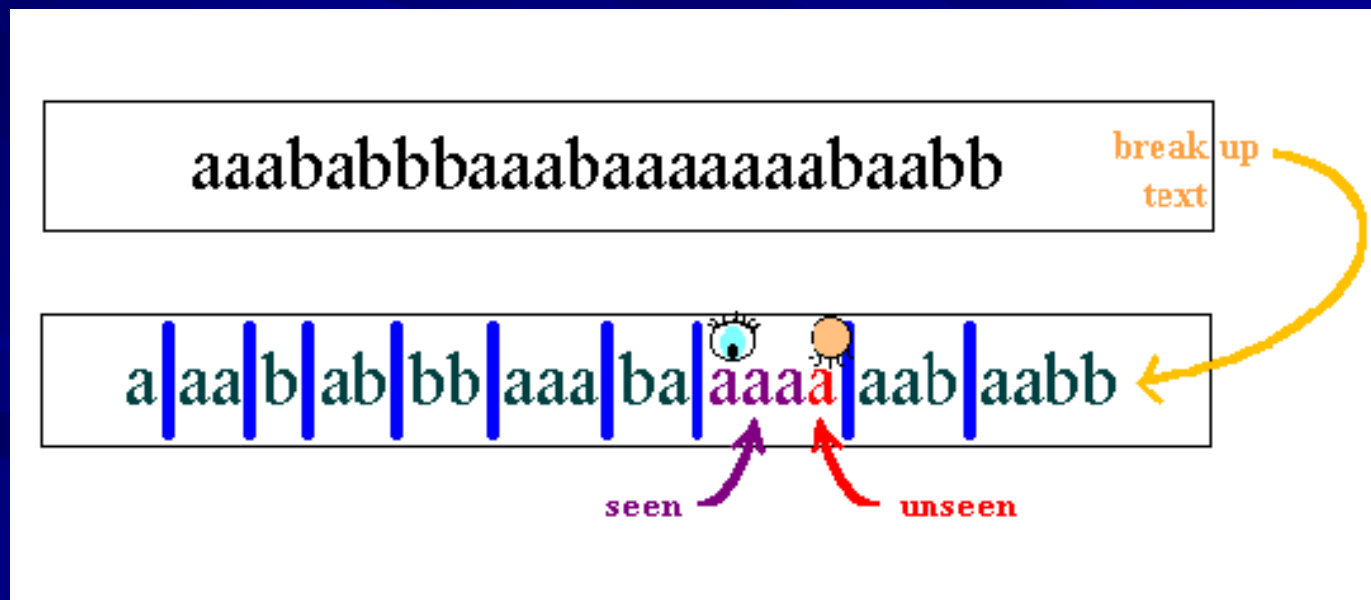


איך (עוד) מספרים סיפור בקיצור

- אומרים: היום, כל מה שקרה לי דומה למה שקרה לי אתמול, או ביום שלישי שעבר, חוץ מתוספת קטנה אחת, והיא ...
- למפל וזיו, שני פרופסורים מהטכניון, פרסמו בשנים 1977 ו-1978, שיטות לדחיסת ספרים (או תוכניות מחשב, או כל קובץ של אותיות) מבלי למצוא קודם את השכיחות היחסית של האותיות (או תתי-המחרוזות) בקובץ אותו רוצים לדחוס.

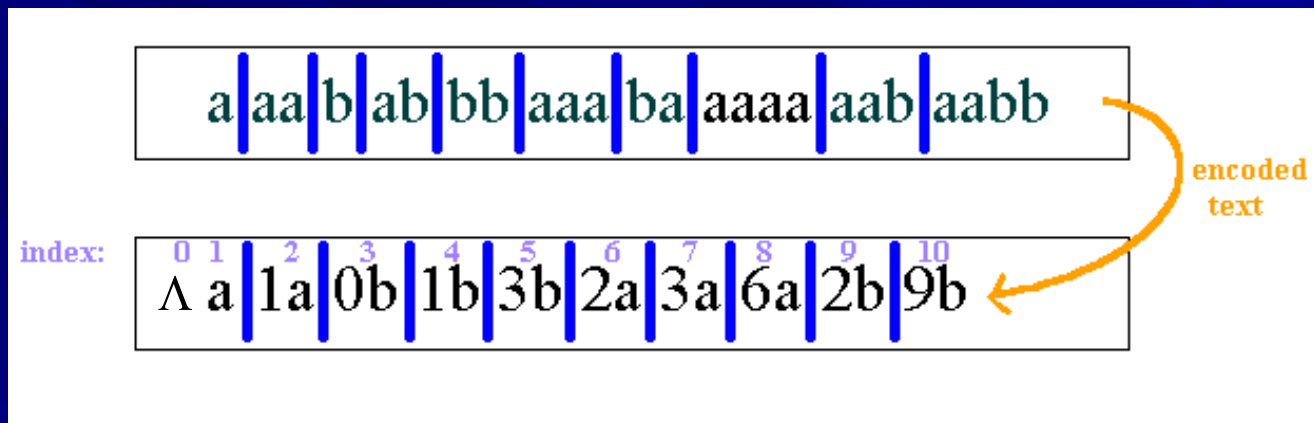
האלגוריתם של למפל וזיו 1978

מפסקים את הסדרה הנתונה לפסקאות שונות זו מזו, פסקה אחרי פסקה, מראשית המחרוזת אל סופה, כך שכל אחת מהפסקאות מורכבת מפסקה שפוסקה קודם לכן פלוס האות שבאה אחריה



האלגוריתם של למפל וזיו 1978

במקום כל פסקה, כותבים את מספר הפסקה שלה היא דומה ("בדיוק כמו ביום שלישי שעבר") ואת האות הנוספת ("חוץ מתוספת קטנה אחת והיא..")



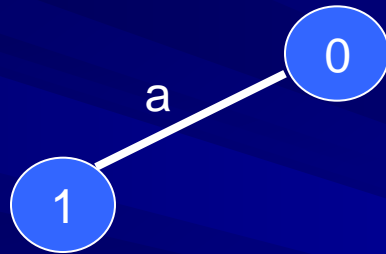
(0 הוא היום בו לא קרה כלום)

גם פה ה-Trie עוזר מאד

aaababbbbaaabaaaaaabaabb

גם פה ה-Trie עוזר מאד

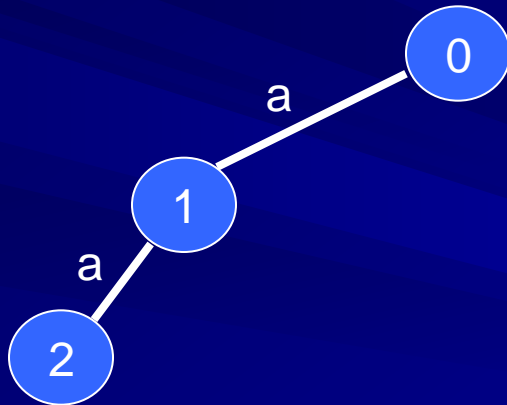
a,aababbbbaaabaabbaabb



(0,a)

גם פה ה-Trie עוזר מאד

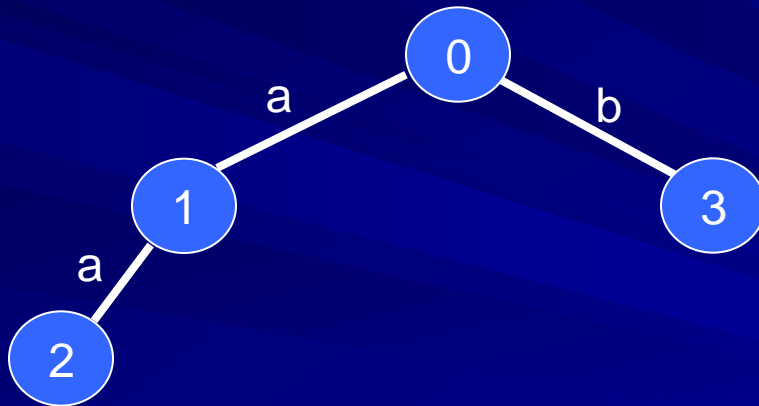
a,aa,babbbbaaabaababb



(0,a)(1,a)

גם פה ה-Trie עוזר מאד

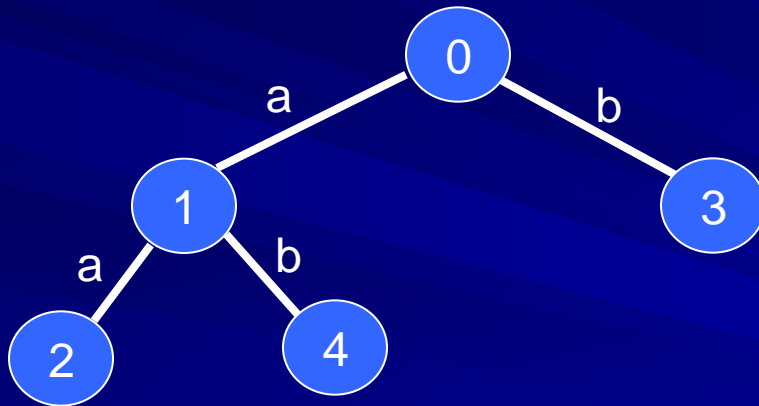
a,aa,b,abbbbaabaaaaaabaabb



(0,a)(1,a)(0,b)

גם פה ה-Trie עוזר מאד

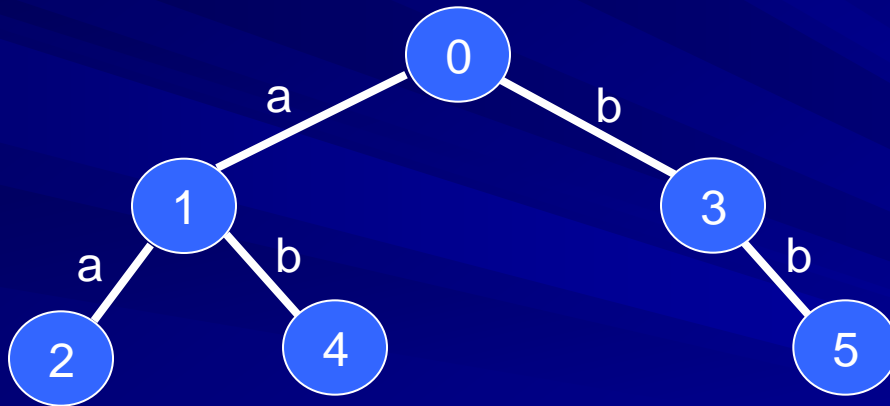
a,aa,b,ab,bbbaabaaaaaabaabb



(0,a)(1,a)(0,b)(1,b)

גם פה ה-Trie עוזר מאד

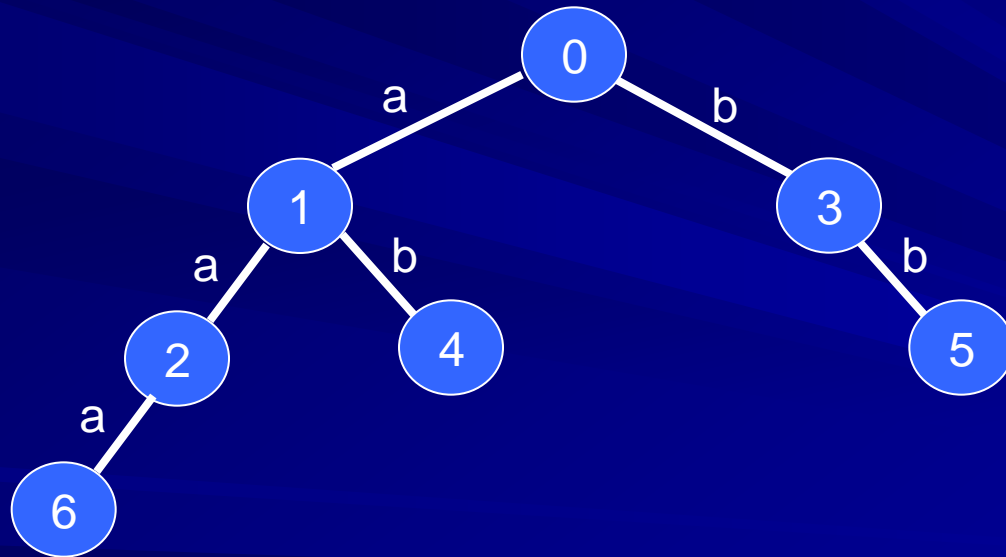
a,aa,b,ab,bb,aaabaaaaaabaabb



(0,a)(1,a)(0,b)(1,b)(3,b)

גם פה ה-Trie עוזר מאד

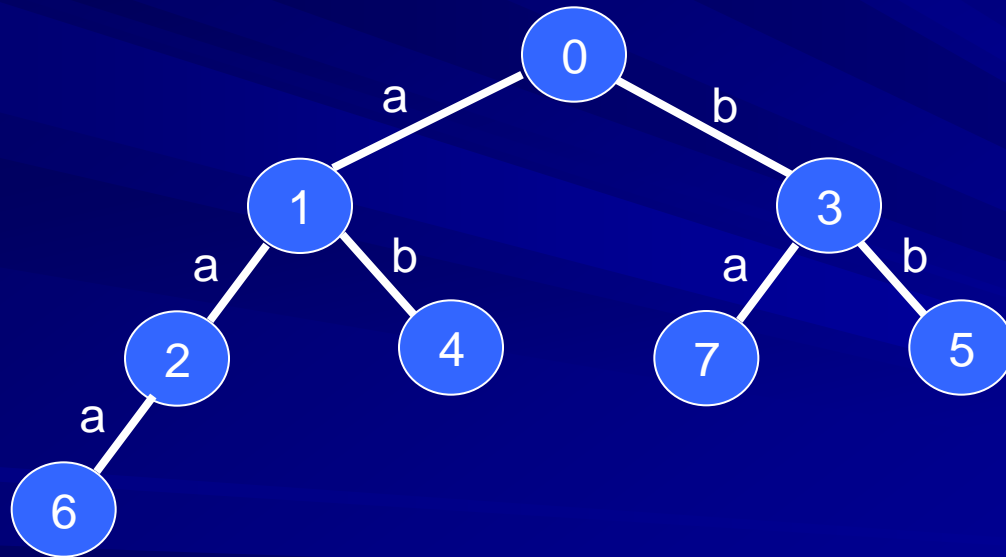
a,aa,b,ab,bb,aaa,baaaaaabaabb



(0,a)(1,a)(0,b)(1,b)(3,b)(2,a)

גם פה ה-Trie עוזר מאד

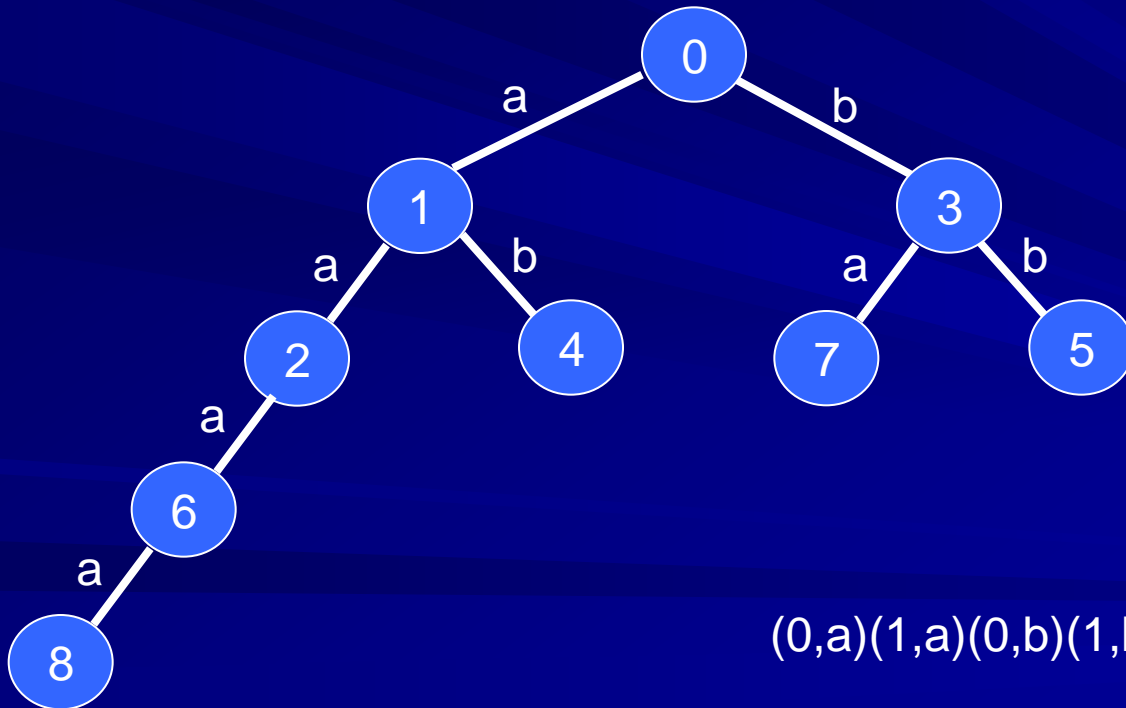
a,aa,b,ab,bb,aaa,ba,aaaaabaabb



(0,a)(1,a)(0,b)(1,b)(3,b)(2,a)(3,a)

גם פה ה-Trie עוזר מאד

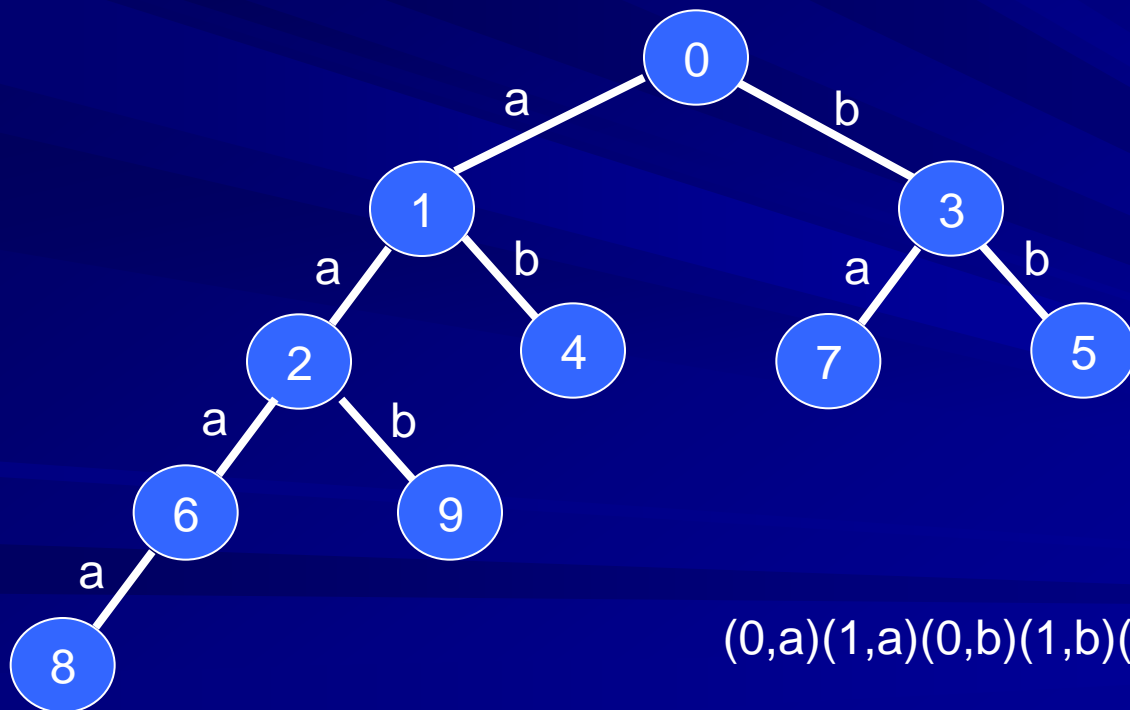
a,aa,b,ab,bb,aaa,ba,aaaa,aabaabb



(0,a)(1,a)(0,b)(1,b)(3,b)(2,a)(3,a)(6,a)

גם פה ה-Trie עוזר מאד

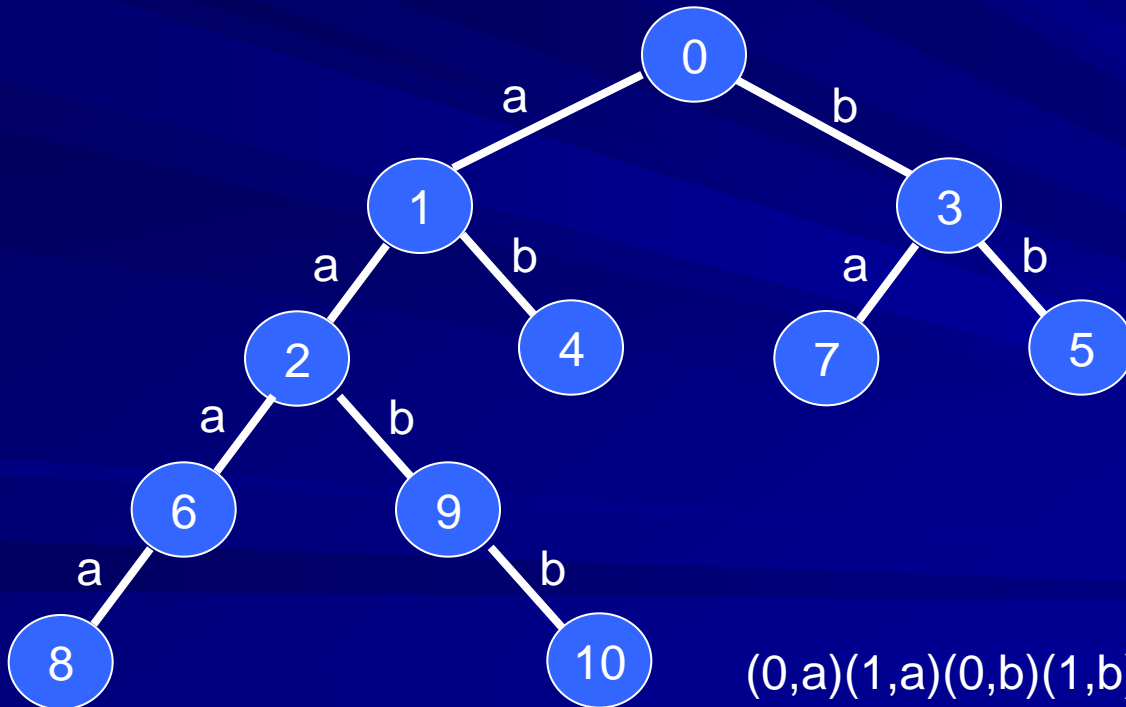
a,aa,b,ab,bb,aaa,ba,aaaa,aab,aabb



(0,a)(1,a)(0,b)(1,b)(3,b)(2,a)(3,a)(6,a)(2,b)

גם פה ה-Trie עוזר מאד

a,aa,b,ab,bb,aaa,ba,aaaa,aab,aabb



(0,a)(1,a)(0,b)(1,b)(3,b)(2,a)(3,a)(6,a)(2,b)(9,b)

הפלט מתורגם לבינארית

(0,a)(1,a)(0,b)(1,b)(3,b)(2,a)(3,a)(6,a)(2,b)(9,b)

1 2 3 4 5 6 7 8 9 10

נמספר את הקטעים המפוסקים מהסדרה

a,1a,00b,01b,011b,010a,011a,110a,0010b,1001b
בזוג ה- i , ניצג את האינדקס, שהוא תמיד מספר בין 0 ל- $i-1$, על פני $\lceil \log_2 i \rceil$ ביטים

0,10,001,011,0111,0100,0110,1100,00101,10011
נחליף כל אות ב- $\lceil \log_2 \sigma \rceil$ ביטים, כאשר σ הוא גודל הא"ב של הטקסט. בדוגמא שלנו,
הא"ב הוא $\{a,b\}$ ו- $\sigma=2$

01000101101110100011011000010110011

נמחק את הפסיקים

ארוך יותר מהטקסט המקורי... אבל לקבצים גדולים ישנה התכנסות מוכחת לאנטרופיה

למעשה, אפשר לייצר פלט בינארי עבור כל קטע מיד לאחר הגדרתו

- כאשר אנו מזהים (תוך ירידה בעץ) קטע חדש --
קטע i , ומזהים אותו כקטע j ($j < i$) ואחריו התו c ,
אנו יכולים להוציא מיד את היצוג של j על פני
 $\lceil \log_2 i \rceil$ ביטים, ולאחריו את היצוג הבינארי של c
על פני $\lceil \log_2 \sigma \rceil$ ביטים נוספים.
- ביטים אלה שנוציא, נשרשר אל קצה המחרוזת
הבינארית שאנו יוצרים.

ואיך מפענחים?

נתונה מחרוזת בינארית, צריך לייצר טקסט

- צעד אחרי צעד, קטע אחרי קטע, כפי שהוגדרו בתהליך ההצפנה:
- לפני ההתחלה: קטע מס' 0 מוגדר כקטע הריק, הקטע באורך 0.
- בצעד ה-1 קוראים $\lceil \log_2 \sigma \rceil$ ביטים מתוך המחרוזת הבינארית ומייצרים את הקטע הראשון, קטע מס' 1, שאורכו אות אחת – היא האות המיוצגת על ידי הביטים שקראנו. נרשום אות זו כאות הראשונה של הטקסט אותו אנו משחזרים.
- נניח שפענחנו ושרשרנו את כל הקטעים עד לקטע $i-1$. כדי לפענח את הקטע ה- i , נקרא, ראשית $\lceil \log_2 i \rceil$ ביטים מהמחרוזת הבינארית. אלה מייצגים את המספר הסיידורי j עבור $j < i$, של קטע שכבר מוכר לנו. נדביק קטע זה אל סוף הטקסט שאנו משחזרים. נקרא עוד $\lceil \log_2 \sigma \rceil$ ביטים מהמחרוזת הבינארית, אלה מייצגים אות שנקרא לה c , ונדביקה אל סוף הטקסט שאנו משחזרים. בכך סיימנו לפענח את הקטע ה- i , אשר מורכב משרשור הקטע ה- j והאות c , ואף הדבקנו אותו אל סוף הטקסט שאנו משחזרים.
- עם סיום קריאת המחרוזת הבינארית כולה, נסיים לשחזר את הטקסט.

מבני נתונים לפיענוח

אחד המבנים היעילים הוא קטע זיכרון רציף T לצורך יצירת הטקסט בתוכו, וטבלה שבה בשורה ה- i שני אינדקסים אל תוך T : אל המקום הראשון והאחרון שתופס בו הקטע ה- i שמגדיר האלגוריתם.

סוד הדחיסה

Variable to Fixed

(Huffman is Fixed to Variable) ■

קטעים קצרים וארוכים מקבלים אותה מילה בינארית: מספר

סידורי אל תוך אוסף הקטעים שנוצר עד כה

קטעים שכיחים מתארכים וקטעים נדירים – לא

רק הקטעים הרלוונטיים גדלים, ולא כולם

■ עם הפמן עבור מקורות שאינם חסרי זיכרון מאריכים את כולם

טוב לקבצים "מלאכת יד" -- אינם חסרי זיכרון, אבל קשה

לתפוס בדיוק את אורך הזכרון, והאם זהה עבור כל ההקשרים

השונים.

האלגוריתמים של למפל וזיו

- אלגוריתם אוניברסאלי – טוב לכל מחרוזת מבלי להתכונן אליה מראש או ללמוד את שכיחות אותיותיה ותת-מחרוזותיה.
- הוכח אופטימאלי לסדרות ארוכות מאד, נמצא בפועל טוב ויעיל. (תופס את תכונתנו לחזור על עצמנו בסדרות שאנו מיצרים)
- 1977 משמש ב-Winzip, 1978 ב-Compress של Unix
- שניהם בסטנדרטים של תקשורת בין מחשבים
- נוצרו חומרות רבות לביצועם
- מהפכה בתחום: עידן שיטות הדחיסה עם מילון, להבדיל מעידן הדחיסה הסטטיסטית.