

נקודות



- **אנו מזכיר** - נסמן n כמספר הפעולות שמבצעים באלגוריתם.
- **אקסום** - כל פעולה מוגדרת כפעולת $\text{move}(x)$ ביחס למשתנה x .
- **הנולות** הן גורם אחד בלבד שגורם לאלגוריתם לפעולת $\text{move}(x)$ ביחס למשתנה x .

מקרה נורמלי: במקרה נורמלי אלגוריתם יבצע את הפעולות כמפורט בפערקציה $T(n)$.

- (1) **הorst case** (best case) - הנולות מוגדרות כך שאלגוריתם יבצע מינימום פעולה.
- (2) **הorst case** (worst case) - הנולות מוגדרות כך שאלגוריתם יבצע מקסימום פעולה.
- (3) **הAverage case** (average case) - הנולות מוגדרות באופן אוניברסלי.

օסירות מילויים: הערך c_0, c_1, \dots, c_m מוגדר כערך שאלגוריתם יבצע בזיהוי i .
 $c_i = (\text{return})$, $1 = (+=, +=, \dots, -, +)$, $0 = (\text{move}(x))$.

- מילויים נורמליים: מילויים שאלגוריתם יבצע מינימום פעולה.
- מילויים נורמיים: מילויים שאלגוריתם יבצע מקסימום פעולה.
- מילויים נורמיים: מילויים שאלגוריתם יבצע אוניברסלי.

$$\begin{aligned} O(g(n)) &= \{f(n) \mid \exists n_0, c \in \mathbb{R}^+ \Rightarrow \forall n \geq n_0, 0 \leq f(n) \leq c \cdot g(n)\} \\ \Omega(g(n)) &= \{f(n) \mid \exists n_0, c \in \mathbb{R}^+ \Rightarrow \forall n \geq n_0, 0 \leq c \cdot g(n) \leq f(n)\} \\ \Theta(g(n)) &= \{f(n) \mid \exists n_0, c_1, c_2 \in \mathbb{R}^+ \Rightarrow \forall n \geq n_0, 0 \leq c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)\} \\ o(g(n)) &= \{f(n) \mid \exists n_0, c \in \mathbb{R}^+ \Rightarrow \forall n \geq n_0, 0 \leq f(n) < c \cdot g(n)\} \\ \omega(g(n)) &= \{f(n) \mid \exists n_0, c \in \mathbb{R}^+ \Rightarrow \forall n \geq n_0, 0 \leq c \cdot g(n) < f(n)\} \end{aligned}$$

- $O(n)$ מילויים נורמיים.
- $\Omega(n)$ מילויים נורמיים.
- $\Theta(n)$ מילויים נורמיים.
- $o(n)$ מילויים נורמיים.
- $\omega(n)$ מילויים נורמיים.

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = C$$

- ($C < \infty$) : $f(n) = O(g(n))$
- ($C > 0$) : $f(n) = \Omega(g(n))$
- ($0 < C < \infty$) : $f(n) = \Theta(g(n))$
- ($C = 0$) : $f(n) = o(g(n))$
- ($C = \infty$) : $f(n) = \omega(g(n))$

Boyer-Moore Majority Voting Algorithm

count $\leq \frac{n}{2}$ \rightarrow Time $\Theta(n)$ Space $O(1)$

```
def majorityElement(self, nums: List[int]) -> int:
    candidate = None
    counter = 0
    for n in nums:
        if counter == 0:
            candidate = n
        counter += 1 if n == candidate else -1
    return candidate
```

- הינה מבחן שבודק אם קיימת איבר אחד שקיים ב- n מינימום פעולות.
- ביחס לאיבר x מוגדר $\text{move}(x)$ כפעולת $\text{move}(x)$ ביחס למשתנה x .
- ביחס למשתנה x מוגדר $\text{move}(x)$ כפעולת $\text{move}(x)$ ביחס למשתנה x .
- אם x מופיע לפחות $\lceil \frac{n}{2} \rceil$ פעמים אז הוא מוגדר $\text{move}(x)$ כפעולת $\text{move}(x)$.
- מבחן שבודק אם קיימת איבר אחד שקיים ב- n מינימום פעולות.

טורים

טור ארכיטמטי (סדרה חשבונית):

$$\sum_{k=1}^n k = 1+2+\dots+n = \frac{1}{2}n(n+1)$$

$$\sum_{k=0}^n x^k = \frac{x^{n+1}-1}{x-1}$$

$$\sum_{k=0}^{\infty} x^k = \frac{1}{1-x} \quad |x| < 1 \quad S_n = \frac{x^{n+1}-1}{x-1}$$

$$H_n = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n} = \sum_{k=1}^n \frac{1}{k} - \ln n + O(1)$$

טור הרמוני:

$$\sum_{k=1}^n k^2 = \frac{n(n+1)(2n+1)}{6}$$

סדרת הריבועים:

$$\sum_{k=1}^n k^3 = \frac{n^2(n+1)^2}{4}$$

$$\sum_{k=1}^n k^4 = \frac{n^2(n+1)^2(2n+1)}{30}$$

$$\sum_{k=1}^n k^5 = \frac{n^2(n+1)^2(2n^2+2n-1)}{120}$$

Pattern Matching

תבנית בדקה

מג'ה: מילא מחרט הגדלת פונט חקוקה תחילה. ואכן הוא מזכיר לנו שפה אוטומטית. ומשם נולש המשמעות של מחרט זה. אם לא מחרט לא ניתן לארוך צורה מסוימת. אם צורה מסוימת לא ניתן לארוך צורה מסוימת, אז צורה מסוימת לא ניתן לארוך צורה מסוימת.

```

For i=1 to n-m+1 do
    match←1
    For j=1 to m do
        If P[j]≠T[i+j-1] then match←0
    End
    If match=1 then output "match at location " i
End

```

תבנית בדקה: מחרט בזאת מילא מחרט, כי מילא מחרט לא ניתן לארוך צורה מסוימת. לא ניתן לארוך צורה מסוימת, אם צורה מסוימת לא ניתן לארוך צורה מסוימת. אם צורה מסוימת לא ניתן לארוך צורה מסוימת, אז צורה מסוימת לא ניתן לארוך צורה מסוימת. אולם לא ניתן לארוך צורה מסוימת, אם צורה מסוימת לא ניתן לארוך צורה מסוימת.

$O(|T| \cdot |P|)$

מבחן בדקה: מילא מחרט מילא מחרט, כי מילא מחרט לא ניתן לארוך צורה מסוימת. לא ניתן לארוך צורה מסוימת, אם צורה מסוימת לא ניתן לארוך צורה מסוימת. אולם לא ניתן לארוך צורה מסוימת, אם צורה מסוימת לא ניתן לארוך צורה מסוימת.

KMP

Knuth-Morris-Pratt

Compute-Prefix-Function(P)

```

m←length[P]
π[1]←0
k←0
for q←2 to m
    do while k>0 and P[k+1]≠P[q]
        do k←π[k]
    if P[k+1]=P[q] then k←k+1
    π[q]←k
return π

```

מבחן בדקה: מילא מחרט מילא מחרט, כי מילא מחרט לא ניתן לארוך צורה מסוימת. לא ניתן לארוך צורה מסוימת, אם צורה מסוימת לא ניתן לארוך צורה מסוימת. אולם לא ניתן לארוך צורה מסוימת, אם צורה מסוימת לא ניתן לארוך צורה מסוימת.

KMP-Matcher(T,P)

```

n←length[T]
m←length[P]
π←Compute-Prefix-Function(P)
q←0
for i←1 to n
    do while q>0 and P[q+1]≠T[i]
        do q←π[q]
    if P[q+1]=T[i] then q←q+1
    if q=m then print "Pattern occurs with shift" i-m
    q←π[q]

```

מבחן בדקה: מילא מחרט מילא מחרט, כי מילא מחרט לא ניתן לארוך צורה מסוימת. לא ניתן לארוך צורה מסוימת, אם צורה מסוימת לא ניתן לארוך צורה מסוימת. אולם לא ניתן לארוך צורה מסוימת, אם צורה מסוימת לא ניתן לארוך צורה מסוימת.

P =	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
T =	0	0	1	2	0	1	(2)	0	1	2	3	4	5	6	(7)	3	4	5	6	7	8
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21

מבחן בדקה: מילא מחרט מילא מחרט, כי מילא מחרט לא ניתן לארוך צורה מסוימת. לא ניתן לארוך צורה מסוימת, אם צורה מסוימת לא ניתן לארוך צורה מסוימת. אולם לא ניתן לארוך צורה מסוימת, אם צורה מסוימת לא ניתן לארוך צורה מסוימת.

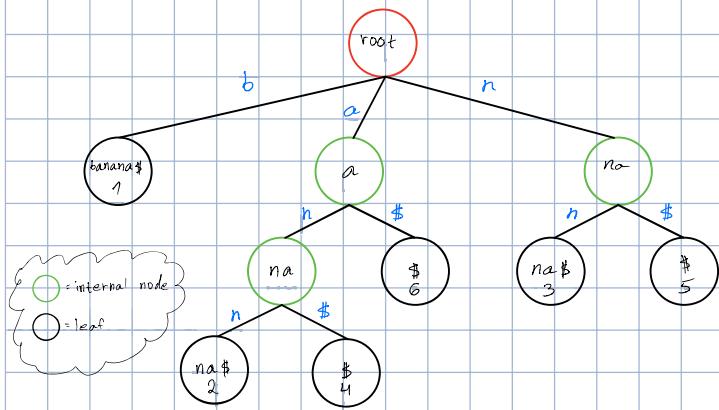
P =	1	2	3	4	5	6	7	8	9
T =	0	1	0	1	(2)	3	4	(5)	2
	1	2	3	4	5	6	7	8	9

$$\rho_{k+3} = \rho_{q+3}$$

$$k = 1 \quad b_1 = \alpha \quad k+2 = 2$$

מבחן בדקה: מילא מחרט מילא מחרט, כי מילא מחרט לא ניתן לארוך צורה מסוימת. לא ניתן לארוך צורה מסוימת, אם צורה מסוימת לא ניתן לארוך צורה מסוימת. אולם לא ניתן לארוך צורה מסוימת, אם צורה מסוימת לא ניתן לארוך צורה מסוימת.

Indexing - Suffix Tree



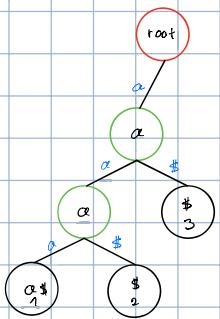
ריבוע תרשים של סט סיסם
 סט סיסם T מוגדר כטביעה של סדרת סיסם S
 סט סיסם T מוגדר כטביעה של סדרת סיסם S
 $|T| = \# \text{nodes}$ סט סיסם T מוגדר כטביעה של סדרת סיסם S
 $|T| - 1$ סט סיסם T מוגדר כטביעת סיסם S
 $|T| + (|T| - 1) + 1 = 2|T|$ סט סיסם T מוגדר כטביעת סיסם S

```

class SuffixTreeNode {
    // begin/end of current sub-string
    int start;
    int end; // end=-1 means $
    // MAX is a constant = 256
    SuffixTreeNode *children[MAX];
    // relevant for leaf nodes only.
    int suffixPos;
}
    
```

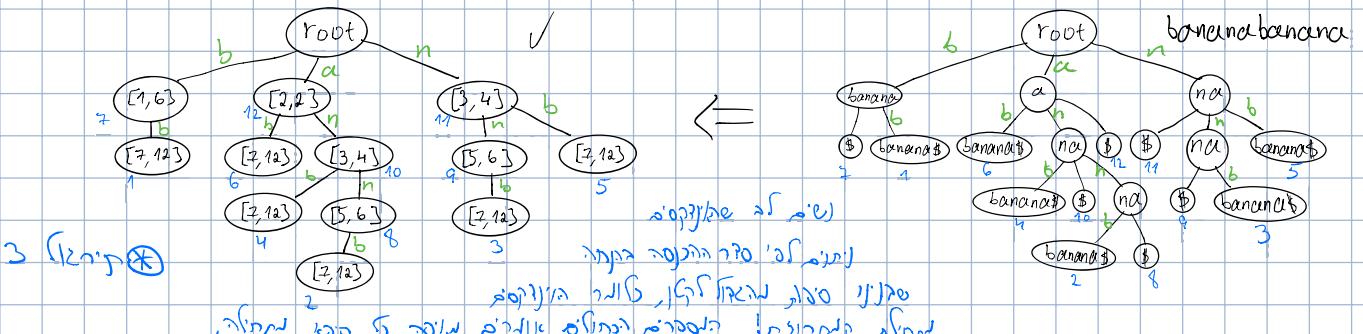
הטענה שsuffixTree(S) מוגדר כטביעת סיסם S היא נכונה. נוכיח זאת על ידי הוכחה אינדוקטיבית על $|S|$.
 בסיסי: $|S| = 1$. הטענה נכונה כי suffixTree(S) מוגדר כטביעת סיסם S .
 מילוי: נניח שהטענה נכונה עבור כל S אשר $|S| < k$. נוכיח שהיא נכונה גם עבור S אשר $|S| = k$.
 נסמן $T = \text{suffixTree}(S)$. נוכיח כי T מוגדר כטביעת סיסם S .

הטענה שsuffixTree(S) מוגדר כטביעת סיסם S היא נכונה. נוכיח זאת על ידי הוכחה אינדוקטיבית על $|S|$.
 בסיסי: $|S| = 1$. הטענה נכונה כי suffixTree(S) מוגדר כטביעת סיסם S .
 מילוי: נניח שהטענה נכונה עבור כל S אשר $|S| < k$. נוכיח שהיא נכונה גם עבור S אשר $|S| = k$.
 נסמן $T = \text{suffixTree}(S)$. נוכיח כי T מוגדר כטביעת סיסם S .



הטענה שsuffixTree(S) מוגדר כטביעת סיסם S היא נכונה. נוכיח זאת על ידי הוכחה אינדוקטיבית על $|S|$.
 בסיסי: $|S| = 1$. הטענה נכונה כי suffixTree(S) מוגדר כטביעת סיסם S .
 מילוי: נניח שהטענה נכונה עבור כל S אשר $|S| < k$. נוכיח שהיא נכונה גם עבור S אשר $|S| = k$.
 נסמן $T = \text{suffixTree}(S)$. נוכיח כי T מוגדר כטביעת סיסם S .

לעתה נוכיח כיsuffixTree(S) מוגדר כטביעת סיסם S . נוכיח זאת על ידי הוכחה אינדוקטיבית על $|S|$.
 בסיסי: $|S| = 1$. הטענה נכונה כיsuffixTree(S) מוגדר כטביעת סיסם S .
 מילוי: נניח שהטענה נכונה עבור כל S אשר $|S| < k$. נוכיח שהיא נכונה גם עבור S אשר $|S| = k$.
 נסמן $T = \text{suffixTree}(S)$. נוכיח כיsuffixTree(S) מוגדר כטביעת סיסם S .



הטענה שsuffixTree(S) מוגדר כטביעת סיסם S היא נכונה. נוכיח זאת על ידי הוכחה אינדוקטיבית על $|S|$.
 בסיסי: $|S| = 1$. הטענה נכונה כיsuffixTree(S) מוגדר כטביעת סיסם S .
 מילוי: נניח שהטענה נכונה עבור כל S אשר $|S| < k$. נוכיח שהיא נכונה גם עבור S אשר $|S| = k$.
 נסמן $T = \text{suffixTree}(S)$. נוכיח כיsuffixTree(S) מוגדר כטביעת סיסם S .

הנחתה ותורת הסתברות

לט' נושא: Las Vegas

לט' נושא: Monte Carlo

לט' נושא: מוגדרות אוניברסליות

$P(A \cup B) = P(A) + P(B) - P(A \cap B)$, $P(\bar{A}) = 1 - P(A)$, $P(\emptyset) = 0$: מוגדרות אוניברסליות

לט' נושא: סכום כוכב ב- A בסיסי

$X: \Omega \rightarrow \mathbb{R}$ מוגדרת כפונקציית סכום כוכב $E(X) = \sum_{i=1}^n x_i \cdot P(X=x_i)$

$x_1=0, x_2=1, P(X=0)=0.5, P(X=1)=0.5 \Rightarrow E(X) = 0 \cdot 0.5 + 1 \cdot 0.5 = 0.5$

$\sum_{i=1}^6 x_i = i, P(X_i) = \frac{1}{6} \Rightarrow E(X) = 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} + 3 \cdot \frac{1}{6} + 4 \cdot \frac{1}{6} + 5 \cdot \frac{1}{6} + 6 \cdot \frac{1}{6} = 3.5$

לט' נושא: (1) אם היחסן i הוא זר ב- Ω , אז $E(X) = \sum_{x_i \in \Omega} x_i \cdot P(X=x_i)$. (2) אם i לא זר ב- Ω , אז $E(X) = \sum_{x_i \in \Omega, x_i \neq i} x_i \cdot P(X=x_i)$.

לט' נושא: מוגדרות אוניברסליות

לט' נושא: מוגדרות אוניברסליות

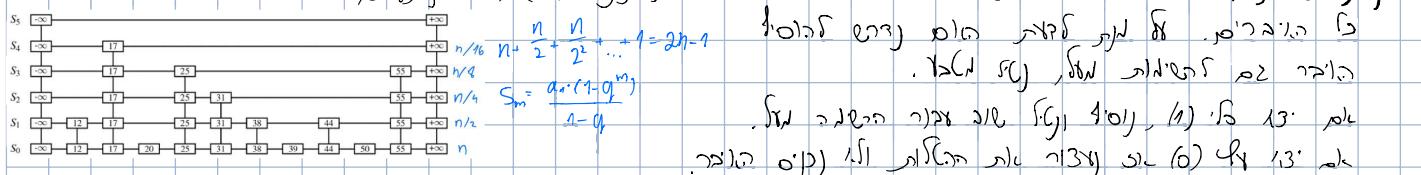
$\Theta(\log n)$: מוגדרות אוניברסליות

- `createDictionary()`
- `insert(value)`
- `remove(value)`
- `find(value)`

$\Theta(n) \leq \Theta(n \log n)$: מוגדרות אוניברסליות

(2) `search(logn)`: מוגדרות אוניברסליות

(3) `delete(logn)`: מוגדרות אוניברסליות



```

PARTITION( $A, p, q$ ) ▷  $A[p..q]$ 
 $x \leftarrow A[p]$  ▷ pivot =  $A[p]$ 
 $i \leftarrow p$ 
for  $j \leftarrow p+1$  to  $q$ 
do if  $A[j] \leq x$ 
then  $i \leftarrow i+1$ 
exchange  $A[i] \leftrightarrow A[j]$ 
exchange  $A[p] \leftrightarrow A[i]$ 
return  $i$ 

```

Running time
 $= O(n)$ for n elements.

Invariant: $\begin{matrix} x & \leq x & \geq x & ? \\ p & i & j & q \end{matrix}$

לט' נושא: $\text{Partition}(A, p, q)$

לט' נושא: מוגדרות אוניברסליות

לט' נושא: מוגדרות אוניברסליות

לט' נושא: מוגדרות אוניברסליות

לט' נושא: מוגדרות אוניברסליות

לט' נושא: $\text{Rand-Partition}(A, p, q)$

לט' נושא: $\text{Quicksort}(A, p, r)$

if $p < r$
then $q \leftarrow \text{PARTITION}(A, p, r)$
 $\text{QUICKSORT}(A, p, q-1)$
 $\text{QUICKSORT}(A, q+1, r)$

Initial call: $\text{QUICKSORT}(A, 1, n)$

לט' נושא: $\text{Quicksort}(A, p, q)$

$a \equiv 2^t \cdot u$ $a \equiv 2^t \cdot u$ $t > 0 \rightarrow 1 \leq t \leq \lfloor \log_2 n \rfloor$.
 $a^2 \equiv (2^t \cdot u)^2 \pmod{n}$ $\Leftrightarrow 2^{2t} \equiv 1 \pmod{n}$

עדות לפירוקות (a, n) : האלגוריתם יזכיר את א"א a שהוא "עדי" לפירוקות של n .

WITNESS(a, n)

1. let $n-1=2^tu$, where $t \geq 1$ and u is odd

2. $x_0 \leftarrow a^u \pmod{n}$

3. for $i \leftarrow 1$ to t

4. do $x_i \leftarrow (x_{i-1})^2 \pmod{n}$ $\leftarrow (a^u \pmod{n})^2 \pmod{n}$

5. if $x_i = 1$ and $x_{i-1} \neq 1$ and $x_{i-1} \neq n-1$

6. then return **true**

7. if $x_i \neq 1$ then return **false**

8. return **false**

גלו של שוטש 1
טרוייאלי של 1
המספר או הטעון

$\alpha^{n-1} \equiv (\alpha^u)^{2^t} \pmod{n}$

$O(\beta^3)$ $\beta = \log n$

אלגוריתם הינה מוגן מכך ש a יגן על n , והוא מומלץ.

אלגוריתם מילר רבין

MILLER-RABIN(n, s)

1. for $i \leftarrow 1$ to s
2. do $a \leftarrow \text{RANDOM}(1, n-1)$
3. if **WITNESS(a, n)**
4. then return **COMPOSITE**
5. return **PRIME**

$((x_{i-1})^2 \pmod{n})$ $\beta = \log n$

$\beta = \log(n-1) \leq \log n$ $\beta = \log n$

$O(\beta^3)$

- סיבוכיותה של פונקציית witness $= O(\beta^3)$

- סיבוכיותה של פונקציית $\beta = \log n$

בנוסף לאם n מוגן או לא-מוגן נקבעת סיבוכיותה של פונקציית witness $= O(s \cdot \log^3 n)$

$\frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{8}$ $s=2$ מוגן n מוגן \Rightarrow מוגן n

$$\frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{8} = \frac{1}{2^3} \quad \text{אברוי}$$

$a^n \equiv 1 \pmod{p}$

הסימן \Leftrightarrow חישוב

$$a^n = a^{n_0+2n_1+2^2n_2+\dots+2^kn_k} = (a^{n_0} \cdot (a^2)^{n_1} \cdot \dots \cdot (a^{2^k})^{n_k})$$

- הסימן \Leftrightarrow חישוב

$$(a^0=1 \Leftrightarrow n_0=0) \quad \text{...} \quad (a^{2^k}=1 \Leftrightarrow n_k=0)$$

- הסימן \Leftrightarrow חישוב

$$a^{2^{i+1}} = (a^{2^i})^2 \quad \text{...} \quad (a^{2^0})^2 = 1$$

- הסימן \Leftrightarrow חישוב

$$\begin{cases} C_0 = a \pmod{p} \\ C_i = C_{i-1}^2 \pmod{p}, \quad i=1, \dots, k \end{cases}$$

- הסימן \Leftrightarrow חישוב

$x \leftarrow 1$

for $i \leftarrow 0$ to k do:

if $b_i \neq 0$ then:

$$x \leftarrow (x \cdot C_i) \pmod{p}$$

end for

$x = a^n \pmod{p}$ \Leftrightarrow חישוב

לע'ן הירקן כלאי

השאלה מילוי טרUTHFULNESS: מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ כך ש $\sum_{i=1}^n b_i x_i = 1$. כלומר, מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ כך ש $\sum_{i=1}^n b_i x_i = 1$.

$S \subseteq \{0,1\}^n$ מילוי טרUTHFULNESS, $1 \leq i \leq n$ $x_i \in \{0,1\}$ $-c \in \mathbb{C}$ $x = (x_1, x_2, \dots, x_n)$ - קיימת גורם $B = (b_1, b_2, \dots, b_n)$ כך ש $\sum_{i=1}^n b_i x_i = 1$. כלומר, קיימת גורם $B = (b_1, b_2, \dots, b_n)$ כך ש $\sum_{i=1}^n b_i x_i = 1$.

תורת המילויים: מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ כך ש $\sum_{i=1}^n b_i x_i = 1$. כלומר, מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ כך ש $\sum_{i=1}^n b_i x_i = 1$.

תורת המילויים: מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ כך ש $\sum_{i=1}^n b_i x_i = 1$. כלומר, מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ כך ש $\sum_{i=1}^n b_i x_i = 1$.

- לא ניתן לcompute את מילוי טרUTHFULNESS ב- $O(n)$. סולוונט נון-דטרמיניסטי (NP-Complete).

מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ כך ש $\sum_{i=1}^n b_i x_i = 1$. כלומר, מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ כך ש $\sum_{i=1}^n b_i x_i = 1$.

תורת המילויים: מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ כך ש $\sum_{i=1}^n b_i x_i = 1$. כלומר, מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ וכך נון-דטרמיניסטי (NP-Complete).

(2) מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ כך ש $\sum_{i=1}^n b_i x_i = 1$. כלומר, מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ וכך נון-דטרמיניסטי (NP-Complete).

מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ כך ש $\sum_{i=1}^n b_i x_i = 1$. כלומר, מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ וכך נון-דטרמיניסטי (NP-Complete).

מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ וכך נון-דטרמיניסטי (NP-Complete).

מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ וכך נון-דטרמיניסטי (NP-Complete).

GREEDY-ACTIVITY-SELECTOR(s, f)

```

1   $n \leftarrow \text{length}[s]$ 
2   $A \leftarrow \{a_1\}$ 
3   $i \leftarrow 1$ 
4  for  $m \leftarrow 2$  to  $n$ 
5    do if  $s_m \geq f_i$ 
       then  $A \leftarrow A \cup \{a_m\}$ 
6     $i \leftarrow m$ 
7  return  $A$ 
```



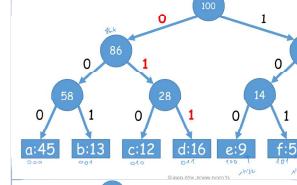
מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ וכך נון-דטרמיניסטי (NP-Complete).

מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ וכך נון-דטרמיניסטי (NP-Complete).

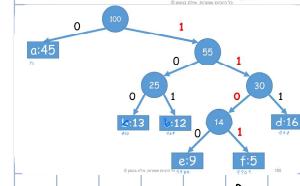
מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ וכך נון-דטרמיניסטי (NP-Complete).

- איגודים: ($O(n \log n)$)

מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ וכך נון-דטרמיניסטי (NP-Complete).



מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ וכך נון-דטרמיניסטי (NP-Complete).



מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ וכך נון-דטרמיניסטי (NP-Complete).

מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ וכך נון-דטרמיניסטי (NP-Complete).

מילוי טרUTHFULNESS מוגדרת כ $x \in \{0,1\}^n$ וכך נון-דטרמיניסטי (NP-Complete).

$$T(n) = \begin{cases} e & n < d \\ \alpha \cdot T(n/b) + f(n) & n \geq d \end{cases}$$

אנו מודים בפונקציית $f(n)$ על מנת לפשט את הטענה.

$$T(n) = \Theta(n^{\log_b a}) \quad \text{אם } \epsilon > 0 \text{ וקיים } f(n) = O(n^{\log_b a - \epsilon}) \text{ כך } \exists c_1, c_2 \text{ כך}$$

$$T(n) = \Theta(n^{\log_b a} \cdot \log n) \quad \text{אם } f(n) = O(n^{\log_b a}) \text{ כך } \exists c_1, c_2 \text{ וכך}$$

$$T(n) = \Theta(n^{\log_b a} \cdot \log^k n) \quad \text{אם } f(n) = O(n^{\log_b a - \epsilon}) \text{ וכך}$$

$$\text{אם } n-1 < b < 1 \quad \forall k \geq 1 \quad a \cdot f(n/b) \leq c \cdot f(n) \quad \text{אם } \epsilon > 0 \text{ וקיים } f(n) = \Omega(n^{\log_b a + \epsilon}) \text{ וכך}$$

$$\lim_{n \rightarrow \infty} \frac{a \cdot f(n/b)}{f(n)} = ? \leq c \quad T(n) = \Theta(f(n))$$

$a > 1$ ו $b > 1$ ו $a > b$ ו $a \cdot f(n/b) \leq c \cdot f(n)$ \Rightarrow $f(n) = \Theta(n^{\log_b a})$ \Rightarrow $\Theta(n^{\log_b a}) \leq \Theta(n^{\log_b a})$

$a > 1$ ו $b > 1$ ו $a < b$ ו $a \cdot f(n/b) \leq c \cdot f(n)$ \Rightarrow $f(n) = \Theta(n^{\log_b a})$ \Rightarrow $\Theta(n^{\log_b a}) \leq \Theta(n^{\log_b a})$

$a > 1$ ו $b > 1$ ו $a = b$ ו $a \cdot f(n/b) \leq c \cdot f(n)$ \Rightarrow $f(n) = \Theta(n^{\log_b a})$ \Rightarrow $\Theta(n^{\log_b a}) \leq \Theta(n^{\log_b a})$

בכל מקרה $T(n) = \Theta(n^{\log_b a})$

$$T(n) = 2T(n/2) + bn =$$

$$= 2^2 T(n/2^2) + 2bn =$$

$$= 2^3 T(n/2^3) + 3bn =$$

$$= 2^i T(n/2^i) + ibn =$$

$$= 2^{\log_2 n} T(1) + bn \cdot \log n = n \cdot bn \cdot \log n = \Theta(n \cdot \log n)$$

במקרה של $n = 2^i$ נקבל $i = \log_2 n$ ו $T(1) = T(n/2^i)$.

במקרה של $n = 2^i$ נקבל $i = \log_2 n$ ו $T(1) = T(n/2^i)$.

$$T(n) = T(\sqrt{n}) + 1 \quad \sqrt{n} = n^{1/2}$$

$$n = 2^m \Leftrightarrow m = \log n \quad \text{במקרה של } n = 2^m$$

$$T(n) = T(2^m) = T(2^{m/2}) + 1 \quad \text{במקרה של } m = \log n$$

$$S(m) = S(m/2) + 1 \quad \Leftarrow S(m) = T(2^m) \quad \Rightarrow S(m) = T(2^m) \Rightarrow S(m/2) = T(2^{m/2})$$

$$a=1, b=2, d(n)=1 \quad \text{ולכן } S(m) = \Theta(m)$$

$$m^{\log_2 1} = 1 = S(m)$$

$$\Theta(\log m) \quad : 2 \text{ כפלי גורם}$$

$$\Theta(\log(\log n)) \quad \text{במקרה של } m = \log n \text{ ו } m = \log n$$

3. מטלות

לעומת הולמת חישובים, מטלות מצריכים לארון תוצאות. מטלות מוגדרות כפונקציות המבצעות פעולה מסוימת על אובייקט מסוים. מטלות מוגדרות באמצעות פונקציית מילוי. מילוי מוגדרת כפונקציה המבצעת פעולה מסוימת על אובייקט מסוים.

מתקני מחשבים מוגדרים באמצעות מטלות. מטלות מוגדרות כפונקציות המבצעות פעולה מסוימת על אובייקט מסוים.

(ב) מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים.

(ג) מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים.

(ד) מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים.

מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים: מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים.

```

max=0.
S_max=0.
for i=1,...,n do
    if v_i > max then
        max=v_i.
        S_max=S.
    end if.
end for.

```

למקרה של מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים, נזכיר את הדוגמה של מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים.

```

Compute-Opt(j)
if j = 0 then return 0
return max(v_j + Compute-Opt(p(j)), Compute-Opt(j-1))

```

• מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים.

• מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים.

Iterative-Compute-Opt

```

M[0]=0
for j=1,2,...,n do
    M[j]=max(v_j+M[p(j)], M[j-1])

```

Find-Solution(j)

```

if j=0 then Output nothing
Else if v_j+M[p(j)] ≥ M[j-1] then
    Output j together with the result of Find-Solution(p(j))
Else
    Output the result of Find-Solution(j-1)

```

***Back-Tracking:** בטלת תוצאות, בזריקת תוצאות.

במקרה שבו מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים, מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים.

M-Compute-Opt(j)

```

if j=0 then return 0
If M[i] is not empty then return M[i]
M[j]=max(v_j+M-Compute-OPT(p(j)), M-Compute-OPT(j-1))
return M[j]

```

• מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים.

- מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים.

• מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים.

• מטלות מוגדרות באמצעות פונקציות המבצעות פעולה מסוימת על אובייקט מסוים.

שאלה גדיין שיברר מחרן מחרן? יא...?

- (1) מחרן מחרן נתקע בדוחן מחרן.
- (2) מחרן רקיוטגיט נתקע בדוחן מחרן.
- (3) מחרן און כהן נתקע בדוחן מחרן.
- (4) מחרן ווילן נתקע בדוחן מחרן.

לצורך:

$$A = (10, 1, 2, 5, \cancel{37}, 100)$$

$$A' = (10, 5, \cancel{37})$$

בצ"ר גע סוגה מחרן מחרן מחרן מחרן?

$a_i \in \mathbb{R}$: נסובב מחרן מחרן מחרן מחרן $A = (a_1, a_2, \dots, a_n)$

$1 \leq i_1 < i_2 < \dots < i_k \leq n$ מחרן מחרן $A' = (a_{i_1}, a_{i_2}, \dots, a_{i_k})$ מחרן מחרן

השאלה: מחרן מחרן מחרן מחרן?

השאלה: מחרן מחרן?

MAX_INDEPENDENT_ARRAY(A[1..i])

```
F[0] = 0
F[1] = max(0, A[1])
for i=2 to n
    F[i] = max(F[i-1], A[i]+F[i-2])
return F[n]
```

Find-Solution(i)

```
if i=0 then Output nothing
if i=1 then Output A[1]

if A[i]+F[i-2] ≥ F[i-1] then
    Output A[i] together with the result of Find-Solution(i-2)
Else
    Output the result of Find-Solution(i-1)
```

MAX_INDEPENDENT_ARRAY(A[1..i])

```
X = 0
Y = max(0, A[1])
for i=2 to n
    Z = max(Y, A[i]+X)
    X=Y, Y=Z
return F[n]
```

Ο(1) ימיהו

Ο(1)

סבוכנות זמן: Θ(n)

Θ(n)

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?

בנוסף לזמן סבוכנות זמן, זמן כפוף לזמן מחרן מחרן?



$(AB) \neq B \cdot A$ (אך $A \cdot B \cdot C = A \cdot (B \cdot C)$)

MATRIX-MULTIPLY(A,B)

```
if columns[A] != row[B] then return error "incompatible dimensions"
for i←1 to row[A]
    for j←1 to columns[B]
        C[i,j]←0
        for k←1 to columns[A]
            C[i,j] += A[i,k] * B[k,j]
```

$$A \cdot B = C \quad (A \cdot B) \cdot C = A \cdot (B \cdot C)$$

$$B \cdot j = \sum_{i=1}^n A_i \cdot B_j \quad B \cdot A = \sum_{j=1}^m A \cdot B_j$$

$$\sum_{k=1}^n C[i,k] \cdot B[k,j] = \sum_{k=1}^n A[i,k] \cdot B[k,j]$$

$$C[i,j] = \sum_{k=1}^n A[i,k] \cdot B[k,j]$$

לפיכם $C[i,j] = \sum_{k=1}^n A[i,k] \cdot B[k,j]$ ו- $A[i,k]$ הוא גורם ב- $\sum_{k=1}^n$ ו- $B[k,j]$ הוא גורם ב- $\sum_{i=1}^m$.

$$P(n) = \sum_{k=1}^{n-1} P(k) \cdot P(n-k)$$

לפיכם $P(n) = \sum_{k=1}^{n-1} P(k) \cdot P(n-k)$

$$P(1) \cdot P(4-1) + P(2) \cdot P(4-2) + P(3) \cdot P(4-3)$$

$$C_n = \frac{1}{n+1} \binom{2^n}{n} = \prod_{k=1}^n \frac{4^k}{n+k}$$

$$P(n) = C_{n-1} = \frac{1}{n} \binom{2^{n-1}}{n-1}$$

כלומר $P(n) = C_{n-1}$

לפיכם $P(n) = \sum_{k=1}^{n-1} P(k) \cdot P(n-k)$

$A_1 \cdot A_2 \cdots A_j$ מוגדר כ- $\sum_{k=1}^{j-1} P(k) \cdot P(j-k)$

$A_1 \cdot A_2 \cdots A_{j-1} \cdot A_j = A_{j-1} \cdot \sum_{k=1}^{j-1} P(k) \cdot P(j-k)$

לפיכם $m[i,j] = \sum_{k=1}^{j-1} m[i,k] \cdot m[k,j]$

$m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

$m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

$m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

$m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

$m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

$m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

$m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

$m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

$m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

$m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

$m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

$m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

$m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

MATRIX-CHAIN-ORDER(p)

```
1 n ← length[p] - 1
2 for i ← 1 to n
3     do m[i,i] ← 0
4 for l ← 2 to n
5     for i ← 1 to n - l + 1
6         do j ← i + l - 1
7         m[i,j] ← ∞
8         for k ← i + j - 1
9             do q ← m[i,k] + m[k+1,j] + p_{i-1} \cdot p_k \cdot p_j
10            if q < m[i,j]
11                then m[i,j] ← q
12                s[i,j] ← k
13 return m and s
```

לפיכם $m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

לפיכם $m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

לפיכם $m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

MATRIX-CHAIN-MULTIPLY(A, s, i, j)

```
1 if j > i
2 then X ← MATRIX-CHAIN-MULTIPLY(A, s, i, s[i,j])
3 Y ← MATRIX-CHAIN-MULTIPLY(A, s, s[i,j] + 1, j)
4 return MATRIX-MULTIPLY(X, Y)
5 else return A_i
```

לפיכם $m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

לפיכם $m[i,j] = \min \{ m[i,k] + m[k,j] : i \leq k < j \}$

$\Theta(n^3)$

MEMOIZED-MATRIX-CHAIN(p)

```
1 n ← length[p] - 1
2 for i ← 1 to n
3     do for j = i to n
4         do m[i,j] ← ∞
5 return LOOKUP-CHAIN(p, 1, n)
```

LOOKUP-CHAIN(p, i, j)

```
1 if m[i,j] < ∞
2 then return m[i,j]
3 if i = j
4 then m[i,j] ← 0
5 else for k ← i to j - 1
6     do q ← LOOKUP-CHAIN(p, i, k)
        + LOOKUP-CHAIN(p, k + 1, j) + p_{i-1} \cdot p_k \cdot p_j
7     if q < m[i,j]
8     then m[i,j] ← q
9 return m[i,j]
```

Longest Common Subsequence

X=A,B,C,B,D,A,B

Y=B,D,C,A,B,A

Z=B,C,A

$\Theta(2^n)$

לצורך חישוב LCS(X,Y) נשתמש במשתנים:
 $X = \langle x_1, x_2, \dots, x_n \rangle$
 $Y = \langle y_1, y_2, \dots, y_m \rangle$

לכל $i < n$ ו- $j < m$ הינה לנו $LCS(i-1, j) = Z_{i-1}$ ו- $LCS(i, j-1) = Z_j$.
 $LCS(i, j) = \max\{LCS(i-1, j), LCS(i, j-1)\}$.

$LCS(i, j) = \begin{cases} 0 & i=0 \vee j=0 \\ \max\{LCS(i-1, j), LCS(i, j-1)\} + 1 & i > 0 \wedge j > 0 \wedge x_i = y_j \\ 1 & i > 0 \wedge x_i \neq y_j \end{cases}$

לכדי לחשב LCS(X, Y) ישנו מערך $LCS[i][j]$ שמייצג $LCS(i, j)$.
 $LCS[0][0] = 0$, $LCS[0][j] = 0 \forall j > 0$, $LCS[i][0] = 0 \forall i > 0$.

במקרה שבו $x_i = y_j$, $LCS[i][j] = LCS[i-1][j-1] + 1$.
במקרה שבו $x_i \neq y_j$, $LCS[i][j] = \max\{LCS[i-1][j], LCS[i][j-1]\}$.

לכל $i < n$ ו- $j < m$ מתקיים $LCS[i][j] \leq \min\{i, j\}$.

LCS-LENGTH(X, Y)

```

1    $m \leftarrow length[X]$ 
2    $n \leftarrow length[Y]$ 
3   for  $i \leftarrow 1$  to  $m$ 
4     do  $c[i, 0] \leftarrow 0$ 
5   for  $j \leftarrow 0$  to  $n$ 
6     do  $c[0, j] \leftarrow 0$ 
7   for  $i \leftarrow 1$  to  $m$ 
8     do for  $j \leftarrow 1$  to  $n$ 
9       do if  $x_i = y_j$ 
          then  $c[i, j] \leftarrow c[i-1, j-1] + 1$ 
10      else if  $c[i-1, j] \geq c[i, j-1]$ 
11        then  $c[i, j] \leftarrow c[i-1, j]$ 
12        else  $c[i, j] \leftarrow c[i, j-1]$ 
13      b[i, j]  $\leftarrow$  " $\nwarrow$ " | " $\uparrow$ " | " $\leftarrow$ " | " $\nwarrow$ " | " $\uparrow$ " | " $\leftarrow$ " | " $\nwarrow, \uparrow$ " | " $\nwarrow, \leftarrow$ " | " $\uparrow, \leftarrow$ " | " $\nwarrow, \uparrow, \leftarrow$ " | " $\nwarrow, \uparrow, \leftarrow, \uparrow$ " | ...
14
15
16
17   return  $c$  and  $b$ 
  
```

PRINT-LCS(b, X, i, X)

```

1   if  $i = 0$  or  $j = 0$ 
2     then return
3   if  $b[i, j] = "$  "" 
4     then PRINT-LCS( $b, X, i-1, j-1$ )
5     print  $x_i$ 
6   else if  $b[i, j] = " $\uparrow"$ "$ 
7     then PRINT-LCS( $b, X, i-1, j$ )
8   else PRINT-LCS( $b, X, i, j-1$ )
  
```

$O(m+n)$ $\Theta(m+n)$ סימן זמן (n^2) סימן זמן וריאנט סימן זמן $\Theta(m \cdot n)$

