שימוש [illegible] המסלול הארוך ביותר בתת העץ האמצעי ואת הגבוה ביותר בתת העץ [illegible] (עם יש).
[illegible] חדש, p צריך להיות אביו ו-g אבא של p. הכנס v כבן נוסף ל-p שלושה בנים – סיימנו. אם ל-p ארבעה ילדים נפצל אותו [illegible] אחד עם שני בנים. נשתול את 'p' החדש (מצד ימין) כבן של [illegible]. מקרה קצה- הגענו לשורש פשוט נוסיף לו אבא וזה יהיה [illegible] חדש!
Amortized – O(1) ע"פ פונקן של מס' צמתים עם דרגה 4.
מרקה: (v הנושא, p אבא, u דוד של v) אם יש לו שני אחים פשוט נשמיט ונסרים. אם יש לו אח בודד אז: אם ל- u שלושה בנים – p מאמץ אחד מבני u, אם ל- u שני בנים – u יאמץ את האח של v ונכשיו p ערירי ונשמיטו רקורסיבית.
ביות ממערך ממויין: (O(n)) ליצור מכל זוג סמוך צומת, ואם נשאר אחרון לצרף לשלישה. ושוב נבצע איחודים עד שנותר איבר בודד. n>...+n/4+n/2

Find -ו Union:

Min Span Tree: 1)נמיין את הקשתות לפי משקל עולה 2)הנסף לגרף קשת (v,w) אם הם בשני רכיבי קשירות שונים. סיבוכיות: O(|E|log|V|)
UnionByRank: בכל צומת נשמור rank. באיחוד הrank הקטן נתלה על הגדול. ואם שווים, תולים ומגדילים את הrank ב1. find – O(logn)
בכל תליה מספר האברים בקבוצה לפחות מוכפל. הסיבוכיות ל-n פעולות find היא nlogn.
PathCompression: אחרי סריקת מסלול כל הצמתים בדרך נתלים על השורש.

Ackerman:

$$A_k(j) = A_{k-1}^{j+1}(j)$$

		1	2	3	4
j+1	A0	2	3	4	5
2j+1	A1	3	5	7	9
	A2	7	23	63	
	A3	2047			
	A4	>10^80			

פול' הפונק: $\alpha(n) = \min\{l / A_l(1) \ge n\}$ $\alpha(n) \le 4$

מזפט: סדרה של m פעולות עם make-set ו max לוקחת $O(m\alpha(n))$ ≥
אם x שורש או דרגתו 0: $\Phi(x)=\alpha(n)\text{rank}(x)$

אחרת: $level(x) = \max\{k / rank(p(x) \ge A_k(rank(x))\}$,

$iter(x) = \max\{i / rank(p(x)) \ge A^i_{level(x)}(rank(x))\}$

$(x)=(\alpha(n)- 0 \le level < \alpha n, \quad 1 \le iter \le rank(x)$

$\Phi(x) = (\alpha(n) - level(x))rank(x) - iter(x)$

הבחנות: 1) דרגה מתחילה מ0 ורקגדלה עד למקסימום n-1
2)מוצת x שלא שורש, דרגה של p(x) גדולה ממש מדרגה של x.
3)דרגות מונטוניות עולות לאורך מסלול 4)דרגת p(x) ממשיכה לגדול לאורך התהליך.
- level אינו יורד במשך התהליך iter אינו יורד כל עוד level קבוע.
- rank ופונק'ם לא משתנים ולכן:
אם level iter לא משתנים Φ לא משתנה. אם Iter גדל ממש או Φ קטן ממש.אם level השתנה, וזה שווה ליחזות rank, אז Φ קטן בaka לפחות.
ואילו iter יכל לגדול בaka rank-1. בשני המקרים Φ קטן בלפחות 1.
עבור find:
אחרי הfind הrank של אביו של x הוא לפחות כמו rank(p(x)). אחרי הfind אז שה(x)iter גדל הlevel לא השתנה או שהlevel גדל. בכל מקרה Φ קטן. (מלבד השורר, צומת הראשון על המסלול ו$\alpha(n)$ צמתים ראשונים, אחר בכל level, כל צומת מקטין Φ בלפחות 1).

$find(z) \le s - (s - \alpha(n) + 2) = O(\alpha(n))$

קרן האינטגרל: חשב הסתברויות, כל פעם שדך את הצעדים הקלים ביותר כאשר משקל הצעד החדש הוא סכום המשורכים. זמן בניה nlogn . נייצג ע"י heap.

נסיונות האלגוריתים של הופמן

יהיה C אלפבית שבו כל תו $c \in C$ הוא בעל שכיחות f[c]. יהיו X,Y 2 התווים בC בעלי השכיחויות הנמוכות ביותר. אזי קיים עבור C קוד תודליות אופטימלי אשר בו מילות הקוד עבור X,Y הם בעלות אותו אורך ונבדלות זו מזו בסיביות האחרונה בלבד.
הוכחה: יהיה B,C 2 תווים היוצאים בT ע"י 2 עלים אחים בעלי עומק מקסימלי נניח כי $f(b) \le f(c)$ וכן $f(x) \le f(y)$ מאחר ש f(x) ו f(y) הם השכיחויות הנמוכות ביותר ו f(b) ו f(c) הם 2 שכיחויות כלשהן מקבלים ש $f(x) \le f(b)$ וגם $f(y) \le f(c)$ נחליף מקומותהם של X ו B בT ונקבל את העץ T' בהמשך נחליף את C ו Y בT' ונקבל את העץ T'' הפרש העליות בין t לT'', הוא:

$$B(T) - B(T'') = \sum_{c \in C} f[c]d_T(c) - \sum_{c \in C} f[c]d_{T''}(c)$$
$$= f[x]d_T(x) + f[b]d_T(b) - f[x]d_T(b) - f[b]d_T(x)$$
$$= f[x]d_T(x) + f[b]d_T(b) - f[x]d_T(b) - f[b]d_T(x)$$
$$= (f[b] - f[x])(d_T(b) - d_T(x)) \ge 0$$

מכך רואים כי החלפת המקומות אינה מגדילה את העלות ולכן נעשה אותו דבר לT'' ונקבל ש: $B(T) \ge B(T') \ge B(T'')$

Trie: עץ שמייצג מחרוזות. אין שתי קשתות זהות מאותו צומת. בסוף יש לדחוס אותו.
SuffixTree: בונים Com.Trie לכל סיומת $+$. שמים סיומת הכה ארוכה. מטיילים לאורך הקשת, אם זו לא מתאים מוציאים צומת חדש. זמן בניה O(n). מקום O(n) ע"י יצוג המקום של הסיומת.
מס' צמתים=יש לפחות n-1 צמתים פנימיים (כי בעץ ביצאר מלא יש כך) ולכן סה"כ יש לפחות 2n-1 צמתים. ולכן יש בערך ככה קשתות.
Pattern-חיפוש: O(m). מס' העלים=מס' הופעות.
GST: עץ שמכיל suffixים של כולם. כל עלה שומרים מאיזה מחרוזת ומקום התחלה.
LongestCommonSubStr-: מחפש צומת פנימי לו עומק מחרוזת max ולו יש בנים משני הסוגים. O(n) "עומק מחרוזת" = סכום מחרוזות על קשתות.
-פלינדרום: בניית GST לS$#R(S) לכל i נשאל מה הLCA של הסיומת שלו: ושל הסיומת n-i+1 R(S) ב(O(1.אורך פלינדרום=אורך מחרוזת
LCA: O(n)
LCA-:מספור מצתים וסריקה בDLR וכתיבת המס' בכניסה ויציאה->מערך בגודל 2|E|+1. LCA יהי הבין i לj בעל העומק המינ.

1)שמירת כל ההותאות מראש. Query-O(1) מקום הכבה:O(n^2)
2) לכל תא שומרים min של 2^j ימינה: שמאלה. נמצא k כך ש $\frac{j-i}{2} \le 2^k \le j-i$. Query-O(1) מקום הכבה:O(nlogn).
3) נבנה ערמה min ב-O(n). נטפס בשני האינדקסים מעלה, כאשר אם האבא שונה ממני הוצה הי בן ימני (עבור i) אז "נחליף" את האבא בערך שלי ונטפס איתו. Query-O(logn) מקום הכבה O(n) .
4) בונים מערך חדש בגודל 2n/logn ובכל תא שומרים את הmin של הדלי המתאים. כל דלי בגודל logn/2. על זה נשתמש במבנ"ת ה(2). בתוך דלי נסתכל על חבנית i-1 – יש $2^{(logn/2)}=n^{0.5}$ אפשרויות. ולכן נקבל סה"כ O(n) שאילתות עדינות שמשמור. Query-O(1) מקום הכבה: O(n).

אנטרופיה: $H_o(s) \le \log n \quad H_o(s) = -\sum_i^k p_i \log(p_i) < \log k \quad p_i = \frac{n_i}{n}$

משפט: $H_o(s)n \le |H_{huff}(s)| \le (H_o(s)+1)n$

משפט: אין אלג' ביוון מכוון כל מחרוזת לפחות מ $nH_o(s)$
טענה: אין אלג' שלכל מחרוזת ביוארית באורך n מסורג ל-n ביטים. (פול חח"ע מקבוצה 2^n ל$2^{(n-1)}$).

מיונים:

שיטות פשוטות (שטובות למספר קטן מאוד של איברים אפילו יותר מהשיטות המסובבות). לכולן הסם עליון ותחתון של n^2:
1. Bubble Sort – הצפת הקל כלפי מעלה.
2. Insertion Sort – אטרציה דוגאה שכל האיברים 1..i ממיינים (נכנים את i למקום הנכון).
3. Selection Sort – בשלב ב- i בוחרים את הקטן ביותר מבין A[i]... A[n] ושמים במקום ה- i.

Cmp	Stb	ממוצע	w.c.	מיון
√	√	-	$\Theta(n^2)$	Bubble
√	X	$\Theta(n^2)$	$\Theta(n^2)$	Selection
√	√	-	$\Theta(n^2)$	Insertion
√	√	-	$\Theta(n \cdot lgn)$	BinTree
√	√	-	$\Theta(n \cdot lgn)$	Merge
√	X	-	$\Theta(n \cdot lgn)$	Heap
√	X	$\Theta(n \cdot lgn)$	$\Theta(n^2)$*	Quick
X	√	$\Theta(n)$	$\Theta(n^2)$	Bucket
X	√	-	$\Theta(n+k)$**	Counting
X	√	-	$\Theta(n \cdot d)$***	Radix

* QuickSort מבוסס חציון לוקח w.c. $\Theta(n \cdot lgn)$
** k - הערך המקסימלי
*** d - מס' הספרות

Quick Sort: בחר פיבוט, מצא וארגן נקודה: נקודה k כך שהערכים הקטנים מופיעים משמאלה והערכים הגדולים מופיעים מימינה. מין את ימין ואת שמאלה רקורסיבית. קלט הכי גרוע – מערך ממוין.
במקרה הגרוע ביותר נבחר כל פעם לפיבוט את הגדול ביותר ואז כל פעם המערך יתחלק ל- 1 ו- n-1. נקבל n^2. נשים לב שהפרטיש הוא ליניארי. במקרה הממוצע נקבל nlogn. אפשר גם לבצע ב- WC של nlogn (ראה OS).
Random – מספור כמה השוואות נעשות בנותחלת. הוסיפה ע"י משתנה מציין אם i ו j הושוו. השוואה קורת בסיכוי 2/j-i+1, ויש n(n-1) אפשרויות. ולכן:

$$E[X] = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\frac{2}{j-i+1} = \sum_{i=1}^{(n-1)}\sum_{k=2}^{(n+i+1)}\frac{2}{k} < \sum_{i=1}^{(n-1)}\sum_{k=1}^{(n)}\frac{2}{k} = \sum_{i=1}^{(n-1)} O(\log n) = O(n\log n)$$

קלט כמעט ממוין: I – מס' הinversions.את החלק הממוין מחזיר בעץRB. נעלה מעלה ימיני עד שנמצא תא עץ בו אמור להיות האיבר, נעלה logd. סה"כ $O(\sum \log I_j + n) = O(n\log\frac{I}{n} + n)$. Am ויפיפה הכבנה O(logd).

HeapSort: להכניס לערימה, לעשות ExtractMax n פעמים. ב- WC nlogn, בממוצע גם אבל קצת יותר גרוע מ- QS. הערה: אם רוצים את k הראשונים: O(n+klogn).

MergeSort: nlogn.

```
MS(A,p,r)
If p<r then {q<-floor((p+r)/2);
    MS(A,p,q); MS(A,q+1,r);
    Merge(A,p,q,r); }
```

מיונים שטובים כשיודעים משהו על הקלט (לא במודל ההשוואות):
מין מניה (Count): $O(n+k)$ למין מספרים בתחום 1 עד k. עבור i מ-1 עד k נשמור ב- C[i] את מספר האברים שקטנים שווים ל-i. נסרוק שוב את A ועבור כל איבר נכנים למערך B בגודל B[C[A[i]]] – n ונפחית 1 מ- A[i] ונפחית 1 מ- C[A[j]].
מין בסיס (radix): $O(dn)$ ממין n מספרים בני d ספרות כל אחד. מתחילים מהספרה הפחות משמעותית ובכל שלב עושים מין יציב (למשל CS).
Bin/Bucket Sort: $O(n)$ מחלקים את הקלט למערך בן n האיברים ל- n קבוצות. נעזרים במערך B בגודל n של n רשימות מקושרות וסורקים את A כדי להכניס כל איבר לרשימה המתאימה. ממיינים את הרשימות בעזרת מין הכנסה.

חסמי מיון:
בונים עץ ההחלטות בו כל צומת מייצג את מרחב האפשרויות. כיוון שהעץ הוא ביוארי קיים מסלול שאורכו לפחות logk, כאשר k הוא מספר העלים. העלים מייצגים סדרים בסיסיים ואת כולם! מספר הסדרים הוא k=n!. $k \le 2^h$ נוציא לוג ונקבל שאורך המסלול המקסימלי גדול מ- $n!>(n/2)^{n/2}$. log(n!) לכן log(n!)=Ω(nlogn). אפשר להראות שהחסם התחתון לזמן ממוצע הוא אותו הדבר.
חסם תחתון-בדיקה: האם יש שני איברים זהים Ω(nlogn).
- בכמה זמן ניתן לבנות עץ חיפוש ביוארי בעזרת השוואות בלבד? חסם עליון: שרשרת ארוכה - nlogn. חסם תחתון: נניח כי ניתן לעשות זאת ב- o(nlogn) (o קטן) אז נבנה עץ, נבצע inorder וקבלנו מיון ב- o(nlogn) – סתירה.
- האם ניתן להפוך ערימה לעח"ב ב- O(n) במודל ההשוואות? נניח שכן. נקח n מספרים, נכניס לערימה, זה לוקח O(n). נהפוך לעח"ב באותה היצילות, נעשה אין-אורדר שוב באותה היצילות וקבלנו מיון ב- O(n).
- קלט: k מערכים ממיונים באורך n. האם אפשר להחזיר מערך גדול ממרין ב- O(n√logn)? לא. נניח שכן, נקח n מספרים מהפוך כל אחד למערך, נפעיל את האלג וקבלנו מיון!
- קלט: n מספרים כך שכל מספר בדיוק פעמיים.
מטרה: חסם התחתון למיון במודל ההשוואות. מספר העלים בעץ ההחלטות הוא $2^{n/2}n!$. לכן קיים מסלול הגדול מ- log של הביטוי האחרון זה nlogn-n/2.
כלומר חסם תחתון nlogn.
- קלט: n מספרים מתוכם n/logn שונים. פלט: מיון. אלג': לפי תרגיל מהאש O(n) בממוצע נקבל את האברים וכמה פעמים הם מופיעים. נמין את n/logn (O(n).
- קלט: n קטעים על קטע סגור. פלט: האם יש זוג שנחתך. אלג': נמין את כל הזוגות (a_i,b_i) לפי $a_1..a_n$ (nlogn), לכל i נבדוק האם (a_i,b_i) נחתך עם (a_{i+1},b_{i+1}). (עוד n).
- קלט: כמו קודם. פלט: למצוא נקודה שמוכלת במספר מקסימלי של קטעים. נמיין את a1..an יחד עם b1 ... bn ואז לעבוד כמו עם אלג' של פתיחת וסגירת סוגריים!
- הצג מ"נ שתומך ב- find,ins,del ב- WC של θ(nlogn) וכן בפעולה k – nearest(x,k) האברים שבצרך מחלט הבי קרובים ל-x. פתרון: עץ 2-3. בעזרת עוקב (logn) נמצא את k האברים הקרובים ביותר. אפשר לייעל אם העלים מוחזקים ברשימה מקושרת (O(k)).
- קלט: n מספרים מהתחום: $\{1...n^2\}$. פלט: מיון. אלג': נשבור כל אחד לשני מספרים באורך n, ונבצע על כל החלקים הראשונים ואח"כ על השניים -count sort (O(n+n)=O(n)).
- קלט: m מערכים שסך גודלם n, כל המספרים בתחום 1..k. נרצה למיין כל אחד בנפרד. נאחד את המערכים, נמיין ב- CS ב- O(n+k) ואז נפצל בחזרה (נשמור זוגות – ערך ושיוך).

נוסחאות רקורסיביות:

$$T(n) = 2T(\frac{n}{2}) + \theta(n) = \theta(n\log n),\ T(n) = 2T(\frac{n}{2}) + n^2 = \theta(n^2)$$

$$T(n) = 2T(\frac{n}{2}) + \theta(1) = \theta(n),\ T(n) = 9T(\frac{n}{7}) + n = \theta(n^{\log_7 9})$$

$$T(n) = \sqrt{n}T(\sqrt{n}) + n = \theta(n\log(\log n)),\ T(n) = 2T(\frac{n}{2}) + n\log^2 n = \theta(n\log^3$$

$$T(n) = 2T(\frac{n}{2}) + \sqrt{n} = \theta(n),\ T(n) = 4T(\frac{n}{2}) + n/n^2/n^3 = \theta(n^2/n^2\log n/n^3)$$

$$T(n) = T(n-1) + \frac{1}{n} = \theta(\log n),\ T(n) = T(2cn) + T((1-c)n) + n = \theta(n^d)$$

$$T(n) = T(\frac{n}{3} - 4) + 1 = \theta(\log n),\ T(n) = T(n-a) + T(a) = \theta(n^2)$$

$$T(n) = T(c \cdot n) + T((1-c) \cdot n) + 1 = \theta(n),\ 0 < c < 1$$

$$T(n) = T(c \cdot n) + T((1-c) \cdot n) + n = \theta(n\log n),\ 0 < c < 1$$

$$T(n) = T(c \cdot n) + T(d \cdot n) + n = \theta(n),\ c + d < 1$$

החלפת משתנים (בסוף לעשות הצבה הוזרת

$$T(n) = 2T(\sqrt{n}) + \lg n$$
$$T(2^m) = 2T(2^{m/1}) + m\ ,\ m = \lg n$$
$$S(m) = 2S(m/2) + m,\ \ S(m) = T(2^m)$$

סדרי גודל

$$\log n \Rightarrow \log^2 n \Rightarrow [\log(\log n)]^{\log(\log n)} \Rightarrow 2^{\log_2 n} \Rightarrow$$

$$(\log n)^{\frac{\log n}{\log(\log n)}}, n \Rightarrow n\log n, \log(n!) \Rightarrow$$

$$n^{\frac{2n+1}{n+1}}, n^2, \frac{n^4+1}{n^2+1} \Rightarrow 2^{\log^3 n} \Rightarrow n^{2\log n} \Rightarrow 2^n$$

$$\Rightarrow 3^n \Rightarrow n! \Rightarrow (n+1)! \Rightarrow n^n$$

שוויוניות שונים

$$1 = O(n^{1/\log n}) = o(\log(\log n)) = o(\sqrt{\log n}) = o(\log n) = o(\log^2 n)$$
$$= o(n^{1/3}) = o(n) = o(2^{\log n}) = o(n^2) = o(\log n)! =$$
$$o(\log n)^{\log n} = o(n^{\log(\log n)}) = o(3/2)^n = o(2^n) = o(n*2^n)$$
$$= O(n!)$$

$$\log(n!) = \Theta(n*\log n) \quad n\log n = o(n^{1.5}) \quad n = O(1.5^n)$$

$$\log^d(n) = o(n^\varepsilon) \quad n! > (n/2)^{n/2}$$

$$\sum_{i=1}^{\log n} 1/i = \Theta(\log(\log n)) \qquad \sum_{i=0}^{\infty} x^i = 1/(1-x)$$

$$\sum_{i=1}^{n} \sqrt{i} = \Theta(n \cdot \sqrt{n}) \qquad \sum_{i=1}^{n} i^k = \Theta(n^{k+1})$$

$$\sum_{i=1}^{n} 1/i = \Theta(\log n) \qquad \sum_{i=1}^{n} \log i = \Theta(n\log n)$$

$$\sum_{i=1}^{\infty} kx^k = \frac{x}{(1-x)^2} \qquad \lim_{n\to\infty}(1+\frac{x}{n})^n = e^x$$

$$\log\frac{n}{2} > \frac{1}{2}\lg n \qquad a^{\log_b n} = n^{\log_b a} \qquad P(Y \ge t) \le \frac{E(Y)}{t}$$

$$n! = \Theta(\sqrt{n}(\frac{n}{e})^n) \to \log(n!) = n\log n \pm O(n)$$

מבנה	search	insert	delete	
מערך	Θ(n)	Θ(1)	Θ(n)	
מערך ממוין	Θ(lgn)	Θ(1)	Θ(lgn)	בשימוש אריה במדנ-
רשימה דו/חד כיוונית	Θ(n)	Θ(1)	Θ(n)	
עץ חיפוש	Θ(h)	Θ(h)	Θ(h)	h גובה העץ
ערימה	Θ(n)	Θ(lgn)	Θ(lgn)	
Hash close	Θ(n)	Θ(n)	Θ(n)	
Open Hash	Θ(1)avg	Θ(1)avg	Θ(1)avg	h(x)rand
	Θ(n)	Θ(1)	Θ(n)	chaining
	Θ(1)avg	Θ(1)	Θ(1)avg	m=Θ(n), h(x) אקראית
Per.H.	Θ(1) wc	Θ(1)avg	Θ(1)wc	

- Hash מגודל 3n מכניסים 2n איברים, כאשר כפליות נשמרות בשרשרת אחת גלובלית. מה צורך הרשיעה? פתרון: צורך רשיעה זה 2n פחות מס' איברים בטבלה. Y=3n*P(cell full) . Y=P(cell full)-1 (cell empty)

$$P(cell\ empty) = (1 - \frac{1}{3m})^{2n}.$$

זמן ריצה:

$f(n)=\Theta(g(n)) \Rightarrow$

$\exists c_1,c_2>0 \forall n>n_0, c_1 g(n)\le f(n)\le c_2 g(n)$

$f(n)=O(g(n)) \Rightarrow$

$\exists c>0 \forall n>n_0, f(n)\le cg(n)$

$f(n)=\Omega(g(n)) \Rightarrow$

$\exists c>0 \forall n>n_0, cg(n)\le f(n)$

w,o

כמו O ו-Ω, אבל מתקיימים עבור כל C>0.אם הראינו o, אז בהכרח גם O מתקיים. כדאי לפתור עם lim (לא לשכוח את לופיטל):

$$f(x)=o(g(x)) \leftrightarrow \lim_{x\to\infty}\frac{f(x)}{g(x)}=0$$

$$\varnothing=o(s(n))\cap\omega(s(n))\quad \varnothing=O(s(n))\cap\omega(s(n))$$

$$\varnothing=o(s(n))\cap\Omega(s(n))$$

זמן Amortized:

- לוקחים מבנה נתונים ריק (חשוב לגבי Del למשל).
- מסתכלים על סדרה של N פעולות שיכולות להיות תערובת של כל אחת מהפעולות.
- נבחר את זאת שלוקחת הכי הרבה זמן
- נחלק את הזמן הזה ב-N לקבלת זמן Amortized

שיטת הבנק: מחיר אמיתי ועוד מס' הפקדות פחות מס' משיכות

שיטת הפוטנציאל: $\Phi(D)=ac(op)+\Delta\Phi$

*אם רוצים להגדיל מערך s->f(s), אז העלות של m פעולות היא:

$$O(m+\frac{m^2}{f(m)-m})$$

פתרון מישוואות רקורסיה: איטרציה – פיתוח רקורסיבי של המשוואה; אינדוקציה; עץ רקורסיה; שיטת האב:

$T(n)=aT(n/b)+f(n)$ $a\ge1,b>1$ $\varepsilon>0$ $t=\log_b a$

1. $f(n)=O(n^{t-\varepsilon}) \Rightarrow T(n)=\theta(n^t)$
2. $f(n)=\theta(n^t) \Rightarrow T(n)=\theta(n^t\log n)$
3. $f(n)=O(n^{t+\varepsilon})$ + קיים $0<c<1$ $a*f(n/b)\le cf(n)$
   $\Rightarrow T(n)=\theta(n^t)$

עץ חיפוש בינארי:

מסלול ממוצע בעת"ב – logn, n -WC.

עץ מלא – לכל צומת 0 או 2 ילדים.**עץ שלם** – צורת משולש.

- בהינתן סריקת DLR ניתן לשחזר את העץ: האיבר הראשון הוא השורש, מחפשים את הל היחידה בה משמאל נמצא מישהו יותר קטן משורש ומימין יותר גדול (וממשיכים רקורסיבית). לא מכל מערך שמייצג DLR אפשר לבנות אבל עץ חיפוש בינארי.
- successor – אם יש בן ימני מחזירים את המינימום מחלק הימני, אחרת עולים למעלה עד שהבנו שx הוא בתת עץ שמאלי שלו - *O(h). ממוצע: O(1) הוכחה – נעבור לי מינום, נעשה n-1 פעמים עוקב, עד שנגיע למקסימום, זה לוקח n*h אבל במצטע n כי עוברים על כל צומת פעמיים -> 2n-2.*
- מחיקה – אם עלה מוחקים, אם בן יחיד, תולים על האבא, אם יש שני בנים מחליפים ערך עם העוקב ומחוקים את העוקב. *O(h)*
- הכנסה – סדרה של m פעולות בהנתן המיקום תעלה O(m), גם סדרה של m הכנסות ומחיקות בהנתן המקום תעלה O(m)
- OrderStatistic(k): נוסיף שדה size מספר הבנים כולל הבומת. אם הסיר של הבן השמאלי שווה ל- k-1 מחזיר את הצומת, אחרת, אם גדול מ-k מחזיר את OS(k,p.l) אחרת OS(k-p.l.size-1,p.r).
- Rank(k): נעלה כל הפעם לאבא. אם אחנו הבן הימני של האבא נעלה מרוח ב- size+1 של הבן השמאלי של האבא (המרוח מאותחל ל- x.l.size+1).
- בשני המימושים האחרונים – להראות שהוספת השדה לא הורסת את ins,del,find.
- Concat: אם כל המפתחות של 1D קטנים מ2D,נחפש צומת במסלול השמאלי שדרגתו כדרגת 1D, נגדיר y אדום אבא של שניהם ואז נוקן למעלה.
- Split: עולים מא לשורש, בכל תת עץ שמאלי משרשרים לT1 וכל תת עץ ימני לT2. זמן O(logn) כי כל פיצול שרשור לוקח |r1-r2|-1 וסכום ההפרשים זה גודל העץ.
- Range: דיווח – O(logn+k). ספירה – שמירה בכל צומת גודל של תת העץ. O(logn)
- D2: עץ ממיין ע"פ x ,לכל לומת פנימי שמירת עץ ממוין ע"פ y. כל עלה מופיע בlogn עצים שונים ולכם סה"כ מקום logn. זמן חיפוש

$$O(\log^d n+k),\ d=2$$

- בניית עץ אדום-שחור ממערך: נסדר כמו ערימה, נעבור ב- inorder רק במקום להדפיס נקח את הערך הבא במערך. נצבע הכל בשחור חוץ משורשה האחרונה אם היא לא פלאה לחלוטין ואז נצבע אותה באדום.

כפל קרטזיריים: למשל – מצא את האדם מעל גובה 1:75 בעל הציון המינימלי. נשתמש בעץ 2-3, כאשר העלים ממיינים לפי הגובה. בכל צומת נשמור את הציון המינימלי של בניו. אלג': נסמן באדום בעץ את המסלולים המובילים לעלים המעניינים אותנו (אלה מעל הגובה 1:75). הצלה: ערכי מינימום של בנים אשר הם השורש של עצים אדומים לגמרי. כאשר שני ערכים עולים הצלה את המינימלי. O(logn)- כי יש לכל היותר תת עץ שבנו האחד שחור ובנו האחר אדום.

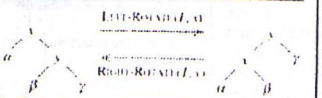
- *אם לכל צומת בעץ הוא עלה או שיש לו בדיוק שני בנים או מספר הצמתים הפנימיים הוא n-1 (n – מס' העלים).*

$$n=2^k-1$$

Dijkstra: מציאת מסלול מינ'. 1)בחרים v Q כך ש-d(v) מינ' 2)מעבירים לS 3)לכל קשת (v,w) כך שw Q עושים relax: אם d(v)+w(v,w)<d(v) אז d(w)=d(v)+w(v,w) ,p(w)=v.

סיבוכיות: O(nlogn+m), כאשר פעולת "decrease key" תקח amc רק O(1).

עצים אצומים-שחורים: עץ בינארי המקיים: כל צומת הוא אדום או שחור; אם צומת הוא אדום, שני בניו שחורים, כל המסלולים הפשוטים מצומת לצאצאים עלים מכילים אותו מספר של צמתים שחורים. במסקנה מתקבל שהוו עץ בערך מאזן! צריך לדאוג שחוקיות העץ תמיד נשמרת בפעולות ins, del חאת מבצעים ע"י רוטציה שמאלית/ ימנית (O(1)):



הכנסה: O(logn). תמיד נכניס אדום כאחד העלים לכן רק תכונת אי סמיכות האדומים יכולה להיות מופרת. על כן נתקן רקורסיבית כלפי מעלה לפי מקרים. מקרה 1: הדוד של הצומת שהכנסנו גם הוא אדום: צובעים את האבא והדוד בשחור, את הסבא בשחור וממשיכים. מקרה 2: הצומת שהכנסנו הוא בן ימני – נבצע רוטציה שמאלית לאביו. מקרה 3: הצומת שהכנסנו הוא בן שמאלי – נבצע רוטציה ימנית לסבו.

מחיקה: יש למחוק כמו בעח"ב. אם מחקנו (בפועל) צומת אדום אז אין בעיה אבל אם מחקנו צומת שחור ייתכן כי הפרנו את האיזון ואז נבצע תיקונים (del_fix): נניח ש-X הוא המקום שמעליו בוצעה השמטה, W אחיו (אחרי המחיקה). מקרה 1: W אדום – החלף צבעי W, אבי X, ובצע רוטציה שמאלית לאביו של X. (מוביל לכל אחד מהמקרים). מקרה 2: W שחור + 2 בנים שחורים – הורד שחור מ- W ו- X והוסף לאבי X. מקרה 3: Wשחור, בן שמאלי אדום ובן ימני שחור – החלף צבעים בין W לבנו השמאלי ובצע רוטציה ימנית (מוליך למצב 4). מקרה 4: W שחור + בן ימני אדום. שנה צבעים ובצע רוט' שמאלית לאבי X (סיום הלולאה).

ערימה: = עץ בינארי מלא, אבא גדול מבניו.

- Heapify – הטמה: הבנים הם ערימות. יורדים למטה ומחליפים כל פעם אם צריך. *O(h)*
- build-heap -- עושים היפיפי מטה שאמצע המערך (משורה שניה ומעלה). *O(n)*
- extract-max – שמים את האיבר האחרון בשורש ועושים לו היפיפי.
- מחיקה – כמו קודם, לוקחים את האחרון ומחליפים. לבדוק אם צריך לעלות או לרדת.
- אפשר לשמור ערימת מקסימום ביחד עם ערימת מינימום (פויינטרים בין איברים זהים).
- קלט: ערימת מקסימום ומספר. פלט: כל אברים הערימה הגדולים מהמספר. אלג': כל עוד אנחנו בערך שגדול מהמספר הדפס והמשך לבנים. *O(k) מספר האברים הגדולים מהמספר.*
- קלט: ערימת מקסימום ו-k. פלט: x הגדולים ביותר. אלג': לולאה I, 1 עד k, נמצא מצביע עם ערך מקסימלי מבין המצביעים , הדפס האיבר, מזק מצביע, הוסף מצביעים לבניו. נשמור המצביעים בערימת עזר *O(klogk)*
- קלט: k מערכים ממוינים שסך גדלם n. פלט: מערך ממוין יחיד. אלג': ערימת מינימום עם מצביעים לאבר הראשון במערך. *n אסרעיות, כל אחת O(logk), סה"כ O(nlogk).*
- קלט: n אברים המכילים logn ערכים שונים. פלט: מיון. אלג': מיון logn הערכים ($\log^2 n$), הכפלת המופעים (n). בשביל הזיהוי עץ חיפוש בינארי מאוזן (למשל 2-3) עם קשורות, מספר אבריו logn, גובהו loglogn לזיהוי nloglon.

טבלאות hash:

מ"נ שתומך ב- find, ins, del ממוצע O(1) (אפשר לקבל תוצאות זהות במערך ביטים אך חבל על זכרון).

- m – גודל טבלת ה-HASH, אופטימלית m=Θ(n)
- α – מקדם העומס, α=m/n

1. **טבלה פתוחה:** כל תא במערך מצביע למבנה נתונים אחר, למשל רשימה מקושרת.

דוגמאות לפונקציית Hashing עבור טבלה פתוחה:

$$h(x)=X \bmod m$$

$$h(x)=\left\lfloor m\cdot\left(x\cdot A-\lfloor x\cdot A\rfloor\right)\right\rfloor \subset 0<A<1$$

לכל פונק' כזו יש קלט עליו הביצועים O(n)

דוגמא לפונקציית Hashing פתוחה:

$$A(x)=a_0+x(a_1+x(a_2+x(a_3+x\cdot a_4)))$$

$$P_4=a_4 \bmod M,\ P_3=(a_3+x\cdot P_4) \bmod M,\ P_1=(a_2+x\cdot P_3) \bmod M,$$

$$P_1=(a_1+x\cdot P_2) \bmod M,\ P=(a_0+x\cdot P_1) \bmod M$$

$$\left.\begin{matrix} x>a_4 \\ M>x \end{matrix}\right\} a_4 \bmod M=a_4$$

2. **טבלה סגורה (שימוש ב- Hashing חוזר):** במקום פונקציית Hashing בודדה, יש סדרה של פונקציות כאלו. מריצים על הערך פונקציה ראשונה, ואם המקום תפוס, מריצים פונקציה שנית, וכן הלאה עד אשר מגיעים למקום ריק. בעיה של מחיקה וגם נטיה ליצור cluster.

דוגמאות לפונקציית Hashing עבור טבלה סגורה:

- Linear Probing:

$$h_i(x)=h(x,i)=\left(h'(x)+i\right) \bmod m$$

- Quadric Probing:

$$h_i(x)=h(x,i)=\left(h'(x)+c_1 i+c_2 i^2\right) \bmod m$$

- Double Hashing:

$$h_i(x)=h(x,i)=\left(h'(x)+i\cdot h''(x)\right) \bmod m$$

במקרה האחרון יש m^2 אפשריות ולא m כמקודם.

הערה: פונקציה 'h ו- ``h וק פונקציות Hashing כלשהן.

ניתוח: מפתח מתמפה לפרמוטציה אקראית. x-מס' דגימות שעושים. אם מפתח לא נמצא:

$$E(x)=1P(x=1)...mP(x=m)\le1(P(x\ge1)-P(x\ge2))...$$

$$\le P(x\ge1)+P(x\ge2)...=1+\frac{n}{m}+\frac{n}{m}\frac{n-1}{m}+...$$

אם

$$\le1+a+a^2+a^3...=\frac{1}{1-a}$$

$$1/1-\frac{i-1}{m}$$ =זמן חיפוש=זמן שלקח לכניס מפתח נמצא:

$$\frac{1}{n}\sum\frac{1}{1-\frac{i-1}{m}}=\frac{1}{n}\sum\frac{1}{1-\frac{j}{m}}=\frac{1}{n}\sum\frac{m}{m-j}=$$

$$=\frac{m}{n}(\frac{1}{m}+\frac{m}{m-1}+...+\frac{1}{m-n+1})=\frac{1}{a}\ln(\frac{1}{1-a})$$

II פונקציה אוניברסלית אם לכל זוג מפתחות X, Y מספר הפונקציות h כך ש h(x) = h(y) הינו |H|/m. כלומר סיכוי להתנגשות 1/m. אורך רשימה ממוצע הוא <= α+1 .

דוג': נקבע קראשוני גדול מ|U|. $|H|=p*(p-1)$.

$$f_{a,b}(k)=\underbrace{((ak+b)(\bmod p))}_{g}(\bmod m)\quad b\in\{0..p-1\}\quad a\in\{1..p-1\}$$ G היא

וזה"ע. כלומר g(k)<>g(l) <= k<>l

ב. העתקה |kl, היא על וזה"ע. כלומר בבחירת (a,b) ומיפוי k,l מתקבלים (r,s) כך

ש<>r. יש $\left\lceil\frac{p}{m}\right\rceil-1$ מס' s שמשאירים אותה שארית r mod m כמו r.

$$\left\lceil\frac{p}{m}\right\rceil-1\le\frac{p+m-1}{m}-1\le\frac{p-1}{m}$$ ולכן סה"כ יש לנו לא יותר מאשר

$$\frac{p(p-1)}{m}$$ זוגות. <= פונק אוניברסלית.

עבור האש סגור ויקח למלא את הטבלה logB בממוצע ליאבר למלא 90% - 10/9ln10 (כלומר לא תלוי ב- B).

אם הטבלה מלאה: עבור האש פתוח אם N>=2B אז כדאי להכפיל הגודל ולהתעדכן. עבור האש סגור אם N>0.9B .

רי-האש עבור התנגשויות: $h_i(x)=h(x)+d_i(\bmod B)$

- עבור $m=n^2$. מחשבים מס' התנגשויות $P(Y\ge1)\le1/2 \Rightarrow \binom{n}{2}\frac{1}{n^2}\le1/2$

לכן, חיפוש יקח W.C O(1).

Expected Average Case	Expected Worst Case	Open Hash (with AVL tree)
O(1+lgα)	O(logn)	Search(Key)
O(1+lgα)	O(logn)	Insert(Key)
O(1+lgα)	O(logn)	Delete(Key)
O(n)	O(n)	Minimum()
O(n)	O(n)	Maximum()

- קלט: n מספרים. פלט: המספרים השונים והריבוי. אלג': לכל נעשה find לטבלה האש, אם מצאנו נעלה קאונטר, אחרת נכניס. הכנסה: O(n) בממוצע, n^2 WC. סריקת הדפסה O(n).
- קלט: n מספרים. פלט: האם כולם שונים. אלג': נשתמש בתרגיל הקודם ונבדוק האם כל הקאונטרים 1.
- קלט: מערכים A,B בגודל n. פלט: האם A הוא פרמוטציה של B. אלג': נכניס את A לטבלת האש עם קאונטרים, אחכ נכניס את B רק שהפעם נוריד מהקאונטרים. בסוף נבדוק שהכל O(n). ממוצע.
- קלט: n מספרים. פלט: האם קיימים שני מספרים שסכומם 10. אלג': נכניס הכל לאהש, לכל x נעשה find ל- 10-x, סה"כ O(n)=O(1)*n. מקרה קצה כאשר מופיע 5 רק פעם אחת.

Perfect Hashing: זמן WC קבוע של O(1) מוכנים להשקיע בבניה O(n) *ממצע.*

מקום: O(n) כי $\sum n_j^2=\sum n_j+2\binom{n_j}{2}=n+2coll.\le3n$

בחירת g – נמפה אותם לתחום [1..ab_i^2] ונבחר a=2. תוחלת מס' ההתנגשויות של מפתח בודד קטן מ- 1/(ab_i). תוחלת מס 'ההתנגשויות של כל המפתחות – 0.5. שוב לפי מרקוב נקבל אותה תוצאה (תוחלת מס' הנסיונות – 2).

סיבוכיות לשיעורין:

- לפ"נ יש סיבוכיות לשיעורין f(n) אם כל סדרה של n פעולות לוקחת O(nf(n)). כלומר בממוצע כל פעולה לוקחת f(n). זה מעניין אם מדברים על סדרה של פעולות.
- מימוש תור בעזרת שתי מחסניות. פעולות על תור לוקחות O(1) ב- WC. נרצה מימוש אמורטייז ב- O(1). בהוצאה נעבור על אברי מחסנית העזור, ונשיך להכניס למחסנית הראשית. ברגע שרוצים להוציא ומחסנית העזר ריקה, שופכים אליה את המחסנית הראשונה. ב- WC O(n):יש n הכנסות ואז n הוצאות. אבל כל סדרה של פעולות לוקחות O(n) כי בכל סדרה של n פעולות עוברים במבנה לכל היותר n אברים ועל כל איבר מבוזבזים לכל היותר 4 פעולות. כלומר סך הכל לכל האברים 4n.

אם נרצה להעביר כל פעם חצי מהאברים, נקח פונק פוטל = ||s1|-|s2||

OS: (סטטיסטיקה סדר)

באמצעות ערימה אפשר למצוא את OS(k) ב- O(n) עבור בניית הערימה + O(klogn) למציאת האיבר המבוקש באמצעות k פעולות extractMax.

שיטה יותר טובה: ביצוע partition באקראי (O(n)) וחיפוש בצד הנכון. בממוצע זה לוקח O(n) אבל ב WC $O(n^2)$. הוכחה:

$$E(T(n))=O(n)+\frac{1}{n}\sum E(T(\max(k-1,n-k)))=O(n)+\frac{2}{n}\sum_{k=n/2}^{n-1}E(T(k))$$

שיטה עוד יותר טובה, SELECT – בחירת האלמנט ה-i הקטן ביותר מתוך n איברים: 1. מחלק ל- n/5 קבוצות של 5 איברים. 2. ומצא את החציון של כ"א מהקבוצות במין פשוט. 3. השתמש ברוב- SELECT למציאת חציון החציונים. 4. בצע partition סביב לחציון החציונים. נניח ש- k הוא מספר האברים בחלק הנמוך ו- n-k מספרים בחלק הגבוה. 5. השתמש ב- select רקורסיבית למציאת i בחלק הנמוך או i-k בחלק הגבוה.

רקורסיה: $T(n)=an+T(\frac{n}{5})+T(\frac{3}{4}n)$ תוצאה אגב: ניתן לבצע QS ב- nlogn WC.

- קלט: n מספרים ומספר k. פלט: k המקסימליים. אלג': נמצא את ה- OS ה- n-k ב- O(n) ואז נעבור על המספרים ונראה מי גדול ממנו.
- קלט: n מספרים. פלט: האם יש מספר שמופיע n/4 פעמים. אלג': נחפש את ה- OS (O(n)) ל- 4n/5 ... 2n/5, n/5. אם קיים איבר כמרשש הוא חייב להיות אחד מ- OS שמצאנו, לכן עבור כל אחד מהארבעה נעבור על המערך ונראה כמה הוא מופיע(4n).
- מציאת חציון משותף של שני(גם שלוש) מערכים ממוינים ללא חזרות. אלג': מציאת שני החציונים ובדיקה מי יותר גדול. במערך עם הגדול נפסול את הגדולים ממנו ובמערך עם הקטן נפסול את הקטנים ממנו ונמשיך רקורסיבית. (O(logn.

עצי 2-3:

לכל צומת פנימי יש m-2m ילדים. לשורש לפחות 2 ילדים. כל העלים באותו מרחק מהשורש. $h\approx\log_m n$

אם מלא – $n=2/4^{h-1}+1$