מיונים

מיון מהיר $O(n\log n)$

בוחרים איבר פיבוט x, ממקמים כך שהאיברים הקטנים ממנו משמאלו והגדולים ממנו מימינו, ואז ממיינים את תת המערך מימינו ואת תת המערך משמאלו. קריאה התחלתית: $QuickSort(A,1,n)$.

מימוש נאיבי:
$O(n^2)$ במקרה גרוע, $O(n\log n)$ במקרה ממוצע.

מימוש פיבוט חציון:
$O(n\log n)$ לכל מקרה. (עץ רקורסיה מאוזן).
נקבל זאת גם עבור פיבוט שהוא קרוב לחציון,
כלומר שממוין בין האיבר $\frac{n}{4}$ לבין האיבר $\frac{3n}{4}$.
לכן גם במימוש פיבוט אקראי נקבל $O(n\log n)$.

```
Quicksort(A, p, r):
❑ If p < r, then
    ❍ q ← Partition(A, p, r)
    ❍ Quicksort(A, p, q − 1)
    ❍ Quicksort(A, q + 1, r)
```

```
Partition(A, p, r):
❑ x ← A[r]
❑ i ← p − 1
❑ For j ← p to r − 1 do
    ❍ If A[j] ≤ x then
        • i ← i + 1
        • A[i] ↔ A[j]
❑ A[i + 1] ↔ A[r]
❑ Return i + 1
```

מיון מיזוג $O(n\log n)$

מיון חצי שמאלי וחצי ימני, ומזג בינהם. ⟸

מיון הכנסה: $O(n^2)$.

עוברים איבר איבר ומחלחלים אחורה עד שהאיבר לפניו לא גדול ממנו.

```
Merge(A,p,q,r)
Let L[1,q] left half array
Let R[q+1,r] right half array
Add cells: L[n_L + 1] = R[n_R + 1] = ∞
i = 1, j = 1
for k = p to r:
        if L[i] ≤ R[j]
                A[k] = L[i]
                i = i + 1
        else
                A[k] = R[j]
```

```
MergeSort(A,p,r)
if p < r
        q = (p+r)/2
        MergeSort(A,p,q)
        MergeSort(A,q+1,r)
        Merge(A,p,q,r)
```

```
InsertionSort(A)
for j = 2 to A.length:
        key = A[j]
        ; insert A[j] into sorted A[1,...,j−1]
        i = j − 1
        while i > 0 and A[i] > key
                A[i + 1] = A[i]
                i = i − 1
        A[i + 1] = key
```

סטטיסטי הסדר $O(n)$

סטטיסטי הסדר הk במערך $A=[a_1 \dots a_n]$
הוא האיבר שנמצא באינדקס k במערך A הממוין מהקטן לגדול.
מימוש נאיבי: מיון, $O(n\ \log n)$. אבל אנחנו רוצים להשתמש בזה עבור מיון מהיר, וזה צריך להיות $O(n)$.
מימוש חכם: $O(n)$, השיטה: כל שלב לעבור על תת המערך הנוכחי ולזרוק חצי ממנו.
ואז כשנשתמש עבור מיון מהיר נקבל: $T(n)=2T\left(\frac{n}{2}\right)+O(n) \rightarrow O(n\log n)$

אלגוריתם Select(A,k):

1. אם A קטן מספיק אז פתור ע"י מיון. (חסרים פה פרטים בבסיס הרקורסיה)
2. חלק את הקלט לחמישיות.
3. מצא חציון בכל חמישייה. נסמן את קבוצת החציונים ע"י B.
4. מצא את החציון של B שיסומן ע"י x, כלומר בצע $Select\left(B,\left\lceil\frac{|B|}{2}\right\rceil\right)$
5. השתמש ב- x בפיבוט, כלומר נחלק את A לשני חלקים:
   - $L=\{a \in A : a \leq x\}$
   - $R=\{a \in A : a > x\}$
6. אם $|L| \geq k$ אז נקרא ל- $Select(L,k)$
   אם $|L| < k$ אז נקרא ל- $Select(R,k-|L|)$

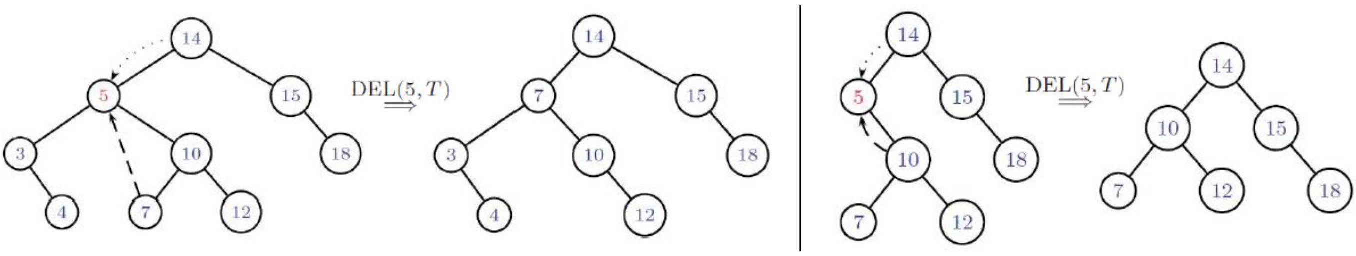
$O(BFS,DFS)=O(|V|+|E|$
$O(\text{חיפוש בינארי})=O(\log n)$
$(\log_a b)^{-1}=\log_b a$
$\log_a n=\dfrac{\log_b n}{\log_b a}$

עצי חיפוש

עץ חיפוש בינרי

עץ שבו לכל צומת הערכים בתת העץ הימני גדולים מערך הצומת, והערכים בתת העץ השמאלי קטנים מערך הצומת.
פעולות: $\mathrm{Member, Min, Max, Insert, Delete}$, סיבוכיות: $O(h)$ כאשר h גובה העץ, לכן נעדיף עצים מאוזנים (2-3, AVL).

מחיקת צומת: מחיקת עלה היא פשוטה, מקרים נוספים:



עצי 2-3 $O(\log n)$

עצי חיפוש בהם לכל צומת יש 2 או 3 ילדים (הימני אופציונלי), וכל המסלולים מהשורש לעלה הם באותו האורך.
איברי הקבוצה נמצאים בעלים, ממוינים משמאל (הקטן) לימין (הגדול). כל קודקוד מחזיק שדות $\mathrm{min_L, min_M, min_R}$.
סיבוכיות: מכיוון ש: $\log_3 n \leq h \leq \log_2 n$ כאשר n מספר העלים, נקבל: $O(h) = O(\log n)$

הכנסה:
- נמצא על ידי MEMBER את P ההורה של הצומת x אותה מוסיפים.
- אם לP 2 ילדים נוסיף ילד x במקום המתאים ונעדכן שדות $\mathrm{min_i}$ כנדרש. אם x מינימלי בP, נעדכן אחורה את $\mathrm{min_1}$.
- אם לP 3 ילדים: נפצל את P ל2 צמתים 'P,P כך שמתוך 3 הילדים וx נחלק לP את 2 הקטנים ול"P את 2 הגדולים.
  יהא G ההורה של P, אם לG ילד אחד פרט לP, נוסיף לו את 'P וסיימנו. אחרת: פצל את G והמשך כך במעלה העץ.
  תנאי עצירה: הגעה לשורש, פיצול שלו, ויצירת שורש חדש.

מחיקה:
- נמצא על ידי MEMBER את P ההורה של הצומת x אותה מוחקים.
- אם לx 2 אחים, נמחוק אותו, נעדכן שדות $\mathrm{min_i}$ כנדרש, וסיימנו.
- אם לx אח יחיד y, נמחוק את x ונתקן את העץ באופן הבא:
  תחילה נחפש אח מימין/משמאל עם שלושה ילדים ונוסיף את הילד הקטן/הגדול שלו לP.
  אם לא מצאנו, לאחיו של P יש 2 ילדים בלבד. נעביר את y לאחיו של P, ונמחוק את P.
  יהא G ההורה של P, אם לG כעת 2 ילדים, סיימנו. אחרת- המשך במעלה העץ באותו תהליך.

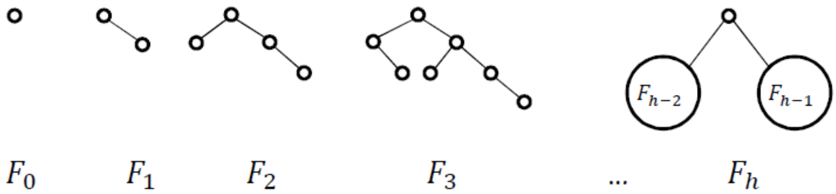
עצי AVL $O(\log n)$

עצי חיפוש בהם לכל צומת ההפרש בין גובה תת העץ השמאלי לגובה תת העץ הימני הוא לכל היותר 1.
נגדיר שדה לכל צומת: $\mathrm{BF}(v) = h_{L(v)} - h_{R(v)} \in \{-1,0,1\}$. נעדכן אותו לאחר פעולות הוספה או מחיקה.

נגדיר סדרת פיבונאצ'י של עצים:

- עץ F_h עץ AVL בגובה h.
- לF_h מספר מינימלי של צמתים מבין שאר עצי AVL בגובה h. מספר הצמתים: (f פונ' פיבונאצ'י ...0,1): $f(h+3)-1$.
- גובה עץ AVL עם n צמתים: $\log n$.

לכן סיבוכיות הפעולות: $O(\log n)$.



לאחר הוספה או מחיקה בעץ AVL יכול להיפגע האיזון ולכן נבצע תיקונים. לאחר הוספה יספיק תיקון יחיד, ולאחר מחיקה לאחר התיקון הראשון נמשיך במעלה העץ לבדוק שלא הופר האיזון כתוצאה מהתיקון שביצענו.

סוגי תיקונים: LL- גלגול ימינה, RR- גלגול שמאלה, RL- גלגול ימינה ואז RR, LR- גלגול שמאלה ואז LL.
ביחס לצומת שהופר בו האיזון, ולצומת שגרם את חוסר האיזון.

טבלאות ערבול

מילון:
קבוצה D של איברים שונים (מפתחות), עם הפעולות: $Insert, Delete, Member$. נניח שהמספרים בתחום $[0, N-1]$.
מימושים: מערך שייכות בגודל N, סיבוכיות $O(1)$, הבעיה היא בזבוז מקום אם N גדול מאוד ביחס לכמות האיברים.

טבלת ערבול:
נחזיק מערך בגודל m, כאשר $m \ll N$ מספר האיברים המקסימלי שנרצה להחזיק במילון.
נמפה את המפתחות בין התחומים בעזרת פנוקציית ערבול: $h: \{0, \dots, N\} \to \{0, \dots, m\}$.
הבעיה היא שמלבד שכדאי שהפונקציה תהיה על כדי לא לבזבז מקום, אם הפונקציה חח"ע אז לא עשינו כלום כי $m = N$,
אבל אם היא לא חח"ע אז נקבל שקיים $h(x_i) = h(x_j)$, ואז ניתקל בבעיה כשנרצה למקם את 2 האיברים במערך.

הנחת ערבול אחיד: פונ' h ממפה כל איבר בהסתברות $\frac{1}{m}$ לכל אחד מהתאים בטווח, באופן בלתי תלוי באיברים אחרים, כלומר הסיכוי שעבור מפתחות שונים x, y נקבל $h(x) = h(y)$ היא $\frac{1}{m}$.

טבלה פתוחה:
אם כמה מפתחות ממופים לאותו תא במערך, נחזיק אותם ברשימה מקושרת שהתא במערך מבציע אליה.
סיבוכיות: במקרה הגרוע כולם ממופים לאותו התא ונקבל $O(n)$, אבל מעשית נקבל לרוב $O(1)$, אם המפתחות מפולגים באופן אחיד ומפוזרים היטב ביחס לפונקציה.
סיבוכיות (תחת ההנחה): לכל חיפוש, כולל הפעלת הפונ', נקבל $O(1 + \alpha)$. עבור מספר איברים $n = O(m)$, נקבל $O(1)$.

שיטת החלוקה: $h(x) = x \bmod m$, כאשר מומלץ m ראשוני.

שיטת הכפל: $A \in [0,1]$, $h(x) = \lfloor m * (xA - \lfloor xA \rfloor) \rfloor$, כאשר מומלץ (Knuth) $A = \frac{\sqrt{5}-1}{2}$.

ערבול אוניברסלי:
עבור פונ' ערבול מסויימת, ניתן לבחור מפתחות כך שימופו לאותו הערך וכך נקבל סיבוכיות $O(n)$ לכל פעולה.
לכן נרצה לבחור באקראי פונקציית ערבול מתוך קבוצה מסויימת, וכך נשיג ביצועים טובים יותר בממוצע לכל סט מפתחות.
הגדרה: H אוסף סופי של פונ' ערבול $h_i: U \to \{0, \dots, m-1\}$ יקרא אוניברסלי אם לכל זוג מפתחות x,y מספר הפונקציות עבורן מתקיים $h(x) = h(y)$ הוא לכל היותר $\frac{|H|}{m}$. לכן עבור פונ' אקראית הסיכוי להתנגשות מפתחות שונים היא לכל היותר $\frac{1}{m}$.
סיבוכיות: כמו קודם תחת ההנחה ועבור מספר איברים $n = O(m)$, נקבל $O(1)$.

שיטת החלוקה: $h_{a,b}(x) = \big((ax + b) \bmod p\big) \bmod m \ \mid p \text{ prime}, a \in Z_p^*, b \in Z_p$, כלומר: $H_{pm} = \{h_{a,b}\}$.

הנחת ערבול אחיד מוכלל: כל איבר בתחום מופנה לכל פרמוטציה h_i בהתפלגות אחידה באופן בלתי תלוי.

טבלה סגורה: ($n \le m$, בניגוד לטבלה פתוחה)
יש סדרת פונ' ערבול כך שעבור מפתח חדש x, נמפה אותו ל h_i הראשון הפנוי עבור i מינימלי.
סיבוכיות: במקרה גרוע: $O(n)$, תחת הנחת ערבול אחיד וטבלה לא מלאה מדי ($n \le \frac{m}{2}$) נקבל $O(1)$.
לינארי: בהינתן פונ' h, נגדיר: $h_i(x) = h(x) + i \pmod m$.
ריבועי: בהינתן פונ' h וקבועים c_1, c_2, נגדיר: $h_i(x) = h(x) + c_1 i + c_2 i^2 \pmod m$.
כפול: בהינתן 2 פונ' h, h', נגדיר: $h_i(x) = h(x) + ih'(x) \pmod m$

נוצרת בעיה כאשר מוחקים איבר, שאז שרשרת החיפוש תתנתק. פתרונות:
נוציא איברים עד המקום הפנוי הבא, נחזיר אותם בלי האיבר אותו רוצים למחוק.
שיטת המצבה: כאשר נמחוק איבר נשאיר סימון 'מצבה', ואם נתקל בה בזמן חיפוש נדע להמשיך לחפש ולא לעצור.
השיטה השניה יותר פשוטה אך חסרונה בכך שזמן החיפוש תלוי כעת גם באיברים שנמחקו.

בעיות:
צריך לדעת מראש סדר גודל למספר האיברים שנרצה להחזיק.
פתרון: כשהטבלה תתמלא נעתיק אותה לטבלה בגודל כפול. כך נשמור על $O(1)$ ומדי פעם נעשה פעולה יקרה.
האם ניתן להנות גם מ $O(1)$ בממוצע וגם מ $O(\log n)$ במקרה ממוצע?
פתרון: טבלה פתוחה כך שכל תא מצביע לעץ חיפוש מאוזן.

סיבוכיות פחת

הרעיון: חישוב סיבוכיות לסדרת פעולות במבנה נתונים, כך שאם יש מעט פעולות יקרות ורוב פעולות זולות, הממוצע טוב.
הגדרה: אם סדרת m פעולות מתבצעת בזמן כולל $T(m)$ אז סיבוכיות הפחת לכל פעולה בסדרה היא $T(m)/m$.

שיטת החסכון:
מחיר פעולה בסיסית: $O(1)=1_{NIS}$, משייכים לכל פעולה עלות פחת, כך שפעולות מהירות משאירות עודפי כסף לשימוש עבור פעולות איטיות, אם לא נכנסים לחוב אז סך התשלומים חוסם את המחיר האמיתי.
דוגמה: מחסנית, $pop=0_{NIS}$, $push=2_{NIS}$, כך בכל דחיפה חוסכים להוצאה עתידית, ואי אפשר להיכנס לחוב.

שיטת פוטנציאל:
עלות פחת לפעולה ה-i: $\hat{c}_i=c_i+\phi(D_i)-\phi(D_{i-1})$, כאשר c_i עלות הפעולה, D_i מבנה הנתונים לאחר הפעולה ה-i.
כמו כן ϕ פונ' פוטנציאל שמשייכת למצב המבנה מספר ממשי, הצריכה לקיים: $\forall i:\phi(D_i)\geq\phi(D_0)$. על מנת לחסום מלמעלה את סך עלות הסדרה: $\sum_{i=1}^{m}\hat{c}_i=\sum_{i=1}^{m}(c_i+\phi(D_i)-\phi(D_{i-1}))=\sum_{i=1}^{m}c_i+\phi(D_n)-\phi(D_0)\Rightarrow\phi(D_m)\geq\phi(D_0)$
דוגמה: עבור מחסנית נגדיר את פונקצית הפוטנציאל להיות מסר האיברים במחסנית.

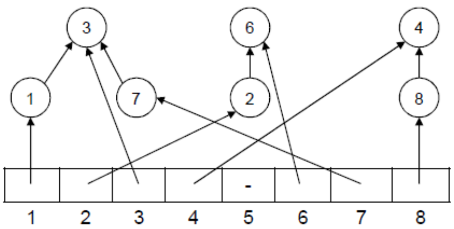
קבוצות זרות

אוסף תתי קבוצות S_i זרות בזוגות של הקבוצה $\{1,\dots,n\}$. פעולות: $Make(x), Find(x), Union(x,y)$.
פעולה Make יוצרת סט חדש עם איבר x, Find מחזירה את נציג הקבוצה, Union מאחדת את הקבוצות של x,y.

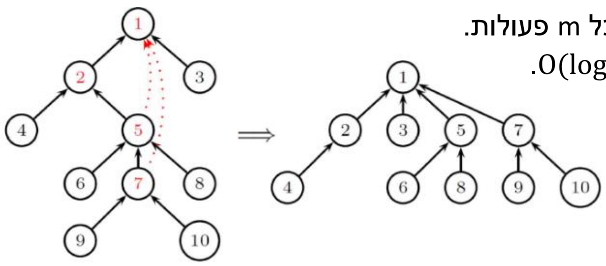
מערך: בגודל n, $A[i]$ הוא הנציג של i. סיבוכיות: $Make, Find: O(1), Union: O(n)$.

רשימה מקושרת: לכל תת קבוצה רשימה מעגלית שבראשה הנציג. כמו כן יש מערך דגלים/מצביעים של האיברים $1,\dots,n$.
סיבוכיות: $Make: O(1), Find, Union: O(n)$, אם Union עובד על נציגים נקבל עבורו $O(1)$.

מערך רשימות: מטריצה n x 2 כך שלכל תא i יש רשומת נציג ורשומת next, האיבר הבא הקבוצה.
סיבוכיות: $Make, Find: O(1)$, $Union: O(n)$, אורך אחת הרשימות עליה עוברים.
לכן כדאי לעבור על הרשימה הקטנה יותר, נוסיף לכל תא רשומה size. (לא מוריד סיבוכיות).
סיבוכיות פחת: לסדרת m פעולות נקבל $O(m\log m)$.



עצים הפוכים: כל תת קבוצה מוחזקת בעץ כשהנציג בשורש (מחזיק גודל ת"ק).
נחזיק מערך בגודל n עם מצביעים לעצים, ומערך רשומות של קבוצות.
סיבוכיות: $Make=O(1), Find=O(h_x), Union=O(h_x+h_y)$.
מכיוון שעומקו כל עץ בכל שלב הוא $\log n$ נקבל סיבוכיות $O(\log n)$.
(כל איחוד הצומת שעומקו גדל נמצא בעץ הקטן, שהכפיל כעת את גודלו).
שיטת האיחוד עליה דיברנו: שורש העץ הקטן מצביע לשורש העץ הגדול. שיטה נוספת: לפי עומק. ואז נקצר מסלולים:
קיצור מסלולים: בכל פעולת Find/Union כאשר עולים במעלה העץ דואגים שכל צומת שעברנו דרכו יצביע ישירות לשורש.
כך נחסוך עבודה בחיפושים עתידיים ונקבל סיבוכיות $O(m\log^* n)$ לכל m פעולות.
כאשר: $\underbrace{\log(\log(..(\log n)..))}_{\min m \text{ time of log}}\leq 1 \equiv m=\log^* n$. $O(\log^* n_{\leq 5})\approx O(1)$.



<u>עץ פורש מינימום</u>
בהינתן גרף ממושקל, עץ פורש מינימום הוא עץ שסכום משקלי הקשתות בו מינימלי ביחס לעצים פורשים נוספים.
קרוסקל: בכל פעם מוסיפים לעץ קשת מינימלית שלא מחברת בין 2 קודקודים שכבר הוספנו לעץ, עד שכל הקודקודים בעץ.
סיבוכיות: במימוש ע"י קבוצות זרות נקבל $O(|E|\log|V|)$.
פרים: נתחיל $U=\{v_0\}$ ובכל פעם נוסיף את הקודקוד הישיג הקרוב ביותר שאינו בU, עד שכל הקודקודים בעץ.
סיבוכיות: $O(|E|+|V|\log|V|)$ (ערימת פיבונאצ'י)

Prim(G,w):
- $U=\{r\}$ /* r is an arbitrary vertex */
- $A\leftarrow\emptyset$
- While $U\neq V$ do
  - let (x,y) be a minimum weight edge that crosses $(U, V\setminus U)$
  - $U\leftarrow U\cup\{y\}$
  - $A\leftarrow A\cup\{(x,y)\}$
- return A

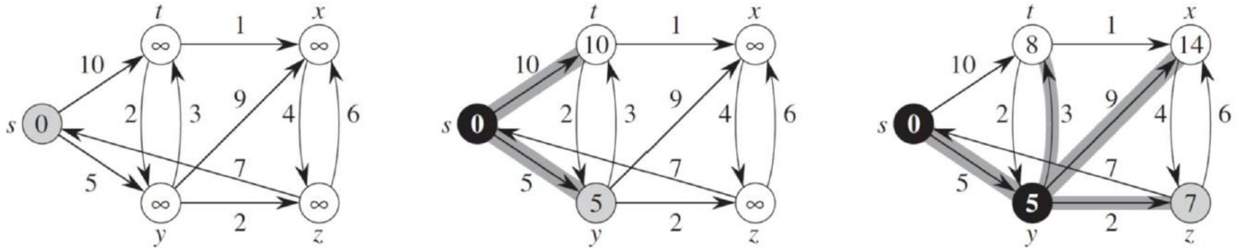
Kruskal(G,w):
- $A\leftarrow\emptyset$
- Let F be a forest of n trees, a tree for each vertex in G
- While $|F|>1$ do
  - Let (x,y) be a minimum weight edge connecting two trees in F
  - $A\leftarrow A\cup\{(x,y)\}$
- return A

מרחקים קצרים ביותר

בהינתן גרף ממושקל (יתכנו משקלים שליליים), נרצה למצוא מסלול קצר ביותר בין קודקודים. אם אין משקלים נעשה BFS.

מרחקים קצרים ביותר בין קודקוד מקור לשאר הקודקודים

דייקסטרה: **לא מתמודד עם משקלים שליליים.** (חמדני), סיבוכיות: $O(|E| + |V| \log|V|)$.
רעיון: כל צומת מחזיק מרחק $D(v)$ שמתעדכן (מאותחל ∞, חוץ מ$D(s) = 0$). בכל שלב נצרף לקבוצה $S = \phi$ את הקודקוד במרחק המינימלי ממנה (בהתחלה s עצמו), ונעדכן כל שכן שלו אם אפשר לקצר את המרחק שלו דרך הקשת בינהם. בנוסף קיים מערך p שמכיל את הצומת האבא לכל צומת, ודרכו ניתן לשחזר את המסלול הקצר ביותר אל הצומת, בנוסף לאורכו.



(a) (b) (c)

ונמשיך z,t,x

בלמן-פורד: **מתמודד עם משקלים (ומעגלים) שליליים**. סיבוכיות: $O(|V||E|)$
רעיון: נבצע $|V| - 1$ פעמים: עבור כל קשת בצע הקלה (אם אפשר). לבסוף נריץ עוד איטרציה כזו ואם קשת הוקלה סימן שיש מעגלים שליליים בגרף ואי אפשר לדבר על מסלול קצר ביותר. בנוסף קיים מערך p שמכיל את הצומת האבא לכל צומת, ודרכו ניתן לשחזר את המסלול הקצר ביותר אל הצומת, בנוסף לאורכו, כך: מסלול הפוך למסלול: $s, \ldots, p[p[v]], p[v], v$.
נשים לב: אם אין מעגלים שליליים, בהכרח בין כל 2 צמתים קיים מסלול קצר ביותר שהוא פשוט, ומכיל עד $|V| - 1$ קשתות.

גרסת תכנות דינמי: (הבסיסית)
נחשב מרחק מינימלי מתעדכן: $d_{s,0} = 0$, $d_{v,0} = \infty$, $d(P_{s\to v}): |P| \le i = d_{v,i} = \min\left\{d_{v,i-1}, \min\limits_{u\to v}\left\{d_{u,i-1} + w(u,v)\right\}\right\}$.
ניתן להתעלם מצמתים לא ישיגים מs ע"י BFS, לכן נניח $|E| \ge |V| - 1$. נקבל (לאחר אתחול):

for $i = 1$ to $n - 1$ do
- for every vertex v do
  - $d[v,i] \leftarrow d[v,i-1]$
  - for every edge (u,v) entering v do
    - if $d[v,i] > d[u,i-1] + w(u,v)$ then $d[v,i] \leftarrow d[u,i-1] + w(u,v)$

Bellman Ford

- $d[s] \leftarrow 0; p(s) \leftarrow \mathrm{NULL}$
- for every $v \neq s$ do $d[v] \leftarrow \infty$
- for $i = 1$ to $n - 1$ do
  - for every edge (u,v) do Relax(u,v)
- for every edge (u,v) do
  - Relax(u,v)
  - if (u,v) was strictly relaxed then return NEG-CYCLE
- Return (d,p)

Procedure Relax(u,v):
- if $d[v] > d[u] + w(u,v)$ then
  - $d[v] \leftarrow d[u] + w(u,v)$
  - $p(v) \leftarrow u$

Dijkstra

- $S \leftarrow \{s\}; D(s) \leftarrow 0$
- For all nodes $v \in N(s)$ do
  - $D(v) \leftarrow w(s,v)$
- While $S \neq V$ do
  - Find $v \notin S$ minimizing $D(v)$
  - $S \leftarrow S \cup \{v\}$
  - For all $u \in N(v) \setminus S$ do:
    - If $D(v) + w(v,u) < D(u)$ then
      - $D(u) \leftarrow D(v) + w(v, u)$
      - $p(u) \leftarrow v$

מרחקים קצרים ביותר בין כל זוג קודקודים

עבור משקלים חיוביים ניתן להריץ מכל קודקוד דייקסטרה.
אם יש משקלים שליליים אז ניתן להריץ מכל קודקוד בלמן-פורד ונקבל $O(|V|^2|E|)$. פתרונות טובים יותר:

פלויד: $O(|V|^3)$
נמספר צמתים $\{1,2,\dots,n\}$, נגדיר $d_{ij}^{(k)}$ המסלול הקצר ביותר $i \to j$ שעובר רק בצמתים $\{1,2,\dots,k\}$.
נקבל נוסחה: $d_{ij}^{(0)} = w_{ij}$, $d_{ij}^{(k)} = \min\left\{d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right\}$. פלט: מטריצת סמיכויות עם משקלים.
ניתן לקבל גם מסלולים: $\pi_{ij}^{(0)} = \begin{cases} \text{NIL if } i=j \text{ or } w_{ij}=\infty \\ i \text{ if } i \neq j \text{ and } w_{ij} < \infty \end{cases}$, $\pi_{ij}^{(k)} = \begin{cases} \pi_{ij}^{(k-1)} \text{ if } d_{ij}^{(k-1)} \le d_{ik}^{(k-1)} + d_{kj}^{(k-1)} \\ \pi_{kj}^{(k-1)} \text{ if } d_{ij}^{(k-1)} > d_{ik}^{(k-1)} + d_{kj}^{(k-1)} \end{cases}$.
שחזור מסלול: $1 \to 2$ נסתכל על $T[1,2] = 3$ כלומר המסלול הוא: $1 \to 3,2$, נמשיך באופן רקורסיבי.

```
Algorithm 3 MOD-FLOYD-WARSHALL(W)
n= W.rows
D^0 = W
π^0 is a matrix with nil in every entry
for i=1 to n do
    for j = 1 to n do
        if i ≠ j and D^0_{i,j} < ∞ then
            π^0_{i,j} = i
        end if
    end for
end for
for k=1 to n do
    let D^k be a new n × n matrix.
    let π^k be a new n × n matrix
    for i=1 to n do
        for j = 1 to n do
            if d^{k-1}_{i,j} ≤ d^{k-1}_{i,k} + d^{k-1}_{k,j} then
                d^k_{i,j} = d^{k-1}_{i,j}
                π^k_{i,j} = π^{k-1}_{i,j}
            else
                d^k_{i,j} = d^{k-1}_{i,k} + d^{k-1}_{k,j}
                π^k_{i,j} = π^{k-1}_{k,j}
            end if
        end for
    end for
end for
```

כפל מטריצות מבוסס בלמן פורד: $O(|V|^3 \log|V|)$, ובמימוש הפרד ומשול (Strassen) אפשר לקבל $O(|V|^3)$.
נחשב בטבלה תלת ממדית: $D_0[i,j] = \infty, D_0[i,i] = 0$, $D_l[i,j] = \min_k\{D_{l-1}[i,k] + w(v_k, v_j)\}$.
כאשר $D_l[i,j]$ הוא המרחק המינמלי $v_i \to v_j$ הכולל לכל היותר l קשתות.
ניצור מטריצת משקלים: $W[i,j] = w(v_i, v_j)$, נגדיר פעולה: $Z = X \odot Y \Longrightarrow Z[i,j] = \min_k\{X[i,k] + Y[k,j]\}$.
ונקבל שמתקיים: $D_l = D_{l-1} \odot W$, ומתוך מבנה D_0 נקבל: $D_{l>0} = W^l$.

ג'ונסון מבוסס דייקסטרה+בלמן פורד: $O(|V|^2 \log|V| + |V||E|)$
רעיון: נחליף את פונ' המשקלים w בפונ' w' שהיא אי שלילית, ומשמרת מסלולים קצרים ביותר, נריץ דייקסטרה מכל קודקוד.
נגדיר: $w'(u,v) = w(u,v) + h(u) - h(v)$, נוסיף לגרף קודקוד s עם קשתות במשקל 0 לכל שאר הצמתים, ונגדיר פונ' גובה: $h(v) = \delta(s,v)$, נחשב את h ע"י הרצת בלמן-פורד, אם אין מעגלים שלילים נריץ, דייקסטרה מכל קודקוד. לבסוף נעדכן את המרחקים שמצאנו: $D[u,v] = D[u,v] - h(v) + h(u)$.

סגור טרנזיטיבי

בהינתן גרף $G(V,E)$ הסגור הטרנזיטיבי שלו הוא: $G^*(V,E^*): E^* = \{(u,v) \mid \text{there is a path } u \to v\}$.
ניתן למצוא על ידי שימוש באלגוריתם פלויד, בשינוי הנוסחה, כך שנקבל $O(|V|^3)$:

$$d_{ij}^{(k)} = d_{ij}^{(k-1)} \vee \left(d_{ik}^{(k-1)} \wedge d_{ki}^{(k-1)}\right), \quad d_{ij}^{(0)} = 1 \text{ if } (i,j) \in E \text{ or } i=j, \text{ else } 0$$

התמרת פורייה מהירה

פולינומים

חישוב חזקה a^n - סיבוכיות $O(\log n)$. בהינתן n השורשים, חישוב מקדמים לוקח $O(n^2)$.

ייצוג טבעי: (וקטור מקדמים)
חישוב ערך פולינום (במעלה n) בנקודה $O(n)$.
חיבור פולינומים: $O(n+m)$.   כפל פולינומים: $O(nm)$.   חילוק פולינומים: $O(m(n-m)), n>m$.

ייצוג ע"י זוגות:
בהינתן $(x_0,y_0),\dots,(x_{n-1},y_{n-1})$ (x_i שונים) הם מייצגים פולינום יחיד ממעלה n-1 לכל היותר שמקיים $P(x_i)=y_i$, ניתן למצוא אותו בסיבוכיות $O(n^2)$.
חיבור פולינומים: $(x_0,z_0),\dots,(x_{n-1},z_{n-1})$, $(x_0,y_0),\dots,(x_{n-1},y_{n-1})$ ב $O(n)$ (חיבור y_i+z_i).
כפל פולינומים: נייצג את הפולינומים ע"י 2n-1 נקודות, נכפול $y_i\cdot z_i$ ונקבל סיבוכיות $O(n)$.

מעבר בין ייצוגים: $O(n\log n)$
ייצוג טבעי טוב לחישוב ערך בנקודה, ייצוג בזוגות טוב לחישוב חיבור וכפל. (כל אלה $O(n)$).
כפל פולינומים: ריפוד באפסים לאורך $\min\{x\geq 2n-1, x=2^a\}$, מעבר לייצוג בזוגות, כפל, מעבר חזרה. $O(n\log n)$.

התמרת פורייה דיסקרטית

ההתמרה של $(a_0,\dots,a_{n-1})$ היא: $y_k=\sum_{j=0}^{n-1}a_j\cdot\left(w_n^k\right)^j$. כאשר: $w_n^k=e^{i\frac{2\pi}{n}k}$.
נגדיר לפולינום $A(x)$ ממעלה $n-1$: $A'(x)=a_0+a_2x+\dots+a_{n-2}x^{\frac{n}{2}-1}$, $A''(x)=a_1+a_3x+\dots+a_{n-1}x^{\frac{n}{2}-1}$.
נקבל שמתקיים: $A(x)=A'(x^2)+xA''(x^2)$, ולכן נקבל: $y_{k+\frac{n}{2}}=y'_k-w_n^k\cdot y''_k$, $y_k(k\leq n/2)=y'_k+w_n^k\cdot y''_k$.
לכן נקבל אלגוריתם הפרד ומשול לחישוב התמרה בסיבוכיות $O(n\log n)$.
ההתמרה ההפוכה של $(y_0,\dots,y_{n-1})$ היא: $a_k=\frac{1}{n}\sum_{j=0}^{n-1}y_j\cdot\left(w_n^k\right)^{-j}$. לכן נשתמש באותו אלגוריתם.

כפל פולינומים: נרפד באפסים את וקטורי המקדמים: $len=\min\{x\geq 2n-1, x=2^a\}$, נתמיר, נכפיל, נתמיר. $O(n\log n)$.

```
Recursive-FFT(a)
❒ if n = 1 then return a
❒ w_n ← e^(i 2π/n)
❒ w ← 1
❒ a' ← (a_0, a_2, ..., a_{n-2})
❒ a'' ← (a_1, a_3, ..., a_{n-1})
❒ y' ← Recursive-FFT(a')
❒ y'' ← Recursive-FFT(a'')
❒ for k = 0 to n/2 − 1 do
    ❍ y_k ← y'_k + w y''_k
    ❍ y_{k+n/2} ← y'_k − w y''_k
    ❍ w ← w · w_n
❒ return y
```

```
Strassen(A, B)
if A.length == 1 then
    return A[1] · B[1]
end if
Let C be a new n by n matrix
A11 = A[1..n/2][1..n/2]
A12 = A[1..n/2][n/2 + 1..n]
A21 = A[n/2 + 1..n][1..n/2]
A22 = A[n/2 + 1..n][n/2 + 1..n]
B11 = B[1..n/2][1..n/2]
B12 = B[1..n/2][n/2 + 1..n]
B21 = B[n/2 + 1..n][1..n/2]
B22 = B[n/2 + 1..n][n/2 + 1..n]
S1 = B12 − B22
S2 = A11 + A12
S3 = A21 + A22
S4 = B21 − B11
S5 = A11 + A22
S6 = B11 + B22
S7 = A12 − A22
S8 = B21 + B22
S9 = A11 − A21
S10 = B11 + B12
P1 = Strassen(A11, S1)
P2 = Strassen(S2, B22)
P3 = Strassen(S3, B11)
P4 = Strassen(A22, S4)
P5 = Strassen(S5, S6)
P6 = Strassen(S7, S8)
P7 = Strassen(S9, S10)
C[1..n/2][1..n/2] = P5 + P4 − P2 + P6
C[1..n/2][n/2 + 1..n] = P1 + P2
C[n/2 + 1..n][1..n/2] = P3 + P4
C[n/2 + 1..n][n/2 + 1..n] = P5 + P1 − P3 − P7
return C
```

הכפלת מטריצות מהירה $O(n^{2.71})$

נרשום על פי הנוסחה: $\begin{pmatrix}A_{11} & A_{12}\\ A_{21} & A_{22}\end{pmatrix}\begin{pmatrix}B_{11} & B_{12}\\ B_{21} & B_{22}\end{pmatrix}=\begin{pmatrix}C_{11} & C_{12}\\ C_{21} & C_{22}\end{pmatrix}=\begin{pmatrix}P_5+P_4-P_2+P_6 & P_1+P_2\\ P_3+P_4 & P_1+P_5-P_3-P_7\end{pmatrix}$

כאשר: $P_1=A_{11}(B_{12}-B_{22}$, $P_2=B_{22}(A_{11}+A_{12})$, $P_3=B_{11}(A_{21}+A_{22})$, $P_4=A_{22}(B_{21}-B_{11})$
$P_5=A_{11}(B_{11}+B_{22})+A_{22}(B_{11}+B_{22})$, $P_6=A_{12}(B_{21}+B_{22})-A_{21}(B_{11}+B_{12})$
$P_7=A_{11}(B_{11}+B_{12})-A_{21}(B_{11}+B_{12})$

במטריצות עבורן $n\neq 2^a$ נרפד באפסים עד לחזקה של 2.

רשתות זרימה

רשת זרימה: $N = (G, s, t, c)$.
פונ' זרימה: $f: V \times V \to R$, סימטריה ניגודית, זרימה בקשת מוגבלת בקיבול, מה שנכנס זה מה שיוצא (פרט ל s, t).
ערך זרימה: סך הזרימה היוצאת מs או לחלופין הנכנסת לt, או לחלופין הזרימה העוברת בחתך כלשהו.
זרימת מקסימום: $|f_{max}| = \min \{c(S, \bar{S})\}$, f מרווה חתך אמ"מ f זרימה מקסימלית.
רשת שיורית: רשת שיורית של רשת N וזרימה f היא: $N_f: c_f(u,v) = c(u,v) - f(u,v)$, $E_f = \{(u,v) | c_{f(u,v)} > 0\}$.
סופר פוזיציה: f_1 זרימה בN, f_2 זרימה ב N_f, אז $f_1 + f_2$ זרימה בN וערכה $|f_1| + |f_2|$. $(f_2 = \max \to f_1 + f_2 = \max)$
f_1, f_2 זרימות בN, f_2, אז $f_1 - f_2$ זרימה ב N_{f_2} וערכה $|f_1| - |f_2|$.

פורד-פולקרסון: כל עוד יש מסלול שיפור (מסלול פשוט $s \to t$ ברשת השיורית), הרווה אותו, המשך עם הרשת השיורית.
סיבוכיות: יש חופש בבחירת המסלולים, במקרה גרוע נרווה כל איטרציה ב1 ונגיע ל $O(|E||f_{max}|) = O(|E||V|c_{max})$.
EK2: בחר מסלול שיפור בעל קיבול גדול ביותר (<u>לקיבולים שלמים</u>). $[T_D: O(m + n\log n)]$ $O(T_{Dijkstra} \cdot m\log\left(\underbrace{nc_{max}}_{F^*}\right)$.
EK1: בחר מסלול שיפור קצר ביותר. $O(|V||E|^2)$.
דיניץ: מימוש יעיל של EK1, בחר מסלול שיפור קצר ביותר בגרעין (=גרף שלבים) של הרשת. $O(|V|^2|E|)$.

<u>שידוך גרף דו צדדי</u> $G(L, R, E)$
שידוך הוא תת קבוצה $M \subseteq E$ של קשתות זרות בצמתים. שידוך מושלם: כל הקודקודים בשידוך: $|L| = |R| = |M|$.
שידוך מקסימלי: לא ניתן להוסיף קשת לשידוך. שידוך מקסימום: שידוך בסכום משקלי קשתות מקסימלי.

שידוך מקסימום: $O(\sqrt{|V|} \cdot |E|)$
גרף חדש: $V = V \cup \{s,t\}, E = E \cup \{\forall v \in L: (s,v), \forall v \in R: (v,t)\}$, משקלי כל הקשתות 1.
כעת נמצא זרימת מקסימום (דיניץ) בגרף החדש וניצור שידוך כך: $M = \{(u,v) \mid u \in L, v \in R, f(u,v) > 0\}$.

חתך מינימום: נריץ אלגוריתם זרימת מקסימום, נריץ BFS על הרשת השיורית, נוסיף כל קודקוד שהגענו אליו לחתך.

בעיות קשות

בעיית 3-צביעה: (2-צביע ניתן לפתור עם BFS בסיבוכיות פולינומית)
נבחר בכל פעם קבוצה צבע א' $|U| \le \lfloor n/3 \rfloor$, נבדוק האם שאר הגרף הוא 2-צביע. $(O(2^n/3 \cdot poly(n))$.

קירובים:
בעיית מינימום: פתרון נקרא קירוב r אם הוא חוקי ומחירו חסום מלמעלה על ידי $r \cdot optimum$.
בעיית מקסימום: פתרון נקרא קירוב r אם הוא חוקי ומחירו חסום מלטה על ידי $r^{-1} \cdot optimum$.

כיסוי צמתים:
קבוצת קודקודים מינימלית שנוגעת בכל הקשתות.
נמצא שידוך מקסימלי M: הוסף קשת לשידוך עד שלא ניתן להוסיף (לכל קשת נעבור על הקודקודים בשידוך $(O(|E||V|)$).
נחזיר כיסוי: הקודקודים בקשתות השידוך. זה פתרון קירוב 2 מכיוון ש: $opt \ge |M|$ ואנחנו מצאנו קבוצה בגודל $2|M|$.

כיסוי בקבוצות:
נתונה קבוצה $U = \{1, \dots, n\}$ ותתי קבוצות $S_1, \dots, S_m$. עלינו למצוא אוסף מינימלי של תתי קבוצות שאיחודן U.
אלגוריתם חמדן: הוסף בכל פעם קבוצה שמוסיפה הכי הרבה איברים חדשים לאיחוד. $(O(|V||E|^2 \ln|V|)$.
האלגוריתם מוצא קירוב $\ln|V|$ של הפתרון, כלומר מספר תתי הקבוצות בפתרון הוא: $|opt| \cdot \ln|V|$.

ראשוניים:
צפיפות: כמות המספרים הראשוניים עד N היא בקירוב $\frac{N}{\ln N}$, לכן אם ננחש נצליח בהסתברות $\frac{1}{\ln N}$, נצטרך $\ln N$ ניחושים.
אלגוריתם מילר רבין: נחש בהתפלגות אחידה $a \in Z_N^+$, חשב משהו, בדוק משהו, אם זה פריק תודיע. (לראשוני לא יודיע).
נקבל שאם N ראשוני, האלגוריתם לא יודיע, ואם N פריק, האלגוריתם יודיע בהסתברות חצי לפחות.
כדי להקטין הסתברות לשגיאה, נריץ t פעמים, ההסתברות שN פריק ולא נגלה היא 2^{-t}. עבור $t = 100$ זה מספיק בטוח.

חתך מקסימום:
נמצא חתך במשקל קשתות מקסימלי: נוסיף כל צומת לחתך בהסתברות $\frac{1}{2}$ באופן ב"ת.
נקבל תוחלת משקל $\frac{1}{2}\sum_{e \in E} w_e$ ומכיוון שמשקל חתך מקסימלי אופטימלי חסום ב-$\sum_{e \in E} w_e$, קיבלנו קירוב 2.