אוניברסיטת חיפה
החוג למדעי המחשב

תכנון וניתוח אלגוריתמים

סיכומי ההרצאות של פרופ' יורי רבינוביץ

נכתב על ידי בר וייסמן

סמסטר א' תשפ"ג

תוכן העניינים

הערה. קובץ זה לא עבר אישור של אף גורם מוסמך, וייתכנו בו טעויות ואי דיוקים מכל סוג.

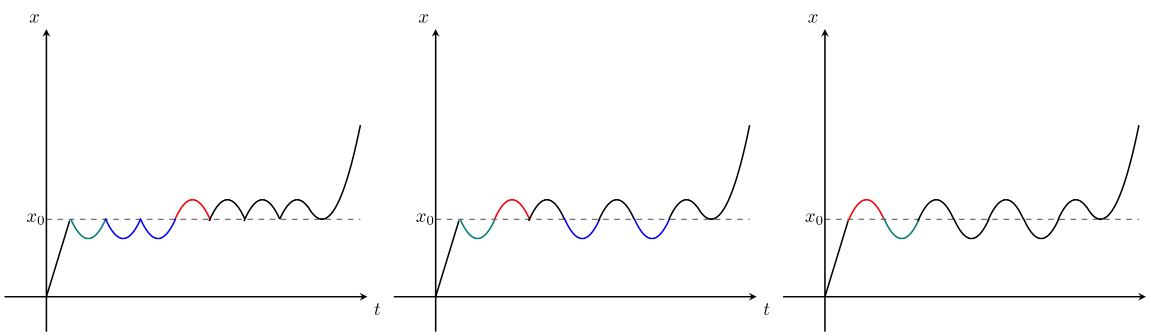
1 מבוא

1.1 גמל מים

- בעיה: גמל במדבר רוצה להגיע כמה שיותר רחוק עם n (טבעי) יחידות מים.
- מודל:
  - הגמל משתמש ביחידת מים אחד ליחידת מרחק אחת
  - הגמל יכול להשאיר מים בכל מקום
  - הגמל יכול לסחוב בכל רגע רק יחידת מים אחת

ננסה להבין את הפתרון האופטימלי על ידי אבחנות:

1. בה"כ, בפתרון האופטימלי נביא מראש כמה שיותר מים לכל נקודה במסלול לפני שנתקדם הלאה.
2. לא משאירים מים מאחור כי כל המים שימושיים להתקדמות.



איור 1: מימין לשמאל: פתרון אופטימלי, לאחר החלפת אדום וירוק, לאחר ביצוע החלפות עם האבחנה

נגדיר פונקציה $In\left(x\right)$ בתור כמות המים שאיתה נכנסים לנקודה x בפעם האחרונה (במסלול הממוין).

- לפי אבחנה 2, הפונקציה רציפה ומונוטונית יורדת.
- נחלק את הבעיה לתתי בעיות ע"פ כמות המים הנתונה: $0, \dots, n$.
  - אם פתרון הבעיה מ-n עד 0 אופטימלי, אז בהכרח גם כל פתרון תת בעיה מ-i עד $i-1$ אופטימלי.
- נחשב את המרחק המקסימלי שניתן לעבור מ-i יחידות מים ל-$i-1$ יחידות מים:
  - בכל מעבר ניתן לעביר לכל היותר יחידת מים אחת.
  - לכן, נצטרך לעבור כל קטע במסלול לפחות i פעמים לכיוון ההליכה
  - בין כל שתי פעמים שאנו מתקדמים הלוך על אותו הקטע נצטרך לחזור חזרה.
  - לכן, לעבור לפחות $i-1$ פעמים חזרה כדי לקחת מים.
- אם אורך הקטע הוא L אזי:
  - נעבור על הקטע $2i-1$ פעמים ולכן המרחק הכולל הוא $(2i-1)\cdot L$.
  - בזבזנו בסך הכל ליטר אחד של מים, הגמל הולך יחידת מרחק אחת עבור כל ליטר ולכן המרחק הכולל שעברנו $1 \geq$.

$$\underbrace{(2i-1)}_{\#\text{traverses}} \cdot L = \text{total\_distance} = \#\text{wasted\_water} \leq 1$$

$$\Longrightarrow L \leq \frac{1}{2i-1}$$

לבסוף, נחשב את המרחק שעברנו ממצב של n ליטרים ועד למצב של 0 ליטרים:

$$\text{dist}\left(n,0\right) = \sum_{i=1}^{n} (i,i-1)\text{dist} \leq \sum_{i=1}^{n} \frac{1}{2i-1} = \sum_{i=1}^{n} \left(\frac{1}{2i-1} + \frac{1}{2i} - \frac{1}{2i}\right)$$

$$= \sum_{i=1}^{2n} \frac{1}{i} + \frac{1}{2}\sum_{j=1}^{n} \frac{1}{j} = \ln\left(2n\right) + c - \frac{1}{2}\left(\ln\left(n\right) + c\right) = \mathcal{O}\left(\log n\right)$$

1.2 בעיית הזיווג היציב

"המטרה היא לא לגרום לאושר מקסימלי - אלא לסבל מינימלי"

נתונה רשימת בנים ורשימת בנות, לכל בן רשימת בנות ולכל בת רשימת בנים.

- המטרה: מציאת זיווג (n= זוגות) בנים-בנות יציב - **ללא בגידות**. יש בגידה כאשר שני בני זוג מזוגות שונים (ומינים שונים) מעדיפים אחד את השני מאשר הזיווג הנוכחי שלהם.

בנים:

1	2	3	4
a	b	b	b
b	a	a	a
c	c	c	d
d	d	d	c

בנות:

a	b	c	d
1	1	1	4
2	2	3	2
3	4	2	1
4	3	4	3

- האלגוריתם:
  - נסתכל תמיד על השורה הראשונה של הבנים
    * אם אין תחרות, (כלומר, לכל בן העדפה אחרת) סיימנו.
    * אם יש תחרות, נמחק את הבת מהשורה של הבן הפחות מועדף.

הוכחה. הוכחת האלגוריתם:

- קיבלנו זיווג - לכל בן מתאימה בת. האם ייתכן שבן ייזרק ולא תימצא לו בת?
  - שמורה - invariant: ברגע שבת קיבלה הצעה - היא נשארת איתה: קבוצת הבנות שקיבלו הצעה לא יכולה לקטון.
  - אם בן נזרק הוא בעצמו הציע לכל בת והפסיד.
  - לכן, יש רק $n-1$ בנים שהציעו ולעומת זאת n בנות - סתירה!
- נראה כי הזיווג יציב.
  - עבור כל בן יש שני סוגי בנות שהוא לא נמצא איתן - תחתונות ועליונות.
  - הבן לא ירצה לבגוד עם הבנות התחתונות.
  - הבן כן ירצה לבגוד עם הבנות העליונות, אך לפי שמורה נוספת - ההצעות שבנות מקבלות הולכות ומשתפרות, הבנות העליונות נמצאות עם מישהו שהן מעדיפות על הבן הנוכחי, והן לא ירצו לבגוד איתו.
- כלומר, אין בגידות והזיווג יציב.

□

סיבוכיות הזמן היא $\mathcal{O}\left(n^2\right)$, לינארי לגודל הקלט (גודל הטבלאות הוא $2n^2$).

2 חמדנות

אלגוריתם חמדן הוא אלגוריתם המבצע בכל שלב את הפעולה שנראית הכי טובה בשלב הנוכחי.

- מגיע ישירות לפתרון האופטימלי.
- קלים ליישום.
- נדיר שעובד.

2.1 הרכבת מספר ממטבעות

בעיה: בהינתן n, המטרה להרכיב את n מכמה שפחות מטבעות ($1, 2, 5, 10$)

- הגישה החמדנית: בכל פעם השתמש במטבע הגדול ביותר.
  - עבור סטים אחרים של מטבעות האלגוריתם החמדן לא בהכרח יהיה אופטימלי.

הוכחה. נוכיח כי האלגוריתם אופטימלי.

- בפתרון האופטימלי אין יותר ממטבע אחד של 5,1 ושני מטבעות 2.
- לכן, בכל המספרים מ-10 ומעלה 10 מופיע, לפי האלגוריתם נבחר ב-10 עד שלא נוכל.
- בנוסף, ניתן להביע בעזרת המכסה של המטבעות האחרים כל מספר בין 1-9.

□

הגדרה. קבוצת מטבעות היא טובה אם היא תומכת בבחירה חמדנית.

דוגמה. $\{1, 2, 5, 10\}$ היא טובה.
..... הוכחה מגעילה אי אפשר לקפוצת $1 - 193 + 104....1$

איך מוצאים פתרון אופטימלי לכל קבוצת מטבעות? (לאו דווקא חמדני) תכנות דינמי.

2.2 מיון בועות

נתונה סדרת מספרים. פעולה מותרת - החלפה בין זוג איברים עוקבים. המטרה: למיין את המספרים.

- מהו מספר הפעולות הדרושות המינימלי?

הגדרה. נקרא לאלגוריתם סביר אם בכל שלב הוא מחליף בין גדול וקטן.

- כלומר, כל החלפות שבוצעו בו הן בכיוון הנכון בדרך למיון: $8, -4 \to -4, 8$.

טענה. כל אלגוריתם סביר הוא אופטימלי.

הוכחה. נגדיר פונקציית פוטנציאל $\Phi(a_1, \ldots, a_n) = \#\{i < j | a_i > a_j\}$ (מספר אי הסדרים בסדרה)

- בכל מעבר פונקציית הפוטנציאל מספר אי הסדרים יכול לקטון או לגדול ב-1.
- אלגוריתם סביר יוריד בכל פעם ב-1 את פונקציית הפוטנציאל.
- לכן, מספר ההחלפות המינימלי הוא כגודל הפוטנציאל ההתחלתי.

□

- בנוסף, ניתן לבצע "סימולציה" לפתרון האופטימלי, להתקזז עם הפתרון האופטימלי ולהיות שקול אליו.

2.3 שיבוץ הרצאות בכיתה

ישנה כיתה אחת ורשימת הרצאות $\left\{\left[\underbrace{s_i}_{\text{start}}, \underbrace{f_i}_{\text{end}}\right)\right\}$. מהו המספר המקסימלי של הרצאות שניתן לקיים (ללא חיתוך בין הרצאה והרצאה)? אבחנות:

1. בה"כ ההרצאה בעלת זמן הסיום המינימלי נמצאת בפתרון האופטימלי (במילים אחרות, קיים פתרון מינימלי בו נמצאת ההרצאה). הוכחה: ניקח פתרון אופטימלי ונחליף את ההרצאה הראשונה בהרצאה עם זמן הסיום המינימלי - $l1$. הפתרון יעמוד בתנאים (אין התנגשות הרצאות) ויש בו אותה כמות הרצאות ולכן אופטימלי.

איור 2: מלמעלה למטה: פתרון אופטימלי, הפתרון לאחר התיקון

באופן כללי כשמוכיחים אבחנה - לוקחים פתרון אופטימלי ו"מתקנים" אותו כך שיתאים לאבחנה

2. אנחנו יכולים לזרוק את כל ההרצאות שמתנגשות עם $l1$ כי הן לא יכולות להיות בפתרון שלנו - אם נזרוק הרצאות אלה נקבל את הבעיה המקורית על קבוצה קטנה יותר.

<u>האלגוריתם:</u>

1. $A = \{\}$.
2. מיין את הקטעים לפי f_i.
3. סמן $t = 0$.
4. לכל $i > 0$:
   (א) אם $s_i \geq t$:
      i. $A = A \cup \{f_i\}$.
      ii. $t = f_i$.
5. החזר את A.

סיבוכיות הזמן של האלגוריתם היא $\mathcal{O}\left(n \log n\right)$.

- מה קורה יש אם שתי כיתות?
  - בחירת תת קבוצה מקסימלית של קטעים בעובי $2 \geq$ (לכל היותר חיתוך של 2 קטעים).
  - סף הזריקה נוצר רק כאשר יש כבר עובי 2.

2.4 מיון חלקים של בחורה

נתונה מטריצה המכילה פרמוטציה של המספרים $1,\ldots,n$. המטרה - למיין אותה (שמאל - המצב ההתחלתי, ימין - הממוין):

3	6	9	1	2	3
2	7	8	4	5	6
5	4	1	7	8	9

- הפעולה החוקית: להחליף בין שני תאים כלשהם. מהו מספר הצעדים המינימלי למיין את המטריצה?
- נייצג את המטריצה באמצעות קבוצה של מעגלים.
  - כל תא מצביע לתא באינדקס של הערך שנמצא בו כרגע. הקבוצות ההתחלתיות במקרה זה:

$$\{3,9,1\}, \{6,8,4,2\}, \{7,5\}$$

- נגדיר את פונקציית הפוטנציאל להיות מספר המעגלים.
- כל פעולה מורידה/מגדילה מספר מעגלים ב-1:
  - אם ההחלפה בתוך המעגל הוא מתפצל (והפוטנציאל גדל ב-1).
  - אם ההחלפה משני מעגלים שונים הם יתאחדו למעגל יותר גדול (והפוטנציאל קטן ב-1).
- כאשר נגיע ל-n מעגלים שונים כל איבר במקומו.

הוכחה. נוכיח כי כל פתרון שמחליף רק בתוך מעגל (ולא בין שניים שונים) הוא אופטימלי.
אחרת - באסה. לכן הפתרון אופטימלי (הפוטנציאל קטן בכל פעם). □

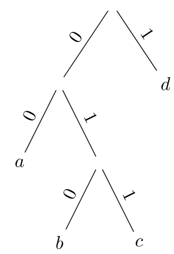
2.5 בעיית k-paging

נתונה סדרת בקשות $a_1,\ldots,a_n; a_i\in K\wedge|K|\le n$ ו-cache בגודל k, אנחנו רוצים לבצע מספר מינימלי של פעולות (הוצאה והכנסה מה-cache) כדי לענות על כולן.

- מקבלים את השאילתות online - ננסה חמדני!
- בייסקלי LRU, הוכחת נכונות פשוטה עם שיקול החלפה.

2.6 עצי האפמן

- נתון טקסט ארוך ונתונה שכיחות של הופעות (מערך מונים של האותיות בטקסט).
  - אילוץ: פענוח יחיד עבור כל צירוף אותיות.
  - אופטימיזציה: אורך הטקסט המקודד מינימלי.
- קוד עץ הוא קוד עם עץ בו כל עלה הוא אות. כאשר מפענחים קוד, מפענחים אות אחר אות ע"י מעקב אחרי המסלולים בעץ.
- נרצה לעבוד עם קודים חסרי רישא - בהם אף מילה היא לא רישא של מילה אחרת.
- המטרה: לבנות עץ בינארי (לא בהכרח מלא) המקיים:
  - העלים נמצאים בהתאמה חח"ע לקלט.
  - פונקציית המטרה היא $\sum_{\sigma\in\Sigma}\underbrace{h(\sigma)}_{\text{עומק}}\cdot\underbrace{f(\sigma)}_{\text{שכיחות}}$.



איור 3: דוגמא לקוד עץ: האותיות בעלים

- העץ מאפשר פענוח קוד בצורה יחידה.

הוכחה. נוכיח כי הפענוח יחיד.

- נניח בשלילה שקיים קוד שניתן לפענח בשתי דרכים $a_1,\dots,a_n$.
- אזי קיימים $i<j$ כך ש-$a_1\dots,a_i$ מפוענח ל-x_1 ו-$a_1,\dots,a_j$ מפוענח ל-x_2, ולכן x_1,x_2 הם עלים.
- אבל, אם נבנה את העץ נקבל ש-x_2 הוא צאצא של x_1, בסתירה לכך ש-x_1 הוא עלה.

- נחפש את הפתרון האופטימלי, כלומר עץ עבורו פונקציית המטרה מינימלית.

<u>אבחנות:</u>

1. העץ האופטימלי הוא בינארי שלם (לכל צומת פנימית בעץ הדרגה היא 3).

(א) הוכחה: אם יש לנו צומת פנימי שדרגתו 2, אפשר להוציא אותו מהעץ ולהחליף אותו בצאצא שלו, מכיוון שהוא אינו עלה הוא לא מייצג אות ולכן נכונות העץ לא נפגעת. בנוסף העומק של כל תת העץ של הצומת הפנימי מתקצר ולכן הפתרון טוב יותר.

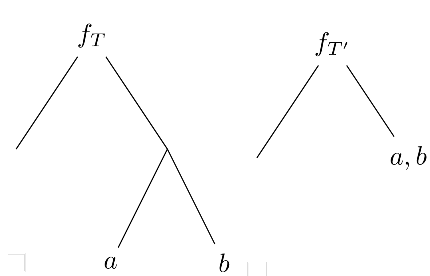
2. בה"כ בעץ האופטימלי שתי האותיות בעלות תדירות מינימלית הן אחיות.

הוכחה. אם אות בעל תדירות יותר גבוהה מאות אחרת תהיה יותר עמוקה מהאות האחרת, נוכל להחליף ביניהם ולקבל שיפור (על פי הנוסחא של פונקציית המטרה). לכן האות בעלת התדירות המינימלית היא בעלת עומק גדול או שווה לשל שאר האותיות. העץ הוא בינארי שלם ולכן חייב להיות לפחות עוד עלה אחד בעל עומק שווה לשל האות עם התדירות המינימלית (לכל עלה יש אח), בה"כ זו תהיה האות השנייה.

3. נניח שידוע שאותיות i,j הן אחיות בפתרון האופטימלי.

$$a_1,\dots,\underbrace{a_i}_{f_i},\underbrace{a_j}_{f_j},\dots,a_n \Longrightarrow a_1,\dots,\underbrace{A_{ij}}_{f_i+f_j},\dots a_n$$

אזי, אם נמצא פתרון אופטימלי למצב השני, ונצמיח ל-A_{ij} שני ילדים נקבל פתרון אופטימלי למצב הראשון.



איור 4: $f_T = f_{T'} + (f(a)+f(b))$

4. ההפרש בין שני העצים הוא קבוע ולכן האופטימליות נשמרת. לכן, המצב הראשון אופטימלי אמ"מ המצב השני אופטימלי וגם המצב הראשון אופטימלי.

5. מאבחנות 2 ו-3 נקבל שאם $f_{T'}$ הוא עץ בו 2 האותיות הכי פחות שכיחות מאוחדות, אז אמ"מ f_T אופטימלי ו-$f_{T'}$ אופטימלי.

נשתמש בחמדנות. <u>האלגוריתם:</u>

- בכל פעם נבחר את שתי האותיות עם התדירות המינימלית, נאחד אותן לתת עץ בו הן אחיות ונריץ את האלגוריתם עם שאר האותיות וצומת האב שלהן (כאשר התדירות שלו היא סכום התדירויות של שתי האותיות).
- נשתמש במבנה נתונים לשמירת האותיות, נרצה שמבנה הנתונים יאפשר הוצאת מינימום והכנסה לכן ערימה נותנת מענה טוב.
- סיבוכיות הזמן היא $\mathcal{O}\left(n\log n\right)$ (ניתן להשתמש בעץ חיפוש בינארי או בערימת מינימום - דרושות הפעולות extract_min, insert).

דוגמא לקידוד חח"ע שאינו חסר רישא: $0\cdot 1^i$.

- נסמן ב-+ איחוד עם חזרות (multiset) וב-· שרשור, אזי מכיוון שהקוד ניתן לפענוח יחיד מתקיים: $\left(w_1+\ldots+w_n\right)^k\subseteq$ $\left(0+1\right)^1+\cdots+\left(0+1\right)^{rk}$; $r=\max_i\left|w_i\right|$. בנוסף, מכאן נובע שכל מחרוזת ב-$\left(w_1+\cdots+w_n\right)^k$ מופיעה פעם בודדת - כדי שההכלה תתקיים.
- לכל w נתאים $f\left(w\right)=\left(\frac{1}{2}\right)^{|w|}$. אזי מתקיים:

$$\left[\left(\frac{1}{2}\right)^{|w_1|}+\cdots+\left(\frac{1}{2}\right)^{|w_n|}\right]^k\leq\underbrace{1+1+\cdots+1}_{\text{פעמים } rk}$$

$$f\left(w\cdot u\right)=\frac{1}{2^{|w|+|u|}}=\frac{1}{2^{|w|}}\cdot\frac{1}{2^{|u|}}=f\left(w\right)\cdot f\left(u\right)$$

$$f\left(w+u\right)=f\left(w\right)+f\left(u\right)$$

- נותר להוכיח כי $\sum_i\left(\frac{1}{2}\right)^{|w_i|}\leq 1$.

$$\left[\sum_i 2^{-|w_i|}\right]^k\ \forall k: rk\geq\left[\sum_i 2^{-|w_i|}\right]^k\Longrightarrow\sum_i 2^{-|w_i|}\leq\sqrt[k]{rk}\xrightarrow[k\to\infty]{}1$$

משפט. עבור סדרה $\left\{l_i\right\}_{i=1}^n$ של מספרים טבעיים, כך ש-$\sum_i 2^{-l_i}\leq 1$ קיים עץ בינארי עם n עלים כך שעומק העלה ה-i הוא l_i.

ולכן, ניתן להתאים כל w_i ל-w_i' חסר רישא, המקיים $|w_i|=|w_i'|$.

הוכחה. באופן כמעט זהה להוכחה של עץ הופמן. אם הסכום שווה ל-1 לא בעיה. אחרת נוכל להשלים את הסכום ל-1, לבנות את העץ ולזרוק את התוספת לאחר מכן. □

2.7 מציאת חציון

נתונים n ערכים שונים, המטרה: למצוא את החציון.

- מותר לבצע השוואה בין איברים בלבד.
- מיון הוא יקר מדי.

באופן כללי - האלגוריתם מוצא את האיבר במקום ה-i במערך הממוין, נדגים אותו על חציון.

- נרצה לבצע ניחוש מושכל לחציון ובכך לצמצם את מספר הפעולות שאנחנו מבצעים.

- נחלק את האיברים לחמישיות (תתי מערכים בגודל 5). נמצא לכל חמישיה את החציון שלה ולאחר מכן נמצא את חציון החציונים:

$$[a_1 - a_5, a_6 - a_{10}, a_{11} - a_{15}, a_{16} - a_{20}, a_{21} - a_{25}]$$

$$\downarrow$$

$$\begin{bmatrix} \text{median}_5 & \text{median}_4 & \text{median}_3 & \text{median}_2 & \text{median}_1 \end{bmatrix}$$

$$\le \text{median}, \ge \text{median}, \text{median}_3 = \text{med}\left(\text{median}_i\right)$$

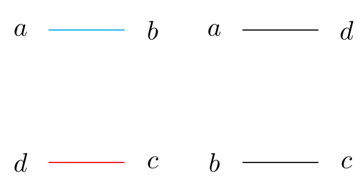
- לאחר מכן נבצע את ההפרדה לפי ה-pivot ונמשיך לחפש בחלק המתאים.
- כך צמצמנו את כמות הערכים שנבדוק ב-$\frac{3n}{10}$
- סיבוכיות הזמן:
  - מציאת חציון של כל חמישיה - $\mathcal{O}(n)$
  - מציאת חציון החציונים - $f\left(\frac{n}{5}\right)$
  - מציאת חציון למספרים שנותרו - $f\left(\frac{7n}{10}\right)$

$$f(n) \le f\left(\frac{n}{5}\right) + f\left(\frac{7n}{10}\right) + \mathcal{O}(n) \underbrace{\le}_{\frac{n}{5}+\frac{7n}{10}\le n} \mathcal{O}(n)$$

2.8 בניית גרף לפי דרגות

הבעיה: מקבלים n מספרים שמייצגים את הדרגות של הצמתים בגרף וצריך לבנות את הגרף.

אבחנה: קיים פתרון בו הקודקוד מהדרגה המקסימלית מחובר לשאר הצמתים בעלי הדרגות המקסימליות. הוכחה:



איור 5: a - הצומת המקסימלי, b - צומת שאנו רוצים לחבר, c, d - שני צמתים אחרים. התיקון מימין לשמאל: נחליף את הקשתות ונקבל פתרון אופטימלי נוסף (סכום הדרגות של כל צומת לא השתנה)

נפתור על ידי חמדנות. האלגוריתם:

- מספר הדרגות שלם וניתן למצוא את הדרגה המקסימלית ב-$\mathcal{O}(n)$, נשתמש ב-bucket sort ונמיין את הצמתים לפי הדרגות.
- בכל פעם ניקח את הצומת עם הדרגה המקסימלית.
  - נחבר אותו לצמתים המקסימליים.
  - נוריד לצמתים המקסימליים את הדרגה ב-1.
  - נוציא את הצומת המקסימלי.
- חיבור כל צומת למספר הצמתים האחרים תלוי ב-$d_{\max}$ ולכן סיבוכיות הזמן היא $\mathcal{O}(n \cdot d_{\max})$.

2.9 ביצוע רשימת מטלות

- נתונה רשימת מטלות $a_i = (f_i, t_i)$. f_i זה הדדליין ו-t_i זה הזמן הדרוש לביצוע המשימה.
- המטרה: לסיים את כל המטלות באיחור מינימלי.
- אילוצים: זמן התחלה $t = 0$, אין להפסיק ביצוע מטלה מרגע התחלתה.
- אבחנה: אין חורים (טריוויאלי).

<u>האלגוריתם:</u>

- נמיין את הרשימה לפי f_i.

הוכחה. נוכיח את נכונות האלגוריתם. □

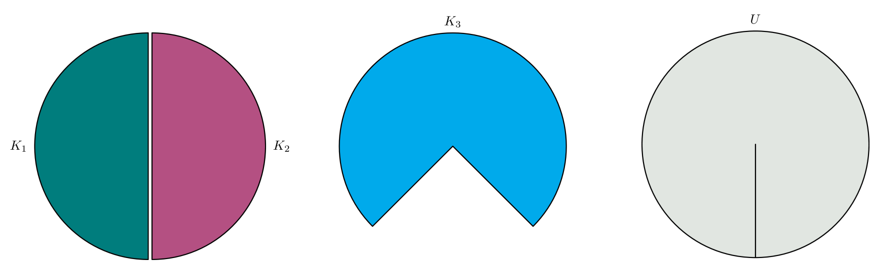
- נניח כי יש לנו פתרון אופטימלי בו המשימות לא ממויינות לפי f_i - קיימים i, j כך ש- $f_i < f_j$ וגם a_j נמצא לפני a_i.
- אזי בוודאות כדאי להחליף בין a_i, a_j: ההחלפה לא תפגע בנכונות הפתרון ובאופטימליות שלו, ובנוסף אם יש שתי משימות שאנחנו לא יכולים לבצע בה"כ נוכל לבצע את זו שמסתיימת מוקדם יותר.

2.10 כיסוי קבוצה על ידי תתי קבוצות

נתון עולם U ובו תת קבוצות $k_1, \ldots, k_n$, כאשר $U = \bigcup_{1 \le i \le n} k_i$. נרצה לכסות את U בעזרת מספר מינימלי של קבוצות. הצעה אפשרית:

- כל עוד $U \neq \emptyset$:
  - בחר את ה-k_i הגדול ביותר.
  - $U \leftarrow U \setminus k_i$.
  - $\forall j : k_j \leftarrow k_j \setminus k_i$.

הפתרון אינו אופטימלי! דוגמא נגדית:



איור 6: ע"פ האלגוריתם צריך 3 קבוצות, הפתרון האופטימלי מכיל את שתי הקבוצות הקטנות

משפט. $|\text{opt}| \ln(n) \ge |A| \ge |\text{opt}|$
נסמן: $x = |\text{opt}|$.

$$|U_1| = n$$

$$|U_2| = n - \max|k_i| \le n - \underbrace{\frac{n}{x}}_{\text{גודל ממוצע של קבוצה ב-}OPT}$$

$$|U_3| \le |U_2| \left(1 - \frac{1}{x}\right) \le n \left(1 - \frac{1}{x}\right)^2$$

האלגוריתם מפסיק כאשר $n\left(1-\frac{1}{x}\right)^r<2$, כלומר כאשר $r=x\ln n$:

$$n\left(1-\frac{1}{x}\right)^{x\ln n}<e$$

ולכן המשפט מתקיים.

3 הפרד ומשול

3.1 הכפלת מספרים בינאריים

נתונים שני מספרים בינאריים A,B באורך n.

- חיבורם יעלה $\mathcal{O}(n)$.
- כיצד נחשב את המכפלה?
  - כפל במאונך: $\mathcal{O}\left(n^2\right)$.
  - נפריד ונמשול.

$$A=\underbrace{1011}_{A_1}\underbrace{0011}_{A_2},B=\underbrace{1001}_{B_1}\underbrace{1101}_{B_2}$$

$$\Longrightarrow A\cdot B=\left(A_1+A_2\cdot 2^{\frac{n}{2}}\right)\left(B_1+B_2\cdot 2^{\frac{n}{2}}\right)=A_1B_1+2^{\frac{n}{2}}\left(A_1B_2+A_2B_1\right)+A_2B_2\cdot 2^n$$

- סיבוכיות הזמן היא $f(n)=4f\left(\frac{n}{2}\right)+O(n)=O\left(n^2\right)$.

$$(a+c)(b+d)=ab+ad+bc+bd$$

  - על ידי פתרון מתוכחם נוכל להקטין את הסיבוכיות - חישובי עזר:

$$(1)\ \ A_2B_2,(2)\ \ A_1B_1,(3)\ \ (A_1+A_2)(B_1+B_2)$$

$$\Longrightarrow(3)-(1)-(2)=A_1B_2+A_2B_1$$

  - כלומר, לא באמת דרוש החישוב של $A_1B_2+A_2B_1$:

$$\Longrightarrow f(n)\leq 3\cdot f\left(\frac{n}{2}\right)+\mathcal{O}(n)=\mathcal{O}\left(n^{\log_2 3}\right)$$

3.2 מרחק מינימלי בין נקודות במישור

"אם אתה שותה כוס תה חם לפני שאתה קופץ ממטוס בלי מצנח זה לא עוזר בכלל"

נתונות n נקודות במישור. במטרה: למצוא זוג הקרוב ביותר.
פתרון - הפרד ומשול - נפריד באמצעות קו באמצע ונמשול.

- צד אחד Δ_1, השני Δ_2. $\Delta=\min\left(\Delta_1,\Delta_2\right)$.
- נותר לבדוק את הזוגות שבהן כל אחת מהנקודות בצד שונה.
  - יש משמעות לבדיקת הנקודות שנמצאות במרחק אופקי לכל היותר Δ בלבד.
  - בריבוע $\Delta\times\Delta$ ניתן להכניס לכל היותר 4 נקודות שהמרחק ביניהן הוא לכל הפחות Δ.
  - לכן, בשני הצדדים ביחד במלבן $2\Delta\times\Delta$ ישבו לכל היותר 8 (6) נקודות.

טענה. אם הנקודות יושבות לפי סדר y יורד אז מספיק להשוות את p_i ל-$p_{i+1},\ldots,p_{i+7}$.

הוכחה. נבדוק את הנקודות מלמעלה למטה, נותר לבדוק רק את הנקודות מתחת ל-p_i. המועמדות היחידות להיות בתוך המלבן שצלעו העליונה על p_i, הן 7 הנקודות הקרובות אליו ביותר מלמטה בלבד, לפי האבחנות שלעיל. שאר הנקודות בוודאות לא יהיו במלבן ולכן אין טעם לבדוק אותן. □

- סיבוכיות הזמן: נעשה preprocessing של מיון רשימות מקושרות. $p_i = (x_i, y_i)$, רשימה מקושרת ממוינת של x_i ושל y_i בנפרד, וכל x_i מקושר עם y_i.

$$x_1 \to x_2 \to \cdots \to x_n \to$$
$$y_{10} \to y_{11} \to \cdots \to y_1 \to$$

- כך ההכנה בכל קריאה רקורסיבית תעלה $\mathcal{O}(n)$, ועלות עיבוד מקדים $\mathcal{O}(n\log n)$. בסך הכל נקבל:

$$f(n) = 2f\left(\frac{n}{2}\right) + \mathcal{O}(n);\ +\mathcal{O}(n\log n)$$

$$\Longrightarrow f(n) = \mathcal{O}(n\log n)$$

3.3 האלגוריתם של Strassen לכפל מטריצות

בעיה: יש שתי מטריצות ריבועיות A, B שאנחנו רוצים להכפיל ($C = A \times B$). נניח שגודלן הוא חזקה שלמה של 2 (מכיוון שתמיד ניתן להגדיל את המטריצה על ידי הוספת אפסים). עלות הכפלה של מטריצות היא מאוד גדולה, נרצה להקטין את העלות.

- הרחבה של האלגוריתם להכפלת מספרים בינאריים.
- נחזור לאלגוריתם כפל המטריצות, נשים לב כי אם נסמן:

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix}, B = \begin{bmatrix} B_{1,1} & B_{1,2} \\ B_{2,1} & B_{2,2} \end{bmatrix}, C = \begin{bmatrix} C_{1,1} & C_{1,2} \\ C_{2,1} & C_{2,2} \end{bmatrix}$$

  - כלומר, נחלק כל מטריצה ל-4 תתי מטריצות שוות בגודלן, אזי יתקיים:

$$\begin{cases} C_{1,1} = A_{1,1}B_{1,1} + A_{1,2}B_{2,1} \\ C_{1,2} = A_{1,1}B_{1,2} + A_{1,2}B_{2,2} \\ C_{2,1} = A_{2,1}B_{1,1} + A_{2,2}B_{2,1} \\ C_{2,2} = A_{2,1}B_{1,2} + A_{2,2}B_{2,2} \end{cases}$$

  - במקרה זה עלינו לעשות 8 הכפלות, סיבוכיות הזמן היא $f(n) = 8f\left(\frac{n}{2}\right) + \mathcal{O}\left(n^2\right) = \mathcal{O}\left(n^3\right)$ (8 תתי בעיות, כל אחת על גודל חצוי ועלות סכימת המכפלות היא כגודל המטריצות)
  - מספר תתי הבעיות מהווה גורם משמעותי על סיבוכיות הזמן - ננסה להקטין אותו
  - מאגיה שחורה - ניצור מטריצות עזר $M_1 - M_7$:

$$\begin{cases} M_1 := (A_{1,1} + A_{2,2})(B_{1,1} + B_{2,2}) \\ M_2 := (A_{2,1} + A_{2,2})B_{1,1} \\ M_3 := A_{1,1}(B_{1,2} - B_{2,2}) \\ M_4 := A_{2,2}(B_{2,1} - B_{1,1}) \\ M_5 := (A_{1,1} + A_{1,2})B_{2,2} \\ M_6 := (A_{2,1} - A_{1,1})(B_{1,1} + B_{1,2}) \\ M_7 := (A_{1,2} - A_{2,2})(B_{2,1} + B_{2,2}) \end{cases} \Longrightarrow \begin{cases} C_{1,1} = M_1 + M_4 - M_5 + M_7 \\ C_{1,2} = M_3 + M_5 \\ C_{2,1} = M_2 + M_4 \\ C_{2,2} = M_1 - M_2 + M_3 + M_6 \end{cases}$$

  - כך מספר הבעיות קטן ל-7 והסיבוכיות השתפרה - $\mathcal{O}\left([7 + o(1)]^n\right) = \mathcal{O}\left(n^{\log 7 + o(1)}\right) \approx \mathcal{O}\left(n^{2.8074}\right)$.

3.4 מציאת קמור של אוסף נקודות

נותר למצוא את שני הדברים הכחולים האלה (קו עליון וקו תחתון בשביל איחוד הקמורים.

- נרדמתי בהרצאה, בכל פעם תפריד ואז אחר כך תמשול.

4 תכנות דינמי

בהינתן בעיה P, נפתור אותה באמצעות יצירת תת בעיות $P_1, \ldots, P_k$ כאשר k לכל היותר פולינומיאלי, כך ש:

1. יש סדר חלקי בין תתי הבעיות P_i.
2. כל תת בעיה היא קלה (נפתרת בזמן ריצה פולינומיאלי) בהינתן פתרונות הבעיות שקדמו לה.
3. אם נפתור את כל תתי הבעיות אז P תפתר.

4.1 תת סדרת עולה מקסימלית LIS

נתון מערך באורך n, יש למצוא את תת הסדרה העולה המקסימלית.

- נחלק לתתי בעיות - P_i היא תת הסדרה המקסימלית המסתיימת ב-S_i
- סדר: אם $j > i$ אז P_j קודם ל-P_i
- פתרון P_i:
  - P_i הוא תת הסדרה העולה המקסימלית מבין תתי הבעיות שקדמו לו, בנוסף ל-a_i
- פתרון P: תת הסדרה המקסימלית בהכרח מסתיימת באחד המספרים במערך, נעבור על כל P_i ונבחר את המקסימלי
- סיבוכיות הזמן: כל תת בעיה לוקחת $\mathcal{O}(n)$ זמן, יש n תתי בעיות ובסך הכל $\mathcal{O}\left(n^2\right)$

<u>שיפור:</u>

- אפשר לשמור לכל ערך של תת סדרה רק את a_i המינימלי מבין אלו ש-P_i שלהם שווה (בה"כ תמיד נבחר אותו בתתי סדרות ארוכות יותר)
- אבחנה: כל ה-a_i המינימליים הם בעצם סדרה עולה
- במצב זה ניתן לשמור פחות נתונים:
  - לכל P_i נסתכל במערך ה-a_i המינימליים עד כה ונמצא את המיקום שלו (חיפוש בינארי $\mathcal{O}(\log n)$)
  - נעדכן את P_i בהתאם למיקום.
  - במידה ו-a_i קטן מהאיבר שנמצא במקום ה-i נשים את האיבר במקום ה-i להיות a_i
- בסך הכל, סיבוכיות הזמן היא $\mathcal{O}(n \log n)$.

4.2 פענוח בינארי

נתון רצף T וקוד e. למשל, $T = 01101011\ldots, e = \{011, 1011, 0001, \ldots\}$. האם T ניתן לפענוח על ידי e?

- נחלק לתתי בעיות: P_i - האם הרישא שנגמרת בתו ה-i ניתנת לפענוח?
- סדר: אם $j > i$ אז P_j קודם ל-P_i.
- פתרון P_i:
  - P_0: בוודאות כן.
  - $P_i, i > 0$: נעבור על האיברים ב-e - אם קיימת תת מחרוזת מהתווים במקומות $i, \ldots, k$ ב-e וגם התשובה ל-P_k היא כן אז התשובה ל-P_i היא גם כן, אחרת לא.
    - דרך יעילה לפתרון היא על ידי בניית עץ סיפות של e.
- פתרון P: התשובה של P_n.
- סיבוכיות הזמן: בכל תת בעיה זמן פתרון תת הבעיות יהיה לכל היותר גובה העץ, יש n תתי בעיות ולכן: $\mathcal{O}(n \cdot h) = \mathcal{O}\left(n \cdot \max_{s \in e}\{|s|\}\right)$.

4.3 שיבוץ הרצאות בכיתה עם עלות מקסימלית

עלינו לשבץ הרצאות בכיתה כאשר לכל הרצאה זמן התחלה, זמן סיום ועלות. המטרה: למצוא את הסידור עם העלות המקסימלית. נפתור בתכנות דינמי:

- נחלק לתתי בעיות - P_k זה הפתרון האופטימלי בקטע $(-\infty, t_k]$, כאשר t_k זה זמן הסיום של הרצאה כלשהי, ו-$P_0 = 0$.
- יחס סדר: נמיין את הבעיות לפי זמני הסיום.
- פתרון תת בעיה: נבדוק מה משתלם יותר, לכלול את ההרצאה ה-k או לא:
  - נשווה בין הפתרון של תת הבעיה הקודמת (שלא כולל את ההרצאה ה-k) ובין עלות ההרצאה ה-k בתוספת לעלות המקסימלית של ההרצאות שמסתיימות לפני זמן ההתחלה של ההרצאה ה-k:

$$P_k = \max \{P_{k-1}, w_k + P_{s_k}\}$$

- פתרון הבעיה כולה הוא התשובה של P_n.
- סיבוכיות הזמן: מיון ההרצאות ומעבר עליהן בזמן לינארי, בסך הכל $\mathcal{O}(n \log n)$.

4.4 תת סדרה בעלת משקל מקסימלי

נתונה סדרת מספרים. הבעיה: מציאת תת סדרה רציפה בעלת משקל מקסימלי. נפתור בתכנות דינמי.

- יחס סדר: לפי האינדקסים בסדר עולה.
- פתרון P_i - תת סדרה רציפה בעלת משקל מקסימלי המסתיימת ב-i:

$$P_i = a_i + \max \{P_{i-1}, 0\}$$

- פתרון P: $P = \max P_i$.
- סיבוכיות הזמן היא $\mathcal{O}(n)$.

4.5 תת קבוצה בלתי תלויה כבדה בעץ

המטרה: למצוא קבוצת צמתים בלתי תלויה בעלת משקל מקסימלי.

1. נבחר צומת אקראי, נהפוך אותה לשורש ונבנה עץ מכוון.
2. ניצור יחס סדרי חלקי: $v_i < v_j$ אם יש מסלול מ-v_i ל-v_j: נמספר את הצמתים מ-1 עד n כך שאם $v_i > v_j$ אז $i > j$.

- P_i - משקל הקבוצה הבלתי המקסימלית עם או בלי i (נסמן עם ב- $P_{1,i}$ ובלי ב-$P_{2,i}$).
- עבור עלה: $P_{1,i} = v_i, P_{2,i} = 0$.
- עבור צומת פנימי:

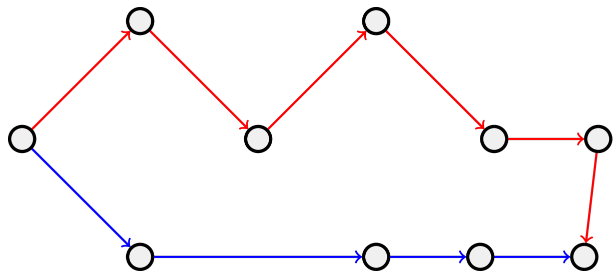
$$P_{1,i} = v_i + \sum_{(v_i,v_j)\in E} P_{2,j}$$

$$P_{2,i} = \sum_{(v_i,v_j)\in E} \max (P_{1,j}, P_{2,j})$$

- סיבוכיות הזמן היא $\mathcal{O}(n)$.

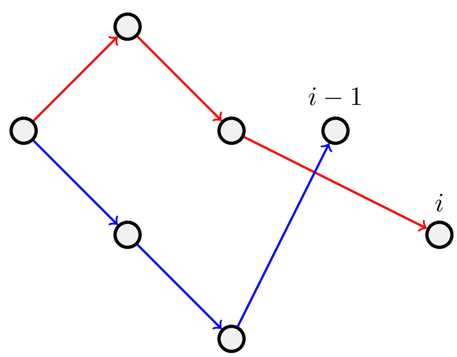
4.6 מעגל קצר בגרף

נרצה ליצור מעגל העובר דרך כל הקודקודים בעל מרחק מינימלי. לנוחות, נפריד אותו לשני תתי מעגלים מכוונים (כל חץ מכוון ימינה)



איור 7: הפרדת המעגל לאדום וכחול, כל הקשתות מכוונות ימינה

- אבחנה: אם נחבר 2 נקודות במסלול אז קיימת דרך יחידה לחבר את כל הנקודות ביניהן.
  - הן יהיו חייבות להתחבר זו לזו לפי ערכי ה-x שלהן (דמיינו מערכת צירים).
- נפתור באמצעות תכנות דינמי.
- סדר חלקי על פי ערך ה-x של כל צומת.
- תת בעיה P_i: כיצד לחבר את הצומת ה-i כך שיווצרו 2 מסלולים מינימליים, כאשר האחד נגמר בנקודה ה-i והשני נגמר בנקודה ה-$i-1$?



איור 8: דוגמא לתת הבעיה P_i

- פתרון הבעיה: ניצור שני עותקים של הצומת האחרון. כאשר נריץ את האלגוריתם שוב על הנקודה האחרונה, נקבל את שני המסלולים המינימליים שנגמרים בצומת ובצומת שאחריו - כלומר נגמרים שניהם בצומת האחרון, וקיבלנו את המעגל הדרוש.
- פתרון תת בעיה P_i:
  - $\mathrm{d}(i,j)$ הוא מרחק בין הצמתים i,j, נמצא לאיזה צומת מתחבר הצומת החדש שהוספנו
  - איך ייראה מסלול שמתחבר ב-$j-1$?
    * נחבר את i ל-$j-1$
    * הצמתים $i,\ldots,j-1$ יתחברו זה לזה לפי האבחנה באופן חח"ע
    * עלות המסלול היא גודל המסלולים עד j (חושב כבר) + סכום הקשתות החדשות שנוספו:

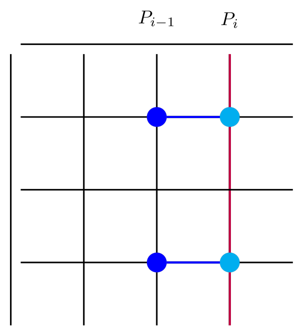
$$\begin{cases} M(1)=0 \\ M(2)=\mathrm{d}(i,j) \\ M(i)=\min_{j<i}\{M(j)+\mathrm{d}(i,j-1)+\mathrm{d}(j,j+1)+\ldots+\mathrm{d}(i-2,i-1)\} \end{cases}$$

    * נסמן $L_i=\sum_{k=2}^{i}\mathrm{d}(k-1,k)$ נקבל $M(i)=\min_{j<i}\{M(j)+L_{i-1}+L_j\}$
    * אם נחשב מראש את כל L_i החישוב הפנימי בתוך ה-min הוא $O(1)$
- ולכן, סיבוכיות הזמן בסך הכל היא $O\left(n^2\right)$

4.7 מסלול על רשת עם עלות מינימלית

תהי רשת בה לכל קשת יש משקל. המטרה: למצוא מסלול קל ביותר מ-A ל-B.

- כפי שהבעיה נראית כרגע, אם נעקוב אחרי המסלול הנוצר - הוא לא בהכרח הולך לאותו כיוון (אפילו יכול לחזור אחורה!), ולכן לא ניתן לפתור ישירות עם תכנות דינמי. ננסה לגרום למסלול להיות חד כיווני.



איור 9: תחילה נחשב את המרחק המינימלי בין נקודות באותה עמודה P_i. כאשר מחשבים את P_i, P_{i-1} כבר חושבה.

כעת נבצע תכנות דינמי למציאת המרחק המינימלי בין P_i ל-A.

4.8 מספר אי הסדרים בפרמוטציה

בהינתן פרמוטציה, מספר אי הסדרים בה הוא מספר הזוגות (לא בהכרח סמוכים) של איברים שמופיעים בסדר לא ממוין.

- מספר אי הסדרים הוא גם מספר ההחלפות של איברים סמוכים שיש לבצע כדי למיין את הפרמוטציה
  - כל החלפה מורידה במקרה הטוב אי סדר אחד, ותמיד יהיה אי סדר סמוך - אחרת לפי bubble sort הפרמוטציה ממוינת

נשתמש בעיקרון הפרד ומשול. בכל פעם נחלק את המספרים לשתי קבוצות שוות, נספור את אי הסדרים בתוכן ונמיין אותן. לאחר מכן נמזג אותן למערך אחד ובמהלך המיזוג נספור את מספר אי הסדרים בין שתי תתי הקבוצות (כלומר - נעשה merge sort ובמיזוג נסכום את אי הסדרים של כל חלק)

- נסמן ב-L את המערך השמאלי הממוין וב-R את המערך הימני הממוין. אזי:
  - $P_1 = 0$ - במערך איבר אחד ולכן הוא ממוין
  - כאשר $i > 1$
    - טרם המיזוג, $P_i = P_L + P_R$
    - תוך כדי המיזוג, בכל פעם שאיבר מ-R נכנס לפני איבר מ-L - P_i++

סיבוכיות הזמן היא $\mathcal{O}(n \log n)$.

4.9 מציאת מסלול המילטוני בגרף

מסלול המילטוני בגרף הוא מסלול בלתי מכוון העובר בכל צומת בדיוק פעם אחת.

- בעיה: בהינתן גרף $G = (V, E)$ עם n קודקודים, נרצה לענות על השאלה - האם קיים מסלול המילטוני בגרף?
- נפתור על ידי תכנות דינמי על תתי הקבוצות של V
- יחס הסדר החלקי: לפי גודל תתי הקבוצות (בסדר עולה)
- תת בעיה $P(S, i, j)$: האם קיים מסלול המילטוני המתחיל ב-i ונגמר ב-j ב-$G \mid S$?
  - בסיס: קבוצות בגודל 2 - אם $(i, j) \in E$ אז כן, אחרת - לא
  - כלל הנסיגה: נסתכל על כל שכן x של i, ונבדוק האם קיים מסלול על פי הפתרון לשאילתא $P(S \setminus \{i\}, x, j)$ (שאותו כבר חישבנו). אם עבור שכן כלשהו קיים מסלול אז התשובה היא כן, אחרת התשובה היא לא

- סיבוכיות הזמן היא $\mathcal{O}\left(2^n n^3\right)$:
  - יש 2^n תתי קבוצות
  - $\binom{n}{2}=\mathcal{O}\left(n^2\right)$ אפשרויות ל-i,j בכל תת קבוצה
  - פתרון כל תת בעיה עולה $\mathcal{O}\left(n\right)$

5 גרפים

"גרף לא מכוון זה מקרה פרטי של גרף לא מכוון"

5.1 הגדרות

הגרפים בהם נעסוק הם מכוונים/לא מכוונים וממשוקלים/לא ממושקלים על הצמתים/קשתות. דרכים לממש גרף:

1. רשימת שכנויות: על ידי מערך לפי צמתים של רשימות מקושרות, לכל צומת רשימה מקושרת של כל הצמתים אשר יש לו קשת אליהם, מתאים לגרפים דלילים. סיבוכיות הבנייה היא $O\left(|V|+|E|\right)$.
2. מטריצת שכנויות: על ידי מערך דו מימדי בוא בתא ה-i,j יסומן האם קיימת קשת מ-i ל-j (מטריצת סמיכויות). מתאים לגרפים עם הרבה קשתות. סיבוכיות הבנייה היא $O\left(|V|^2\right)$.

נגדיר גרף פשוט בתור גרף ללא לולאות עצמיות וללא צלעות כפולות.

- סימון: $\underline{G}$ הוא גרף בו כיווני הקשתות הפוכים מ-G.
- יצירת גרף $\underline{G}$ בהינתן גרף G
  - אם מיוצג באמצעות מערך - עבור על כל רשימה מקושרת, אם יש קשת מ-i ל-j נוסיף לגרף החדש בתא ה-j ברשימה את הצומת i.
  - אם מיוצג באמצעות מטריצת שכנויות, שחלף אותה.

5.2 צמצום קשתות כפולות בגרף

ניתן לתחזק מערך ולעבור על כל הקשתות.

- תחילה, נאפס את המערך.
- בכל פעם שנוגעים בקשת נבדוק את הערך של צומת היעד במערך - אם הוא מכיל ערך ששווה לצומת המקור, מחק את הקשת מהגרף.
  - אחרת, עדכן את השדה במערך להיות צומת המקור.

דרך נוספת - במקום לסמן את צומת היעד נשתמש בדגלים 0 או 1.

- לאחר כל מעבר ננקה את המערך - נשמור את האינדקסים ש-"לכלנו" (סימנו ב-1) ונאפס אותם במעבר חוזר.
- בכך נכפיל את סיבוכיות הזמן, אך מקבלים בכל מעבר על קודקוד מערך נקי.

5.3 מיון טופולוגי, BFS ו-DFS

5.3.1 אלגוריתם חיפוש לרוחב BFS

האלגוריתם מתחיל מצומת שרירותי בגרף ועובר על כל הצמתים במרחק צלע אחת ממנו, ואז על כל הצמתים במרחק שתי צלעות ממנו וכן הלאה.

- חיפוש לרוחב סורק את הגרף בדרך שמבטיחה שכל צומת שנמצא באותו רכיב קשירות כמו השורש יבדק, וסריקה זו נעשית בזמן אופטימלי
- מימוש: תור - הצומת הראשון שנבדוק יהיה גם הראשון שנסיים לבדוק
- דרך פעולה רקורסיבית:
  - מצב התחלתי - דחוף את s לתור
  - תנאי העצירה - לאחר שעברנו על כל הצמתים
  - רקורסיה:

1. הוצא צומת מהתור
2. ··· הבדיקה המתבצעת על ידי האלגוריתם ···
3. עבור כל בן של הצומת, אם עוד לא ביקרנו בבן - נסמן אותו כצומת שבוקר ונכניס אותו לתוך התור

- סיבוכיות הזמן היא $O\left(|V|+|E|\right)$

אבחנות - על הגרף המכוון שנוצר על ידי החיפוש (נסמן את השורש הנבחר בתור s):

1. רשימת העומקים היא מונוטונית לא יורדת, כלומר אם צומת X מופיע אחרי צומת Y אז העומק של X גדול או שווה לעומק של Y.
2. לאחר סיום המעבר על כל הגרף, נראה כי כל צומת שסומן סומן על ידי צומת יחיד וקיים מסלול מ-s אליו שעברנו עליו באלגוריתם (נובע מהגדרת האלגוריתם).
3. לא ייתכנו מעגלים - נובע מאבחנות 1,2: אם קיים מעגל ו-s בו מונוטוניות המרחקים לא מתקיימת, ואם s לא בו יש סתירה עם העובדה שניתן לשחזר מסלול ממנו לצמתים.
4. תכונה עיקרית - $d\left(x\right)$ (העומק של x) שווה לאורך המסילה הקצרה ביותר מ-s ל-x.

<u>הוכחת אבחנה 4:</u>

- נגדיר $L\left(x\right)$ בתור המסילה הקצרה ביותר מ-s ל-x, ויהי $l\left(x\right)=\left|L\left(x\right)\right|$ אורכה.
- נוכיח כי $d\left(x\right)\geq l\left(x\right)$:
  - נבחר את המסילה מ-s ל-x שאורכה הוא $d\left(x\right)$, שהיא המסילה $x\rightarrow\pi\left(x\right)\rightarrow\pi^{2}\left(x\right)\rightarrow\ldots\rightarrow s$. זו מסילה חוקית שאורכה $d\left(x\right)$, מכיוון ש-$l\left(x\right)$ הוא אורך המסילה הקצרה ביותר אזי בהכרח $d\left(x\right)\geq l\left(x\right)$
- נוכיח כי $d\left(x\right)\leq l\left(x\right)$:
  - אבחנה - עבור כל צלע $\left(x,y\right)\in E$ מתקיים כי $d\left(y\right)\leq d\left(x\right)+1$
    * עבור כל x,y - או שכאשר y הגיע x הוכנס ואז $d\left(y\right)=d\left(x\right)$ או ש-x לא הוכנס ואז $d\left(y\right)=d\left(x\right)+1$
  - נפעיל אבחנה זו $\left|d\left(x\right)\right|$ פעמים וכך נקבל כי $\left|d\left(x\right)\right|\leq l\left(x\right)$
- בסך הכל קיבלנו כי $d\left(x\right)\leq l\left(x\right)\wedge d\left(x\right)\geq l\left(x\right)$ ולכן $\boxed{d\left(x\right)=l\left(x\right)}$.

שימושים:

- בדיקות בהן המטרה היא אופטימליות.

5.3.2 חיפוש לעומק DFS

האלגוריתם מתחיל את החיפוש מצומת שרירותי בגרף (s) ומתקדם לאורך הגרף עד שהוא נתקע (לא נותרו בנים), לאחר מכן הוא חוזר על עקבותיו עד שהוא יכול לבחור להתקדם לצומת אליו טרם הגיע.

- מימוש: מחסנית - הצומת האחרון שנבדק הוא גם הצומת הראשון שנסיים לבדוק
- כדי לפשט את ההסבר, נחלק את הצמתים לשלושה צבעים:
  1. לבן - הצומת <u>טרם נבדק</u>
  2. אפור - הצומת <u>או</u> צאצאיו <u>בתהליכי בדיקה</u>
  3. שחור - הצומת <u>וגם</u> צאצאיו <u>נבדקו</u>

דרך הפעולה היא רקורסיבית:

- מצב התחלתי - נדחוף את s למחסנית
- תנאי העצירה - המחסנית ריקה
- רקורסיה:

1. הוצא צומת מהמחסנית וצבע אותו באפור (כלומר - התחלת הבדיקה של צומת זה)
2. ··· הבדיקה המתבצעת על ידי האלגוריתם ···
3. עבור על כל הבנים של הצומת - אם בן לבן, הכנס אותו למחסנית
4. צבע את הצומת בשחור

- סיבוכיות הזמן היא $O\left(\left|V\right|+\left|E\right|\right)$.

אבחנות:

הערה. אם בהצגה לא מכוונת של גרף יש מעגלים, יש לנו כמה דרכים להגיע לצומת ולכן ייתכן ש-v צאצא של u אבל מצאנו את v בדרך אחרת. לכן, בהוכחה שלנו נסתכל על הגרף שמסלול החיפוש יצר, שהוא בלי מעגלים (קשתות שמגיעות לצומת שכבר נבדק לא משפיעות על החיפוש)

1. v צאצא של u אמ"מ זמן ההתחלה שלו וזמן הסיום של v מוכלים באלה של u

(א) כיוון צאצא $\Longleftarrow$ מוכל: נכון עבור צמתים שהיחס שלהם הוא של אב ובן ולכן אינדוקטיבית נכון על כל הצאצאים

(ב) כיוון מוכל $\Longleftarrow$ צאצא: גם כן, הדבר מתקיים באופן טריוויאלי על אב ובן ולכן אינדוקטיבית על כל הצאצאים

2. קטעי מחייה (כלומר - פרק הזמן בו צומת הוא אפור) לא יכולים להיחתך ללא הלכה

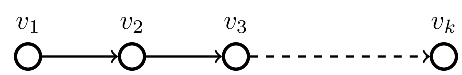
(א) לפי האבחנה הקודמת, אם צומת מתחיל בזמן פעולה של צומת אחר הוא צאצא שלו ולכן בהכרח לפי האבחנה מוכל בו

שימושים:

- עוזר כאשר יש היוריסטיקה (כלומר הנחה נוספת) שעוזרת לחיפוש - אז כבר ניתן בהתחלה לגלות את הדבר הרצוי.
- טוב כאשר המטרה היא איסוף מידע כללי, ולא יעילות.

5.3.3 משפט המסילה הלבנה

נניח שיש לנו מסילה מכוונת מ-v_1 ל-v_k שכולה לבנה, אזי v_k הוא צאצא של v_1 בעץ ה-DFS.



איור 10: מסילה מכוונת מ-v_1 ל-v_k

הוכחה. נניח כי יש לנו קטעים על ציר זמן כך שהקטע ה-k מסמן את משך הזמן בו הצומת ה-k היה אפור. קיים שני מצבים:

1. הקטע k מוכל כולו בתוך הקטע הראשון
2. הקטע k מחוץ לקטע הראשון לגמרי

נראה כי המצב 2 לא אפשרי - באינדוקציה על אורך המסילה:

- בסיס: $i = 1$: לפני ש-v_1 ישחיר הוא יבדוק את כל שכניו. או ש-v_k כבר הושחר, או ש-v_1 השחיר אותו. בכל מקרה הטענה מתקיימת.

□

- צעד: נסתכל על הצומת הראשון שהתגלה במהלך החיים של v_i.
  - כעת נשתמש בהנחת האינדוקציה עבור המסילה $v_i \rightsquigarrow v_k$.
  - באופן טרנזיטיבי קיבלנו כי v_k הוא צאצא של v_1.

5.3.4 מיון טופולוגי

"זה לא יער, זה דג"

משפט. לכל DAG יש מיון טופולוגי ואלגוריתם לינארי לגילוי שלו.

הוכחה. נוכיח את נכונות הטענה הראשונה.

- מאחר והגרף הוא DAG, יש קודקוד עם דרגת כניסה 0.
- בכל פעם נוציא את הקודקוד שדרגת הכניסה שלו היא 0, ונסמן אותו במספר הסידורי הרץ.

– לאחר ההוצאה הגרף יישאר DAG.
– נבצע את הפעולה n פעמים וקיבלנו מיון טופולוגי של הגרף.

□

נתון גרף מכוון חסר מעגלים DAG. המטרה: למיין את הקודקודים - לבנות יחס סדר חלקי לפתרון בעיות.

- כיצד מוגדר יחס הסדר החלקי? אם u אב קדמון של v אז u קטן מ-v
- נשתמש בזמני הסיום של ה-DFS:

– הוכחנו שב-DFS זמן סיום של צומת גדול מזמן הסיום של הצאצאים שלו, ולכן אם נעשה DFS ונמיין בסדר הפוך מסדר זמני הסיום - יתקיים שאב יקבל מספר קטן ממספר של בנו (כי זמן הסיום שלו גדול יותר) ולכן יהיה קטן ממנו ביחס סדר זה.

- פתרון נוסף - נעבור על כל הגרף ונספור לכל צומת את דרגות הכניסה שלו.

– נבחר צומת בעל דרגת כניסה 0, נסמן אותו במספר הסידורי הרץ, נוריד אותו מהגרף ונמשיך רקורסיבית
– בכל צומת בודקים את כל השכנים שלו, ולכן האלגוריתם חסום על ידי מספר דרגות היציאה + מספר הצמתים, ולכן סיבוכיות הזמן היא $O\left(|V|+|E|\right)$.

5.4 רכיב קשירות חזקה

נתון גרף מכוון ונתון יחס: $v\sim u$ אמ"מ קיים מסלול מ-u ל-v וקיים מסלול מ-v ל-u.

- זהו יחס שקילות על V. רכיבים של יחס זה נקראים רכיבי קשירות חזקה.
- אבחנה: הגרף המושרה מהיחס (יצירת סופר-קודקוד מכל רכיב קשירות חזקה) הוא DAG.
- בנוסף, רכיבי הקשירות של $G,\overleftarrow{G}$ זהים.

המטרה: בהינתן גרף G למצוא את רכיבי הקשירות שלו. אלגוריתם:

- נבצע DFS(1) על G ונסדר את הקודקודים לפי זמן סיום יורד.
- נבצע DFS(2) על $\overleftarrow{G}$, נעבור על הקודקודים לפי הסדר מ-DFS(1).
- פלט: עצי ה-DFS(2).

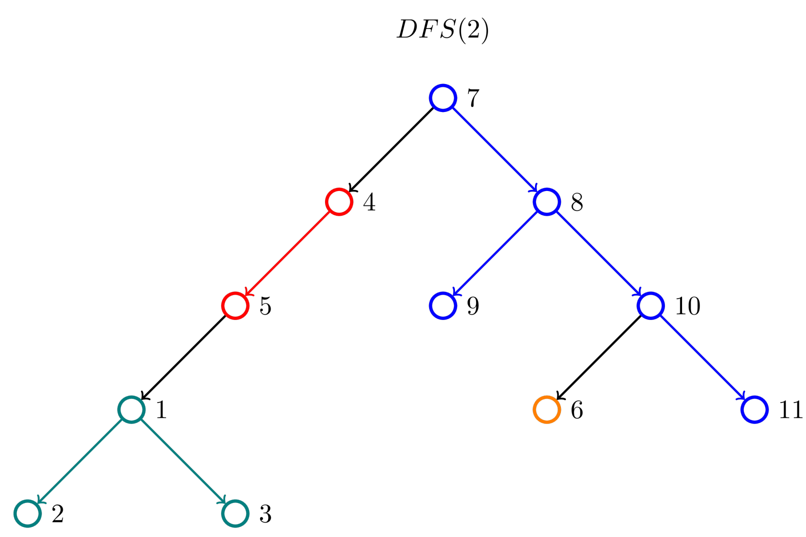


Figure 11: הגרף DFS (2) ממוספר לפי יחס הסדר של DFS (1), רכיבי הקשירות החזקה לפי הצבעים השונים

הוכחה. נכונות האלגוריתם: נסתכל על כל תתי העצים של DFS(2).

- יהי תת עץ k_i ב-DFS(2) שהשורש שלו הוא a_1. אזי, הוא האחרון שימות. אם s_{a_i} מינימלי ב-k_i אז f_{a_i} מקסימלי.
  - הדבר נובע באופן מיידי ממשפט המסילה הלבנה.
- אם יש קשת מ-k_i אל k_j אזי בהכרח $f_{a_i} > f_{a_j}$.
  - אם הקשת היא ל-a_j ניתן להפעיל את משפט המסילה הלבנה.
  - אחרת, a_j התגלה לפני שהצומת התגלה והוא בהכרח התחיל לפני a_i.
    * מכך נגרר כי יש מסילה $a_i \rightsquigarrow a_j$ בנוסף למסילה הקודמת מ-k_i ל-k_j, כלומר הרכיבים הם אותו הרכיב.
- a_i הוא הראשון (והאחרון) ב-k_i ב-DFS(2), והוא מכיל את כל רכיב k_i.
- נראה כי k_i לא כולל אף צומת אחר:
  - נחלק את הצמתים (בגרף ההפוך) שאינם ב-k_i ל-3 גלקסיות: אלה שיש להן קשת ל-k_i, אלה שיש ל-k_i קשת אליהם, וכל השאר.
- גלקסיית הצמתים שיש קשת מ-k_i אליהם: נראה כי כל הצמתים בה אינם לבנים.
  - כל הצמתים בגלקסייה ראו את k_i ב-DFS(1), כלומר זמן הסיום שלהם יהיה גדול מזמן ההתחלה של k_i.
  - לפי הלולאה החיצונית בוודאות כל הגלקסייה תופעל לפני ש-k_i יופעל.
- בוודאות אי אפשר לצרף צמתים מ-k_i לגלקסייה שיש קשת אל k_i (מאחר והסופר גרף הוא DAG) או לשלישית.

□

- סיבוכיות הזמן של האלגוריתם היא $O\left(|V|+|E|\right)$.

5.5 גרף k-קשיר

גרף נקרא גרף -2קשיר <u>צלעית</u> אם הורדת (כל) צלע לא פוגעת בקשירות שלו. ניתן לבדוק האם G הוא כזה בזמן לינארי ע"י DFS: נריץ DFS, ונכוון כל צלע בכיוון המעבר של ה-DFS - הפלט: הגרף $\widetilde{G}$.

- בגרף לא מכוון כל הצלעות שלא בעץ ה-DFS הן מצאצא לאב קדמון שלו (back edges)

טענה. G 2-קשיר צלעית $\Longleftrightarrow$ $\widetilde{G}$ קשיר חזק.

הוכחה. נוכיח כי הטענה נכונה, ונקבל אלגוריתם לבדיקה האם גרף הוא 2-קשיר.

הוכחה. נוכיח כי G 2-קשיר צלעית$\Longrightarrow$ $\widetilde{G}$ קשיר חזק:

- יהי זוג שכנים בגרף u, v. מאחר והגרף קשיר חזק יש מסילות $u \rightsquigarrow v, v \rightsquigarrow u$.
- נסתכל על שתי המסילות, ונזרוק את כל הקשתות שהשתמשנו בהן פעמיים.
  - זרקנו מעגל (דרגת הכניסה שווה לדרגת היציאה בכל קודקוד) ונשארנו עם מעגל (התכונה נשמרה)
- יורי לא מצליח, נקבל לתרגיל בית.
  - קבלו ביטול, הכל טוב (נסו בבית - זה חמוד).

הוכחה. נוכיח כי G 2-קשיר צלעית $\Longleftarrow$ $\widetilde{G}$ קשיר חזק.

- נניח בשלילה כי $\widetilde{G}$ אינו קשיר חזק, כלומר קיימים $u, v \in V$ כך שאין מסילה $u \rightsquigarrow v$.
- נסמן ב-S את קבוצת הקודקודים ש-v יכול להגיע אליהם ב-$\widetilde{G}$, ב-T את שאר הקודקודים.
- נובע מכך שכל הקשתות מ-T ל-S הן בכיוון $S \to T$.

- תחילה, חתך זה מכיל לפחות 2 קשתות (מאחר והגרף 2-קשיר צלעית, אם נסיר צלע אחת הגרף בוודאות יישאר קשיר כלומר קיימת אחת נוספת)
- אחרי המעבר הראשון ב-DFS מ-T ל-S, לא הייתה חזרה. הקודקוד הראשון ב-S שהגיע מ-T הוא אב קדמון של כל הצמתים ב-S, וכאשר הוא הגיע לכל צומת בחתך כל קשת נוספת (backedge) בחתך וניסה לעבור בחזרה מ-S ל-T.
- כלומר, לא הגיוני שהחתך מכוון כולו מ-T ל-S והגענו לסתירה.

□

□

□

5.6 מסילות קלות ביותר בגרף

"הגל עושה תכנות דינמי"

5.6.1 דייקסטרה

בעיה: מציאת אורך המסילה בעלת העלות המינימלית מ-a ל-b כאשר יש משקלים אי שליליים על הצלעות.

פתרון: נתחזק "חזית" של הצמתים הרחוקים ביותר אליהם מגיעים עם הגבלת המשקל x.

- מצא y שעבורו בכל שלב ושלב הקודקוד ה-z (בשלב הראשון $z=a$) נותן הצעה לכל שכניו:

$$\forall\left(z,u\right)\in E,u\text{outside}\quad|\;\delta\left(z\right)+w\left(z\to u\right)$$

וכל שכן מקבל את ההצעה אם היא משפרת את ההצעה הקודמת שלו. הקודקוד החיצוני עם ההצעה המינימלית יבחר.

הוכחה. נכונות האלגוריתם: הוא אורך המסלול הקצר ביותר מ-s אליו:
נבצע אינדוקציה על הזמן ונראה שלכל צומת v מתקיים שברגע הבחירה הערך השמור בו הוא אורך המסלול הקצר ביותר מ-s אליו:

- בסיס: אכן, $d\left(s,s\right)=0$ ולכן הוא המינימלי.
- צעד האינדוקציה: נניח כי עד כה הטענה מתקיימת ובשלב הזה מצטרף צומת v דרך הקשת $\left(u,v\right)$ עם משקל w_e, שיקבל את $d\left(s,u\right)+w_e$ - הטענה נכונה גם עבורו.
- תחילה, לפי הנחת האינדוקציה $d\left(s,u\right)$ מסמל את המסילה המינימלית מ-s ל-u, ולכן בנוסף לקשת w_e המשקל $d\left(s,u\right)+w_e$ מסמל משקל של מסילה <u>כלשהי</u> מ-s ל-v.
- נסתכל על המסילה הקצרה ביותר מ-s ל-v, נראה כי אורכה גדול או שווה ל-$d\left(s,u\right)+w_e$.
- מאחר ו-v מחוץ לצמתים שקיבלו הצעה ו-s בפנים, בנקודה מסוימת הייתה קשת $\left(x,y\right)$ מקבוצת הצמתים שקיבלו הצעה לקבוצת הצמתים שלא קיבלו הצעה.
- תחילה, הוצעה ל-y הצעה (לפחות מ-x)
- נבחר v, ולכן בפרט ההצעה של v היא לא פחות טובה מזאת של y.
- לפי הנחת האינדוקציה $d\left(s,x\right)$ מכיל את המרחק המינימלי מ-s ל-x.
- אורך המסילה של x ו-y היא ההצעה שהוגשה ל-y + משקל $k\geq 0$ ($y\rightsquigarrow v$)
- מאחר וההצעה של y גדולה או שווה מההצעה של v (אחרת סתירה מתכונת תת-אופטימליות), אם נוסיף לה k בוודאות ההצעה של v תישמר יותר טוב.

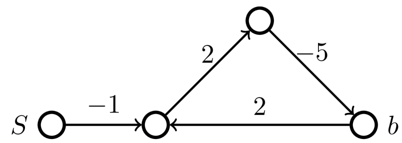
$\square$

מימוש:

- יש לנו שתי פעולות - עדכונים (לכל היותר כמספר הקשתות) והוצאת מינימום (לכל היותר כמספר הצמתים)
- לכן, המבנה המתאים הוא תור עדיפויות - ערימה.
- סיבוכיות הזמן של האלגוריתם היא $O\left(\left(|V|+|E|\right)\log|V|\right)$ אם נשתמש בערימה בינארית, במידה ונשתמש בערימת פיבונאצ'י הסיבוכיות תהיה $O\left(|E|+|V|\log|V|\right)$.

5.6.2 בלמן פורד

בעיה: מציאת מסלול בעל עלות מינימלית כאשר יש קשתות עם משקל שלילי.



איור 12: נשים לב לכך שהמרחק מ-S ל-b אינו מוגדר כאשר מגיעים למעגל שערכו הכולל שלילי: $\delta(S,b)=-\infty$

מטרה: בהינתן גרף מכוון G עם משקלים חיוביים ושליליים - למצוא את המרחקים מ-s לצומת v או להכריז כי קיים מעגל שלילי הניתן להגעה מ-s.

אבחנה: אם אין בגרף מעגלים שליליים, נגדיר $d(s,u)=\min_{P:s\rightsquigarrow u} w(P)$, והמינימום מושג ממסילה פשוטה.

- המסילה ללא מעגלים שליליים, במידה ויש בה מעגל חיובי נוכל להוריד אותו ולא להגדיל את משקל המסילה.

אלגוריתם:

- מצב התחלתי: $\forall v\in V\setminus\{s\}:\delta(s,v)=\infty,\delta(s,s)=0$.
- לולאה שנריץ $|V|-1$ פעמים:
  - $\forall (v,u)\in E:\delta(s,u)>\delta(s,v)+w(u,v)\implies \delta(s,u)=\delta(s,v)+w(v,u)$
- לולאת ביקורת: עבור על כל קשת בגרף פעם נוספת:
  - אם לא היה שינוי, כלומר $\delta(s,u)\ngtr\delta(s,v)+w(u,v)$ פלוט $\{d(s,u)\}$.
  - אחרת, יש מעגל שלילי.

הוכחה. נכונות האלגוריתם: נוכיח כי אם קיים מעגל שלילי שניתן להגיע אליו מ-s - נמצא אותו, ואם אין - נמצא את המרחקים המינימליים.

- נראה כי אם אין מעגל אז מתקיים $\delta(s,u)=d(s,u)$:
- קיימת מסילה פשוטה מ-u בעלת משקל מינימלי אפשרי.
  - באיטרציה הראשונה של האלגוריתם כל הצמתים שקיים להם מסלול מינימלי באורך 1 יעודכנו לאורך מינימלי זה, עבור כל איטרציה אורך המסלול שיתעדכן יגדל.
  - באיטרציה ה-i של הלולאה הראשית $\delta(s,u_i)=d(s,u_i)$ כאשר u_i הקודקוד ה-i במסילה (מרחקו מ-s הוא i).
  - מכיוון שאנחנו עוברים על כל המסלולים המינימליים בכל האורכים מ-1 ועד $|V|-1$, בהכרח נפגוש במסלול המינימלי וניתן להוכיח באינדוקציה שהטענה מתקיימת.
  - יש להוכיח באינדוקציה כי ערכי ה-δ מייצגים אורכי מסילות.
- אם קיים המעגל נזהה אותו באמצעות האלגוריתם - יהיה מסלול מינימלי אינסופי, כלומר תמיד אפשר יהיה לשפר ונמצא אותו בלולאה.

□

סיבוכיות הזמן: נעבור על כל הקשתות $\mathcal{O}(|V|)$ פעמים, ולכן $\mathcal{O}(|V|\,|E|)$.

5.6.3 גוהנסון

אלגוריתם שמוצא בין כל שני צמתים מסלול קצר ביותר, בגרף כללי. יעיל יותר מפליד ווארשל עבור גרפים דלילים.

- האלגוריתם מתבסס על הרצת דייקסטרה על כל צומת ($O\left(|V|\,|E|+|V|^2\log|V|\right)$), אך עם עיבוד מקדים כדי לטפל במשקלים שליליים על הקשתות.

- מוסיפים משקל h לכל צומת ומעדכנים את המשקל של כל קשת כך:

$$w'(u,v) = w(u,v) + h(u) - h(v)$$

- הוספה זו מבטיחה שהמסלול הקצר ביותר יישאר הקצר ביותר
- בוחרים את משקלי הצמתים בצורה הבאה:
  - מוסיפים צומת s ומחברים אותו עם משקל 0 לכל הצמתים בגרף
  - מריצים בלמן פורד, נקבל עבור כל צומת מרחק - זהו ה-h של כל צומת
- בלמן פורד לא יפגע בסיבוכיות האלגוריתם, ובסך הכל היא $O\left(|V||E| + |V|^2 \log|V|\right)$.

5.6.4 פלויד ווארשל

אלגוריתם שמוצא בין כל שני צמתים מסלול קצר ביותר, בגרף כללי.

- עבור שני צמתים: עובר על כל הצמתים ובוחר בכל פעם צומת דרכה נעבור (אם יש קשת, בודק גם את האפשרות הזו)
- חלוקה לשני מסלולים ובחירת המקרה בו סכימת המסלולים מינימלית (או את הקשת הישירה בין הצמתים, אם קיימת)
- שומר כל מסלול שנבדק לגמרי בתכנות דינמי

5.7 עץ פורש מינימלי

יהי G גרף לא מכוון וקשיר עם משקלים על הקשתות. המטרה: למצוא עץ פורש בעל משקל מינימלי.

הגדרה. עץ פורש הוא תת גרף של G שהוא עץ המכיל את כל V.

- לנוחות ההוכחה, נניח כי כל המשקלים שונים זה מזה.
- קשת ב-G תיקרא טובה אם היא מינימלית בחתך כלשהו (כי הכרחי לשמור לפחות על קשת אחת מבין החתכים לשמירה על קשירות, וזו הקשת עם העלות המינימלית)
  - חתך - חלוקה של G לשתי תתי קבוצות קודקודים לא ריקות, כל הקשתות שמחברות בין הקבוצות הן קשתות בחתך
- קשת בG תיקרא רעה אם היא מקסימלית במעגל כלשהו (כי היא הקשת שהכי טוב להעיף מהמעגל, בשביל לשמור על קשירות ולהקטין את העלות)

טענה. לא ייתכן כי קשת היא טובה ורעה בו זמנית.

הוכחה. נניח בשלילה שקשת g היא טובה ורעה בו זמנית.

- אזי, קיימת קשת נוספת f הנמצאת עם g במעגל ובחתך.
- מחד גיסא, g מינימלית בחתך ולכן קטנה מ-f.
- מאידך, g מקסימלית במעגל ולכן גדולה מ-f.
- סתירה! ולכן לא ייתכן שקשת היא טובה ורעה בו זמנית.

□

טענה. כל הצלעות הטובות מהוות עץ פורש.

הוכחה. נוכיח כי הגרף הנוצר הוא עץ פורש.

- אין מעגלים: אם היה מעגל הייתה בו קשת מקסימלית ולכן קשת שהיא טובה ורעה בו זמנית, והדבר אינו אפשרי.
- קשיר: נניח כי הגרף אינו קשיר, נבחר רכיב קשירות באקראי ונהפכו לקבוצה בחתך.
  - אזי, בהכרח קיימת קשת טובה בחתך זה.
  - כלומר רכיב הקשירות מקושר ברכיב קשירות נוסף - סתירה!
- פורש: ניתן ליצור חתך המכיל צומת אחד באחת הקבוצות.
  - החתך יכיל קשת מינימלית ואחד מקצוותיה יהיה הצומת המדובר.
  - ניתן ליצור חתך כזה עבור כל צומת ולכן יש קשת טובה היוצאת מכל צומת - הגרף פורש.

□

טענה. כל קשת היא טובה או רעה.

הוכחה. נסתכל על כל הקשתות הטובות - הן יוצרות עץ פורש.

- תהי קשת e לא טובה ב-G.
- e בהכרח סוגרת מעגל עם הקשתות הטובות (כי הן יוצרות עץ).
- במעגל זה יש קשת רעה כלשהי, שאינה אף אחת מהקשתות הטובות.
- לכן בהכרח הקשת הרעה היא e.

□

טענה. אוסף הקשתות הטובות הוא MST (עץ פורש מינימלי).

הוכחה. נניח בשלילה כי יש ב-MST קשת רעה.

- אזי, בהכרח קיימת קשת טובה שאינה ב-MST.
- נוסיף את הקשת הטובה ל-MST ונקבל מעגל.
- ברור כי עדיף להוריד את הקשת הרעה מהמעגל ולא את הטובה.
- נוריד את הקשת הרעה ונקבל עץ פורש בעל עלות קטנה יותר מהMST, סתירה!

□

סיבוכיות הזמן:

- עלות כל שלב - $\mathcal{O}(|E|)$.
- מספר השלבים הוא $\mathcal{O}(\log|V|)$ - מספר הצמתים קטן פי 2 בכל שלב.

בסך הכל, סיבוכיות הזמן היא $\mathcal{O}(|E|\log|V|)$.

5.7.1 אלגוריתם קרוקסל

נמיין את הקשתות וניקח בכל פעם את הקשת המינימלית שעוד לא בדקנו.

- אם הקשת לא מחברת בין אותו רכיב קשירות לעצמו (ויוצרת מעגל), נוסיף אותה.
- בכל פעם הקשת שבחרנו היא קשה טובה בחתך שיוצר רכיב הקשירות.

סיבוכיות הזמן:

- נפתור באמצעות Union-Find וכך סיבוכיות הזמן תהיה $\mathcal{O}(|E|\log|E|)=\mathcal{O}(|E|\log|V|)$.

הוכחה. נוכיח את נכונות האלגוריתם.

- כל צלע שנזרקת סגרה מעגל, מאחר וכל הצלעות הקודמות קטנות ממנה היא מקסימלית במעגל ולכן רעה.
- כל הצלעות שלא נזרקו הן לא רעות, ולכן טובות - כלומר כל הצלעות הטובות נכנסו.
- כלומר, כל צלע טובה נכנסה וכל צלע רעה לא נכנסה - והאלגוריתם מוצא MST.

□

קרוקסל הפוך:
כל פעם נסתכל על הצלע המקסימלית
נתון שהגרף קשיר - אם הורדת צלע פוגעת בקשירות הגרף חייב להוסיף אותה
מימוש: גרף דינמי שניתן להוריד צלע ולהוסיף צלע וכל פעם לבדוק קשירות, פר פעולה $\log n\left(\log\log n\right)^3$. (Thorup, Dynamic Graphs)

5.7.2 האלגוריתם של פרים

- נבחר צומת רנדומלי ונבחר את הקשת המינימלית שלו, כעת יש לנו שני צמתים (בקצוות הקשת).
- לאחר מכן, נבחר את המינימלית מבין הקשתות של שני הצמתים שלנו ונמשיך הלאה עד שנעבור על כל הצמתים.
- כל הקשתות שנבחרו הן הקשתות הטובות ביותר בחתך שמכיל את הצמתים שכבר מצאנו, ולכן האלגוריתם יפלוט עץ פורש מינימלי.

סיבוכיות הזמן:

- נפתור באמצעות ערימת פיבונאצ'י וכך סיבוכיות הזמן תהיה $O(|E|+|V|\log|V|)$.

הוכחה. נוכיח את נכונות האלגוריתם.

- בשלב כלשהו בריצת האלגוריתם, נוכיח כי הצלע הבאה שתיבחר מינימלית בחתך.
- כל הצלעות האחרות שנמצאות בחתך כבר קיבלו הצעה - נבחר את המינימלית שהיא מינימלית בחתך ולכן צלע טובה.
- כלומר, נבחר בכל פעם צלעות טובות בלבד.

□

5.7.3 אלגוריתם Boruvka

"היה פעם פטל בישראל, מיץ פטל וריבת פטל ואין יותר. זה עוזר מאוד נגד שיעול וחבל מאוד"

נפתור בצורה רקורסיבית:

1. נעבור על כל הצמתים בגרף ונסמן את הקשתות המינימליות היוצאות מהם - כולן בהכרח טובות ולכן יהיו ב-MST.
2. נכווץ את הצלעות ונקבל גרף מכווץ.
   (א) אם ניתקל בצלעות כפולות, נשאיר את המינימלית.
   (ב) אם יש בגרף המכווץ יותר מקודקוד אחד הפעל עליו את האלגוריתם.

<u>סיבוכיות הזמן:</u> $\mathcal{O}\left(E\log V\right)$. בכל פעם מספר הרכיבים קטן לפחות פי 2.

הוכחה. נוכיח את נכונות האלגוריתם.

- בכל שלב כל הקשתות שנוספות הן טובות: מינימליות בחתך של הסופר-קודקוד לעומת שאר הגרף.
- בנוסף, בסופו של דבר מתקבל עץ - בכל פעם רכיב הקשירות גדל, עד למצב בו יש רכיב קשירות אחד והגרף קשיר. הגרף חסר מעגלים ולכן גם עץ.

□

הגדרה. גרף מישורי הוא גרף שניתן לצייר אותו במישור בלי שצלעות שלא אמורות להיחתך ייחתכו.

<u>צלקת:</u> בכל גרף מישורי קיים קודקוד שדרגתו קטנה מ-6.

כיווץ של גרף שומר על מישוריות. בנוסף, בגרף מישורי $|E| < 3|V|$. לכן, במידה ונפעיל את האלגוריתם על גרף מישורי סיבוכיות הזמן תהיה $\mathcal{O}\left(n\right)$:

- שלב הכיווץ $\mathcal{O}\left(V\right)$, ומספר הקשתות קטן בכל פעם ויהיה $\mathcal{O}\left(V\right)+\mathcal{O}\left(\frac{V}{2}\right)+\cdots+\mathcal{O}\left(1\right)=\mathcal{O}\left(V\right)$.

שילוב של MST ודייקסטרה

"כמו הפרסומת לדלתות פנדור, זה מתאים למדינה כמו צרפת"

נסתכל על גרף לא מכוון $G=\left(V,E\right)$, בעל משקלים אי שליליים וקשיר.

- לכל $v\in V$ מגדיר D_v שהוא עץ הדייקסטרה המוגדר ממנו (עץ הקשתות שהתקבלו בהצעה), וה-MST שמושרש בו.
- האם ה-MST יהיה תמיד D_v? לא!
- האם ה-D_v יהיה תמיד MST? לא! גרף מלא בו כל הצלעות הן 1.
- האם קיים פתרון ביניים - כלומר לכל היותר פי 10 יותר גרוע מ-MST, ולכל היותר פי 10 יותר גרוע מדייקסטרא? כן!

<u>הרעיון:</u>

- ניקח עץ פורש ונהפוך אותו למעגל, שעובר בכל צלעות ה-MST פעמיים.
  - נמשקל את צלעות המעגל לפי הגרף. נתחיל מקודקוד v.
  - כל עוד $2\cdot d\left(v,r_i\right) < d\left(v,\ \underbrace{r_{i-1}}_{\text{קודם מסומן}}\right)+\underbrace{s}_{\text{המרחק ב-}MST\text{ בין v ו-}r\text{-1}}$, סמן את r.
- ניקח את כל הצמתים המסומנים, ואת החלק מעץ הדייקסטרה שמחבר ביניהם.
- עבור כל הלא-מסומנים, הוסף את הקשתות מהמעגל לעץ (נלביש על העץ את ה-MST).
- נריץ דייקסטרה מ-v, ונחזיר את הפלט.

הוכחה. שני אי שוויונות:

$$\Longrightarrow \underbrace{2\cdot d\left(v,r_i\right) < d\left(v,r_{i-1}\right)+s}_{\text{מצורת מסימון}} \wedge \underbrace{d\left(v,r_{i-1}\right)\leq d\left(v,r_i\right)+s}_{\text{אי שוויון המשולש}}$$

$$\Longrightarrow d\left(v,r_i\right) < 2s$$

כלומר, כל המרחקים בין הצמתים המסומנים הם לא יותר מפעמיים אורך המעגל.

$$\Longrightarrow W\left(\widetilde{G}\right)\leq W\left(MST\right)+4MST\leq 5MST$$

- עבור הצמתים המסומנים הנכונות ברורה.
- עבור הצמתים הלא מסומנים:
  - כל צומת לא מסומן מקיים $2\cdot d\left(v,r_i\right)\geq d\left(v,r_{i-1}\right)+s$.
- כלומר, כל המרחקים מ-v גדלו לכל היותר פי 2.
- הדייקסטרה יכול רק לשפר את המרחקים, ולא יכול להעלות את המשקל.
- כלומר בסך הכל המרחקים גדלים לכל היותר פי 2, ומשקל ה-MST יגדל לכל היותר פי 5.
- בסך הכל נמצא תת גרף (אחרי הדייקסטרה תת עץ)

□

המגניב הגדול: לפרוש את ה-MST למעגל כזה.

- אגב פריסת כבישים, עץ שטיינר

בעיית Steiner

"ברגע שאתה נותן למישהו זכויות שווים, הלך"

- נתון גרף $G=(V,E)$ עם פונקציית משקל $w>0$, נרצה למצוא תת גרף קשיר של קבוצת קודקודים כחולים עם משקל המינימלי ביניהם.
- הבעיה היא NP-Complete.
  - עץ פורש מינימלי נותן משקל לכל היותר פי 2 יותר גרוע.
  - קיימים קירובים בזמן פולינומיאלי, ניתן לקרב עד סף.
- אם יש שני קודקודים, ניתן לפתור בזמן פולינומיאלי: דייקסטרה.
- אם יש k קודקודים: יש לכל היותר קודקוד פיצול אחד.
- גישה לא חכמה:
  - דרגת כל הקודקודים הפנימיים היא לפחות 3, ולכן יש לכל היותר $k-2$ צמתים פנימיים (מאחר והגרף הוא עץ) ונקודות פיצול.
  - נמצא עץ פורש מינימלי עבור כל אפשרות של צמתים.
  - בעץ האופטימלי יש בין k ו-$k+(k-2)=2k-2$.
  - בסך הכל, עיבוד מקדים של $\mathcal{O}\left(n^3\right)$ - כל המרחקים, ו-$n^{k-2}\cdot k^2$ עבור מציאת כל ה-MST-ים.
- גישה חכמה (קריצה למועד א'): המטרה - $f\left(k\right)\cdot Poly\left(n\right)$.
  - נתונה קבוצה S של הצמתים הכחולים. תכנות דינמי (מהקטן לגדול):
  - עבור כל קודקוד $v\in V$ נמצא MST של כל $S'\cup\{v\}$ ($S'\subseteq S$, נעשה זאת בזמן $4^k\cdot n$)

- העץ האופטימלי יכול להיראות בכמה אופנים:
  * v צומת פנימי בעץ, העץ מורכב מהרכבה של שני עצים יותר קטנים ($n \cdot \underbrace{2^k}_{\text{מציאת ה-}MST} \cdot \underbrace{2^k}_{\text{תתי הקבוצות של } S}$), ניעזר בתכנות הדינמי.
  * v עלה בעץ, או שהוא מחובר לקודקוד שאינו כחול, ועוברים לעץ ללא v, או שהוא מחובר לכחול וזה בקטנה.
- בסך הכל, בערך $4^k \cdot Poly\left(n\right) = \underbrace{Poly\left(n\right)}_{\text{מציאת MST}} \cdot \underbrace{2^k}_{\text{מספר החלוקות לבעיה}} \cdot \underbrace{2^k}_{\text{מספר הבעיות}}$.

5.8 קבוצות קודקודים עם מרחק מקסימלי (clusters)

נתונות n נקודות עם מרחקים ביניהן. המטרה: למצוא k קבוצות קודקודים כך שהמרחק בין כל 2 קבוצות הוא מקסימלי.

- מרחק בין 2 קבוצות קודקודים א', ב' הוא המרחק המינימלי בין קודקוד בקבוצה א' לקודקוד בקבוצה ב'

פתרון: נמצא עץ פורש מינימלי ונמחק ממנו את $k-1$ הקשתות הגדולות ביותר.

הוכחה. נכונות הפתרון.

- המרחקים המינימליים בהכרח יהיו בעץ הפורש המינימלי (כי קבוצת קודקודים יוצרת חתך והמרחק המינימלי יהיה קשת בחתך, ולכן תהיה קשת טובה ותופיע ב-MST
- בשביל לחלק ל-k קבוצות בעץ נצטרך למחוק $k-1$ קשתות
- נניח בשלילה כי בפתרון האופטימלי מחקנו קשת שאינה מבין $k-1$ הקשתות הגדולות ביותר
  - אזי, עדיף לנו למחוק את הקשת הגדולה ביותר שלא מחקנו מבין $k-1$ הקשתות הגדולות ביותר (אשר קיימת לפי הבולֶט הקודם).
  - נקבל פתרון טוב יותר מכיוון שהמרחקים יהיו גדולים יותר - סתירה!

□

סיבוכיות הזמן:

- סיבוכיות הזמן שווה לסיבוכיות האלגוריתם למציאת ה-MST.

6 רשתות זרימה

6.1 הגדרות

"ארבע מינוס שתיים זה שתיים"

בתורת הגרפים, רשת זרימה היא סוג מיוחד של גרף מכוון, המשמש למידול בעיות שמערבות מעבר של חומר בין מקומות.

- $N=(G,s,t,c)\,, G=(V,E)$
  - s - צומת המקור.
  - t - צומת היעד, הבור.
  - $c:E\to\mathbb{R}^+$ פונקציית קיבול אי שלילית.
- מספר הנחות מפשטות:
  - $.d_{in}(s)=d_{out}(t)=0$
  - $.c:E\to\mathbb{N}$
  - אין לולאות עצמיות או קשתות מקבילות.
- פונקציית $s-t$ זרימה $f:E\to\mathbb{R}^+$ היא חוקית אם:
  - אילוץ הקיבול: $.\forall e\in E:0\le f(e)\le c_e$
  - שימור הזרימה: $\forall v\in V\setminus\{s,t\}:\sum_{e\in N^+(v)}f(e)=\sum_{e\in N^-(v)}f(e)$, סכום הזרימה הנכנסת שווה לסכום הזרימה היוצאת.
- הערך של פונקציית הזרימה מוגדר על ידי $.|f|=\sum_{v\in V}f(s,v)$
- ניתן להניח כי תחת אילוצי רשת הזרימה מתקיים גם $.|f|=\sum_{v\in V}f(v,t)$
- נרצה למצוא פונקצייה f כך ש-$|f|$ מקסימלי.

הגדרה. זרימה פשוטה $s-t$ היא זרימה המורכבת ממסילה פשוטה.

הגדרה. סירקולציה פשוטה $s-t$ היא זרימה על מעגל פשוט.

טענה. כל זרימה $s-t$ היא סכום של זרימות וסירקולציות פשוטות.

הוכחה. בהינתן גרף G, נבדוק האם הוא זרימה חוקית.

- בכל פעם נחסר זרימה מתוך הגרף (נחסר כמות קבועה מה-f של מסלול/סירקולציה כלשהם.
- כלומר, ניקח מסלול כלשהו ונוריד מכל קשת במסלול את המשקל המינימלי של קשת במסלול (אם אין מסלולים נבצע את הדבר על סירקולציה).
- לפחות f של קשת אחת יתאפס (כלומר קשת זו תיסתם), ונוכל להוריד קשת אחת ועדיין לשמור על G בתור זרימה אם הוא במקור זרימה. הסיבה לכך היא שלכל צומת במסלול אנחנו מורידים את אותו הסכום בכניסה וביציאה, ולכן הזרימה נשמרת.
- נמשיך עד שייגמרו המסלולים או הסירקולציות (עליהן נבצע בערך את אותו הדבר - לאחר מחיקת קשת מסירקולציה היא הופכת לזרימה פשוטות).
- אם בסופו של דבר ה-f של כל הקשתות הוא 0, כלומר אם נשארנו עם גרף "ריק", הגרף הוא זרימה.
- אחרת, אם נגמרו המסלולים והסירקולציות אך יש קשתות בהן ה-f אינו 0, G אינו זרימה.

<u>**סיבוכיות הזמן:**</u> $.\mathcal{O}(VE)$ □

הגדרה. $S-T$ - חתך.

חתך הוא חלוקה של הגרף ל-2 קבוצות, הקשתות בחתך הן הקשתות העוברות מקבוצה אחת לשנייה.

- בזרימה, אנחנו מחלקים לקבוצה S המכילה את s ולקבוצה T המכילה את t.

- נגדיר את ערך הזרימה לפי חתך $S-T$:

$$f(S,T)=\sum_{e:S\to T}f(e)-\sum_{e:T\to S}f(e)$$

טענה. בזרימה, $\forall S,T:f(S,T)=|f|$.

הוכחה. נראה תחילה כי הטענה נכונה לזרימות/סירקולציות פשוטות.

- בזרימות פשוטות, יש מספר אי זוגי של קשתות בחתך ולכולן גודל השווה לגודל הזרימה.
  - תהיה קשת אחת יותר לטובת S, השאר מתבטלות ולכן הזרימה היא כגודל הקמשת.

□

- בסירקולציות פשוטות המצב דומה, רק שהפעם יש מספר זוגי של קשתות (כי צומת ההתחלה והסיום זהה) ולכן הערך יהיה 0.

טענה. אם הטענה נכונה עבור זרימות f_1,f_2 פשוטות הדבר מתקיים גם עבור f_1+f_2.

- לינאריות של הסכומים בהגדרת ערך החתך.

<u>המטרה:</u> מציאת זרימה מקסימלית אפשרית תחת ההנחות.

הגדרה. <u>קשת רוויה</u> היא קשת e בה $f_e=c_e$.

נוסיף אפשרות להורדת זרימה מצלעות הפוכות - פוטנציאל השיפור הוא הכמות שאפשר להזרים לעבר t, כך למשל קשת הפוכה במסילה פשוטה 7 / 4 נוכיל <u>להוריד</u> 4.

- מדוע גם כאשר ניתן להוריד זרימה, היא נשארת חוקית?
  - הכל מצטמצם - מה שהורדנו מהקשת ההפוכה יורד מהקשת שאליה נכנס, והזרימה נשמרת.
- בעייתי - לא בהכרח נעצר אחרי E איטרציות.

משפט. אם אין מסילות משפרות (בשני הכיוונים) אזי הזרימה מירבית.

<u>אבחנה:</u> לכל זרימה $s-t$, לכל חתך $S-T$, $|f|\le C(S,T)$.

$$|f|=f(S,T)=\sum_{e:S\to T}f(e)-\sum_{e:T\to S}f(e)\le\sum_{e:S\to T}c_e-0=c(S,T)$$

הגדרה. רשת שיורית.

- רשת זרימה המתארת כמה עוד ניתן להזרים.
- נוצרת על ידי לקיחת כל זוג צמתים והגדרה של קיבול שיורי עבורם, המתאר כמה עוד ניתן להזרים ביניהם - $c^f_{u,v}=c_{u,v}-f_{u,v}$.
- גרף הרשת השיורית $G^f=\left(V,E^f\right)$ מורכב מצמתי הגרף המקורי, ומהקשתות שקיבוליהן השיורי חיובי.
- לבסוף, צומת המקור והבור הם אלה מהרשת המקורית.

הגדרה. מסלול משפר.

- נניח שברשת שיורית קיים מסלול על פני קשתות בעלות קיבול (שיורי) חיובי ממש.
- נאמר שמסלול זה הוא מסלול משפר לזרימה ברשת המקורית.
- מסלול משפר אינו בהכרח מאבד את כיוון הרשת.
  - אם e בכיוון שהגענו ממנו, $\Delta(e) = c_e - f_e$, אחרת $\Delta(e) = f_e$.

אבחנה: אם ברשת השיורית קיים מסלול משפר, וקיבול המינימום בגרף השיורי הוא c, אזי אפשר להגדיל את הזרימה ברשת המקורית ב-c על ידי הזרמה נוספת במסלול המשפר.

הגדרה. שיפור על פני מסלול משפר.

- תהי m מסילה משפרת, נגדיר $\Delta(m) = \min_{e \in m} \Delta(e)$.
- נקבל כי $f' = f + \Delta(m)$ וגם כי f' היא זרימה חוקית.
- אם לא קיימת מסילה משפרת ל-f אז f זרימה מקסימלית.

מסקנה. הזרימה המקסימלית שווה לחתך המינימלי, $\text{maxflow} = \text{mincut}$. בהינתן maxflow ניתן למצוא את mincut בזמן לינארי.

רעיון מאחורי הפתרון:

- נתחיל מזרימה התחלתית 0.
- כל עוד ניתן נבחר מסלול לא רווי (מסלול שהזרימה העוברת דרכו קטנה מהקיבולת שלו), ונשפר על פי מסלול זה.
- כלומר, אם נגדיר את פוטנציאל השיפור כקיבולת צלע פחות הזרימה העוברת דרכה, אזי נשפר בהתאם לפוטנציאל השיפור המינימלי מבין כל הצלעות במסלול זה.

טענה. אם $\forall e : c_e \in \mathbb{N}$ אז קיימת זרימה מקסימלית שלמה ($\forall e : f_e \in \mathbb{N}$).

הוכחה. הזרימה ההתחלתית היא 0. מאחר וכל הקיבולות שלמות, בכל פעם הזרימה תגדל בגודל שלם. כלומר, maxflow היא סכום של שלמים ולכן $|f| \in \mathbb{N}$. $\square$

- פורד פלקרסון - לשפר לשפר לשפר (מה שעשינו מקודם).

6.2 אגמונדס קארפ

נשתמש בפורד פלקרסון, אך בכל פעם נבחר במסילה המשפרת הקצרה ביותר. כעת בדוק מתכנס וגם מהיר.

- אינטואיציה לנכונות. נניח כי בשלב ה-i שיפרנו מסילה באורך L_i.
  - אם המסילה שציירנו בשלב ה-$i+1$ לא הולך בצלע נגדית ב-i בוודאות $L_{i+1} \geq L_i$.
  - גם אם הלכנו בכיוון הנגדי, אם משתמשים במה שהשיפור ה-i פתח בוודאות תאריך את המסלול שלך (תספור את הקשת שהלכת בנגדי 3 פעמים).
  - בסך הכל, קיבלנו כי $L_{i+1} \geq L_i$.
- בכל שיפור אנחנו סותמים יותר ויותר צלעות, ולכן בכל כמה זמן בוודאות נלך בכיוון הנגדי ונגדיל את אורך המסילה.
- בסך הכל, קיבלנו כי האלגוריתם מתכנס (מאחר והמסילה המשפרת הארוכה ביותר בגודל $V-1$).

בעיה: השוואה בין שכנים בלבד. אין בהכרח טרנזיטיביות. אם $i, i+1$ ו-$i+1, i+2$ מסכימים, לא בהכרח $i, i+2$ מסכימים. שני מקרים אפשריים:

- חזרנו ופתחנו קשת (והזרימה גדלה)
- כל הקשתות נסתמו וסיימנו.

נרצה לא להיות תלויים בשכנים. $d_i(s,v), d_i(t,x)$ הם אורך המסילה המשפרת הקטנה ביותר מ-s ל-x בשלב i מ-x ל-t. נרצה להראות כי:

$$d_{i+1}(s,x) \geq d_i(s,x) \wedge d_i(x,t) \leq d_{i+1}(x,t)$$

נניח כי $i > j$; L_i, L_j השתמשו באותה צלע בכיוונים שונים. אזי - $|L_j| \geq |L_i| + 2$ ולכן $|L_j| > |L_i|$.
במקרה זה, מספר השלבים קטן מ-$\frac{(E+1)(V-1)+1}{2}$. כל שלב עולה זמן לינארי, ולכן בסך הכל $\mathcal{O}\left(|V||E|^2\right)$.

6.3 שידוך מקסימלי בגרף דו צדדי

נרצה למצוא זיווג מקסימלי - אוסף צלעות זרות בקודקודים. חמדן לא עובד! נתרגם את השאלה לזרימה.

- חיבור צומת s לכל הצמתים בצד אחד, t לצד השני.
- הקיבולות בתוך הגרף לא משנות, שאר הקיבולות 1.
- נמצא maxflow ולאחר מכן את הזיווג ב-G: כל הצלעות בין שני הצדדים שהזרימה בהן הוא 1. זה פתרון, והוא אופטימלי.

הוכחה. נוכיח שקיבלנו פתרון והוא אופטימלי.

- מכל צומת תיבחר לכל היותר קשת אחת בתוך G, מאחר והיא קיבלה מ-s 1 והיא תתן אותו לצומת אחת מהצד השני.
- לכל צומת תגיע לכל היותר קשת אחת, אחרת הזרימה ממנה ל-t תהיה יותר מהקיבולת - סתירה!

כלומר, קיבלנו זיווג.

- גודל הקבוצה שמצאנו $|M|$ שווה לערך הזרימה המקסימלית $|f_{max}|$.

$$\forall M' \text{ זיווג } \exists f_{M'} : |f_{M'}| = |M'|$$

$$\Longrightarrow |f_{max}| \geq |M_{OPT}|$$

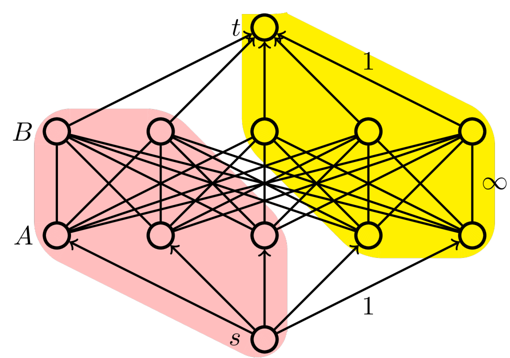
$$\Longrightarrow |M| \geq |M_{OPT}| \Longrightarrow \boxed{|M| = |M_{OPT}|}$$

- והפתרון אופטימלי.

<u>סיבוכיות הזמן:</u> בכל פעם משפרים ב-1: $\mathcal{O}\left((E+V)\cdot V\right) = \mathcal{O}\left(|V|\,|E|\right)$. □

6.4 קבוצה בלתי תלויה מקסימלית בגרף דו צדדי

כעת נגדיר את הקיבולות בין שני צדדי הגרף להיות ∞. אזי $|f_{max}| = |M_{OPT}| = |\text{mincut}|$.



איור 13: נסתכל על חתך $S-T$ כלשהו. נסמן $A' = A\cap S, B' = B\cap S$, ו-$A'' = A\setminus A', B'' = B\setminus B'$.

$$|C| = |\{e\in E | e: s\to A''\}|\cdot 1 + |\{e\in E|e: A'\to B''\}|\cdot\infty + |\{e\in E|e: B'\to t\}|\cdot 1$$

הזרימה היא סופית ושלמה. בנוסף, ב-mincut אין צלעות מ-A' ל-B'' (קיימים חתכים סופיים, והוא החתך המינימלי): $A'\cup B''$ היא קבוצה ב"ת I.

$$\Longrightarrow \underbrace{|C|}_{|M_{OPT}|} = |A''| + |B'|$$

$$\Longrightarrow \underbrace{|M_{OPT}|}_{|A''|+|B'|} + \underbrace{|I|}_{|A'|+|B''|} = |V(G)|$$

טענה. לכל גרף G, לכל J ב"ת וזיווג M מתקיים $|M| + |J| \leq |V(G)|$.

הוכחה. במקרה הטוב ביותר J מכילה את כל הלא-מזווגים, ו-1 מכל זוג בזיווג. ולכן בכל מקרה אי השוויון מתקיים. □

טענה. כאשר $|M_{OPT}| + |I| = |V(G)|$, I היא קבוצה ב"ת מקסימלית.

מסקנה. בגרף G דו צדדי, אם נמצא f_{max} נקבל את $MaxInd$ ב-G.

$$\Longrightarrow |M_{OPT}| + |MaxInd| = |V(G)|$$

6.5 משפט Dilworth

"זה יפריע לתחתונים"

נתון יחס סדר חלקי.

- שרשרת היא סדרת איברים עולה: $a_1 < a_2 < \cdots < a_k$.
- אנטי-שרשרת היא סדרה $b_1, \ldots, b_r$ כך שאין יחס בין אף שני איברים בה.

משפט. ניתן לכסות את כל האיברים ע"י r שרשראות, כאשר r הוא גודל האנטי-שרשרת המקסימלית.

הוכחה. נבצע רדוקציה.

- ניצור גרף דו צדדי, עבור כל איבר v יהיה v_d, v_u (מלשון up ו-down).
- אם $v > u$ תהיה קשת $v_d \rightarrow u_u$.
- תחילה, ברור כי גודל הכיסוי גדול או שווה מגודל האנטי-שרשרת (מאחר וכל איבר באנטי-שרשרת המקסימלית חייב להיות בשרשרת משל עצמו, ולא עם אף אחד מחבריו לאנטי-שרשרת).
- נסתכל על חלוקת הגרף לשרשראות, ונמצא זיווג מקסימלי ב-$G_{\geq}$.
- כעת, נמצא זיווג מקסימלי.
- נסתכל על הקבוצה הב"ת המקסימלית.
  - כל האיברים שלא מופיעים בזיווג נמצאים שם בוודאות, ומכל זיווג איבר אחד בדיוק (לפי $|M_{OPT}| + |MaxInd| = |V(G)|$).
  - עבור כל שרשרת, עד מקום מסוים נבחר בקודקודים עליונים, ואחריו תחתונים בלבד (כל שרשרת מתחילה בקודקוד עליון ומסתיימת בתחתון שלא בזיווג)
  - מכל שרשרת קודקוד אחד ייבחר פעמיים.
  - כל הקודקודים בהם בחרנו פעמיים מהווים אנטי-שרשרת: אין ביניהם יחס מאחר והם בבלוקים שונים.
  - גודל האנטי-שרשרת הוא מספר הבלוקים, שהוא גם גודל הכיסוי.
  - כלומר, גודל הכיסוי שווה לגודל האנטי-שרשרת.
  - מאחר וכל כיסוי גדול או שווה מכל אנטי-שרשרת, הכיסוי מינימלי והאנטי-שרשרת מקסימלית.

□

6.6 דייניץ

כעת, בכל שלב ושלב ניתן להגיע מ-s ל-t במסילה הכי קצרה במשקל d.

- אתר את כל הצלעות שנמצאות במסילה שמשקלה d.
  - הרץ דייקסטרא על G מ-s, ועל $\overline{G}$ מ-t.
  - עבור על כל קשת, היא נמצאת במסילה שמשקלה $d\left(s,u\right)+w_e+d\left(v,t\right)$.
  - פשוט הולכים קדימה.
- סיבוכיות הזמן היא $\mathcal{O}\left(\left|V\right|^2\left|E\right|\right)$.

6.7 תמונה שחור לבן

"נשים את השחורים למטה כי הם הרעים"

נתונה תמונה. המטרה: להפוך את התמונה לשחור לבן, כך שיישמר מידע על התמונה.

- לכל פיקסל (=קודקוד) v יש שני שדות $w_v,b_v\geq 0$, הרווח שנקבל אם נבחר שהפיקסל יהיה שחור או לבן בהתאמה.
- לכל צלע (התמונה היא grid) e יש שדה z_e שיכיל את הרווח מלחתוך את הצלע.

המטרה: לחלק את הקודקודים לשתי קבוצות X,Y, X קבוצת הלבנים ו-Y השחורים ($X\sqcup Y=V$), המשיגות אופטימום של $W\left(X\right)+B\left(Y\right)-Z\left(X-Y\right)$.

פתרון. פתרון לבעיה, באמצעות זרימה.
תחילה, הופכים את הגרף מלא מכוון למכוון.
נוסיף קודקודים b,w. הקיבולת של הקשת מ-b ל-v היא b_v, ומ-v ל-w היא w_v. בנוסף, הקיבולת של הקשת $e=\left(u,v\right)$ היא z_e.
נמצא זרימה מקסימלית מ-b ל-w, וכך גם את החתך המינימלי, שמכיל את כל הצלעות הפנימיות שלא הורדנו, ועוד הצלעות מ-w ל-X ומ-b ל-Y.

$$\Longrightarrow W\left(Y\right)+Z\left(X-Y\right)+B\left(X\right)$$

נשים לב, כי $W\left(Y\right)+Z\left(X-Y\right)+B\left(X\right)=\left(W\left(V\right)+B\left(V\right)\right)-\left(W\left(X\right)-Z\left(X-Y\right)+B\left(Y\right)\right)$. כלומר, כאשר החתך מינימלי פונקציית המטרה שלנו $W\left(X\right)+B\left(Y\right)-Z\left(X-Y\right)$ מקסימלית.

6.8 משפט Hall

משפט. יהי $G=\left(A,B,E\right)$ גרף דו צדדי. G מכיל זיווג מושלם (יעני - לכולם יש זוג) אמ"מ לכל תת קבוצה $S\subseteq A$ מתקיים $\left|\Gamma\left(S\right)\right|\geq\left|S\right|$ (Γ - כמות השכנים של כל הקבוצה).

הוכחה. נוכיח את שני כיווני ההוכחה.

- נוכיח את הכיוון הראשון $\Longleftarrow$.
  - נניח ש-G מכיל זיווג מושלם.
  - אזי, לכל קבוצה $A'\subseteq A$ יש לפחות $\left|A'\right|$ קשתות ל-$\left|A'\right|$ צמתים שונים, כי לכל צומת ב-A' יש איבר בB שהוא מזווג אליו, ולכן $\left|\Gamma\left(A'\right)\right|\geq\left|A'\right|$.
- נוכיח את הכיוון השני $\Longrightarrow$.
  - נסתכל על מצב בו $\left|A\right|=\left|B\right|=n$ (התנאי הכרחי כי קיימת התאמה חח"ע ועל מכל צומת ב-A לזיווגו)
  - ידוע כי $\text{maxMatch}=\text{maxFlow}=\text{minCut}$, נותר להראות ש-$\text{minCut}\geq n$.
  - נסתכל על חתך כלשהו.

$$c\left(\text{cut}\right)=\underbrace{\left|B'\right|}_{\geq\left|B'\cap\Gamma\left(A'\right)\right|}+\left|A\right|-\left|A'\right|+\underbrace{E\left(A'\to B''\right)}_{\geq\left|B''\cap\Gamma\left(A'\right)\right|=\left|\Gamma\left(A'\right)\right|}\geq\left|A\right|-\left|A'\right|+\left|\Gamma\left(A'\right)\right|$$

$$\geq\left|A\right|-\left|A'\right|+\left|\Gamma\left(A'\right)\right|\geq\left|A\right|-\left|A'\right|+\left|A'\right|=\left|A\right|=n$$

- קיבלנו כי קיבולת החתך גדולה מ-n, ולכן גם קיבולת החתך המינימלי גדולה מn וקיימת זרימה שגודלה n - זרימה שבה לכל צומת יש זוג, כנדרש.
- כלומר, הגרף מכיל זיווג מושלם.

□

משפט. יהי $G=(A,B,E)$ גרף דו צדדי d-רגולרי (כל דרגותיו הן d), כאשר $d>0$. אזי, יש ב-G זיווג מושלם.

הוכחה. נסתכל על תת קבוצה $A'\subseteq A$. יוצאות מ-A' $d\cdot|A'|$ צלעות.

- נניח בשלילה כי $|A'|>|\Gamma(A')|$.
- אזי, מתקיים כי ל-$\Gamma(A')$ נכנסות $d\cdot|A'|$ צלעות אבל יוצאות ממנה $d\cdot|\Gamma(A')|$ - סתירה!

□

7 אלגוריתם KMP

"יש תולעת שנשענת על האחוריים שלה"

נתונה טקסט T ומחרוזת W, נרצה אלגוריתם לינארי למציאת כל ההופעות הרציפות של W ב-T. שיטת העבודה - sliding window.

- שלב ההכנה - W נגד W: לכל מיקום ב-W, נמצא את הרישא הארוכה ביותר ב-W שמתאימה לו.
  - נניח כי רישא של W (עד התו ה-k) היא סיפא של חלק מסוים של T. נרצה להזיז את W קדימה.
  - נזיז את W קדימה עד למקום הראשון $0 \leq \phi(k) < k$ שבו רישא קטנה יותר של W היא סיפא של W עד למקום מסוים.
    * אם יש הסכמה, ממשיך להזיז קדימה את החזית.
    * אחרת, חוזר על התהליך.
- שלב השימוש - W נגד T.
  - מתבצע באופן דומה מאוד לשלב ההכנה.
  - בכל פעם או שהחזית תתקדם או שהאחורה, לכן בסך הכל $2|T|$ צעדים לכל היותר וסיבוכיות הזמן היא $\mathcal{O}(n)$ (לשיעורין).

חישוב $\phi(k)$, $W = xyyxyyxyx$.

- תחילה, מאחר ו-$0 \leq \phi(k) < k$ מתקיים $\phi(1) = 0$ (אין התאמה, אסור להישאר).

$$\phi(1) = \phi(2) = \phi(3) = 0$$

$$\phi(4) = 1$$

$$\phi(5) = 2$$

$$\phi(6) = 3$$

$$\phi(7) = 4$$

$$\phi(8) = 5$$

- כעת, אין הסכמה בתו התשיעי. נצטרך לזוז $\phi(\phi(9-1))$. גם $\phi(5)$ אין הסכמה, נלך ל-$\phi(\phi(5)) = \phi(2) = 0$, ולכן אין תרומה.
  - כלומר, $\phi(9) = 1$.

8 התמרת פורייה ו-FFT

"אני חושב שאני רוצה לישון"

נתונים שני פולינומים $P_1(x), P_2(x)$ כך ש-$\deg P_1 = d_1, \deg P_2 = d_2$. המטרה: לחשב את $P_3(x) = P_1(x) \cdot P_2(x)$.

- הפתרון הנאיבי יעלה $\mathcal{O}(n^2)$.
- גישה נוספת - ניתן לקבע פולינום ממעלה d עם $d+1$ נקודות.
  - נניח שאנחנו יודעים את ערכם של שני הפולינומים ב-$d_1 + d_2 + 1$ נקודות.
  - הערך של מכפלת הפולינומים בכל אחת מהנקודות שווה למכפלת הערכים של P_1, P_2.

כלומר, משני פולינומים P_1, P_2 בהצגת מקדמים, נעבור לייצוג לפי ערכים, להכפיל, ולחזור לייצוג מקדמים של פולינום המכפלה.

- אממה, אם נבחר נקודות כלשהן לא נרוויח ונשלם $\mathcal{O}\left(n^2\right)$ זמן בכל מקרה.
- לכן, נבחרי בשורשי היחידה.
  - נניח כי $2^{k-1} < d_1 + d_2 < 2^k = N$.
  - בנוסף, נסתכל על שורשי היחידה $1, \varepsilon^1, \varepsilon^2, \ldots, \varepsilon^{N-1}$ כאשר $\varepsilon = e^{2\pi i/N}$.
- נוכל לחשב את ערכי הפולינומים בשורשי היחידה ב-$\mathcal{O}\left(n \log n\right)$.

דוגמה. עבור פולינום מסדר 4.
תחילה, עבור 1 החישוב פשוט: $a_0 + 1 \cdot a_1 + 1 \cdot a_2 + a_3 \cdot 1$
עבור ε: $a_0 + a_1\varepsilon + a_2\varepsilon^2 + a_3\varepsilon^3$
עבור ε^2: $a_0 + a_1\varepsilon^2 + a_2 \cdot 1 + a_3\varepsilon^2$
עבור ε^3: $a_0 + a_1\varepsilon^3 + a_2\varepsilon^2 + a_3\varepsilon$
נסתכל על העמודות הזוגיות והאי זוגיות בנפרד: דרוש לנו לחשב את $a_0 + a_2$ ואת $a_0 + a_2\varepsilon^2$.
בנוסף, בעמודות האי זוגיות לחשב את $a_1 + a_3, \varepsilon\left[a_1 + a_3\varepsilon^2\right], \varepsilon^2\left[a_1 + a_3\right], \varepsilon^3\left[a_1 + a_3\varepsilon^2\right]$.

כלומר, נוכל לחשב את FFT מסדר n באמצעות שני FFT מסדר $n/2$, ובזמן לינארי לסכום את התוצאות.

$$\mathcal{T}\left(n\right) = 2\mathcal{T}\left(n/2\right) + \mathcal{O}\left(n\right)$$

$$\Longrightarrow \boxed{\mathcal{T}\left(n\right) \in \mathcal{O}\left(n \log n\right)}$$

9 נספחים

9.1 פתרון משוואות רקורסיביות

נתונה משוואה מהצורה:

$$T\left(n\right) = \sum a_i T\left(b_i n\right) + g\left(n\right)$$

כאשר $0 \le b_i \le 1, a_i > 0$. מה הנוסחא הסגורה ל-$T\left(n\right)$? נמצא מספר p שמקיים:

$$\sum_{i=1}^{k} a_i b_i^p = 1$$

ואז נקבל כי:

$$T\left(n\right) = \Theta\left(n^p\left(1 + \int_1^n \frac{g\left(x\right)}{x^{p+1}} dx\right)\right)$$