i}\ldots P_{j-1}$



הקו המקווקו מתאר את פונקצית הכישלון.

עתה נחשב את f(j+1)

נבחן האם Pi=Pj?

- אם $Pi=Pj$ אזי $f(j+1)=f(j)+1$,
כיוון ש- P_j $P_{j-i-1}\ldots P_{j-1}$ = $P_0\ldots P_{i-1}$ P_I

- אם $Pi\neq Pj$ לאור האמור לעיל נמצא את ה- r הקטן ביותר כך שיתקיים:

א. $P_j=P_u$ ו- $f^{(r)}(j) = u$

או

ב. $P_j\neq P_0$ ו- $f^{(r)}(j) = 0$

במקרה א' $f(j+1) = u + 1$

במקרה ב' $f(j+1) = 0$

<u>לסיכום להלן אלגוריתם לחישוב f</u> (פונקציית הכשלון)

נניח שהתבנית P הינה $P=P_0\ldots P_{m-1}$, כלומר אורכה m.

1. אורך התבנית $m\leftarrow(P)$
2. $f[1]\leftarrow 0$;=0
3. עבור j מ- 2 עד m, בצע:
   3.1 $i\leftarrow f(j-1)$
   3.2 כל עוד $P[j-1]\neq P[i]$ וגם $i>0$
   3.2.1 בצע: $i\leftarrow f(i)$
   3.3 אם $P[j-1]\neq P[i]$ וגם $i=0$
   3.3.1 אז בצע: $f(j)=0$
   3.3.2 אחרת בצע: $f(j)=i+1$
4. סוף האלגוריתם.

טענה 1: האלגוריתם המתואר מעלה מחשב את פונקצית הכישלון f.

ניתן להוכיח את נכונות הטענה באינדוקציה על j לחישוב f(j), עבור כל j.
נשאיר את ההוכחה כתרגיל ל"קוראים מתקדמים" ולהוכיח לכל j ש- f(j) הוא המספר הגדול ביותר I, כאשר I<j ומתקיים :
$P_0 \ldots P_{I-1} = P_{j-I-1} \ldots P_{j-1}$ ואם אין I כזה אזי f(I) יקבל ערך 0.

<u>ניתוח זמן הריצה של האלגוריתם לחישוב f (פונקצית הכישלון)</u>

באמצעות ניתוח לשיעורין נקבל כי זמן הריצה של COMPUTE-PREFIX-FUNCTION הוא O(m).

בניתוח זה נתייחס ל-i כאל k , ל-j כאל q ואל הפונקציה f כאל π.

נייחס פוטנציאל k למצב הנוכחי k של האלגוריתם.

- על-פי שורה 2, ערכו ההתחלתי של הפוטנציאל הוא 0.
- בכל פעם ששורה 3.2.1 מתבצעת, ערכו של k קטֵן, שכן $\pi[k] < k$.
- אולם, מכיוון ש- $\pi[K] \geq 0$ עבור כל k, אזי k לעולם אינו יכול להפוך למספר שלילי.
- השורה הנוספת היחידה המשפיעה על k היא שורה 3.3.2 , שבה ערכו של k גדל ב- 1 לכל היותר במהלך כל ביצוע של גוף לולאת ה-for.
- מכיוון שעם הכניסה ללולאת ה-for מתקיים k<q, ומכיוון שערכו של q גדל ב-1 בכל איטרציה של גוף לולאת ה-for, אזי k<q מתקיים תמיד. (טיעון זה מצדיק גם את הטענה ש $\pi[q]$ קטן מ-q , על פי שורה 9.)
- נוכל לשלם עבור כל ביצוע של גוף לולאת ה- While בשורה 3.2 באמצעות הירידה המתאימה בערכה של פונקצית הפוטנציאל, שכן $\pi[k] < k$. שורה 3.3.2 מגדילה את ערכה של פונקצית הפוטנציאל ב-

1 לכל היותר, כך העלות לשיעורין של גוף הלולאה שבשורת 3.1 עד 3.3 היא O(1).

- מכיוון שמספר האיטרציות של הלולאה החיצונית הוא O(m), ומכיוון שפונקצית הפוטנציאל הסופית גדולה לפחות כמו פונקצית הפוטנציאל ההתחלתית, הרי שזמן הריצה הכולל בפועל של COMPUTE-PREFIX-FUNCTION שמקרה הגרוע הוא O(m).

<u>סופית</u> זמן הריצה של האלגוריתם לחישוב פונקצית הכישלון הוא 0(m).

<u>בניית אוטומט סופי דטרמיניסטי</u>

עבור התבנית P, האוטומט הסופי Mp, ימצא במצב I, אחרי שקרא k תוים $T_0,\ldots T_{k-1}$ מטקסט T, אם ורק אם $P_0P_1\ldots P_{I-1}$ היא הרישא הארוכה ביותר של P שהוא סיפא של $T_0\ldots T_{k-1}$.

עתה נבנה אוטומט סופי דטרמיניסטי $M=(Q,\sum, \delta,q_0,F)$ המזהה אוסף כל המחרוזות המסתיימות בתבנית $P=P_0\ldots P_{m-1}$, כלומר $\sum^{*}\cdot P$, כאשר :

$Q=\{0,1,\ldots.m\}$
$F=\{m\}$
$q_0=\{0\}$

ופונקצית המעבר תבנה כך :

1. **עבור j מאחד עד m בצע: $\delta(J\text{-}1,P_{J-1})\leftarrow j$**

/* כאן בונים את השלד העיקרי*/

2. **לכל סימן a, כאשר $a\in\sum$ ו- $a\neq P_0$ $\delta(0,a)\leftarrow 0$**

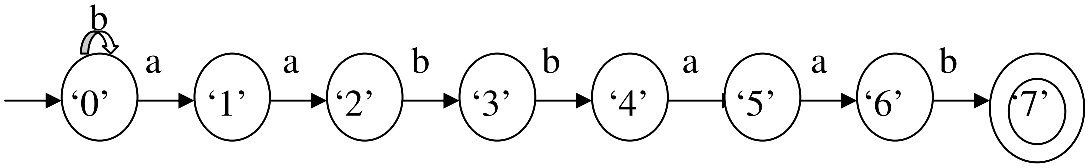
/* במצב התחלתי כל סימן השונה מ- P_0, התו הראשון של התבנית, מוביל את החישוב שוב למצב התחלתי*/

3. **עבור j מאחד עד m בצע:** /* עוברים על כל מצב j */

3.1 עבור כל סימן a ב-$\sum$ בצע:

אם $a \neq P[j]$ אז בצע: $\delta(j,a)=\delta(f(j),a)$

נבהיר את אופן הבניה של אוטומט סופי דטרמיניסטי בעזרת התבנית הבאה: P=aabbaab.
בעזרת הצעדים 1 ו- 2 נקבל את השלד הבא:



ראינו בעבר שפונקצית הכישלון במקרה זה הינה:

J	1	2	3	4	5	6	7
F(j)	0	1	0	0	1	2	3

<u>נבחן מספר מצבים:</u>

<u>עתה האוטומט נמצא במצב 1</u> ונקלוט את התו b מהקלט ולא את התו a כמצופה, לכן פונקצית הכישלון מובילה למצב 0, כי $f(1)=0$, אך
$\delta(0,b)=0$
כלומר $\delta(1,b)= \delta(f(1),b)= \delta(0,b)=0$
(*)

<u>עתה האוטומט נמצא במצב 2</u>
ונקלוט את התו a מהקלט ולא את התו b כמצופה, לכן פונקצית הכישלון מובילה את האוטומט למצב 1, כי $f(2)=1$, אך $\delta(1,a)=2$ כלומר
$\delta(2,a)= \delta(f(2),a)=\delta(1,a)=2$
(**)

<u>עתה האוטומט נמצא במצב 3</u> והתו הנקלט הוא a, ולא התו b כמצופה.
לכן פונקצית הכישלון מובילה את האוטומט למצב 0, כי $f(3)=0$, אך
$\delta(0,a)=1$ כלומר

$\delta(3,a)=\delta(f(3),a)=\delta(0,a)=1$
(***)

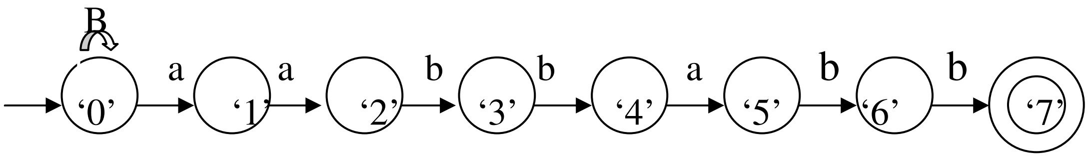
עתה האוטומט נמצא במצב 4 והתו הנקלט הוא b לכן
$\delta(4,b)=\delta(f(4),b)=\delta(0,b)=0$
וכך ממשיכים:
$\delta(5,b)=\delta(f(5),b)=\delta(1,b)=0$

לפי (*)

$\delta(6,a)=\delta(f(6),a)=\delta(2,a)=2$

לפי (**)
$\delta(7,a)=\delta(f(7),a)=\delta(3,a)=1$

לפי (***)
$\delta(7,b)=\delta(f(7),b)=\delta(3,b)=4$
וסופית האוטומט הסופי הדטרמיניסטי



ניתוח זמן הריצה לבניית אוטומט סופי דטרמינסטי

עתה ננתח את זמן הריצה של האלגוריתם שראינו לבניית אוטומט סופי דטרמיניסטי. לפני תחילת האלגוריתם מפעילים את השגרה לחישוב פונקצית הכישלון, אשר תדרוש זמן $0(m)$.

שורה 1 דורשת זמן $0(m)$.

שורה 2 דורשת זמן $0(|\Sigma|)$

שורה 3 דורשת זמן (בגלל עיקרון הכפל) $0(m\bullet|\Sigma|)$ כיוון שפונקצית הכישלון כבר מחושבת עוד לפני תחילת האלגוריתם הנדון.

לסיכום זמן הריצה של האלגוריתם לבניית האוטומט הסופי דטרמיניסטי הינו:

$$O(m+|\Sigma|+m\bullet|\Sigma|+m)=o(m\bullet|\Sigma|)$$

שורה1 שורה2 שורה3 קדם עיבוד,חישוב

אם מספר התוים הוא קבוע, כלומר $|\Sigma|=c$, כאשר c קבוע כלשהו אזי סיבוכיות זמן הריצה לבניית האוטומט סופי דטרמיניסטי הוא $0(m)$, אחרת $0(m\cdot|\Sigma|)$.

פעולת האוטומט הסופי להתאמת מחרוזות

נתונה תבנית $P=[P_0,P_1,\ldots P_{m-1}]$, וטקסט $T=[T_0,\ldots T_{n-1}]$ וכמו כן ניתן להניח שקיים אוטומט סופי דטרמיניסטי Mp עם Q מצבים, כאשר $Q=\{0,\ldots,m\}$ והמצב 0 הינו התחלתי והמצב $\{m\}$ הינו מצב סופי (מצב מקבל). הנחה זו ראליסטית מאחר וראינו בסעיף הקודם שניתן לבנות אוטומט סופי כזה בזמן $0(m)$.

עתה נבהיר את אופן הפעולה של אוטומט סופי Mp בעזרת קטע קוד המדמה את התנהגותו של Mp. כזכור המטרה העיקרית למצוא את המופעים של התבנית P בטקסט T. להלן קטע קוד:

1. אורך הטקסט $n\leftarrow T$
2. אורך התבנית $m\leftarrow P$
3. $q\leftarrow 0$

/* q מייצג מצב וכאמור בהתחלה האוטומט נמצא במצב התחלתי */

4. עבור j מאחד עד n בצע:

/* עבור על כל הטקסט */

4.1 $q\leftarrow \delta(q,T[I])$

4.2 אם $q=m$ /* הגענו למצב סופי וזיהינו את המופע של P בטקסט T */

אז בצע:

4.2.1 הדפס: קיים מופע של P בטקסט T החל מהמקום: I-m

5. סוף האלגוריתם

ניתוח זמן הריצה להתאמת מחרוזות

עתה ננתח את זמן הריצה לבעיית התאמת מחרוזות. קודם בונים אוטומט סופי דטרמיניסטי. ראינו כי סיבוכיות זמן הריצה לבניית אוטומט סופי דטרמיניסטי הינו:

$$\begin{cases} 0(m) & \text{אם } |\Sigma| \text{ קבוע} \\ 0(m*|\Sigma|) & \text{אחרת} \end{cases}$$

עתה נראה הזמן הנדרש בכל שורה של האלגוריתם האחרון:

שורה 1 תדרוש זמן $0(1)$, כי אורך הטקסט ידוע מראש.

שורה 2 תדרוש זמן $0(1)$, כי אורך התבנית ידוע מראש.
שורה 3 תדרוש זמן $0(1)$.
שורה 4 תדרוש זמן $0(n)$.

לסיכום זמן הריצה של בעיית התאמת מחרוזות הינו :
$0(m\cdot|\Sigma|)+0(1)+0(n)=0(n+m\cdot|\Sigma|)$

מסקנה : אם $|\Sigma|$ קבוע אזי זמן הריצה של האלגוריתם הנידון הינו $0(n+m)$,

אחרת הוא $0(n+m\cdot|\Sigma|)$.

<u>אלגוריתם ג' – בעיית התאמת מחרוזות</u>

עתה נציג אלגוריתם של Knuth_Morris_Pratt (בקיצור KMP) שרץ בזמן $0(n+m)$.
באלגוריתם החדש KMP לא מחשבים את פונקצית המעבר δ של אוטומט הסופי Mp, אך כן משתמשים בפונקצית הכישלון, f, אותה תיארנו בסעיף הקודם.
כמו כן ראינו כיצד מחשבים אותה מראש מן התבנית הנתונה P בזמן $0(m)$, כאשר $P=P_0\ldots P_{m-1}$, וכאשר $m=length(P)$. כזכור חישוב פונקצית המעבר-δ דרש זמן $0(m\cdot|\Sigma|)$ וכך נרצה לחסוך גורם של $|\Sigma|$. כפי שראינו בסעיף הקודם הערך $f(j)$ הוא אורכה של הרישא (prefix) הארוכה ביותר של P שהוא סיפא (suffix) ממש של $P_0\ldots P_{j-1}$, ואם אין רישא כזו אזי ערכו של $f(j)$ שווה לאפס.
לאור זאת הערך $f(j)$ מכיל את המידע הדרוש לחישוב $\delta(q,a)$ שלא תלוי ב- a ואין צורך לחשב את $\delta(q,a)$.

להלן האלגוריתם (KMP):

1. חישוב פונקצית הכישלון – f.
2. $q \leftarrow 0$

/*q מייצג מצב, וכאמור בהתחלה האוטומט נמצא במצב התחלתי */

3. עבור i מאפס עד n-1 בצע:

/*למצוא את המצב q כך ש - $P_0 \ldots P_{q-1} = T_{i-q-1} \ldots T_{i-1}$ */

3.1 כל עוד $q>0$ ו- $P[q] \neq T[i]$ בצע: $q \leftarrow f(q)$

/* עתה נבדוק האם $P_q = T_i$?*/

3.2 אם $T[i]=P[q]$

אז $q \leftarrow q+1$

/*עתה נבדוק האם הגענו למצב סופי – מקבל?*/

3.3 אם $q=m$ אז בצע:

3.3.1 הדפס: ישנה התאמה ממיקום i-m

3.3.2 $q=f(q)$

4. סוף האלגוריתם.

סיבוכיות זמן הריצה של האלגוריתם: $0(m+n)$

שורה 3 שורה 1

אלגוריתם קנות'-מוריס-פראט רץ בזמן של O(m+n) הקריאה ל- COMPUTE-PREFIX-FUNCTION, צורכת זמן של O(m) כפי שראינו זה עתה, וניתוח לשיעורין דומה, תוך שימוש בערכו של q כפונקצית הפוטנציאל, מראה שיתרת השגרה KMP-MATCHER מתבצעת בזמן O(n).

בהשוואה ל- FINITE-AUTOMATON-MATCHER על ידי שימוש ב- π במקום ב- δ קיצרנו את זמן העיבוד המקדים של התבנית מ- $O(m|\Sigma|)$ ל- O(m), תוך שמירה על חסם של O(m+n) על הזמן שעורכת ההתאמה עצמה.