אלגוריתמים 1

חוברת הרצאות

יולי 2015

שלמה מורן

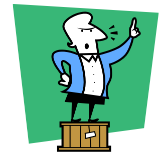
החוברת מכילה תקצירי הרצאות של הדס שכנאי ושלי, שניתנו בסמסטר חרף 2006-7, בתוספת מספר הרצאות של ספי נאור ושלי מסמסטר חורף 2012-2013. בסוף החוברת מצורפים דפי נוסחאות שצורפו לבחינות בשנים 2013-14.

בכתיבת חוברת זו נעזרתי בחוברת של גיל כהן מ2007, שהכילה סיכומים של ההרצאות מאותה שנה, ובמספר סיכומי הרצאות של סטודנטים בקורס.

ברצוני להודות לשמואליק זקס על הערותיו, וליונתן גולדהירש על הערותיו ועל הכנת גירסה ראשונית של דף הנוסחאות.
תודות גם לסטודנטים שלא התביישו לשאול ולהעיר, ובכך תרמו לשיפור החוברת.

ייתכן שהחוברת עדיין מכילה טעויות דפוס. מי שמגלה טעות כזו מוזמן ליידע אותי ובכך להביא לתיקונה.

שלמה מורן



הרצאה מס' 1

אלגוריתמים לחיפוש בגרפים

אלגוריתמים לחיפוש בגרפים באים לענות על שאלה בסיסית: בהינתן צומת בגרף, מהם הצמתים הנגישים (בני הגעה, או reachable) ממנו.

קלט: גרף (סופי) מכוון או לא מכוון $G=(V,E)$, וצומת מקור $s\in V$.

פלט: עץ מכוון $T=(R,E')$ עם שרש s, המכיל את כל הצמתים ב- V הנגישים מ- s. עץ כזה נקרא "עץ חיפוש". במיוחד, $R\subseteq V$, ו E' הן קשתות מכוונות המייצגות קשתות ב E.

הערה: גירסאות מסויימות של אלגוריתמי חיפוש יחזירו **יער** פורש כאשר הגרף לא קשיר.

האלגוריתמים בפרק זה יוגדרו עבור גרפים לא מכוונים, אך בשינויים המתבקשים הם נכונים גם לגרפים מכוונים.

אלגוריתם חיפוש גנרי

$R := \{s\};\ E'=\Phi$
while there is an edge (u,v) s.t. $u\in R$ *and* $v\notin R$
$\qquad R := R\bigcup\{v\};\ \ E' := E'\cup\{(u\to v)\};$
$\qquad$ *Parent(v) := u;*

בהנחה שניתן להוסיף צמת לR ולבדוק אם צמת נמצא בR בזמן קבוע, כל איטרציה של פסוק ה while ניתנת לביצוע בזמן $O(E)$ [†]. מאחר ובכל איטרציה מספר הצמתים בR גדל ב1, יש O(V) איטרציות. מכאן שסיבוכיות הזמן של האלגוריתם היא $O(VE)$. שני המימושים שנציג- BFS ו DFS - יהיו בעלת סיבוכיות קטנה יותר – $O(V+E)$.

למה 1.0:

א. הגרף $T=(R,E')$ הוא עץ מכוון ששרשו s.

ב. R מכילה את כל הצמתים הנגישים מ s.

<u>הוכחה</u>: א.: בגרף T לכל צומת מלבד s יש דרגת כניסה 1, ול s דרגת כניסה 0. כמו כן אינדוקציה פשוטה מראה שיש מs מסלול מכוון של קשתות מ E' לכל צומת ב R. לכן, על סמך משפט איפיון של עצים מכוונים, T הוא עץ מכוון.

ב.: אם צמת u הנגיש מs איננו בעץ, אז במסלול מs לu קיימת קשת (x,y) כך ש $x\in R,\ y\notin R$, וזה לא ייתכן בסיום האלגוריתם. לכן כל צמת u כזה נמצא בעץ. מצד שני, אינדוקציה פשוטה מראה שאם u נמצא בעץ אז קיים מסלול מ s ל u ולכן u נגיש מ s. ■

---

† בחוברת זו V ו-E מייצגים, בהתאם להקשר, או את קבוצות הצמתים והקשתות בגרף, או את העוצמות של קבוצות אלו.

נלמד שני מימושים של האלגוריתם הגנרי: "חיפוש לרוחב" ו"חיפוש לעומק".

חיפוש לרוחב (Breadth First Search - BFS):

<u>הגדרה:</u> מרחק בין שני צמתים בגרף לא ממושקל הוא המספר המינימאלי של קשתות במסלול ביניהם, או ∞ אם אין מסלול כזה. מסלול שארכו שווה למרחק הוא מסלול קצר ביותר.■

<u>אי שוויון הקשת</u>: לכל צומת s וקשת (u,v) מתקיים $dist(s,v) \le dist(s,u)+1$ (למה?)

מסלול מ s ל- v שאורכו $dist(s,v)$ יקרא "מסלול קצר ביותר מ- s ל v ".

בעץ חיפוש לרחב T, המרחק מ- s לכל צומת ב-T הוא המרחק ביניהם בגרף המקורי.

האלגוריתם בונה את העץ ב"שכבות". שכבה i מכילה את כל הצמתים שמרחקם מ-s הוא i:

```
Layer(0) := {s};   i := 0;
While Layer(i) ≠ Φ
        Layer(i+1) := Φ;
        While there is an edge (u,v) s.t. u ∈ Layer(i) & v ∉ ⋃_{k≤i} Layer(k)
                Layer(i+1) := Layer(i+1) ⋃ {v};  Parent(v) := u;
        i := i+1;
```

מימוש BFS בעזרת תור

התור (שיסומן Q) מכיל בהתחלה רק את s.

כל זמן שהתור לא ריק, מוציאים את הצומת u שבראש התור, מכניסים לסוף התור את כל שכניו של u שעוד לא התגלו, ו-u נקבע כאביהם של השכנים האלו בעץ ה BFS שנבנה.

לכל צמת v מוגדרים שני שדות: המרחק של v מ- s – $d(v)$, והאב של v בעץ ה-BFS– $parent[v]$. $adj(v)$ מסמן את קבוצת השכנים של v.

האלגוריתם:

BFS(G,s):

```
for any u in V do
        d(u):= ∞;
        parent(u) := nil;
Q :=[s]; d(s):=0;
while Q is not empty do
        u := dequeue(Q);
        for each v in adj(u) do
                if d(v) = ∞ then
                        d(v):= d(u) + 1;
                        parent(v) := u;
                        enqueue(Q,v);
```

נגיד שצומת x ***גילה*** את צומת y אם התבצעה הפקודה $d(y):=d(x)+1$

זמן ריצה סיבוכיות האתחול היא $O(V)$. כל צומת מוכנס ל- Q לכל היותר פעם אחת (למה?).

כאשר צומת u מוכנס לתור, האלגוריתם מבצע מספר קבוע של פעולות לכל צומת ב- $adj(u)$.

סה"כ הזמן הנדרש לבצוע פעולות אלו במהלך האלגוריתם הוא $O\left(\sum_u |adj(u)|\right)=O(E)$.

מכאן שהסיבוכיות הכוללת היא $O(V+E)$. במיוחד האלגוריתם תמיד עוצר, ולכן כל צומת שמוכנס לתור גם מוצא ממנו (כי בזמן העצירה התור ריק).

הוכחת נכונות

נראה כי BFS מוצא את המרחק הקצר ביותר (בקשתות) לכל צומת שהוא בן הגעה מ- s .

נסמן ב- $dist(s,v)$ את המרחק מ- s ל- v . טענה בסיסית שנשתמש בה היא:

<u>למה 1.1:</u> בכל שלב בביצוע האלגוריתם מתקיים לכל צומת v אי השוויון $d(v)\ge dist(s,v)$.

<u>הוכחה:</u> באינדוקציה על מספר פעולות העדכון של ערכי $d(v)$ שהאלגוריתם מבצע (לאחר שנקבעים הערכים לראשונה).

<u>בסיס</u>: אחרי האתחול $d(s)=0=dist(s,s)$ ולגבי שאר הצמתים $d(v)=\infty\ge dist(s,v)$.

<u>צעד האינדוקציה</u>: נניח הטענה נכונה אחרי k פעולות עדכון, ותהי $d(y):=d(x)+1$ פעולת העדכון ה k+1.

לגבי כל צומת u למעט y הטענה נכונה מהנחת האינדוקציה (כי ערך $d(u)$ לא השתנה).

לגבי y מתקיים לאחר העדכון: $d(y)=d(x)+1\ge dist(s,x)+1\ge dist(s,y)$

כאשר אי השוויון הראשון נובע מהנחת האינדוקציה והשני נובע מאי שוויון הקשת.■

<u>למה 1.2:</u> . לכל זוג צמתים x, y , כך ש $dist(s,x)<dist(s,y)<\infty$ x מוכנס ל-Q לפני y.

<u>הוכחה:</u> ראשית נשים לב שמלמה 1.0, שני הצמתים נמצאים בעץ החיפוש לאחר ביצוע האלגוריתם, ומכאן ששניהם מוכנסים תור.

הוכחה באינדוקציה על $dist(s,x)$. בסיס האינדוקציה $dist(s,x)=0$: הטענה מתקיימת כי s הוא היחיד המקיים השוויון, והוא מוכנס לתור בתחילת האלגוריתם.

נניח שהטענה נכונה עבור k, ויהי x המקיים $dist(x)=k+1$ . .

יש לא שכן x' *המקיים* $dist(s,x')=k$ (למשל הצומת הקודם ל x במסלול קצר ביותר מ s ל x). x הוכנס לתור לכל המאוחר כאשר x' היה בראש התור.

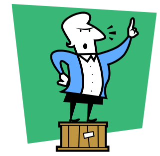
מצד שני, יהי y' הצומת שהכניס את y לתור. y' מקיים $dist(s,y')\ge dist(s,y)-1>k$ (אי שוויון הקשת). לכן, מהנחת האינדוקציה, x' הוכנס לתור לפני y'. ולכן x' מגיע לראש התור לפני y'. מכאן שסדר הפעולות הוא: x מוכנס לתור (לכל המאוחר כאשר x' היה בראש התור) ← y' בראש התור ← y מוכנס לתור, ובמיוחד x הוכנס לפני y.■

משפט 1.3: כאשר מריצים את BFS(G,s) על גרף G עם צומת מקור s, בסיום ההרצה לכל צומת v מתקיים: $d(v)=dist(s,v)$. כלומר, קבוצת הקשתות $(parent(v),v)$ מגדירה עץ BFS של G עם מקור s.

הוכחה: מלמה 1.0 נובע כי קבוצת הקשתות הזו היא עץ חיפוש ששרשו s. מלמה 1.1 נובע כי לאורך ביצוע האלגוריתם $d(v)\ge dist(s,v)$ לכל צומת v. נותר להוכיח שבסיום האלגוריתם מתקיים אי השוויון ההפוך $d(v)\le dist(s,v)$. אי שוויון זה טריביאלי אם $dist(s,v)=\infty$.

נניח בשלילה שקיים y עבורו בסיום האלגוריתם $d(y)>dist(s,y)$. אז קיים צומת y כזה עבורו $dist(s,y)=k$ עבור k מינימאלי אפשרי (שימו לב ש-$k>0$).

מלמה 1.2 נובע שכל הצמתים x עבורם $dist(s,x)\le k-1$ הוכנסו לתור לפני ש-y הוכנס אליו. לפחות אחד מצמתים אלו הוא שכן של y (למשל שכן על מסלול קצר ביותר מ-s ל-y). מבין כל הצמתים האלו, יהי x_0 השכן הראשון של y שהוכנס לתור. כאשר בוצעה לולאת ה- while עבור x_0, התבצעה פעולת העדכון $d(y):=d(x_0)+1$ והצומת y הוכנס לתור. בזמן זה התקיים $d(y)=d(x_0)+1\le k=dist(s,y)$. כמו כן ערכו של $d(y)$ לא השתנה יותר עד סיום האלגוריתם – סתירה להנחת השלילה.■



הרצאה מס' 2

חיפוש לעומק (DFS) Depth First Search -

חיפוש לעומק (DFS) הוא אלגוריתם חיפוש שהוגדר בצורתו הנוכחית על ידי Tarjan ו Hopcroft ב 1972, על סמך טכניקה ל"הליכה במבוך" מ1872 של Tremaux.

עבור גרף (מכוון או לא מכוון) $G=(V,E)$, הרצת DFS על G תחזיר יער $F=(V,E')$ של עצי חיפוש זרים בצמתים. היער F מכיל את כל צמתי הגרף ו $E' \subseteq E$.

טכניקת האלגוריתם: "התקדמות לעומק" - כאשר נבקר בצומת u , אם יש קשת (u,v) לצומת v שעוד לא נתגלה – נקבע ש u הוא אב של v ביער F, נחצה את הקשת ונמשיך את החיפוש מ-v . אם אין קשת כזו ו u איננו השורש של העץ הנוכחי, ניסוג לאביו של u.

הצומת u שממנו נעשה החיפוש נקרא "מרכז הפעילות".

DFS – מימוש על ידי מחסנית

<u>קלט</u>: גרף מכוון או לא מכוון $G=(V,E)$

<u>פלט</u>: יער DFS של G וגם לכל $v \in V$ זמן הגילוי של v.

סימון: $k[v]$ זמן הגילוי של v, ו-$p[v]$ – האב של v ביער ה DFS.

```
for all v in V
        k[v] := 0 ;  p[v] := nil;
i := 0;
while there is a vertex s with k[s] = 0
        STACK := [s]; i := i+1;  k[s] := i;
        While STACK ≠ Ø
                u:= head(STACK) ; {u is the "center of activity"}
                if there is an edge (u,v) s.t. k[v]=0
                        i := i+1; k[v] := i; p[v] := u ;
                        push(STACK,v) ;
                else
                        pop(STACK);
```

הגרף המכוון $F=(V,E')$ שנוצר על ידי הרצת האלגוריתם הוא הגרף המורכב מהקשתות $(p(u),u)$. כעת נוכיח מספר תכונות של האלגוריתם, ובמיוחד שגרף זה הוא יער מכוון המכיל את כל צמתי הגרף.

למה 2.1: א. כל צומת מוכנס לכל היותר פעם אחת למחסנית.

ב. אם האלגוריתם מסתיים כל צומת שמוכנס למחסנית מוצא ממנה.

הוכחה: א. צומת u מוכנס רק אם $k(u)=0$, ולאחר שהוכנס $k(u)\neq 0$. ב. נובע מכך שבסיום האלגוריתם המחסנית ריקה. ■

מסקנה מלמה 2.1: האלגוריתם מסתיים לאחר לכל היותר $2V$ איטרציות של לולאת הwhile הפנימית (כי בכל איטרציה מוכנס או מוצא צומת מהמחסנית). סבוכיות הזמן שלו היא $O(V+E)$: $O(V)$ פעולות pop ו push, ו$O(E)$ לצורך סריקת כל שכניו של כל צומת שהוכנס למחסנית.

למה 2.2: יהי $STACK=[s=u_0,\ldots,u_k]$ תכן המחסנית בשלב מסויים בביצוע האלגוריתם. אז לכל $i=0,\ldots,k-1$, מתקיים $u_i=p[u_{i+1}]$. (משמעות: u_i הוא אביו של u_{i+1} ביער הDFS).

הוכחה: אינדוקציה על מספר פעולות push ו pop שמתבצעות במהלך האלגוריתם.■

משמעות למה 2.2 היא שבכל שלב באלגוריתם, סדרת הצמתים הנמצאים במחסנית מהווה מסלול מכוון המתחיל בשורש העץ הנוכחי ומסתיים בצומת האחרון שהוכנס למחסנית.

למה 2.3: תהי U קבוצת הצמתים שהוכנסו למחסנית, ותהי F קבוצת הקשתות $\{(p(u),u)\}$. אז $U=V$ והגרף (V,F) הוא יער מכוון המכיל את כל צמתי הגרף.

הוכחה: $U=V$ משום שהאלגוריתם עוצר רק לאחר שכל הצמתים הוכנסו למחסנית.

תהי X קבוצת הצמתים המוכנסת למחסנית באיטרציה אחת של לולאת הwhile החיצונית (שבתחילתה צומת s מוכנס למחסנית). (א) בגרף F יש מסלול מכוון מs לכל צומת ב X (נובע מלמה 2.2). (ב) לs יש דרגת כניסה 0 ולכל צומת אחר בX יש דרגת כניסה 1. תנאים אלו מגדירים עץ מכוון ששרשו s המכיל את כל צמתי X. מאחר וכל צומת מוכנס למחסנית לכל היותר פעם אחת, העצים המתקבלים הם זרים בצמתים.■

למה 2.4: v הוא צאצא של u ביער $F \leftrightarrow v$ הוכנס למחסנית כאשר המחסנית הכילה את u.

הוכחה: תהי $P=(s=v_0,v_1,\ldots,v_k=v)$ סדרת הצמתים במחסנית כאשר v הוכנס למחסנית.

מלמות 2.2 ו 2.3 נובע ש s הוא השורש של העץ T המכיל את v, וP הוא המסלול היחיד ב T מ s ל v. לכן u אב קדמון של v אםם u נמצא ב P.■

בדיון להלן, צומת ייקרא ***לבן*** אם עדיין לא הוכנס למחסנית.

למה 2.5: אם כאשר u הוכנס למחסנית היה בגרף הקלט G מסלול $(u=u_0,u_1,..,u_k=v)$ של צמתים לבנים, אז v הוא צאצא של u ביער F.

הוכחה: על סמך למה 2.4, יספיק להוכיח שכל צמת במסלול הנ"ל הוכנס למחסנית לפני ש u הוצא מהמחסנית. את זה מוכיחים באינדוקציה על סדר הצמתים במסלול, על סמך העובדה ש צומת מוצא מהמחסנית רק לאחר שכל שכניו הוכנסו למחסנית (בגלל הפסוק if…else… באלגוריתם).■

למה 2.6 כאשר G גרף לא מכוון, יער הDFS $F=(V,E')$ המוחזר על ידי האלגוריתם מקיים את <u>תכונת הקשת האחורית</u>: לכל קשת (x,y) ב E, x הוא צאצא של y ב F או להפך.

הוכחה: נניח ב.ה.כ. ש x התגלה (הוכנס למחסנית) לפני y. אז הקשת (x,y) מקיימת את ההנחה של למה 2.5 ולכן y הוא צאצא של x בF.■

הגדרה: עץ פורש מושרש של גרף לא מכוון וקשיר G המקיים את תכונת הקשת האחורית נקרא ***עץ DFS* של *G***.

סיווג קשתות DFS על גרף מכוון

יער DFS על גרף מכוון $G=(V,E)$ מגדיר ארבעה סוגי קשתות בE:

- <u>קשתות עץ</u>: (u,v) קשת עץ אםם $u = p(v)$.
- <u>קשתות אחוריות</u>: קשת (u,v) שמחברת את u לאב קדמון של u בעץ DFS (לולאה עצמית תחשב כקשת אחורית).
- <u>קשתות קדמיות</u>: קשתות (u,v) מ- u לצאצא של u בעץ DFS, שאינן קשתות עץ.
- <u>קשותות חוצות (cross)</u>: כל הקשתות האחרות ב– G: קשתות בין צמתים באותו עץ DFS ללא יחס אב קדמון – צאצא, או קשתות בין עצי DFS שונים.

הרצאה מס' 3

שימוש בDFS: מציאת רכיבים קשרים היטב בגרף מכוון

הגדרה: רכיב קשיר היטב (רק"ה בקיצור) בגרף מכוון $G=(V,E)$ הוא קב' מקסימלית של צמתים C מ- V כך שלכל זוג צמתים u,v ב- C יש מסלולים מכוונים מ- v ל- u ומ- u ל- v. מהמקסימליות של C נובע שכל מסלול מכוון בין זוג צמתים בC מכיל רק צמתים מ C.

דוגמה: בגרף מכוון אציקלי (חסר מעגלים מכוונים) כל צומת הוא רק"ה.

נגדיר יחס "חברות" בין צמתים: u חבר של v אם u ו v נמצאים באותו רק"ה. יחס החברות הוא יחס שקילות (רפלקסיבי, סימטרי, טרנסיטיבי). מכאן שהוא מחלק את צמתי הגרף לקבוצות זרות ומשלימות. כל קבוצה כזו משרה רק"ה של הגרף.

הגדרה: נניח שהרק"ה ב- G הם $C_1,\ldots,C_k$ (מן הנכתב לעיל $\bigcup C_i=V, C_i\cap C_j=\Phi$).

גרף הרכיבים קשירים היטב של G –, שיסומן $G^*=(V^*,E^*)$, מוגדר באופן הבא.

$V^*=\{v_1,\ldots,v_k\}$

$E^*=\{(v_i,v_j):$ יש בE קשת שיוצאת מצומת ב- C_i אל צומת ב- C_j . $\}$

מטרתנו למצוא אלגוריתם יעיל למציאת גרף הרק"ה של גרף נתון G. הרעיון הוא לקבוע סדר (פרמוטציה) π על הצמתים של G, כך שכאשר מריצים DFS על G וסדר הכנסת הצמתים למחסנית ריקה (בלולאת ה while החיצונית) נקבע על פי π, יוחזרו כל הרק"ה של G.

הדיון להלן מתייחס לגרף מכוון $G=(V,E)$ כלשהו.

הגדרה: הגרף ההפכי של גרף $G=(V,E)$, הוא הגרף G^T המתקבל ע"י הפיכת כיווני הקשתות ב G, כלומר: $G^T=(V,E^T)$ כאשר: $E^T=\{(v,u):(u,v)\in E\}$.

טענה 3.1: ל G^T ולG יש אותם רק"ה.

למה 3.2: גרף הרק"ה *G הוא אציקלי.

הוכחה: בשלילה. אם קיים ב*G מעגל המכיל לפחות שני צמתים המייצגים שני רק"ה C ו

'C, אז קל לראות שקיים מסלול מכוון בין כל זוג צמתים ב $C \cup C'$ (בכל כיוון), ולכן C ו 'C מוכלים באותו רק"ה – סתירה להנחה שהם שני רק"ה שונים. ■

<u>למה 3.3:</u> בכל ריצת אלגוריתם DFS, כל רכיב קשיר היטב, C, מוכל במלואו באחד מעצי ה DFS שהאלגוריתם מחזיר.

<u>הוכחה:</u> ע"ס למה 2.5, הצומת הראשון מ C שמוכנס למחסנית יהיה אב קדמון בעץ הDFS של כל הצמתים האחרים מ C.■

כנכתב לעיל, הפלט של ריצת DFS על גרף G תלוי בפרמוטציה של הצמתים ב V בה משתמש האלגוריתם. כעת נאפיין פרמוטציות המבטיחות שכל איטרציה של האלגוריתם תחזיר רק"ה יחיד.

<u>הגדרה:</u> רק"ה C ***גדול*** מרק"ה 'C (נסמן $C > C'$) אם קיים מסלול מכוון מצמת ב C לצמת ב 'C.

תהי $\pi = (\pi_1, ..., \pi_n)$ פרמוטציה של V. נגיד **שרק"ה C מופיע ב π לפני רק"ה 'C** אם הצומת הראשון של C מופיע ב π לפני הצומת הראשון של 'C .

<u>הגדרה:</u> פרמוטציה π של צמתי V תיקרא ***כשרה*** אם היא מקיימת את התנאי הבא: לכל זוג רק"ה C, 'C, אם $C<C'$ אז C מופיע ב π לפני C'.

דוגמה: אם הגרף הוא מסלול מכוון $(1 \to 2 \to ... \to n)$, אז כל רק"ה מכיל צומת בודד והפרמוטציה הכשרה היחידה היא $(n, n-1, ..., 2, 1)$ (n ראשון, 1 אחרון).

<u>למה 3.4:</u> נניח שהרצנו אלגוריתם DFS על גרף מכוון, כך שסדר הצמתים בביצוע האלגוריתם הוא פרמוטציה כשרה π. אז כל ביצוע של לולאת הwhile החיצונית מחזיר עץ המכיל את הצמתים של רק"ה יחיד של G.

<u>הוכחה</u>: יהי s הצומת הראשון שהוכנס למחסנית בביצוע של לולאת הwhile החיצונית, יהי C הרק"ה של s, ויהי T עץ ה DFS המוחזר ע"י ביצוע הלולאה. מלמה 3.3 נובע שכל צמתי C נמצאים ב T, לכן יספיק להוכיח שאף צומת אחר לא נמצא ב T: מאחר ו π פרמוטציה כשרה, כל צומת u בר השגה מ s שאינו ב C שייך לרק"ה שמופיע לפני C, ולכן (ע"ס למה 3.3) הוכנס למחסנית לפני s, ולכן (ע"ס למה 2.4) אינו צאצא של s, ולא מופיע ב T.■

<u>מסקנה</u> כדי למצוא את כל הרק"ה של גרף G, מספיק למצא פרמוטציה כשרה של צמתי G ולהריץ עליה DFS.

האלגוריתם שנציג משתמש ב DFS כדי למצא פרמוטציה כשרה של הגרף ההפכי G^T, שעל סמך טענה 3.1 מורכב מאותם רק"ה של G. האלגוריתם מתבסס על הלמה הבאה:

למה 3.5 נניח שהורץ DFS על G. לכל צומת u נגדיר f[u] (זמן הסיום של u) כזמן בו הוצא u מהמחסנית. הפרמוטציה π המתקבלת על ידי סידור הצמתים לפי סדר יורד של f[u] היא פרמוטציה כשרה של G^T.

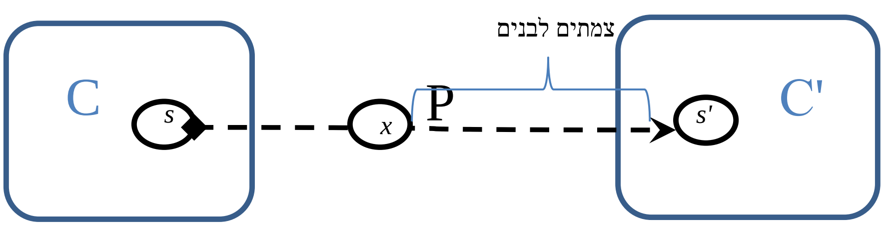
הוכחה: ל G ול G^T יש אותם רק"ה אך הסדר ביניהם מתהפך. לכן צריך להוכיח כי אם ב G מתקיים ש C > C' אז הצמת הראשון מ C מופיע ב π לפני הצומת הראשון מ C'.

יהיו s ו s' הצמתים הראשונים מ C,C' שהוכנסו למחסנית בהתאמה. אז הם גם הצמתים האחרונים מ C, C' בהתאמה שהוצאו מהמחסנית. על פי הגדרת π, צ.ל. כי f[s] > f[s'], כלומר ש s הוצא מהמחסנית ***אחרי*** s'.

יהי P מסלול כלשהו מ s ל s' (קיים מסלול כזה כי $C > C'$). יהי x הצומת הראשון ב P שהוכנס למחסנית. נבדיל בין שני מקרים:

א. $x=s$. אז כאשר s הוכנס למחסנית כל שאר הצמתים ב P היו לבנים ולכן s' הוכנס והוצא מהמחסנית לפני ש s הוצא ממנה (בדומה להוכחה של למה 2.5).

ב. $x \neq s$. במקרה זה x איננו ב C (כי s הוא הצומת הראשון מ C שהוכנס למחסנית, ו x הוכנס לפני s), ולכן אין מסלול מכוון מ x ל s, בעוד שיש מסלול מכוון של צמתים לבנים מ x ל s' (ראו ציור). מכאן ש s' הוכנס והוצא מהמחסנית לפני ש s הוכנס למחסנית (קל וחומר לפני ש s הוצא מהמחסנית).■



האלגוריתם המשתמע מהנכתב לעיל הוא:

אלגוריתם למציאת רכיבים קשירים היטב

Strongly_Connected_Components(G):

```
(1) Call DFS(G) to compute f[u] for  all u in V

(2) Compute G^T

(3) Call DFS(G^T) on the vertices ordered in decreasing order of
f[u] (as computed in (1))

(4) output the vertices in each DFS tree generated in (3)
```

זמן ריצה

זמן הריצה הוא $O(V+E)$ (סיבוכיות DFS). שימו לב שלצורך שלב (3) צריך לשמור את צמתי G בסדר יורד של ערכי $f[u]$ בזמן שהם מחושבים.

נכונות

תהי π הפרמוטציה של הצמתים שנקבעת ע"י סדר יורד של $f[v]$. מלמה 3.5 נובע ש π היא כשרה עבור G^T, ולכן האלגוריתם מחזיר את הרכיבים הקשירים של G^T (למה 3.4), שהם הרכיבים הקשירים של G (טענה 3.1).

כדי לבנות את גרף הרק"ה של G אפשר לעבור על הקשתות של G, ואם יש קשת מכוונת מצומת בC_i לצומת ב C_j, להוסיף קשת (v_i,v_j) ל E^*.

הרצאה מס' 3 אלטרנטיבית

מציאת צמתי הפרדה ורכיבים אי פריקים

ההרצאה עוסקת באפליקציה נוספת של DFS, הפעם לגרפים לא מכוונים. לאורך כל ההרצאה נניח ש $G=(V,E)$ הוא גרף לא מכוון וקשיר. נפתח במספר הגדרות.

צומת הפרדה (separating vertex) : יהי $G=(V,E)$ גרף נתון. $v\in V$ הוא צומת הפרדה אם קיימים צמתים u,w השונים מ v, כך שכל מסלול מ u ל w עובר ב v.

גרף אי פריק: גרף לא מכוון G הוא ***אי פריק*** אם הוא קשיר ואינו מכיל צומת הפרדה.

תת גרף מושרה: תהי $U\subseteq V$. $G(U)$, תת הגרף המושרה ע"י U, מוגדר ע"י:

$$G(U)=(U,E'),\ \text{where } E'=\{(x,y): x,y\in U,(x,y)\in E\}$$

רכיב אי פריק (irreducible component): רכיב אי פריק של G הוא תת גרף מושרה $G(U)$ שהוא אי פריק, אבל כל תת גרף של G המכיל אותו ממש הוא פריק.

גרף הרכיבים האי פריקים של G: $H=(X,E')$ מוגדר על ידי:

$$X=\{u: u \text{ is a separating vertex}\}\bigcup\{C: C \text{ is an irreducible component}\}$$

$$E'=\{(u,C): u\in C\}$$

הטענה הבאה ממחישה מהו מבנה גרף הרכיבים האי פריקים:

__טענה__: H, גרף הרכיבים האי פריקים של G, הוא עץ לא מכוון.

תקציר ההוכחה : א. H קשיר משום שG קשיר (הוכיחו..) ב. צומת הפרדה אינו יכול להיות על מעגל ב H, ולכן בH אין מעגלים. מכאן ש H גרף קשיר חסר מעגלים, כלומר עץ.■

בהמשך ההרצאה נציג אלגוריתם המתבסס על DFS למציאת צמתי הפרדה ורכיבים אי פריקים, ולבניית גרף הרכיבים האי פריקים.

הגדרה: יהי T עץ DFS של $G=(V,E)$ ו $u\in V$. קשת $(v,w)\in E$ ***עוקפת*** את u אם v הוא צאצא אמיתי של u וw הוא אב קדמון אמיתי של u בעץ T. (שימו לב שקשת עוקפת היא בהכרח קשת אחורית).

הגדרה: יהי $T=(V,E)$ עץ מכוון ויהי $v\in V$. $T(v)$ הוא תת העץ של T המושרה על ידי v וכל צאצאיו של v ב T. שימו לב ש v הוא השורש של $T(v)$.

__למה 3.6__: יהי T עץ DFS של G עם שורש s. צמת u הוא צמת הפרדה ב G אםם מתקיים אחד מהתנאים הבאים:

1. $u=s$ ויש לו יותר מבן אחד.
2. $u\neq s$, ויש ל u בן v כך שלא קיימת קשת היוצאת מצומת ב$T(v)$ ועוקפת את u. במקרה זה u מפריד בין s לv .

__הוכחה:__ 1. ← : מאחר ובעץ DFS אין קשתות חוצות, כל מסלול בין שני בנים שונים של s

חייב לעבור דרך s.

$\rightarrow$: אם לs יש רק בן אחד, הרי קיים מסלול בין כל שני צמתים השונים מ s המשתמש רק בקשתות עץ ואינו עובר דרך s.

2. $\leftarrow$:אם לא קיימת קשת אחורית מצומת ב$T(v)$ העוקפת את u, כל מסלול מ v ל s חייב לעבור דרך u (u הינו הצומת הראשון במסלול כזה שאיננו שייך ל$\mathrm{T}(v)$).

$\rightarrow$: אם לכל בן v של u קיימת קשת אחורית כנ"ל, הרי שניתן להגיע מכל צמת $a \in V \setminus \{u\}$ לשרש s בלי לעבור דרך u, ולכן בין כל שני צמתים יש מסלול שאינו עובר דרך u. לכן u אינו צומת הפרדה.■

כדי לבדוק אם תנאי 2 של למה 3.6 מתקיים, מחשבים לכל צומת u את $\mathrm{L}(u)$, ה lowpoint של u, המוגדר להלן:

$$\mathrm{L}(u) = \min\{k(v) : [v = u] \text{ OR } [\text{ there is a back edge from a vertex in } T(u) \text{ to } v]\}$$

<u>למה 3.7</u>: יהי u צמת שאינו שרש ויהי v בן של u. אזי:

[קיימת קשת היוצאת מצומת ב $\mathrm{T}(v)$ ועוקפת את u] אםם [$\mathrm{L}(v) < k(u)$].

<u>הוכחה:</u> נניח שקיימת קשת (x, y) היוצאת מצומת x ב $\mathrm{T}(v)$ ועוקפת את u. אז y הוא אב קדמון של u. מהגדרת $\mathrm{L}(u)$ מקבלים: $L(u) \le k(y) < k(u)$.

מצד שני, אם $\mathrm{L}(v) < k(u)$ אז קיימת קשת מצמת ב $\mathrm{T}(v)$ לצמת y כך ש $k(y) < k(u)$. מאחר והקשת היא קשת אחורית, y בהכרח אב קדמון של u. ■

<u>למה 3.8</u>: אם u איננו השרש s, אז [u צומת הפרדה] $\leftarrow\rightarrow$ [קיים בן v של u כך ש $L(v) \ge k(u)$].

הוכחה: מלמה 3.7, התנאי בצד שמאל אומר שלא קיימת קשת היוצאת מ $\mathrm{T}(v)$ ועוקפת את u. ע"ס למה 3.6 (סעיף 2) זה שקול לכך ש u צמת הפרדה (המפריד בין s ל v). ■

<u>למה 3.9</u>: לכל צמת u מתקיים השוויון:

$$L(u) = \min\left[\{k(u)\} \bigcup \{k(v) : (u,v) \text{ is a back edge}\} \bigcup \{L(v) : v \text{ is a child of } u \text{ in } T\}\right]$$

<u>תקציר הוכחה:</u> על פי ההגדרה של $\mathrm{L}(u)$ (מומלץ לוודא שאתם יודעים להשלים את הפרטים החסרים...).■

אלגוריתם למציאת צמתי הפרדה ע"י DFS

<u>קלט</u>: גרף לא מכוון וקשיר $G=(V,E)$ וצומת s.

<u>פלט</u>: עץ DFS של G עם שורש s, וצמתי ההפרדה של G.

האלגוריתם משתמש במשתנים הבאים: $k[v]$- זמן הגילוי של v, $p[v]$ – האב של v, $L(v)$ – ערך ה lowpoint של v.

```
for all v in V
      k[v] := 0, p[v] := nil, L(v):=∞
for all e in E
      mark e "new"
STACK:= {s}, i:= 1,  k[s]:= 1, L[s]:= 1;
While STACK ≠ Ø
      u:= head(STACK)
      if there is an edge e=(u,v) marked "new"
            mark e "used"
            if k[v]=0
                  i:=i+1, k[v]:=i, L(v):=i, p[v]:=u,
                  push(STACK,v)
            else  \* k[v]>0, i.e. (u,v) is back-edge
      (a)         L[u]:= min{L[u],k[v]}
      else \* all edges leaving u are "used"
            if  (u≠s)
                  if ( p[u]≠s & L[u]≥k[p[u]] )
                        mark p[u] as separating vertex
                  if ( p[u]=s & s has a "new" edge )
                        mark s as separating vertex
      (b)         L[p[u]] := min{L[p[u]],L[u]}
            pop(STACK)
```

הערה: אם v הוא צומת הפרדה השייך ל k רכיבים אי פריקים, האלגוריתם יגדיר את v כצומת הפרדה k-1 פעמים.

למה 3.10: האלגוריתם לעיל מוצא את צמתי ההפרדה של G.

תקציר הוכחה: מוכיחים באינדוקציה על מספר פעולות המחסנית את הטענה הבאה:

העדכונים בשורות (a), (b) מבטיחים שלכל צומת u, כאשר u מוצא מהמחסנית הערך L(u) שחושב על ידי האלגוריתם הוא lowpoint(u) . הוכחת צעד האינדוקציה משתמשת בלמה 3.9. מכאן נובע (על סמך למה 3.6(1) ולמה 3.8) שצמת מסומן כseparating vertex אםם הוא צומת הפרדה. ■

כעת נציג אלגוריתם המחזיר רכיבים אי פריקים בזמן O(V) על ידי שימוש בפלט של האלגוריתם למציאת צמתי הפרדה.

האלגוריתם סורק את עץ הDFS שהוחזר, ומכניס למחסנית כל צמת שמתגלה (בדומה ל DFS).

כאשר האלגוריתם מתקדם לצמת v בעץ, הוא בודק אם מתקיים אחד התנאים על פיהם $p(v)$ הוא צומת הפרדה, ואם $p(v)$ צמת הפרדה הוא מסמן את v.

כאשר האלגוריתם נסוג מצומת מסומן v הוא: א. מוציא מהמחסנית את v וכל הצמתים שמעליו; ב. רושם רכיב אי פריק המכיל את $p(v)$ ואת כל הצמתים שהוצאו מהמחסנית.

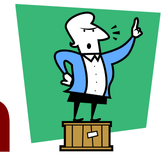
אלגוריתם לרכיבים אי פריקים

קלט: עץ מכוון T עם שרש s, שהוא עץ DFS של G , ולכל צומת u הערכים k[u] (זמן הגילוי של u) וL[u] (ה lowpoint של u) שהוחזרו ע"י האלגוריתם למציאת צמתי הפרדה.
פלט: הרכיבים האי פריקים של G.

האלגוריתם (בעמוד הבא) משתמש במחסנית STK לצורך בנית הרכיבים האי פריקים. p[u] הוא האב של u בT. צמת v מסומן *special* אם p[v] הוא צמת המפריד את v וכל צאצאיו בT משאר צמתי הגרף.

```
mark all vertices of T "new"
STK :={s}, u:=s,
while STK≠∅
      if u has a "new" child v
             mark v "used"; push(STK, v)
                  if u≠s & L[v]≥k[u] mark v "special"
                  if u=s and s has another "new" child mark v "special"
                  u:=v
      else \* all children of u are used
            if u≠s
                  if u is marked "special"
                        Let C be u and all the vertices above u in STK;
                        Remove C from STK and declare C∪{p[u]} an
                        irreducible component
                   u:=p[u]
            else \* u=s
                  declare all the vertices in STK irreducible
                  component, and remove them from STK.
```

כדי להחזיר את גרף הרכיבים האי פריקים H, ניתן לחשב (בזמן לינארי) את הקשתות ב H מרשימת הצמתים ברכיבים האי פריקים.



הרצאה מס' 4

עצים פורשים מינימאלים (Minimum Spanning Trees)

גרף ממושקל (weighted graph) הוא גרף שבו לכל קשת e יש משקל $w(e)$.

כזכור, עץ פורש של גרף לא מכוון וקשיר G=(V,E) הוא תת גרף של G שהוא עץ המכיל את כל צמתי G. משקל של עץ פורש של גרף ממושקל הוא סכום משקלי קשתותיו:

$$w(T)=\sum_{e\in T} w(e).$$

בעית עפ"מ:

קלט: גרף ממושקל, פשוט, לא מכוון וקשיר G=(V,E).

פלט: עץ פורש של G בעל משקל מינימאלי אפשרי.

אלגוריתם גנרי למציאת עפ"מ

הגדרה: חתך (cut) בגרף $G=(V,E)$ הוא חלוקה של V לשתי קבוצות זרות ולא ריקות X ו- $V\setminus X$. קשתות החתך הן הקשתות ב E שקצה אחד שלהן ב X והקצה השני ב $V\backslash X$.

להלן שני קשרים בסיסיים בין עצים פורשים, מעגלים וחתכים, עליהם מסתמך האלגוריתם:

למה 4.1: יהי T עץ פורש של G.

(1) הוספת קשת ל T תיצור ב T מעגל (פשוט) יחיד.

(2) הסרת קשת e מ T מחלקת את T לשני תתי עצים, שצומתיהם מהווים חלוקה של צמתי הגרף. קשתות החתך המוגדר על ידי חלוקה זו מכילות את e ואינן מכילות קשת אחרת של T.

הוכחה: תרגיל פשוט לבית. ■

כעת נציג אלגוריתם גנרי לעפ"מ. האלגוריתם בונה את העפ"מ T על ידי סדרה של פעולות: בכל פעולה מוסיפים קשת "קלה" לT (כלל כחול) או פוסלים קשת "כבדה" מלהיות ב T (כלל אדום).

האלגוריתם הגנרי לעפ"מ Generic MST Algorithm

```
הכלל האדום (לקשתות כבדות):

תנאי: קיים ב G מעגל C חסר קשתות אדומות, וe היא קשת לא צבועה בעלת
משקל מקסימאלי (מבין הקשתות הלא צבועות) ב C.

פעולה: צבע את e באדום.

הכלל הכחול (לקשתות קלות):

תנאי: קיים ב  G חתך D חסר קשתות כחולות, וe היא קשת לא צבועה בעלת
משקל מינימאלי (מבין הקשתות הלא צבועות) ב D.

פעולה: צבע את e בכחול.

האלגוריתם הגנרי:

בחר קשת כלשהי עליה ניתן להפעיל את הכלל האדום או הכלל הכחול, והפעל
כלל זה. עצור כאשר אין בG קשת כזו.
```

למה 4.2: האלגוריתם הגנרי מסתיים לאחר E צעדים, כאשר כל הקשתות צבועות.

הוכחה: מאחר ובכל צעד נצבעת קשת, האלגוריתם מסתיים לאחר לכל היותר E צעדים. כדי להוכיח שבסיום האלגוריתם כל הקשתות צבועות, נוכיח כי כל זמן שיש בגרף קשת לא צבועה, ניתן לצבוע קשת על פי אחד הכללים.

נניח כי $e=(u,v)$ היא קשת לא צבועה. נבחין בין 2 מקרים:

- קיים מסלול כחול בין u ל v. יהי C המעגל הנוצר על ידי הוספת e למסלול. ניתן לצבוע את e על פי הכלל האדום.
- לא קיים מסלול כחול מ u ל v. תהי X קבוצת הצמתים שיש אליהם מסלול כחול מ u. החתך המוגדר על ידי X ו $V \setminus X$ מכיל את e ואין בו קשתות כחולות. לכן ניתן לצבוע את אחת מקשתותיו על פי הכלל הכחול.■

כעת נוכיח שכאשר האלגוריתם מסתיים הקשתות הכחולות חייבות להיות עפ"מ.

הגדרה: גרף G שחלק מקשתותיו כחולות וחלק אחר אדומות מקיים את ***שמורת הצבע*** אםם קיים בG עפ"מ שמכיל את כל הקשתות הכחולות ואף אחת מהקשתות האדומות.

למה 4.3: בכל שלב בביצוע של האלגוריתם הגנרי על גרף לא מכוון ממושקל וקשיר G (שקשתותיו מלכתחילה אינן צבועות), הגרף G מקיים את שמורת הצבע.

הוכחה: באינדוקציה על מספר הקשתות שהאלגוריתם הגנרי צבע.

בסיס: כאשר אף קשת לא צבועה כל עפ"מ מקיים את שמורת הצבע (באופן ריק).

צעד: נוכיח שאם הטענה נכונה לאחר צביעת k-1 קשתות, היא נכונה גם לאחר צביעת הקשת ה k. יהי T העפ"מ המקיים את שמורת הצבע לאחר צעד k-1. אם בצעד ה k צובעים באדום קשת שאינה שייכת ל T או צובעים בכחול קשת השייכת לT הרי ש T מקיים את שמורת הצבע גם לאחר הצעד ה k. אחרת יש שני מקרים אפשריים:

- מקרה א: בצעד ה k צבענו בכחול קשת $e=(u,v)$ שאינה שייכת לT. נקרא לחתך עליו הופעל הכלל הכחול=D $(X,V\setminus X)$. נסתכל על המסלול בעץ T שמחבר בין הצמתים u,v. המסלול חייב להכיל קשת e' שחוצה את D, ולכן אינה כחולה. e' גם אינה אדומה (מאחר ו $e'\in T$). כלומר e' אינה צבועה. מכאן ש $w(e)\le w(e')$ (כי ע"פ הכלל הכחול e היא קשת לא צבועה בעלת משקל מינימאלי ב D). נשמיט את e' ונוסיף את e ל- T. נקבל תת גרף חדש T' שגם הוא עץ (קשיר ומכיל n-1 קשתות), ומתקיים $w(T')=w(T)-w(e')+w(e)\le w(T)$. לכן T' הוא עפ"מ, והוא מכיל את כל הקשתות הכחולות כנדרש. (שימו לב שלמעשה חייב להתקיים $w(T)=w(T')$, ולכן גם $w(e)=w(e')$).
- מקרה ב: בצעד ה k צבענו קשת $e=(u,v)$ באדום על ידי הפעלת הכלל האדום על מעגל C, ו $e\in T$. במקרה זה השמטת e מ T מחלקת את T לשני עצים זרים T_1, ו T_2 (למה 4.1 (2)). צמתי עצים אלו מהווים חלוקה של צמתי הגרף G. החתך D המוגדר ע"י חלוקה זו כולל את הקשת e ולפחות קשת נוספת מהמעגל C, שתקרא e'. הקשת e' אינה שייכת לT (למה 4.1 (2)), ולכן מהנחת האינדוקציה אינה כחולה. מאחר ו- e' שייכת למעגל Cעליו הופעל הכלל האדום, היא גם לא אדומה. לכן $w(e)\ge w(e')$. תת הגרף T' המתקבל ע"י השמטת e והוספת e' ל T הוא עץ (למה?). העץ T' מקיים $w(T')=w(T)+w(e')-w(e)\le w(T)$, ולכן $w(T)=w(T')$ ובמיוחד T' הוא עפ"מ המקיים את השמורה.■

מסקנה: כאשר האלגוריתם הגנרי מסתיים הקשתות הכחולות מהוות עפ"מ.

הוכחה: על פי למה 4.3, כאשר האלגוריתם מסתיים הקשתות הכחולות מוכלות בעפ"מ. אם הן לא מהוות עפ"מ, היה ניתן להשלימן לעפ"מ על ידי הוספת קשתות נוספות שהן בהכרח אדומות (למה?) – בסתירה לכך שהקשתות האדומות אינו מוכלות בעפ"מ.■

האלגוריתם של Prim

- מפעילים את הכלל הכחול בלבד עד שמתקבל עפ"מ.
- הכלל הכחול מופעל על החתך בין צמתי העץ שהולך ונבנה ושאר צמתי הגרף.

קלט: גרף לא מכוון קשיר ממושקל G=(V,E)

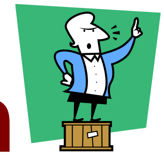
האלגוריתם של Prim

אתחול – כל הקשתות אינן צבועות; $U:=\{r\}$.

- כל עוד $U \neq V$ בצע:
  הפעל את הכלל הכחול על חתך $(U, V\setminus U)$ וצבע בכחול קשת $e=(u,v)$
  מינימאלית בחתך זה כך ש- $u\in U, v\in V\setminus U$; בצע $U:=U\cup\{v\}$.

טענה: האלגוריתם של פרים מסתיים כאשר הקשתות הכחולות הן עפ"מ.

הוכחה: כאשר האלגוריתם מסתיים, הקשתות הכחולות מהוות עץ פורש (V-1 קשתות המקשרות בין כל הצמתים ב V). ע"ס למה 4.3 זהו עץ המוכל בעפ"מ, ולכן הוא עפ"מ. ■



הרצאה מס' 5

מימוש האלגוריתם של פרים

זמן האתחול תלוי במימוש. אחריו יש לבצע V-1 איטרציות. בכל איטרציה צריך לבחור קשת $e=(u,v)$ מינימלית בחתך $(U,V\setminus U)$, לצבוע אותה בכחול, ולצרף את v ל U. לצורך זה:

- ✓ לכל צומת $v\in V\setminus U$ יוחזק מפתח $key[v]$ = המשקל המינימלי של קשת שמחברת את v לצומת ב U, ואם אין כזו אז $key[v]=\infty$.
- ✓ כאשר $key[v]\neq\infty$ נשמור קשת $(p[v],v)$ מ- v לצומת ב- U שמשקלה $key[v]$.

האלגוריתם משתמש במשתנה $Q=V\setminus U$ לשמירת הצמתים שעדיין לא צורפו לעץ.

> לכל $v\in V$ בצע $key[v]=\infty,\ p[v]=nil$
> $Q:=V,\ key[r]:=0$
> כל עוד $Q\neq\varnothing$
>   הוצא מ- Q צומת u כך ש $key(u)$ מינימאלי
>   לכל שכן v של u שמופיע ב Q:
>   אם $w(u,v)<key[v]$ אז $p[v]:=u,key[v]:=w(u,v)$

קשתות העפ"מ הן הקשתות $\{(p(u),u):u\neq r\}$.

סיבוכיות האלגוריתם תלויה במימוש של Q. נראה שני מימושים:

מימוש Q בעזרת מערך של צמתי V

א. שומרים לכל צומת v במערך את $key(v)$ ואת $p[v]$, וביט המציין אם v שייך ל Q.

ב. בכל איטרציה:

- מוצאים $v\in Q$ עבורו $key[v]$ מינימאלי, ומוציאים אותו מ Q.
  (סיבוכיות: $O(V)$ לאיטרציה, $O(V^2)$ סך הכל.)
- עבור כל קשת (v,w) כך ש $w\in Q$, מעדכנים את ערכי $key[w]$ ו $p[w]$.
  (סיבוכיות: $O(d(v))$ לאיטרציה, כלומר $O(E)$ בסך הכל.)

סיבוכיות כוללת של מימוש בעזרת מערך $O(V^2+E)=O(V^2)$.

מימוש Q בעזרת ערימה

נזכיר כי ערימה (Heap) היא עץ בינארי מאוזן (הפרש בין עומק שני עלים ≤ 1), שבו המפתח בשורש כל תת עץ קטן מכל המפתחות בתת העץ, ובפרט, השורש מכיל את המפתח המינימאלי (המפתח המינימלי יכול להופיע גם בצמתים נוספים בעץ). הפעולות שניתן לבצע על ערימה:

- בניית הערימה - $O(V)$ צעדים.
- הוצאת אבר עם מפתח מינימאלי - $O(\log V)$ צעדים

ג. Decrease Key (הקטנת ערכו של מפתח של אבר בערמה) - $O(\log V)$ צעדים.

בכל איטרציה מבוצעות הפעולות הבאות:

- מציאת $v \in Q$ עבורו $key[v]$ מינימאלי והוצאתו מ-Q. (סיבוכיות: $O(\log V)$ לאיטרציה)
- עדכון ערכי $key[w]$ ו $p[w]$ עבור כל קשת (v,w) כך ש $w \in Q$.

  (סיבוכיות: $O(d(v)\log V)$ לאיטרציה, כלומר $O(E\log V)$ בסך הכל.)

סיבוכיות כוללת של מימוש בעזרת ערימה: $O(V)+O(V\log V)+O(E\log V)=O(E\log V)$

<u>הערה</u>: אפשר לשפר עוד את הסיבוכיות על ידי שימוש בערימות פ'יבונאצ'י. פעולת הוצאת המינימום ב- $O(\log V)$ צעדים ופעולת decreaseKey ב-O(1) צעדים בממוצע (על ידי Amortized Analysis), ולכן הסיבוכיות היא $O(E+V\log V)$.

האלגוריתם של Kruskal

- מיין את הקשתות בסדר לא יורד לפי משקל.
- עבור על הרשימה הממוינת, ולכל קשת $e=(u,v)$ בצע:
  - אם יש מסלול כחול מ u ל v, צבע את e באדום,
  - אחרת צבע את e בכחול.

משפט 5.1

כל צביעת קשת באלגוריתם של Kruskal נעשית על פי הכלל הכחול או הכלל האדום של האלגוריתם הגנרי, ולכן בסוף האלגוריתם הקשתות הכחולות מהוות עפ"מ.

הוכחה

- אם יש מסלול כחול מ u ל v אז e היא קשת לא צבועה יחידה במעגל ששאר קשתותיו כחולות, ולכן צביעתה באדום נעשית על פי הכלל האדום.
- אם אין מסלול כזה, נגדיר את X_u כקבוצת הצמתים שיש אליהם מסלול כחול מ u $(u \in X_u)$. מהגדרה זו נובע שהחתך $(X_u, V \setminus X_u)$ אינו מכיל קשתות כחולות, ומכיל את הקשת e. מאחר שבעת ביצוע הצעד e היא קשת קלה ביותר בגרף שאיננה צבועה, היא ודאי קשת קלה ביותר שאינה צבועה בחתך $(X_u, V \setminus X_u)$. לכן צביעתה בכחול נעשית על פי הכלל הכחול.■

מימוש האלגוריתם של קרוסקל

- מיון הקשתות בתחילת האלגוריתם דורש $O(E \log V)$. לאחר מכן בכל צעד יש לקבוע אם יש מסלול כחול בין u לv. לצורך ביצוע צעד זה נחזיק במבנה נתונים את הקבוצות הזרות של צמתים שנמצאות באותו עץ כחול. בשלב האתחול ניצור לכל צומת v קבוצה S_v שמכילה רק אותו (יש V קבוצות כאלו, O(V) זמן אתחול). בכל איטרציה יבוצע:
- בדיקה עבור 2 קצוות הקשת $e=(u,v)$ אם $S_u = S_v$.
- אם לא, יצירת הקבוצה $S_u \cup S_v$ המייצגת את העץ שנוצר ע"י צביעת e בכחול.

פעולות אלו יבוצעו על ידי מבנה נתונים התומך בפעולות הבאות:

- find_set(v) – מחזירה את שם הקבוצה (היחידה) שמכילה את v.
- Union(u,v) – איחוד הקבוצות שמכילות את u,v בהתאמה.

שני מבני נתונים אפשריים למימוש יעיל של הפעולות האלו:

1. מערך שבו ליד כל צומת רשום שם הקבוצה המכילה אותו, ולכל קבוצה כזו רשימה מקושרת של הצמתים הנמצאים בה.

   במימוש זה כל פעם שמאחדים 2 קבוצות, מצרפים את הצמתים בקבוצה הקטנה יותר לקבוצה הגדולה יותר.

מימוש זה מבטיח זמן O(1) לכל פעולת find_set(v), וזמן כולל של O(Vlog V) לסה"כ פעולות Union(u,v) (פעולת איחוד דורשת זמן קבוע לכל צרוף של צומת לקבוצה חדשה, וכל צומת יעבור לכל היותר $\log_2 V$ צרופים כאלו, משום שבכל פעולה גודל הקבוצה אליה הוא שייך גדל לפחות פי שניים).

2. ייצוג כל קבוצה ע"י עץ מושרש, כאשר השורש מכיל את שם הקבוצה.

במימוש זה Union($\underline{u}$,v) נעשה על ידי הוספת השורש של הקבוצה הקטנה יותר כבן של השורש של הקבוצה הגדולה יותר. מספר צמתי עץ שנוצר בפעולת איחוד כזו גדול לפחות פי שניים ממספר צמתי העץ הקטן מבין השניים. מכאן ניתן להוכיח באינדוקציה שעמקו של עץ המייצג קבוצה בעלת k צמתים הוא לכל היותר ($\log_2 k$).

פעולת find_set(v) נעשית על ידי "הליכה" מהצומת לשורש העץ.

מימוש זה מבטיח זמן $O(1)$ לפעולת Union(u,v), וזמן $O(\log V)$ לפעולת find_set(V), משום שעמקו של העץ כאמור $O(\log V)$.

שימו לב ששני המימושים ניתנים לביצוע בזמן $O(E\log V)$, שנקבע על ידי הזמן הדרוש למיון הקשתות. מן הראוי לציין שיש מבנה נתונים המממש את הפעולות Union, Find תוך יצוג קבוצות על ידי עצים העובד בזמן "כמעט לינארי" (במספר הפעולות).

אלגוריתמים למסלולים קלים בגרפים

יהי $G=(V,E)$ גרף פשוט, מכוון או לא מכוון, שבו לכל קשת (u,v) משקל $w(u,v)$.

המשקל של מסלול $p=\langle v_0,v_1,\ldots,v_k\rangle$ הוא סכום המשקלות של קשתות המסלול:

$$w(p)=\sum_{i=1}^{k} w(v_{i-1},v_i).$$

מסלול קל ביותר מ- s ל- v הוא מסלול p^* מ- s ל- v המקיים:

לכל מסלול p מ s ל v, $w(p^*)\le w(p)$.

לא תמיד קיים מסלול קל ביותר: ייתכן שאין כלל מסלול מ- s ל- v. לחילופין ייתכן שקיימים אין סוף מסלולים מ- s ל- v, שאף אחד מהם אינו קל ביותר. הכלל הוא:

- אם אין ב- $G=(V,E)$ ***מעגלים שליליים*** (שסכום משקלי קשתותיהם שלילי) שניתנים להגעה מ- s אזי לכל v שניתן להגעה מ s קיים מסלול קל ביותר מ s ל v (למה?).
- אם קיים מסלול מ- s ל- v שמכיל מעגל שלילי, אזי לכל מסלול p מ s ל v קיים מסלול קל ממנו, ולכן לא קיים מסלול קל ביותר מ- s ל- v. במקרה זה כאמור $dist(s,v)=-\infty$

.

לכן הגדרה מלאה של מרחק מ s ל v היא

$$dist(s,v)=\begin{cases}\min\{w(p) \mid p \text{ is path from } s \text{ to } v\} & \text{if there is a shortest path from } s \text{ to } v\\ \infty & \text{if there is no path from } s \text{ to } v\\ -\infty & \text{if there is path but no shortest path from } s \text{ to } v\end{cases}$$

תתי מסלולים של מסלולים קלים

למה 5.2: יהי $G=(V,E)$ גרף ממושקל (מכוון או לא) עם פונקצית המשקל $w:E\to\mathbb{R}$. יהי $p=\langle v_1,v_2,\ldots,v_k\rangle$ מסלול קל ביותר מ- v_1 ל- v_k, ויהי $p_{ij}=\langle v_i,v_{i+1},\ldots,v_j\rangle$ תת המסלול ב- p מ- v_i ל- v_j. אז p_{ij} הוא מסלול קל ביותר מ- v_i ל- v_j.

הוכחה

נפרק את p לתתי מסלולים $v_1\xrightarrow{p_{1i}}v_i\xrightarrow{p_{ij}}v_j\xrightarrow{p_{jk}}v_k$. אזי $w(p)=w(p_{1i})+w(p_{ij})+w(p_{jk})$. נניח בשלילה כי קיים מסלול p_{ij}' כך ש $w(p_{ij}')<w(p_{ij})$ - אז ניתן להגדיר מסלול p' כך $v_1\xrightarrow{p_{1i}}v_i\xrightarrow{p_{ij}'}v_j\xrightarrow{p_{jk}}v_k$ כך ש $w(p')<w(p)$, סתירה. ■

עץ מסלולים קלים ממקור יחיד s של גרף $G=(V,E)$ הוא תת גרף $G'=(V',E')$ כך ש- $V'\subset V$ ו- $E'\subset E$ ומתקיים:

- V' הוא אוסף הצמתים שניתנים להגעה מ- s.
- G' הוא עץ מושרש ששורשו s.
- לכל $v\in V'$ – המסלול היחיד ב- G' מ- s ל- v הוא מסלול קל ביותר ב- G מ- s ל- v.

דוגמה: כאשר לכל הקשתות אותו משקל, עץ BFS הוא עץ מסלולים קלים.

טענה: קיים עץ מסלולים קלים ביותר מ s אםם אין מעגל שלילי נגיש מ s.

הוכחת הטענה: תרגיל בתורת הגרפים.

סוגי אלגוריתמים לבעיית המסלול הקל

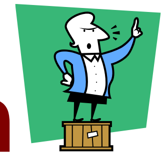
הפלט הנדרש מאלגוריתם למסלולים קלים הינו בדרך כלל אחד משלושת הסוגים הבאים:

- מסלול קל ביותר בין זוג צמתים: "מקור" ו-"בור".

- מסלולים קלים ממקור יחיד – בהנתן צומת מקור s מצא את המסלולים הקלים ביותר מ-s לכל צמתי הגרף. גירסה מצומצמת של הבעיה: מצא את ***המרחקים*** מ s לכל צמתי הגרף.
- מסלולים קלים ביותר בין כל זוגות הצמתים.

בכל אחת מהשאלות הנ"ל מבדילים בין שני מקרים: (א) כל המשקלים אי שליליים, (ב) ייתכנו גם משקלים שליליים.

בהמשך נניח שגרף הקלט הוא גרף מכוון. גרף לא מכוון ניתן לייצוג על ידי גרף מכוון בו לכל קשת קיימת קשת אנטי-מקבילה בעלת אותו משקל.



הרצאה מס' 6

אלגוריתמים למציאת מסלול קל ביותר ממקור יחיד s

קלט: גרף $G=(V,E)$, פונ' משקל $w:E\rightarrow\mathbb{R}$, וצומת s.

פלט: לכל $v\in V$ המרחק (משקל המסלול הקל ביותר) מ – s ל v, שיסומן $dist(s,v)$. לעתים נרצה שהאלגוריתם יחזיר גם עץ מסלולים קלים ביותר.

בחלק מהאלגוריתמים שנלמד מניחים שאין בגרף הקלט מעגלים שליליים נגישים מ s, ומכאן שלכל צומת v, או שקיים מסלול קל ביותר מ s ל v, או ש $dist(s,v)=\infty$.

אלגוריתם גנרי למסלולים קלים ביותר ממקור יחיד

א. קבע חסמים עליונים התחלתיים על ארכי המסלולים.

ב. שפר חסמים אלו על ידי "שיפורים מקומיים", עד שלא ניתן לשפרם עוד.

הנכונות של האלגוריתם הגנרי מתבססת על תכונות של החסמים העליונים הנשמרות על ידי השיפורים המקומיים. תכונות אלו מפורטות בהגדרה הבאה, המתייחסת לגרף קלט $G=(V,E)$ וצומת מקור s:

פונקציית חסם עליון על מרחקים (בקיצור פח"ע): זוהי פונקציה $d:V\rightarrow \mathrm{R}\cup\{\infty\}$ המקיימת שני תנאים:

- לכל צומת $v\in V$ מתקיים $d(v)\ge dist(s,v)$ (מכאן השם חסם עליון).
- $d(s)\le 0$.

כעת נגדיר את השיפורים המקומיים של האלגוריתם הגנרי:

קשת מקצרת: (u,v) היא קשת מקצרת ביחס לd אם $d(u)<\infty$ ו $d(v)>d(u)+w(u,v)$.

שיפור מקומי על קשת (u,v) – relaxation: בהינתן פ"חע d, שיפור מקומי על קשת (u,v), שייקרא שיפור $(d;u,v)$, הוא הכלל הבא:

אם (u,v) קשת מקצרת ביחס ל d בצע $d(v)\leftarrow d(u)+w(u,v)$.

כאשר d ברור מההקשר, נכתוב בקיצור "שיפור (u,v)".

כדי לבנות עץ מסלולים קלים ביותר, כאשר (u,v) היא קשת מקצרת נבצע בשיפור $(d;u,v)$ גם את הפקודה $parent(v)\leftarrow u$.

כעת ניתן ניסוח מדויק של האלגוריתם הגנרי:

א. קבע פח"ע $d:V\to R\cup\{\infty\}$.
ב. כל זמן שקיימת קשת (u,v) מקצרת ביחס ל d, בצע שיפור $(d;u,v)$.

טענה 6.0: תהי נתונה פח"ע d, ויהי C מעגל (מכוון) המכיל צומת u כך ש $d(u)<\infty$. אם אין ב C קשת מקצרת ביחס ל d, אז C איננו מעגל שלילי.

הוכחה : יהי $C=(u_0,\dots,u_{k-1},u_k=u_0)$ מעגל ב G. ראשית נשים לב ש $d(u_i)<\infty$ לכל צומת u_i במעגל (נמקו...). לכן ניתן לסכום את אי השוויונות $w(u_{i-1},u_i)\ge d(u_i)-d(u_{i-1})$, ולקבל את אי השוויון שצריך להוכיח:

$$\sum_{i=1}^{k} w(u_{i-1},u_i)\ge \sum_{i=1}^{k}\left[d(u_i)-d(u_{i-1})\right]=0. \qquad \blacksquare$$

למה 6.1: תהי $d:V\to \mathbb{R}\cup\{\infty\}$ פח"ע עבור G ו s. אז 2 הטענות הבאות נכונות:

א. d נשארת פח"ע גם אחרי ביצוע שיפור $(d;u,v)$.

ב. אם אין קשת מקצרת ביחס ל d, אז לכל צומת v מתקיים $d(v)=dist(s,v)$.

הוכחה

א. נניח שבוצע שיפור $(d;u,v)$: אם (u,v) איננה קשת מקצרת $d(x)$ לא השתנה לאף צומת x, ולכן מהנחת האינדוקציה מתקיים $d(x)\ge dist(s,x)$. אחרת רק $d(v)$ שונה ע"י העדכון $d(v)\leftarrow d(u)+w(u,v)$, והטענה נובעת מאי השוויונות הבאים:

$$d(v)=d(u)+w(u,v)\ge \underbrace{dist(s,u)+w(u,v)}_{\text{זהו אורך מסלול מ } s \text{ ל } v}\ge dist(s,v)$$

אי השוויון השמאלי נובע מההנחה ש $d(u)\ge dist(s,u)$, והימני נובע מהכתוב במסגרת. כמו כן ברור שלאחר השיפור $d(s)\le 0$, משום ש d יכולה רק לקטון.

ב. צ"ל שלכל צומת v מתקיים $dist(s,v)=d(v)$. ידוע ש- $d(v)\ge dist(s,v)$ (מכך ש d פח"ע), לכן מספיק להוכיח ש $d(v)\le dist(s,v)$. נניח בשלילה שקיים צומת v כך ש $d(v)>dist(s,v)$, כלומר קיים מ- s ל- v מסלול שאורכו קטן מ $d(v)$. נסמן מסלול זה

ב- $P=(u_0=s,u_1,\dots,u_k=v)$. יהי $dist_p(s,u)$ המרחק על המסלול p מ- s לצומת u במסלול. מהנחת השלילה $d(v)>dist_p(s,v)$. כעת –

$$dist_p(s,s)=d(s)=0$$

כי מטענה 6.0 נובע שאין מעגל שלילי המכיל את s

$$\vdots$$

$$dist_p(s,v)<d(v)$$

מכאן שקיים i כך ש

$$dist_p(s,u_i)\ge d(u_i)$$
$$dist_p(s,u_{i+1})<d(u_{i+1})$$

ולכן

$$d(u_{i+1})>dist_p(s,u_{i+1})=dist_p(s,u_i)+w(u_i,u_{i+1})\ge d(u_i)+w(u_i,u_{i+1})$$

והקשת (u_i,u_{i+1}) היא קשת מקצרת ביחס ל-d, בניגוד להנחה.■

<u>הערה:</u> למה 6.1 תקפה גם אם בגרף G יש מעגלים שליליים.

מימושים שונים של האלגוריתם הגנרי נבדלים בסדר הביצוע של שיפורים מקומיים. המימוש הראשון שנלמד מרשה קשתות בעלות משקל שלילי:

אלגוריתם בלמן-פורד

<u>קלט</u>: גרף מכוון $G=(V,E)$ ממושקל, ללא מעגלים שליליים, וצומת $s\in V$.
<u>פלט</u>: לכל צומת v - $dist(s,v)=d(v)$.
<u>האלגוריתם</u>:
אתחול: לכל $v\in V\setminus\{s\}$ אתחל $d(v)\leftarrow\infty$, $parent(v)\leftarrow nil$
$d(s)\leftarrow 0$;
עבור $i=1$ עד $V-1$ בצע
לכל קשת (u,v) בצע שיפור $(d;u,v)$

זמן ריצה

$$O(V)+O(VE)=O(VE)$$

הוכחת נכונות

נסתמך על שתי הלמות הבאות:

למה 6.2: לכל צומת v, אם קיים מסלול קל ביותר מ- s ל- v המכיל k קשתות אז לאחר k איטרציות $d(v)=dist(s,v)$.

הוכחה באינ' על k. עבור $k=0$ רק s מקיים את הנחת האינדוקציה ואכן לאחר 0 איטרציות (בזכות האתחול) מתקיימת הטענה - $d(s)=0=dist(s,s)$.

<u>צעד</u>: נניח נכונות עבור k ונוכיח עבור $k+1$. יהי v צומת המקיים שיש מסלול קל ביותר מ- s ל- v המכיל $k+1$ קשתות.

תהי (u,v) הקשת האחרונה, כלומר ה $k+1$ במסלול הנ"ל. כעת k הקשתות הקודמות ל- (u,v) הן מסלול קל ביותר מ- s ל- u (כזכור תת מסלול של מסלול קל ביותר הוא מסלול קל ביותר), ולכן מהנחת האינדוקציה לאחר איטרציה k מתקיים $d(u)=dist(s,u)$. בזמן האיטרציה ה-$k+1$ ביצענו שיפור (u,v) (כי ביצענו שיפור כזה על כל הקשתות). לאחר ביצוע זה מתקבל

$$d(v)\le d(u)+w(u,v)=dist(s,u)+w(u,v)=dist(s,v)$$

כי בוצע
שיפור (u,v)

הנחת האינדוקציה
על $dist(s,u)$

כי (u,v) הקשת
האחרונה במסלול
הקל ביותר מ s ל v

למה 6.3: אם ב- G אין מעגלים שליליים נגישים מ- s אז לכל צומת v הנגיש מ s, קיים מסלול קל ביותר מ- s ל- v המכיל לכל היותר $V-1$ קשתות.

הוכחה

אם אין מעגלים שליליים אז קיים מסלול קל ביותר ללא מעגלים (הוכיחו...), ומסלול ללא מעגלים מכיל לכל היותר $V-1$ קשתות (כי כל צומת מופיע בו לכל היותר פעם אחת).■

מנכונות שתי הלמות הנ"ל נובעת נכונות האלגוריתם של בלמן פורד: מלמה 6.3 נובע כי לכל צומת הנגיש מ- s קיים מסלול קל ביותר מ s המכיל לכל היותר $V-1$ קשתות, ומלמה 6.2 מספיקות $V-1$ איטרציות בכדי למצוא כל מסלול כזה.

דרישה חזקה יותר משל בלמן-פורד, ולכן אלג' מהיר יותר

אלגוריתם דיקסטרה (Dijkstra)

<u>קלט</u>: גרף ממושקל מכוון ללא משקלים שליליים, וצומת s.

<u>פלט</u>: לכל צומת v - $d(v)=dist(s,v)$

<u>האלגוריתם</u>:

<u>אתחול</u>: קבע פח"ע: לכל $v\in V$: $parent(v)\leftarrow nill, d(v)\leftarrow\infty$; $d(s)\leftarrow 0$

$Q\leftarrow V$ // Q מכיל את כל הצמתים עבורם $d(v)\neq dist(s,v)$ (ואולי צמתים נוספים)

כל זמן ש $Q\neq\varnothing$ בצע

מצא ב- Q צומת u כך ש $d(u)$ מינימאלי.

הוצא את u מ- Q ולכל קשת (u,v) בצע שיפור (u,v).

ניתוח סיבוכיות

כאשר Q ממומש במערך - $O\left(V^2\right)$

כאשר Q ממומש בערימת מינימום - $O(E\log V+V)$. ניתן גם ב $O(E\log V)$ ע"י כך שמאתחלים את Q לקבוצת הצמתים הנגישים מs, אותם מוצאים ב BFS בסיבוכיות O(E).

(ניתוח הסיבוכיות דומה לזה של האלגוריתם של פרים לעפ"מ).

הוכחת נכונות

על סמך למה 6.1 ב על נכונות האלגוריתם הגנרי, מספיק להראות שבסיום האלגוריתם לא קיימת קשת מקצרת ביחס ל d. את זה נוכיח בשתי הלמות הבאות. בלמות אלו $\bar{d}(u)$ הוא הערך של $d(u)$ כאשר u הוצא מ- Q.

למה 6.4: אם v הוצא מ Q מיד אחרי u, אז $\bar{d}(u)\leq\bar{d}(v)$.

הוכחה

אם הערך של $d(v)$ לא עודכן בין הוצאתו של u להוצאתו של v מתקיים $\bar{d}(v)\geq\bar{d}(u)$, מפני ש $\bar{d}(u)$ היה מינימלי בעת ש u הוצא מQ. אם $d(v)$ כן עודכן בזמן זה מתקיים $\bar{d}(v)=\bar{d}(u)+w(u,v)\geq\bar{d}(u)$ (מאחר ו $w(u,v)\geq 0$). בשני המקרים $\bar{d}(u)\leq\bar{d}(v)$. ■

מסקנה: אי השוויון $\bar{d}(u)\leq\bar{d}(v)$ מתקיים גם אם v הוצא בזמן כלשהו אחרי u, כי אפשר להפעיל את הטענה על סדרה של הוצאות מהתור - $\bar{d}(u)=\bar{d}(u_1)\leq\cdots\leq\bar{d}(u_k)=\bar{d}(v)$.

מכאן נובע, בגלל אי שליליות הקשתות, ש $\bar{d}(u)$ לא שופר לאחר ש u הוצא, ולכן $\bar{d}(u)$ הוא ערכו של $d(u)$ גם בסיום האלגוריתם.■

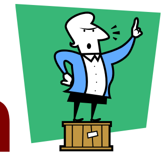
למה 6.5 בסיום האלגוריתם לא קיימת קשת מקצרת.

הוכחה

תהי (u,v) קשת כלשהי. נוכיח שבסיום האלגוריתם זו אינה קשת מקצרת.

בזמן הוצאת u מהתור התבצע שיפור (u,v). מיד אחרי ביצוע שיפור זה התקיים $d(v) \le d(u) + w(u,v)$. מאחר ו $d(u)$ לא שונה יותר עד סיום האלגוריתם ו $d(v)$ יכול היה רק לקטון, הרי גם בסיום האלגוריתם התקיים $d(v) \le d(u) + w(u,v)$. ■

מלמה 6.5 ומלמה 6.1 ב על נכונות האלגוריתם הגנרי נובע שעם סיום ריצת האלגוריתם מתקיים $d(v) = dist(s,v)$ לכל $v \in V$.



הרצאה מס' 7

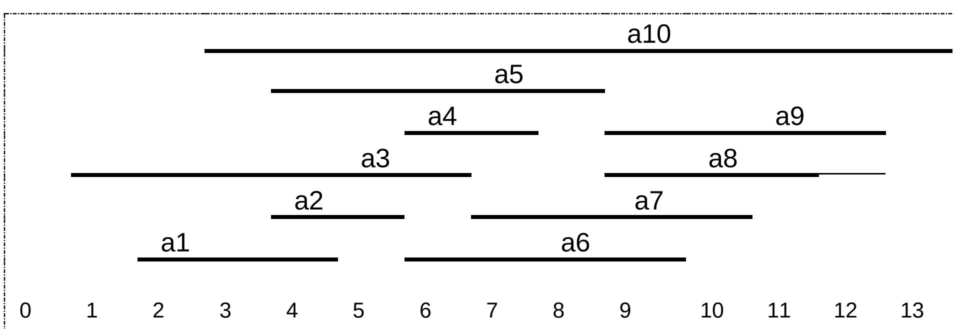
אלגוריתמים חמדניים (Greedy Algorithms)

אלגוריתם חמדני (greedy) הוא אלגוריתם שבונה פתרון ע"י סדרה של "שיפורים מקומיים". האלגוריתם הגנרי למסלולים קלים ביותר הוא חמדני, וגם האלגוריתם הגנרי לעפ"מ. בהרצאה זו ובבאה נראה עוד מספר אלגוריתמים כאלו.

מציאת מספר מקסימלי של קטעים זרים בזוגות

נתון אוסף מטלות $\{a_1,...,a_n\}$. לכל מטלה יש זמן התחלה s_i וזמן סיום f_i. המטרה היא לשבץ מספר מקסימלי של מטלות על משאב יחיד כך ששתי מטלות לא תחתכנה[‡] (דוגמה: אוסף של תכניות טלוויזיה מעניינות ביום שידורים אחד, חלקן חופפות, והמטרה למקסם את מספר התכניות שניתן לראות במלואן).

כל משימה מוגדרת כקטע (s_i, f_i) על ישר הזמן, והמטרה למצוא מספר מקסימאלי של קטעים זרים הדדית (לכן בעיה זו נקראת גם "מציאת קבוצה בלתי תלויה מקסימאלית בגרף אינטרוואלי"). דוגמה לקלט:



הגדרת הבעיה במונחי קלט-פלט

קלט: קבוצת קטעים על הישר $S=\{a_1,...,a_n\}$. כל קטע $a_i=(s_i, f_i)$ מוגדר ע"י זמן סיום וזמן התחלה.

פלט: סדרה $A=(b_1,b_2,...,b_k)$ המכילה מספר מקסימלי אפשרי של קטעים זרים בזוגות מ S, מסודרים על פי זמני התחלה (או זמני סיום – הקטעים כאמור זרים בזוגות).

---

‡ משימות נחתכות אם קיים קטע זמן השייך לשניהן

האלגוריתם:

מיין את הקטעים לפי סדר עולה של זמני סיום. (מעכשיו נניח כי הקטעים מסודרים כך ש $f_1 \le f_2 \le \cdots \le f_n$.)

אתחל $A \leftarrow \varnothing$, $i \leftarrow 0$ ו- $f_0 \leftarrow -\infty$. \\ $a_i = (s_i, f_i)$ הוא הקטע האחרון שנכלל בפתרון.

עבור $m=1$ עד n בצע:

אם $f_i \le s_m$ אז $A \leftarrow A \bullet (a_m)$ ו- $i \leftarrow m$.

סיבוכיות

המיון עולה $O(n\log n)$, והלולאה המרכזית $O(n)$ לכן סה"כ $O(n\log n)$.

למה 7.1: תהי $(b_1,\ldots,b_k)$ סדרת הקטעים הזרים בזוגות המוחזרת ע"י האלגוריתם. עבור כל $0 \le i \le k$, $(b_1,\ldots,b_i)$ הם i קטעים זרים בזוגות, המהווים רישא של פיתרון אופטימאלי (כלומר, הם i הקטעים הראשונים בפתרון אופטימאלי).

הוכחה

באינדוקציה על i.

בסיס: עבור $i=0$ הטענה מתקיימת באופן ריק.

צעד: נניח נכונות עבור $0 \le i < k$ ונוכיח עבור $i+1$. תהי $(b_1,\ldots,b_i)$ הסדרה המקיימת את טענת האינדוקציה עבור i. מהנחת האינדוקציה קיים פתרון אופטימאלי כלשהו $(b_1,\ldots,b_i,c_{i+1},\ldots,c_k)$. ראשית ברור שסדרת הפלט מכילה אבר b_{i+1} (כי הקלט מכיל קטעים המתחילים אחרי b_i). האלגוריתם יחזיר קטע b_{i+1} המסתיים מוקדם ככל האפשר, ולכן לא אחרי c_{i+1}. לכן b_{i+1} אינו נחתך עם אף אחד מהקטעים המופיעים אחרי c_{i+1}. מכאן שהסדרה $(b_1,\ldots,b_i,b_{i+1},c_{i+2},\ldots,c_k)$ המתקבלת על ידי החלפת c_{i+1} ב b_{i+1} היא גם פתרון אופטימאלי, והיא מקיימת את הטענה. ■

נכונות האלגוריתם נובעת מהמקסימאליות (מבחינת הכלה) של הסדרה המוחזרת $(b_1,\ldots,b_k)$

שיבוץ משימות על משאב יחיד עם איחור מינימאלי

נתונה סדרה של n משימות $A=(a_1,\ldots,a_n)$. לכל משימה נתון d_i - הזמן המאוחר ביותר (deadline) בו היא אמורה להסתיים, ו t_i - משך הזמן הדרוש לבצע אותה. כל משימה יש לבצע ברציפות, ואי אפשר לבצע שתי משימות במקביל (דוגמה: אוסף דירות

ששיפוצניק יחיד צריך לשפץ בחודש נתון, כך שהעבודה בכל דירה צריכה להיעשות ברציפות, ולכל דירה יש מועד סיום רצוי. מעבר בין דירה לדירה לוקח זמן זניח). יש לסדר את כל המשימות בסדרה ולבצע אותן על פי הסדר. סידור של המשימות מוגדר על ידי פרמוטציה (תמורה) π של $\{1,2,..,n\}$. $A_\pi = (a_{\pi(1)},...,a_{\pi(n)})$, זמן הסיום של משימה $a_{\pi(k)}$ הוא

$f_{\pi(k)} = \sum_{i=1}^{k} t_{\pi(i)}$. האיחור של משימה $a_{\pi(k)}$ בסידור זה הוא $l_{\pi(k)} = f_{\pi(k)} - d_{\pi(k)}$ (איחור שלילי פרושו סיום לפני הזמן). האיחור המקסימלי בסידור זה הוא $L(A_\pi) = \max_{1\le k\le n}\{l_{\pi(k)}\}$.

מטרתנו למצוא סידור $\pi*$ עבורו האיחור המקסימלי של משימה קטן ככל האפשר, כלומר $L(A_{\pi*}) = \min_{\pi\in S_n} L(A_\pi)$. (S_n היא קבוצת התמורות על $\{1,2,..,n\}$.)

הפתרון מבוסס על ההבחנה שאם זמן סיום של משימה גדול מזה של משימה הבאה מיד אחריה, ניתן להחליף בין המשימות מבלי להגדיל את האיחור המקסימאלי:

למה 7.2 (החלפת משימות סמוכות)

תהי $A = (a_1,...,a_n)$ סדרה נתונה של משימות. אם קיים $k < n$ כך ש $d_k > d_{k+1}$, אז הסדור $A' = (a_1,..,a_{k+1},a_k,a_{k+2},..,a_n)$ המתקבלת על ידי החלפת סדר המשימות a_k, a_{k+1} מקיים $L(A') \le L(A)$.

הוכחה

נסמן ב f_i', l_i' את זמני הסיום והאיחור ב A'. קל לראות שעבור $i \notin \{k,k+1\}$ מתקיים $f_i = f_i'$ ולכן $l_i = l_i'$. לכן יספיק להוכיח: $\max\{l'_k, l'_{k+1}\} \le \max\{l_k, l_{k+1}\}$. נוכיח כי למעשה $\max\{l'_k, l'_{k+1}\} < l_{k+1}$:

$$l'_{k+1} = f'_{k+1} - d_{k+1} = (f_{k+1} - t_k) - d_{k+1} < f_{k+1} - d_{k+1} = l_{k+1}$$

כמו כן, מאחר $d_{k+1} < d_k$, ו $f'_k = f_{k+1}$ מתקיים גם

$$l'_k = f'_k - d_k = f_{k+1} - d_k < f_{k+1} - d_{k+1} = l_{k+1} \qquad \blacksquare$$

מסקנה: מיון המשימות בסדר עולה על פי זמני סיום נותן פיתרון אופטימאלי לבעיה (שימו לב שלכל הסידורים הממוינים לפי זמני סיום יש אותו איחור מקסימלי).

הוכחה: מאחר ומספר הסידורים סופי, קיים סידור אופטימאלי. אם סידור אופטימאלי $A = (a_1,...,a_n)$ אינו ממוין על פי זמני סיום, קיים i כך ש $d_i > d_{i+1}$. על סמך למה 7.2 הסידור A' המתקבל על ידי החלפת d_i ב d_{i+1} מקיים ש $L(A') \le L(A)$. לאחר לכל היותר

$\binom{n-1}{2}$ החלפות כאלו יתקבל סידור A^* בו המשימות ממוינות לפי זמני סיום (למה?). סידור

זה המקיים $L(A^*) \le L(A)$. מאחר ו A אופטימאלי, גם A^* אופטימאלי.■

בעית עצי Huffman או קודים פרפיקסיים אופטימלים

בעיה קלאסית במדעי המחשב היא אחסון מידע (סדרה של אותיות מעל א"ב נתון) תוך צריכה מינימאלית של זיכרון מחשב.

נתון א"ב סופי $\Sigma=\{s_1,...,s_n\}$. "קידוד של Σ" הוא מיפוי $C:\Sigma\to\{0,1\}^+$ המתאים לכל אות ב Σ מילה בינארית $C(s_i)=w_i$.[§] קבוצת המלים $\{w_1,...,w_n\}$ היא "קוד", שגם הוא נקרא C.

לכל מילה u ב Σ^+, $C(u)$ היא המלה הבינארית המתקבלת משרשור הקידודים של האותיות ב u . הקוד צריך להיות "חד פענח": לכל זוג מלים שונות $u_1,u_2\in\Sigma^*$, צריך ש $C(u_1)\ne C(u_2)$.

דוגמה לקוד חד פענח: קוד המכיל מלים בינאריות שונות בעלות אותו אורך.

דוגמה כללית יותר : קוד פרפיקסי (חסר רישות) - זהו קוד בו אף מילת קוד אינה פרפיקס (רישא) של מילת קוד אחרת.

נתונה סדרת אותיות S מ Σ שברצוננו לאחסן (S יכולה להיות למשל אסף מחרוזות DNA של יצורים שונים). כמו כן נתון מיפוי של אותיות לתדירויות $f=\{f(s_1),...,f(s_n)\}$ כאשר $f(s_i)$ היא מספר הפעמים ש s_i מופיעה בS. עבור קידוד נתון C של Σ, נסמן ב $l(w_i)$ את האורך (מספר ביטים) של המילה $C(s_i)=w_i$. אז סה"כ מספר הביטים הדרוש לקידוד S על ידי C הוא $B(C)=\sum_{i=1}^{n} l(w_i)f(s_i)$.

בדוגמא להלן S מכילה 100000 אותיות. הקוד בעל הארך הקבוע דורש 300 אלף ביטים, הקוד הפרפיקסי דורש 224 אלף.

אות	a	b	c	d	e	f
תדירות (באלפים)	45	13	12	16	9	5
קוד בעל ארך קבוע	000	001	010	011	100	101

[§] ניתן להכליל את הדיון לקידודים שאינם בינאריים, הממפים כל אות ב Σ למילים בנות k אותיות, עבור k כלשהו הקטן n.

קוד פרפיקסי	0	101	100	111	1101	1100

בהינתן סדרה S כנ"ל, מטרתנו למצא קוד חד פענח C המקודד את S במספר מינימאלי של ביטים.

הגדרה פורמאלית של הבעיה:

- קלט: $[\Sigma, f]$, כאשר $\Sigma=\{s_1,...,s_n\}$ הוא א"ב ו f המגדיר לכל אות $s_i \in \Sigma$ תדירות $f(s_i)$.
- פלט: קוד פרפיקסי (בינארי) של Σ : $C(s_i)=w_i$ הממזער את הסכום

$$B(C)=\sum_{i=1}^{n} l(w_i) f\left(s_i\right)$$

קיימים גם קודים חד פענח שאינם פרפיקסיים. הסיבה שההגדרה מסתפקת בקודים פרפיקסיים נובעת מהמשפט הבא, שמובא ללא הוכחה:

משפט: יהי $\{w_1,...,w_n\}$ קוד חד פענח כלשהו. אז קיים קוד פרפיקסי $\{u_1,...,u_n\}$ כך שלכל i מתקיים $l\left(u_i\right)=l\left(w_i\right)$.

למתעניינים - משפט זה נובע מאי שוויון קראפט Kraft (שמוכח למשל בספר Graph Algorithms של אבן).■

עצי האפמן Huffman

<u>יצוג קודים בינאריים פרפיקסים ע"י עצים בינאריים:</u> כזכור, עץ בינארי הוא עץ מסודר בו לכל צומת פנימי שני בנים לכל היותר, אחד שמאלי והשני ימני. קשת לבן שמאלי מסומנת ב 0 וקשת לבן ימני מסומנת ב 1.

יהי $C=\{w_1,...,w_n\}$ קוד פרפיקסי של א"ב $\Sigma=\{s_1,...,s_n\}$, כאשר $w_i=C(s_i)$, ויהיו $f=\{f(s_1),...,f(s_n)\}$ התדירויות המתאימות בקובץ S.

ניתן לייצג את C ע"י עץ בינארי $T=(V,E)$ עם n עלים המותאמים באופן 1-1 ערכי לאותיות ב Σ באופן הבא:

$$V=\{u: u \text{ is a prefix of } w_i \text{ for some } i\in[1,n]\}$$
$$E=\{(u,ub): u\in\{0,1\}^*, b\in\{0,1\}, ub\in V\}$$

עבור כל אות s_i, מילת הקוד המתאימה w_i היא סדרת הביטים המופיעים על קשתות המסלול מהשורש של T לעלה המתאים ל s_i. נסמן ב- $d_T\left(s_i\right)$ את עומק העלה המתאים ל s_i. מההגדרה נובע ש $d_T\left(s_i\right)=l(w_i)$. לכן מספר הביטים הדרוש לקידוד הקובץ S הוא:

$$B(T)=\sum_{i=1}^{n} d_T(s_i)\,f(s_i)$$

T הוא עץ אופטימלי עבור הקוד והתדירות הנתונים אם $B(T)$ הוא המינימאלי האפשרי. עץ אופטימאלי כזה נקרא עץ האפמן Huffman.

למה 7.3

קיים עץ האפמן בו שתי אותיות בעלות תדירויות מינימאליות מתאימות לזוג עלים אחים בעלי עומק מקסימאלי.

הוכחה

יהי T עץ אופטימלי כלשהו. ראשית נוכיח שיש ב T שני עלים אחים בעלי עמק מקסימאלי: מהאופטימאליות של T נובע שלכל צמת פנימי יש שני בנים (הוכיחו...). לכן לכל עלה בעל עומק מקסימלי יש אח שהוא גם עלה.

יהיו a,b שתי אותיות המתאימות לזוג עלים אחים בעלי עומק מקסימלי, ויהיו x,y שתי אותיות בעלות תדירות מינימאלית.

נניח בה"כ כי

$$\begin{cases} f(x)\le f(y) \\ f(a)\le f(b) \end{cases}$$

ולכן ניתן להניח שקיימים $\varepsilon,\delta\ge 0$ כך ש:

$$\begin{cases} f(a)=f(x)+\varepsilon \\ f(b)=f(y)+\delta \end{cases}$$

יספיק להוכיח כי העץ T' המתקבל ע"י החלפת x עם a ו y עם b הוא אופטימלי (כי בעץ T' האותיות x,y מתאימות לעלים אחים בעומק מקסימאלי – ראו ציור).

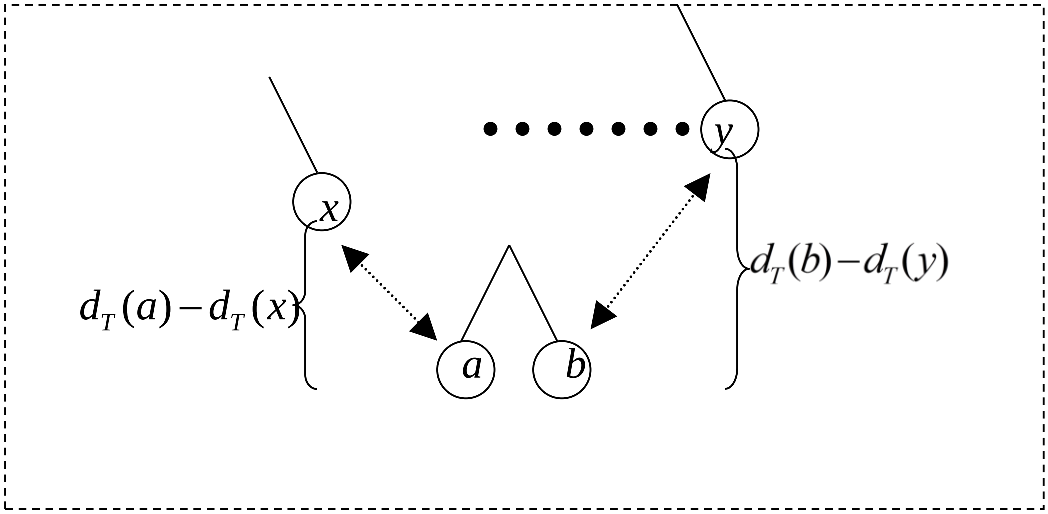
החלפת x עם a מקטינה את הסכום $B(T)$ ב $\left(d_T(a)-d_T(x)\right)\varepsilon\ge 0$.

באופן דומה החלפת b עם y מקטינה את $B(T)$ ב - $\left(d_T(b)-d_T(y)\right)\delta\ge 0$.

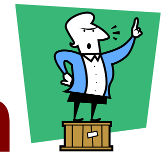
ומכאן מקבלים

$$B(T')=B(T)-\left(d_T(a)-d_T(x)\right)\varepsilon-\left(d_T(b)-d_T(y)\right)\delta\le B(T)$$

מאחר ו- T אופטימלי גם T' אופטימלי (ולמעשה מתקיים שוויון).■



x
y
a
b
$d_T(a)-d_T(x)$
$d_T(b)-d_T(y)$

 הרצאה מס' 8

סיכום של בעיית עצי האפמן עד כה

- קלט: $[\Sigma, f]$ כאשר Σ הוא א"ב ו f מתאימה לכל אות $s \in \Sigma$ תדירות (מספר ממשי חיובי) $f(s)$.
- פלט: עץ האפמן בינארי עבור $[\Sigma, f]$, כלומר עץ בינארי T שעליו הם האותיות ב Σ, הממזער את הסכום

$$B(T)=\sum_{s\in\Sigma} d_T(s)\, f(s)$$

כאשר $d_T(s)$ הוא עומק העלה המתאים לאות s בעץ T.

האלגוריתם למציאת עצי האפמן מתבסס על למה 7.3 . הוא מעביר קלט $[\Sigma, f]$ עם $|\Sigma|=n$ לקלט $[\Sigma', f']$ עם $|\Sigma'|=n-1$, מוצא בצורה רקורסיבית עץ האפמן T' עבור $[\Sigma', f']$, ומ T' מגדיר עץ האפמן T עבור $[\Sigma,F]$. נכונות האלגוריתם נובעת מלמה 8.1 :

למה 8.1 יהי $[\Sigma, f]$ קלט לבעיית עצי האפמן, ויהיו x,y אותיות בעלות תדירות מינימאלית ב Σ. יהי $[\Sigma', f']$ הקלט המוגדר על ידי:

- $\Sigma'=\Sigma\setminus\{x,y\}\bigcup\{z\}$ כאשר $z\notin\Sigma$;
- $f'(z)=f(x)+f(y)$, ולכל אות אחרת $f'(s)=f(s)$.

יהי T' עץ האפמן עבור $[\Sigma', f']$, אז העץ T המתקבל מ-T' ע"י הפיכת העלה המתאים ל- z לצומת פנימי ולו שני בנים x,y הוא עץ האפמן עבור $[\Sigma, f]$.

הוכחה

יהיו T, T' העצים המוגדרים בלמה, ויהי h העומק של העלים x,y ב T. מהבניה מתקבל:

$$B(T)=B(T')-(h-1)f(z)+h(\underbrace{f(x)+f(y)}_{f(z)})=B(T')+f(z)$$

כמו כן, קיים עץ האפמן T_0 עבור $[\Sigma, f]$ בו x,y הם שני עלים אחים בעלי עומק מקסימאלי . יהי T_0' העץ המתקבל מ- T_0 ע"י "תלישת" x,y והקצאת תדירות $f(z)=f(x)+f(y)$ לאבא שלהם, שהפך להיות עלה. מאותם שיקולים מתקבל: $B(T_0)=B(T_0')+f(z)$.

מהאופטימאליות של T' נובע כי $B(T')\le B(T_0')$. ולכן:

$$B(T)=B(T')+f(z)\le B(T_0')+f(z)=B(T_0)$$

ומאחר ו T_0 אופטימאלי ו $B(T)\le B(T_0)$ הרי שגם T אופטימאלי.■

אלגוריתם Huffman לקוד פרפיקסי אופטימלי (גרסה רקורסיבית)

<u>Recursive_Huffman(</u>$[\Sigma, f]$<u>)</u>:

<u>קלט</u>: א"ב Σ בעל 2 אותיות לפחות, ותדירות $f(s)$ לכל אות $s\in\Sigma$.

<u>פלט</u>: עץ האפמן של $[\Sigma, f]$

אם $|\Sigma|=2$ החזר עץ בינארי בעל שני עלים שהם שני איברי Σ .

אם $|\Sigma|>2$:

יהי $[\Sigma', f']$ הקלט הנוצר על ידי החלפת שתי אותיות x,y בעלי תדירות מינימאלית ב Σ באות חדשה z שתדירותה $f'(z)=f(x)+f(y)$.

$T' :=$ Recursive_Huffman($[\Sigma', f']$).

הוסף לעלה המתאים ל- z ב- T' את x ו- y כבנים, והחזר את העץ שהתקבל T .

סיבוכיות זמן

הקלט $[\Sigma, f]$ יוחזק בערמת מינימום, כאשר המפתח של אות x הוא תדירותה $f(x)$. בנית הערימה נעשית ב $O(n)$, ומציאת אבר בעל תדירות מינימאלית ב $O(\log n)$.

אם נסמן ב- את סיבוכיות הנוהל עבור n איברים, אז נקבל משוואת נסיגה:

$t(n)=d\log n+t(n-1), t(2)=c$ (c,d קבועים כלשהם).

פתרון משוואה זו נותן את אי השוויון $t(n)<dn\log n+c=O(n\log n)$.

<u>תכנות דינמי</u>

תכנות דינמי (או תכנון דינמי) הוא שיטת תכנות הפותרת בעיה בשלבים: מתחילים בפתרון גרסה פשוטה של הבעיה, וכל שלב פותרים גרסה (או גרסאות) מסובכות יותר של הבעיה. הפתרון של כל גרסה משתמש בפתרונות של גרסאות פשוטות יותר, שנמצאו בשלבים מוקדמים יותר. בשלב האחרון פותרים את הבעיה המקורית.

דוגמה ראשונה שנראה היא מציאת המרחקים (והמסלולים הקלים ביותר) בין כל זוגות הצמתים בגרף ממושקל חסר מעגלים שליליים:

אלגוריתם פלויד וורשל Floyd–Warshall למציאת כל המרחקים בגרף ללא מעגלים שליליים

<u>קלט</u>: גרף מכוון $G=(V,E)$ עם פונקצית משקל $w:E\rightarrow R$ ללא מעגלים שליליים.

<u>פלט</u>: המרחקים $dist(i,j)$ עבור כל זוגות הצמתים i,j.

(גרסה מורחבת של האלגוריתם גם מחזירה מסלולים קלים ביותר בין כל זוגות הצמתים .)

הערה: האלגוריתם של בלמן פורד מוצא בסיבוכיות זמן $O(VE)$ את המרחק מצומת יחיד לכל שאר הצמתים. לכן הרצת בלמן פורד מכל צומת פותרת את הבעיה בסיבוכיות זמן $O(V^2E)$.

האלגוריתם שנציג משפר זאת על ידי מציאת כל המרחקים באופן סימולטני, והמבנה שלו דומה לזה של בלמן פורד.

בשלב ראשון נגדיר הכללות של פח"ע ושל כלל השיפור המקומי שהאלגוריתם משמש בהן.

פונקצית חסם עליון על ארכי כל המסלולים

לצורך הגדרתה של פונקצית חסם העליון על ארכי כל המסלולים נכליל ראשית את פונקצית המשקל w לפונקצית משקל מוכללת $\bar{w}$, המוגדרת לכל זוג צמתים.

$$\bar{w}(i,j)=\begin{cases}0 & i=j\\ w(i,j) & (i,j)\in E\\ \infty & otherwise\end{cases}$$

פח"ע על ארכי כל המסלולים בG, בקיצור פחע"מ (פונקצית חסם עליון מוכללת) על G, היא פונקציה - $d:V\times V\rightarrow R\cup\{\infty\}$ המקיימת $dist(i,j)\le d(i,j)\le \bar{w}(i,j)$.

שימו לב שאם בגרף אין מעגלים שליליים, $dist(i,i)=\bar{w}(i,i)=0$ לכל i, ומכאן $d(i,i)=0$.

שיפור מקומי מוכלל

שיפור מקומי מוכלל עבור שלשה של צמתים (i,k,j) הוא הכלל:

אם $d(i,k)+d(k,j)<d(i,j)$ אז $d(i,j)\leftarrow d(i,k)+d(k,j)$.

אם מעוניינים שהשיפור המקומי המוכלל יאפשר גם לשחזר מסלולים קלים ביותר, כל פעם שמעדכנים את $d(i,j)$ מבצעים גם את העדכון $parent_i(j)\leftarrow parent_k(j)$. משמעות העדכון:

$parent_i(j)$ הוא הצומת הקודם ל- j במסלול הקל ביותר מ- i ל- j.

למה 8.2 יהי $G=(V,E)$ גרף ממושקל ללא מעגלים שליליים, ותהי d פחע"מ על G. שתי הטענות הבאות נכונות:

א. d נשארת פחע"מ גם אחרי ביצוע שיפור (i,k,j).

ב. אם לכל שלשה צמתים $\{i,j,k\}$ מתקיים $d(i,k)+d(k,j)\geq d(i,j)$, אז לכל זוג צמתים i,j מתקיים $d(i,j)=dist(i,j)$.

הוכחה: הוכחת א. דומה להוכחת למה 6.1א על מסלולים קלים ביותר ממקור יחיד. הנכונות של ב. נובעת מנכונות למה 6.1ב (משום ההנחה של ב. גוררת את ההנחה של למה 6.1ב).■

אלגוריתם פלויד וורשל למציאת כל המסלולים הקלים ביותר

<u>קלט</u>: $G=(V,E)$ עם פונקציית משקל $w:E\to R$, ללא מעגלים שליליים.

בה"כ נניח כי $V=\{1,...,n\}$.

<u>פלט</u>: לכל זוג צמתים $i,j\in V$ מוחזר המרחק מ- i ל- j - $dist(i,j)$.

<u>אתחול</u>: לכל זוג צמתים i,j בצע $d(i,j)\leftarrow \bar{w}(i,j)$.

עבור $k=1$ עד n בצע

  לכל זוג צמתים i,j בצע שיפור (i,k,j).

הערה: אם רוצים גם למצא את המסלולים הקלים ביותר, צריך באתחול להוסיף: לכל זוג צמתים (i,j) בצע: אם קיימת קשת (i,j) אז $parent_i(j)\leftarrow i$, אחרת $parent_i(j)\leftarrow nil$.

זמן ריצה

אתחול ב- $O(V^2)$.

בכל איטרציה הלולאה מבצעת $O(V^2)$ פעולות, ויש V איטרציות, סה"כ $O(V^3)$.

<u>הוכחת נכונות</u>

תזכורת: צומת במסלול נקרא "פנימי" אם איננו אחד מצמתי הקצה של המסלול. לצורך הוכחת הנכונות נגדיר היררכיה על כל המסלולים בגרף, הנקבעת על ידי הצמתים הפנימיים במסלולים:

<u>רמה 0</u>: קבוצת המסלולים ללא צמתים פנימיים (כלומר המכילים לכל היותר קשת אחת).

<u>רמה 1</u>: כוללת את רמה 0, ובנוסף גם מסלולים המכילים את 1 כצמת פנימי.

<u>רמה 2</u>: קבוצת המסלולים המכילים כצמתים פנימיים רק את 1 ו\או 2 (או אף אחד משניהם).

......

<u>רמה k</u>: קבוצת המסלולים שמכילים בתור צמתים פנימיים רק צמתים מהקבוצה $\{1,2,...,k\}$

.....

<u>רמה n</u>: קבוצת כל המסלולים בגרף.

שימו לב שקבוצת המסלולים ברמה $k+1$ מכילה את קבוצת המסלולים ברמה k

הגדרה $dist^{(k)}(i,j)$ הוא משקל המסלול המינימלי מ- i ל- j מבין כל המסלולים ברמה k.

אם אין מסלול מ- i ל- j ברמה k אז $dist^{(k)}(i,j)=\infty$.

למה 8.3 מתקיים

$$dist^{(k)}(i,j)=\min\{dist^{(k-1)}(i,j),dist^{(k-1)}(i,k)+dist^{(k-1)}(k,j)\}$$

הוכחה

- אם לא קיים מסלול ברמה k מ- i ל- j, שני צידי המשוואה הם ∞ (בדקו..).
- אחרת, אם קיים מסלול קל ביותר ברמה k מ- i ל- j שלא עובר דרך k אז זהו מסלול קל ביותר גם ברמה ה- $k-1$, ולכן מתקיים השוויון $dist^{(k)}(i,j)=dist^{(k-1)}(i,j)$.
- אחרת – קיים מסלול כזה העובר דרך k פעם אחת (כי אין מעגלים שליליים). תת המסלול מ- i ל- k הוא מרמה $k-1$, וכנ"ל לגבי תת המסלול מ- k ל- j. ולכן במקרה זה $dist^{(k)}(i,j)=dist^{(k-1)}(i,k)+dist^{(k-1)}(k,j)$ ■

ההגדרה והלמה הבאות מתייחסות לריצה נתונה של האלגוריתם.

הגדרה $d^{(k)}(i,j)$ הוא הערך של $d(i,j)$ לאחר סיום האיטרציה ה k-ית של האלגוריתם.

למה 8.4 לאחר האיטרציה ה- k-ית של האלגוריתם מתקיים $d^{(k)}(i,j)=dist^{(k)}(i,j)$.

הוכחה באינדוקציה על k:

- עבור $k=0$ הטענה נובעת מכך ש $d^{(0)}(i,j)=\bar{w}(i,j)$.
- נניח נכונות עבור $k-1$ ונוכיח עבור k. באיטרציה ה- k-ית האלגוריתם מבצע שיפור (i,k,j), ולאחר ביצוע שיפור מתקיים:

$$d^{(k)}(i,j)=\min\left\{d^{(k-1)}(i,j),\ d^{(k-1)}(i,k)+d^{(k-1)}(k,j)\right\}$$
$$=\min\left\{dist^{(k-1)}(i,j),\ dist^{(k-1)}(i,k)+dist^{(k-1)}(k,j)\right\}=dist^{(k)}(i,j)$$

מהגדרת
שיפור (i,k,j)

מהנחת האינדוקציה
עבור k-1

מלמה 8.3

■

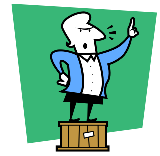
נכונות האלגוריתם של פלויד וורשל נובעת מלמה 8.4 עבור $k=n$, שכן בסיום הריצה מתקיים

$$dist(i,j)=d^{(n)}(i,j)=dist^{(n)}(i,j)=dist(i,j)$$

כי d פחע"מ

למה 8.4 עבור
$k = n$

כי הרמה ה n מכילה את
כל המסלולים בגרף



הרצאה מס' 9

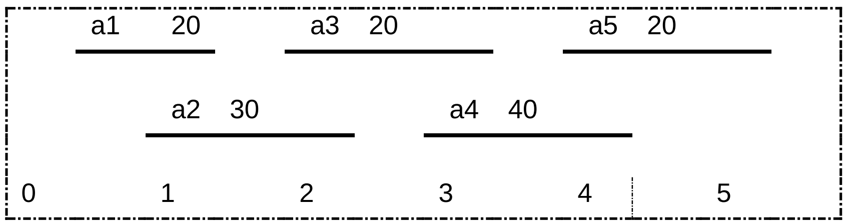
קבוצה בלתי תלויה של קטעים בעלת משקל מקסימלי

<u>קלט</u>: אוסף של מטלות $S=\{a_1,...,a_n\}$, כאשר מטלה a_i מאופיינת ע"י זמן התחלה s_i, וזמן סיום f_i, ומשקל w_i. (ניתן להתייחס אל (s_i,f_i) כאל קטע, ומכאן הכותרת).

<u>פלט</u>: תת קבוצה $A\subset S$ של קטעים זרים בזוגות שסכום משקליהם מקסימאלי אפשרי.

ניתן להניח שכל המשקלים אי שליליים, שכן פתרון אופטימאלי לא יכיל קטע שמשקלו שלילי (האלגוריתם שיוצג נכון גם ללא הנחה זו).

דוגמה לקלט לבעיה:



אלגוריתם לפתרון הבעיה מתחיל במיון המטלות על פי זמני סיום. כלומר $f_1\le f_2\le\cdots\le f_n$.

ביחס לסידור הזה, נגדיר לכל i את המושגים $opt(i),A(i)$ ו $pred(i)$:

$opt(i)$: מחיר פתרון אופטימלי עבור מטלות $\{a_1,...,a_i\}$.

$A(i)$: קבוצת אינדקסים של קטעים המממשים פתרון אופטימלי כנ"ל.

$$pred(i)=\begin{cases}\max\{j: f_j\le s_i\} & \text{if such a } j \text{ exists}\\ 0 & \text{otherwise}\end{cases}$$

כלומר, $a_{pred(i)}$ היא המטלה האחרונה (ברשימה הממוינת על פי זמני סיום) המסתיימת לפני ש a_i מתחיל. אם אין מטלה כזו אז $pred(i)=0$.

למה 9.1

$opt(i)$ מקיים את השוויון הבא:

$$opt(i)=\max\{opt(i-1),\ opt(pred(i))+w_i\}$$

ובהתאמה $A(i)=A(i-1)$ או $A(i)=A(pred(i))\cup\{i\}$.

<u>הוכחה</u>: אם קיים פתרון אופטימאלי עבור מטלות $\{a_1,...,a_i\}$ שאינו מכיל את a_i הרי ש $opt(i)=opt(i-1), A(i)=A(i-1)$. אחרת הפתרון האופטימאלי מכיל את a_i ובנוסף לו רק אברים מהקבוצה $\{a_1,a_2,...,a_{pred(i)}\}$, ולכן

$$opt(i)=opt(pred(i))+w_i, A(i)=A(pred(i))\cup\{i\}. \blacksquare$$

האלגוריתם

<u>אתחול:</u>

- מיין את המטלות לפי סדר עולה של זמני סיום. לאחר המיון $f_1 \le f_2 \le \cdots \le f_n$.
- אתחל $A(0) \leftarrow \varnothing$, ו- $opt(0) \leftarrow 0$.
- לכל i מצא את $pred(i)$.

<u>גוף האלגוריתם:</u>

- עבור $i=1$ עד n בצע
  - אם $opt(i-1) < opt(pred(i))+w_i$ אז
    - $A(i) \leftarrow A(pred(i)) \cup \{a_i\}$.
    - $opt(i) \leftarrow opt(pred(i))+w_i$.
  - אחרת
    - $A(i) \leftarrow A(i-1)$.
    - $opt(i) \leftarrow opt(i-1)$.

החזר $A(n)$.

נכונות האלגוריתם נובעת ישירות מלמה 9.1.

ניתוח סיבוכיות

- מיון ב- $O(n\log n)$.
- מציאת pred(i) לכל i: מאחר והמטלות ממוינות לפי זמני סיום, ניתן למצוא את pred(i) בזמן $O(\log i)$, לכן סה"כ $O\left(\sum_{i=1}^{n}\log(i)\right)=O(n\log n)$.
- גוף האלגוריתם - $O(n)$.

סה"כ $O(n\log n)$.

הערה: אם בכל שלב היינו מחשבים את $opt(i), opt(pred(i))$ ע"י קריאה רקורסיבית לאלגוריתם, היינו מקבלים אלגוריתם בעל סיבוכיות אקספוננציאלית. נראה זאת גם בדוגמה הבאה לתכנות דינאמי:

מציאת סדר אופטימלי למכפלת n מטריצות

תזכורת: מכפלת מטריצה A מסדר $p\times q$ עם מטריצה B מסדר $q\times r$ נותנת מטריצה C מסדר $p\times r$ ודורשת (בביצוע ה"נאיבי") pqr כפלים סקלריים♠ (ומספר דומה של פעולות חיבור – אך פעולות חיבור זולות מפעולות כפל ולכן נתעלם מהן).

כאשר רוצים לכפול יותר משתי מטריצות, יש מספר דרכים שונות לעשות זאת. ניתן לתאר ביצוע המכפלה ב-2 צורות:

- שימוש בסוגריים (למשל, ל3 מטריצות יש שתי אפשרויות: $(AB)C$ או $A(BC)$).
- עץ בינארי בו העלים הם מטריצות הקלט, וצומת פנימי מייצג את מכפלת שתי המטריצות המיוצגות ע"י בניו.

מספר הכפלים (הסקלריים) שצריך לבצע תלוי בעץ שבחרנו.

תאור הבעיה: מצא סדר למכפלה של n מטריצות שממזער את מספר הכפלים הסקלאריים שצריך לבצע.

נדגים את הבעיה עבור כפל של 3 מטריצות $A_{10\times100}\bullet B_{100\times5}\bullet C_{5x50}$.

$(AB)C$ יתן 10*100*5+10*5*50=7500 כפלים, ולעומת זאת:

$A(BC)$ יתן 100*5*50+10*100*50=75000 כפלים – כלומר פי 10(!) מחישוב $A(BC)$.

נתייחס למספר הכפלים הדרוש לביצוע מכפלה בסדר נתון כ"מחיר" של המכפלה בסדר זה.

הגדרת הבעיה:

קלט: סדרה של n מטריצות $(A_1,...,A_n)$. כאשר מטריצה A_i ממימד $p_{i-1}\times p_i$.

פלט: המחיר המינימאלי האפשרי של ביצוע המכפלה $A_1\cdot A_2\cdot\cdots\cdot A_n$.

הערה: בפועל נרצה גם למצא עץ אופטימאלי המגדיר סדר אופטימלי לביצוע המכפלה. בהרצאה נתמקד רק במציאת המחיר המינימאלי, ונתאר בקצרה איך הפתרון מוכלל גם למציאת העץ.

הפתרון הוא על ידי תכנות דינאמי: מציאת פתרון אופטימאלי לסדרה של n מטריצות תעשה על ידי שמוש בפתרונות אופטימאליים לתתי סדרות שארכיהן קטנים מ n.

נתחיל במספר הגדרות.

הגדרה: עבור קלט $(A_1,..,A_n)$, $m[i,j]$ הוא המחיר המינימאלי של מכפלת המטריצות $A_i\cdots A_j$. שימו לב שהפתרון שאנו מחפשים הוא $m[1,n]$.

♠ כפל סקלרי: כפל של שני מספרים בשדה מעליו מוגדרות המטריצות

למה 9.2 $m[i,j]$ מקיים את השוויון:

$$m[i,j]=\begin{cases}0 & i=j\\ \min\limits_{i\le k<j}\{m[i,k]+m[k+1,j]+p_{i-1}p_kp_j\} & i<j\end{cases}$$

הוכחה: יהי T עץ שמשיג את $m[i,j]$. תהי A_k המטריצה הימנית ביותר בתת העץ השמאלי של T ($i\le k<j$). בגלל האופטימאליות של T, המחיר של מכפלות המטריצות בשני תתי העצים הוא $m[i,k]$ ו- $m[k+1,j]$ בהתאמה. בשלב האחרון מכפילים שתי מטריצות שממדיהן $p_{i-1}\times p_k$ ו $p_k\times p_j$, במחיר של $p_{i-1}p_kp_j$ כפלים.■

חישוב $m[1,n]$ לפי הנוסחה הרקורסיבית באופן ישיר דורש זמן אקספוננציאלי: יהי $T(n)$ הזמן הדרוש לחישוב $m(i,i+n-1)$ לפי נוסחה זו. מתקיים:

$$T(2)\ge 2,\ \text{and for } n>2:$$

$$T(n)\ge 2(n-1)+2\sum_{j=1}^{n-1}T(j)>2T(n-1)\ge 2^{n-1}$$

הסבה לסיבוכיות הגבוהה במימוש הרקורסיבי היא שאנחנו מחשבים בכל שלב מחדש ערכים שחישבנו בשלבים קודמים. לכן עדיף להשתמש בתכנות דינמי, שכאמור שומר את ערכי הערכים שחושבו בשלבים קודמים.

האלגוריתם

<u>קלט</u>: n מטריצות $A_1,\ldots,A_n$. כאשר מטריצה A_i ממימד $p_{i-1}\times p_i$.

<u>פלט</u>: $m[1,n]$ - מחיר (מספר הכפלים הסקלריים מינימאלי) המכפלה $A_1\times\cdots\times A_n$.

- (אתחול) עבור $i=1$ עד n בצע $m[i,i]\leftarrow 0$.
- עבור $j=1$ עד $n-1$ בצע
  אורך הרצפים הוא $j+1$
  - עבור $l=1$ עד $n-j$ בצע
    A_l היא המטריצה השמאלית ביותר
    - $m[l,l+j]\leftarrow \min\limits_{l\le k<l+j}\{m[l,k]+m[k+1,l+j]+p_{l-1}p_kp_{l+j}\}$

זמן ריצה

אתחול $O(n)$. הלולאה החיצונית מבוצעת n-1 פעמים, ובכל פעם הלולאה הפנימית מבוצעת פחות מ n פעמים. בכל בצוע של הלולאה הפנימית מוצאים מינימום מתוך קבוצה של פחות מ n מספרים, שכל אחד מהם מחושב בזמן קבוע. סה"כ $O(n^3)$.

דוגמת הרצה על קלט של 4 מטריצות :

$A_1{:}35\times15 \quad A_2{:}15\times5 \quad A_3{:}5\times10 \quad A_4{:}10\times20$

בטבלה להלן, $(p_{i-1}\times p_{i+j})$ הם ממדי מטריצת המכפלה $A_i\cdot A_{i+1}\cdots A_{i+j}$. הכניסה השמאלית העליונה (בצהוב) היא התוצאה המבוקשת. המספרים בכחול הם האינדקסים k שמשיגים את המינימום בנוסחה ל$m[l,l+j]$, לצורך בניית העץ האופטימלי:

j	$m[1,1+j]$, $(p_0\times p_{1+j});k$	$m[2,2+j]$, $(p_1\times p_{2+j});k$	$m[3,3+j]$, $(p_2\times p_{3+j});k$	$m[4,4+j]$, $(p_3\times p_{4+j});k$
3	m[1,4]=7125, (35×20);2			
2	m[1,3]=4375, (35×10);2	m[2,4]=2500, (15×20);2		
1	m[1,2]=2625, (35×5);1	m[2,3]=750, (15×10);2	m[3,4]=1000, (5×20);3	
0	m[1,1]=0, (35×15)	m[2,2]=0, (15×5)	m[3,3]=0, (5×10)	m[4,4]=0, (10×20)
	A_1:35×15	A_2:15×5	A_3:5×10	A_4:10×20

לדוגמה, החישוב של $m[1,3]$ נעשה באופן הבא:

$$m[1,3]=\min\left\{\begin{array}{l} m[1,1]+m[2,3]+p_0p_1p_3=0+750+35\times15\times10=6000 \\ m[1,2]+m[3,3]+p_0p_2p_3=2625+0+35\times5\times10=4375 \end{array}\right\}=4375$$

תוספת אפשרית להרצאה מס' 9

התאמת מחרוזות Sequence Alignment

נתונות שתי מילים מעל א"ב Σ (המילים יכולות להיות רצפים של אותיות DNA מעל א"ב בן ארבע אותיות $\{A,G,C,T\}$, או שתי מילים באנגלית שאחת היא שיבוש של השניה). "התאמת מחרוזות" sequence alignment מגדירה סדרת "פעולות עריכה" המעבירות מילה אחת לשניה. הפעולות המותרות הן: הצבה (החלפת אות באות), מחיקה, או הוספה (של אות אחת). מטרתנו למצא התאמה בעלת מחיר מינימאלי, כפי שיוגדר להלן.

דוגמה: התאמה אפשרית בין המילים occurrence, ocurancea

o	c	c	u	r	r	e	n	c	e	-
o	-	c	u	r	-	a	n	c	e	a

הגדרה: התאמה בין שתי מילים $X = x_1x_2...x_m,\ Y = y_1y_2...y_n$ מעל א"ב Σ היא מטריצה של שתי שורות $A = (E(X);E(Y))$, כאשר $E(x) = z_1z_2...z_k, E(y) = w_1w_2...w_k$ הן מילים שוות אורך מעל א"ב $\Sigma \cup \{-\}$ ("$-$" היא אות המייצגת רווח, שאינה ב Σ), כך ש

א. מחיקת ההופעות של "$-$" במילה $E(X)$ (שבשורה הראשונה) תיתן את X.

ב. מחיקת ההופעות של "$-$" במילה $E(Y)$ (שבשורה השנייה) תיתן את Y.

ג. A אינה מכילה עמודה עם שני "$-$" (בהתאמה אין רווח מעל רווח)

כל עמודה ב $A = (E(X);E(Y))$ מגדירה "פעולת עריכה" במעבר מהמילה X למילה Y.

המחיר של כל פעולת עריכה מוגדר על ידי פונקצית מחיר σ, המוגדרת עבור כל צמד תווים ב $(\Sigma \cup \{-\}) \times (\Sigma \cup \{-\}) \setminus \{(-,-)\}$:

$$\sigma : [(\Sigma \cup \{-\}) \times (\Sigma \cup \{-\}) \setminus \{(-,-)\}] \ \mapsto \ \Re$$

משמעות פונקצית המחיר:

- $\sigma(x,y)$ הוא מחיר החלפת x ב y (ייתכן ש $x = y$)
- $\sigma(x,-)$ הוא מחיר של מחיקת האות x
- $\sigma(-,y)$ הוא מחיר של הוספת האות y

x	x	-
y	-	y

בדרך כלל מניחים ש $\sigma(x,x) = 0$ לכל אות $x \in \Sigma$.

מחיר ההתאמה $A = (E(X);E(Y))$ הוא סכום מחירי העמודות בA:

$$\text{cost(A)=cost(E(X);E(Y))=cost}(z_1z_2...z_k;w_1w_2...w_k) = \sum_{i=1}^{k} \sigma(z_i,w_i)$$

"מרחק העריכה" (edit distance) בין מילה X למילה Y הוא המחיר המינימאלי של התאמה בין X ו Y.**

$$d(X,Y)=\min\{\text{cost}(A): A \text{ is an alignment of } X,Y\}$$

כעת נגדיר את בעיית התאמת המחרוזות:

<u>קלט</u>: זוג מחרוזות $(X,Y)=(x_1...x_m, y_1...y_n)$ מעל $\Sigma^* \times \Sigma^*$, ופונקציית מחיר $\sigma:[(\Sigma\cup\{-\})\times(\Sigma\cup\{-\})\setminus\{(-,-)\}]\mapsto\Re$.

<u>פלט</u>: $d(X,Y)$, המרחק בין X ו Y.

לעתים נרצה גם להחזיר התאמה אופטימאלית $A^*=(E^*(X);E^*(Y))$ כך ש: $d(X,Y)=\text{cost}(A^*)$.

האלגוריתם לפתרון בעיה זו מבוסס על הלמה הבאה:

<u>למה 9.3</u>: יהיו $X=x_1...x_m, Y=y_1...y_n$ מחרוזות לא ריקות מעל א"ב Σ, ותהי σ פונקציית מחיר נתונה. אז $d(X,Y)$ מקיימת את השוויון הבא:

$$d(X,Y)=\min\begin{cases} d(x_1...x_{m-1}, y_1...y_{n-1})+\sigma(x_m,y_n) \\ d(x_1...x_{m-1}, y_1...y_n)+\sigma(x_m,-) \\ d(x_1...x_m, y_1...y_{n-1})+\sigma(-,y_n) \end{cases}$$

<u>הוכחה</u>: תהי $A=(E(X);E(Y))$ התאמה אופטימאלית של (X,Y) כלומר $d(X,Y)=\text{cost}(A)$.

נניח ש ב A יש k עמודות.

המחיר של A הוא [מחיר k-1 העמודות הראשונות ב A] + [מחיר העמודה האחרונה].

העמודה האחרונה ב A היא אחת משלוש העמודות המתאימות ל$match$, $insert$ או $delete$:

x_m	-	x_m
y_n	y_n	-

ובהתאם לכך, k-1 העמודות הראשונות של A הן התאמה אופטימאלית של $(x_1...x_{m-1}, y_1...y_{n-1})$ או של $(x_1...x_m, y_1...y_{n-1})$ או של $(x_1...x_{m-1}, y_1...y_n)$.

במקרה $match$ המחיר הוא $d(X,Y)=d(x_1...x_{m-1}, y_1...y_{n-1})+\sigma(x_n,y_n)$,

** לעתים מגדירים "מרחק עריכה" כמרחק המוגדר ע"י פונקציית המחיר $\sigma(x,x)=0$, ו $\sigma(x,y)=1$ לכל $x\neq y$.

במקרה insert המחיר הוא $d(X,Y)=d(x_1...x_m, y_1...y_{n-1})+\sigma(-,y_n)$,

במקרה delete המחיר הוא $d(X,Y)=d(x_1...x_{m-1}, y_1...y_n)+\sigma(x_m,-)$ ■

האלגוריתם מחזיר מטריצה $D(0:m,0:n)$ כך שעבור $0\le i\le m, 0\le j\le n$ מתקיים

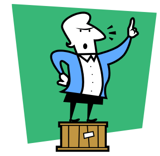
$D(i,j)=d(x_1...x_i, y_1...y_j)$. במיוחד $D(m,n)=d(X,Y)$.

<u>אתחול</u> (אתחל ערכי שורה 0 ועמודה 0): $D(0,0):=0$ עבור $k=1$ עד $k=m$ בצע $D(k,0):=D(k-1,0)+\sigma(x_k,-)$ עבור $k=1$ עד $k=n$ בצע $D(0,k):=D(0,k-1)+\sigma(-,y_k)$ **<u>איטרציות</u>** עבור $i=1$ עד $i=m$ עבור $j=1$ עד $j=n$ $Match:=D(i-1,j-1)+\sigma(x_i,y_j)$ $Delete:=D(i-1,j)+\sigma(x_i,-)$ $Insert:=D(i,j-1)+\sigma(-,y_j)$ $D(i,j):=\min(Match, Delete, Insert)$ החזר את $D(m,n)$.

ניתן למצוא התאמה אופטימאלית על ידי שחזור סדרת הפעולות שמביאה לפתרון האופטימאלי, מהסוף להתחלה.

<u>סיבוכיות</u> זמן: שלב האתחול דורש $O(m+n)$. שלב האיטרציות דורש $O(mn)$, וזהו חסם על סיבוכיות אלגוריתם כולו.

<u>נכונות:</u> ניתן להוכחה באינדוקציה כפולה (על i ו j), בהסתמך על למה 9.3, ש $D(i,j)=d(x_1,...,x_i,y_1,..,y_j)$ עבור $i=0,1,...,m$ ו $j=0,1,..,n$.



הרצאה מס' 10

התמרת פורייה מהירה (FFT) וכפל מהיר של פולינומים

1. חזרה על פולינומים: פולינום מדרגה קטנה מ n מעל שדה F (במקרה שלנו: המספרים הממשיים או המרוכבים):

$$A(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{n-1} x^{n-1}$$

לכל ערך $x_0 \in F$ הפולינום A מתאים ערך $y_0 = A(x_0)$. ניתן לחשב את $A(x_0)$ בזמן ליניארי באמצעות כלל $Horner$: $A(x_0) = a_0 + x_0\left(a_1 + x_0\left(a_2 + \ldots + x_0\left(a_{n-2} + x_0 a_{n-1}\right)\ldots\right)\right)$

2. ייצוגים של פולינומים:

2.1. ייצוג סטנדרטי ע"י וקטור המקדמים:

$$A(x) = \sum_{j=0}^{n-1} a_j x^j$$
$$a = (a_0, a_1, a_2, \ldots, a_{n-1})$$

2.2. ייצוג ע"י ערכי הפולינום בנקודות שונות.

משפט יחידות האינטרפולציה הפולינומית: לכל n זוגות של ערכים $(x_0, y_0), (x_1, y_1), (x_2, y_2), \ldots, (x_{n-1}, y_{n-1})$, כך ש- $x_i \neq x_j$ לכל $i \neq j$, קיים פולינום יחיד מדרגה קטנה מ n, $A(x)$ כך ש: $A(x_i) = y_i$ לכל $i = 0,1,\ldots,n-1$.

יחידות: נובעת מכך שלפולינום מדרגה קטנה מ n יש פחות מ n שרשים.

קיום: נוסחת לגרנז' להלן מחשבת את פולינום האינטרפולציה (שדרגתו קטנה מ n) בסבוכיות $\Theta(n^2)$:

$$A(x) = \sum_{k=0}^{n-1} y_k \frac{\prod_{j \neq k}(x - x_j)}{\prod_{j \neq k}(x_k - x_j)}$$

הערה: כשהשדה F אין סופי יש אינסוף אפשרויות לבחור n זוגות ערכים ליצוג פולינום נתון.

יתרון של ייצוג זה: אם נתונים שני פולינומים $A(x), B(x)$, ע"י ערכיהם באותן נקודות, אז יצוג ע"י ערכים של פולינום המכפלה $C(x) = A(x) \cdot B(x)$ באותן נקודות ניתן לחישוב בזמן לינארי במספר הנקודות ע"י ביצוע n המכפלות $C(x_i) = A(x_i) \cdot B(x_i)$

3. מכפלת פולינומים והגדרת הבעיה:

נתונים 2 פולינומים מדרגה קטנה מ n מעל שדה F , מיוצגים על ידי :

$$A(x) = a_0 + a_1x + a_2x^2 + \ldots + a_{n-1}x^{n-1}$$
$$B(x) = b_0 + b_1x + b_2x^2 + \ldots + b_{n-1}x^{n-1}$$

פולינום המכפלה $C(x) = A(x) \cdot B(x)$ הוא פולינום מדרגה קטנה מ $2n$-1 המוגדר על ידי:

$C(x) = c_0 + c_1x + c_2x^2 + \ldots + c_{2n-2}x^{2n-2}$, כאשר $c_j = \sum_{k=0}^{j} a_k b_{j-k}$

דוגמה: כאשר $A(x) = 1 - 2x + x^4, B(x) = 2 - x + x^2$, אז למשל:

$c_4 = a_4b_0 + a_3b_1 + a_2b_2 + a_1b_3 + a_0b_4 = 2 + 0 + 0 + 0 + 0 = 2$. (שימו לב ש $b_4 = b_3 = 0$).

הוקטור $c = (c_0, \ldots, c_{2n-2})$ הוא הקונבולוציה של $a = (a_0, \ldots, a_{n-1})$ ו $b = (b_0, \ldots, b_{n-1})$. מסמנים $c = a \otimes b$. זמן חישוב פולינום המכפלה (או הקונבולוציה) באופן ישיר: $\Theta(n^2)$.

<u>מטרתנו לבצע את המכפלה בסיבוכיות $\Theta(n \log n)$</u>. לצורך זה נשתמש בעובדה שכאשר מייצגים את הפולינומים A ו B באמצעות ערכיהם ב N - נקודות, ניתן להגיע ל"יצוג באמצעות ערכים" של C בזמן $O(N)$ ע"י $C(x_i) = A(x_i) \cdot B(x_i)$ עבור $i = 0,..,N-1$ (שימו לב שצריך להתקיים $N \geq 2n-1$ כדי להבטיח יצוג יחיד של C). מה שדרוש הוא מעבר מ"יצוג על ידי מקדמים" ל"יצוג ע"י ערכים" ובחזרה. לצורך זה משתמשים בהתמרת פורייה המהירה (FFT), המשתמשת ביצוג הפולינומים באמצעות ערכיהם על שרשי היחידה.

4. תזכורת מספרים מרוכבים ותכונות של שרשי היחידה:

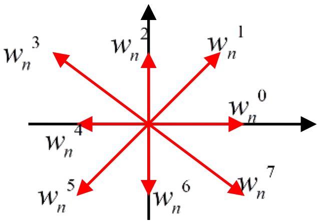
מספר מרוכב ניתן ליצוג $z = |z| e^{i\theta} = |z| (\cos\theta + i \sin\theta)$, כאשר $i = \sqrt{-1}$.

נוסחת De Moivre: $(\cos\theta + i\sin\theta)^n = \cos n\theta + i \sin n\theta$.

מספר מרוכב w הוא שורש יחידה מסדר n אם $w^n = 1$. מנוסחת De Moivre נובע שיש בדיוק n שורשי יחידה מרוכבים מסדר n , והם נתונים ע"י $e^{i\frac{2\pi k}{n}}$, $k = 0, 1, 2, \ldots, n-1$.

אם בנוסף $w^k \neq 1$ עבור $0 < k < n$, w הוא שורש יחידה <u>פרימיטיבי</u> מסדר n. נסמן את שורש היחידה הפרימיטיבי $e^{i\frac{2\pi}{n}}$ ב w_n . אז n שורשי היחידה הם: $w_n^{\,0}, w_n^{\,1}, w_n^{\,2}, \ldots, w_n^{\,n-1}$.

הציור להלן מתאר את שרשי היחידה עבור n=8.



5. הרעיון הכללי של האלגוריתם – שימוש בהתמרת פורייה המהירה FFT:

5.1. נייצג את הפולינומים A ו B ע"י ערכיהם ב $2n$ שורשי היחידה מסדר $2n$. המעבר למיצוג ע"י מקדמים ליצוג זה נקרא התמרת פוריה הבדידה (DFT או DFT_{2n}). נראה איך לממשו ע"י אלגוריתם FFT בזמן $\Theta(n\log n)$. כפי שנראה המימוש היעיל של FFT דורש ש n יהיה חזקה של 2.

5.2. נכפיל את שני הפולינומים A ו B בייצוג הנ"ל ונקבל ייצוג של המכפלה C ע"י ערכים ב $2n$ שורשי היחידה מסדר $2n$. סבוכיות $\Theta(n)$.

5.3. נעבור מייצוג של C על ידי ערכיו ליצוג ע"י וקטור מקדמיו. המעבר נקרא התמרת פוריה ההפוכה $\left(DFT^{-1}\right)$ וממומש ע"י FFT (עם שינויים קלים) בזמן $\Theta(n\log n)$ (במקום בזמן ריבועי ע"י נוסחת לגרנז').

שלושת השלבים האלו הם "משפט הקונבולוציה": עבור 2 וקטורים a,b באורך n מתקיים:

$$a\otimes b=DFT_{2n}^{-1}\left(DFT_{2n}(a)\cdot DTF_{2n}(b)\right)$$

ניתן לתארם בסכימה הבאה:

$$\begin{array}{ccc}
\begin{array}{c} a_0,a_1,a_2,\dots,a_{n-1} \\ b_0,b_1,b_2,\dots,b_{n-1} \end{array} & \xrightarrow{\Theta(n^2)} & c_0,c_1,c_2,\dots,c_{2n-2} \\
\Big\downarrow{\scriptstyle \Theta(n\log n)} & & \Big\uparrow{\scriptstyle \Theta(n\log n)} \\
\begin{array}{c} A\left(w_{2n}^{\ 0}\right),A\left(w_{2n}^{\ 1}\right),\dots A\left(w_{2n}^{\ 2n-1}\right) \\ B\left(w_{2n}^{\ 0}\right),B\left(w_{2n}^{\ 1}\right),\dots B\left(w_{2n}^{\ 2n-1}\right) \end{array} & \xrightarrow{\Theta(n)} & C\left(w_{2n}^{\ 0}\right),C\left(w_{2n}^{\ 1}\right),\dots,C\left(w_{2n}^{\ 2n-1}\right)
\end{array}$$

6. FFT (Fast Fourier Transform) (התמרת פוריה המהירה):

נתון פולינום $A(x)$ שדרגתו קטנה מ n וצריך לחשב את ערכיו ב n שורשי היחידה מסדר n :

כלומר צריך לחשב את הערכים $y_k = A\left(w_n{}^k\right) = \sum_{j=0}^{n-1} a_j \cdot \left(w_n{}^k\right)^j$ עבור $k = 0,1,2,...n-1$

הוקטור $y = (y_0, y_1, ..., y_{n-1})$ הוא התמרת פוריה הבדידה (Discrete Fourier Transform) של $a = (a_0, a_1, ...a_{n-1})$ המסומנת: $y = DFT(a)$.

נניח ש n חזקה של 2 (אם יש צורך נוסיף אפסים מובילים מימין). נחלק את $A(x)$ לחזקות זוגיות ואי זוגיות:

$$A(x) = (a_0 + a_2x^2 + a_4x^4 + ... + a_{n-2}x^{n-2}) + x(a_1 + a_3x^2 + a_5x^4 + ... + a_{n-1}x^{n-2})$$

באמצעות חלוקה זו נגדיר שני פולינומים מדרגה קטנה מ $\frac{n}{2}$:

$$A_{even}(z) = a_0 + a_2z + a_4z^2 + a_6z^3 + ... + a_{n-2}z^{n/2-1}$$
$$A_{odd}(z) = a_1 + a_3z + a_5z^2 + a_7z^3 + ... + a_{n-1}z^{n/2-1}$$

נציב $z = x^2$ ונקבל: $A(x) = A_{even}\left(x^2\right) + x \cdot A_{odd}\left(x^2\right)$. נסמן שוויון זה ב (*).

כדי לחשב את $A(w_n^k)$ עבור k=0,..,n-1, ניתן להשתמש ב (*) ובערכי A_{even} ו A_{odd} בנקודות $\{w_n^{2k} \mid k = 0,1,...,n-1\}$. מאחר ו n זוגי, הרי קבוצה זו היא בדיוק קבוצת שרשי היחידה מסדר $\frac{n}{2}$: $\{w_n^{2k} \mid k = 0,1,..,n/2-1\}$. לכן ניתן לחשב את ערכי הפולינום A ב n שרשי היחידה מסדר n באמצעות הערכים של כל אחד מהפולינומים A_{even} ו A_{odd} ב $\frac{n}{2}$ שרשי היחידה מסדר $\frac{n}{2}$, שאותם ניתן למצוא על ידי פתרון שתי בעיות FFT, כל אחת מהן בגודל $\frac{n}{2}$. זה מכתיב אלגוריתם הפותר את הבעיה עבור קלט בגודל n על ידי פיתרון רקורסיבי של 2 בעיות בגודל $\frac{n}{2}$. להלן האלגוריתם:

אלגוריתם RECURSIVE_FFT

קלט: a=(a_0,a_1,…,a_{n-1}) - סדרת מקדמי הפולינום $A(x)$, n חזקה של 2[††].

פלט: y=(y_0,y_1,…,y_{n-1}), כך ש $y_k = A(w_n^k)$, כלומר y=DFT$_n(a)$.

[††] כאמור, כל אחד ממקדמי הפולינום יכול להיות 0.

אם $n=1$ החזר $y=(a_0)$, אחרת:

קריאה לאלגוריתם עבור שני קלטים $\begin{cases} y^{[0]}=(y_0^{[0]},\dots,y_{n/2-1}^{[0]}) \leftarrow RECURSIVE\_FFT(a_0,a_2,\dots,a_{n-2}) \\ y^{[1]}=(y_0^{[1]},\dots,y_{n/2-1}^{[1]}) \leftarrow RECURSIVE\_FFT(a_1,a_3,\dots,a_{n-1}) \end{cases}$

בגודל $n/2$.

עבור $k=0$ עד $k=\frac{n}{2}-1$ בצע: $y_k=y_k^{[0]}+w_n^k y_k^{[1]}$; $y_{\frac{n}{2}+k}=y_k^{[0]}-w_n^k y_k^{[1]}$

[השתמשנו בעובדה ששורש יחידה פרימיטיבי מסדר n מקיים $w^{\frac{n}{2}+k}=-w^k$.]

החזר $y=(y_0,y_1,..,y_{n-1})$

סיבוכיות: $\begin{cases} T(n)=2T(n/2)+\Theta(n) \\ T(1)=O(1) \end{cases} \Rightarrow T(n)=\Theta(n\log n)$

7. נותר עדיין להראות איך לעבור מהיצוג של פולינום $A(x)$ מדרגה קטנה מ n על ידי וקטור y של ערכיו ב n שרשי היחידה, ליצוג שלו באמצעות וקטור a של מקדמיו. לצורך זה נציג את y באמצעות כפל מטריצה ון דר מונדה של שרשי היחידה V_n בוקטור עמודה a.

כתיבה מפורשת של הכפל $V_n a=y$ מופיעה כאן:

$$\underbrace{\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & w_n & w_n^{\,2} & \dots & w_n^{\,n-1} \\ 1 & w_n^{\,2} & (w_n^{\,2})^2 & \dots & (w_n^{\,2})^{n-1} \\ \vdots & & & & \\ 1 & w_n^{\,n-1} & (w_n^{\,n-1})^2 & \dots & (w_n^{\,n-1})^{n-1} \end{pmatrix}}_{V_n} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{n-1} \end{pmatrix}$$

המטריצה V_n אינה סינגולרית, לכן a מקיים את השויון $a=V_n^{-1}\cdot y$. כעת נראה ש V_n^{-1}היא מטריצה בעלת מבנה דומה לזה של V_n כאשר במקום w_n משתמשים ב w_n^{-1} , ולכן ניתן לחשבה באמצעות FFT. לצורך זה נשתמש בלמה הבאה:

<u>**למה (סכום שרשי היחידה):**</u> יהי $w\neq 1$ שורש יחידה מסדר n. אז $\sum_{k=0}^{n-1} w^k=0$

הוכחה: ע"פ נוסחת טור הנדסי $\sum_{k=0}^{n-1} w^k=\frac{w^n-1}{w-1}=\frac{1-1}{w-1}=0$

טענה: $[V_n^{-1}]_{j,k}=\frac{w_n^{-jk}}{n}$ עבור $0\le j,k\le n-1$, כלומר:

$$\underbrace{\frac{1}{n}\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & w_n^{-1} & ((w_n)^{-1})^2 & \dots & ((w_n)^{-1})^{n-1} \\ 1 & ((w_n)^{-1})^2 & \left(((w_n)^{-1})^2\right)^2 & \dots & \left(((w_n)^{-1})^2\right)^{n-1} \\ \vdots & & & & \\ 1 & \left((w_n)^{-1}\right)^{n-1} & \left(((w_n)^{-1})^{n-1}\right)^2 & \dots & \left(((w_n)^{-1})^{n-1}\right)^{n-1} \end{pmatrix}} \cdot \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{n-1} \end{pmatrix} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{pmatrix}$$

הוכחה: נראה ש: $V_n^{-1} \cdot V_n = I$. לצורך זה נחשב את ערך הכניסה ה j', j במטריצת המכפלה

$$(V_n^{-1} \cdot V_n)_{j',j} = \sum_{k=0}^{n-1} \frac{w_n^{-j'k}}{n} \cdot w_n^{jk} = \frac{1}{n} \sum_{k=0}^{n-1} w_n^{k(j-j')}$$

אם $j' = j$ אז התוצאה היא 1. אחרת $w_n^{j-j'}$ הוא שורש יחידה מסדר n שונה מ 1 (כי $0 < |j-j'| < n$), ולכן לפי למת סכום שרשי היחידה, התוצאה היא 0. מ.ש.ל.

מסקנה: $a = V_n^{-1} \cdot y$ כלומר המקדם a_k נתון ע"י:

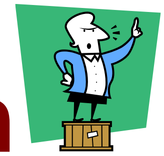
$$a_k = \sum_{j=0}^{n-1} \left(V_n^{-1}\right)_{kj} \cdot y_j = \frac{1}{n} \sum_{j=0}^{n-1} y_j \cdot w_n^{-jk}$$

$$y_k = \sum_{j=0}^{n-1} a_j \cdot \left(w_n^{k}\right)^j$$

כלומר המקדמים a_k, $k = 0,..,n-1$ הם ערכי הפולינום שמקדמיו $\frac{y_j}{n}$ ב n שורשי היחידה מסדר n: $\left(w_n^{-1}\right)^0, \left(w_n^{-1}\right)^1, ..., \left(w_n^{-1}\right)^{n-1}$

מכאן שניתן לחשב את מקדמי הפולינום $A(x)$ מוקטור y של ערכיו ב n שרשי היחידה על ידי הכנסת השינויים הבאים באלגוריתם ל FFT שניתן בתחילת ההרצאה:

1. להחליף תפקידים בין a ו y.
2. להחליף שורש היחידה הפרימיטיבי w_n בשורש הפרימיטיבי ההופכי w_n^{-1}.
3. לבצע FFT עם הקלט המוגדר בסעיפים 1 ו 2.
4. לחלק את אברי וקטור התוצאה ב n.

 הרצאה מס' 11

רשתות זרימה, פונקציות זרימה וזרימת מקסימום

רשת זרימה היא רביעיה $N=(G,s,t,c)$, כאשר $G=(V,E)$ הוא גרף מכוון ללא לולאות או קשתות מקבילות. $s,t\in V$ כאשר s הוא צומת "מקור" ו- t הוא צומת "בור". $c:E\to R^{+}$ היא פונקציית קיבול המגדירה לכל קשת $e\in E$ קיבול אי שלילי $c(e)$. לכל צומת v, $in(v)$ היא קבוצת הקשתות הנכנסות לצומת ו $out(v)$ היא קבוצת הקשתות היוצאות ממנו.

פונקצית זרימה $f:E\to R^{+}$, היא פונקציה המגדירה לכל קשת e את כמות הזרימה $f(e)$ העוברת בה. פונקצית זרימה חייבת לקיים שני אילוצים:

(i) <u>אילוץ הקיבול</u> (נקרא גם <u>חוק הקשת</u>) לכל קשת מתקיים $0\le f(e)\le c(e)$, כלומר הזרימה דרך קשת היא אי שלילית ואינה יכולה לחרוג מקיבול הקשת.

(ii) <u>שימור הזרימה</u> (נקרא גם <u>חוק הצומת</u>). לכל צומת $v\in V\setminus\{s,t\}$ (כלומר שאינו מקור או בור) מתקיים $\sum\limits_{e\in out(v)} f(e)=\sum\limits_{e\in in(v)} f(e)$, כלומר סה"כ הזרימה הנכנסת לצומת שווה לסה"כ הזרימה היוצאת מממנו.

<u>ערך (או חוזק) פונקצית הזרימה</u> הוא הזרימה נטו הנכנסת לבור, ומסומן ב $|f|$:

$$|f| = \sum_{e\in in(t)} f(e)-\sum_{e\in out(t)} f(e)$$

<u>השאלה שבה נעסוק</u>: בהינתן רשת זרימה, מהי הזרימה המקסימלית האפשרית בה?
<u>הערה</u>: על מנת לפשט את ניתוחי הסיבוכיות נניח שגרף התשתית קשיר, ולכן מתקיים $O(V+E)=O(E)$.

חתך s-t ברשת זרימה

חתך $(S,\bar{S})$ בגרף הוא חלוקה של V לקבוצות זרות ומשלימות: $S\subset V,\ \bar{S}=V-S$.

אם החתך מקיים $s\in S,\ t\in\bar{S}$, הוא נקרא "חתך s-t".

כל החתכים שנעסוק בהם בפרק זה הינם חתכי s-t, אלא אם נכתב במפורש אחרת.

יהי $(S,\bar{S})$ חתך נתון. $(S\to\bar{S})$ היא קבוצת הקשתות החוצות את החתך מ s ל $\bar{S}$ (ב"כיוון הקדמי"): $(S\to\bar{S})=\{(u,v)\in E: u\in S, v\in\bar{S}\}$.

$(\bar{S}\to S)=\{(v,u)\in E: u\in S, v\in\bar{S}\}$ היא קבוצת הקשתות החוצות אותו ב"כיוון האחורי".

למה 11.1

לכל חתך $(S,\bar{S})$ ולכל פונקצית זרימה f מתקיים

$$|f| = \sum_{e\in(S\to\bar{S})} f(e) - \sum_{e\in(\bar{S}\to S)} f(e)$$

הוכחה

נחשב בשתי דרכים את הסכום הבא:

$$(*) = \sum_{v\in\bar{S}}\left(\sum_{e\in in(v)} f(e) - \sum_{e\in out(v)} f(e)\right)$$

דרך א): נשים לב כי משימור הזרימה נקבל לכל $v\in\bar{S}-\{t\}$:

$$\sum_{e\in in(v)} f(e) - \sum_{e\in out(v)} f(e) = 0$$

לכן על פי ההגדרה של $|f|$:

$$(*) = \sum_{e\in in(t)} f(e) - \sum_{e\in out(t)} f(e) = |f|$$

דרך ב): נשים לב שלכל קשת $e=(u,v)$, אם $u,v\in\bar{S}$, או $u,v\in S$, התרומה של e ל $(*)$ היא אפס (במקרה הראשון $f(e)$ מופיע פעמיים בסכום – פעם אחת עם "+" ופעם שניה עם "-", ובמקרה השני $f(e)$ אינו מופיע כלל). לכן, $|f|$ מתקבל על ידי סכימת ערכי $f(e)$ עבור קשתות החתך: קשתות קדמיות עם סימן "+", ואחוריות עם סימן "-":

$$|f| = (*) = \sum_{e\in(S\to\bar{S})} f(e) - \sum_{e\in(\bar{S}\to S)} f(e)$$

■

כאשר מציבים בלמה 11.1 $S=\{s\}$ מתקבל שערך הזרימה שווה לזרימה נטו היוצאת מהמקור.

<u>קיבול של חתך</u> $(S,\bar{S})$ הוא סכום קיבולי הקשתות החוצות אותו בכיוון הקדמי:

$$c(S,\bar{S}) = \sum_{e\in(S\to\bar{S})} c(e)$$

<u>חתך מינימום</u> הוא חתך (s,t) בעל קיבול מינימאלי.

למה 11.2

לכל חתך $(S,\bar{S})$ ולכל פונ' זרימה f מתקיים: $|f| \le c(S,\bar{S})$.

הוכחה

$$|f| = \sum_{e\in(S\to\bar{S})} f(e) - \sum_{e\in(\bar{S}\to S)} f(e) \le \sum_{e\in(S\to\bar{S})} f(e) \le \sum_{e\in(S\to\bar{S})} c(e) = c(S,\bar{S})$$

למה 11.1

כי $f(e)\ge 0$

מסקנה

אם קיים חתך $(S,\bar{S})$ כך ש $F=c(S,\bar{S})$, אז f היא זרימת מקסימום, ו $(S,\bar{S})$ הוא חתך מינימום (בעל קיבול מינימאלי).

הגרף השיורי ומסלולי שיפור

האלגוריתמים שנציג למציאת זרימת מקסימום הם אלגוריתמים חמדניים, אשר בכל איטרציה מגדילים את הזרימה באמצעות "מסלול שיפור" מהמקור לבור. לצורך הצגתם נחוצות מספר הגדרות:

רשת זרימה G ופונקציית זרימה f המוגדרת עליה מגדירים לכל קשת $e=(u,v)$ עם קיבול $c(e)$ וזרימה $f(e)$ שתי קשתות אנטי מקבילות עם קיבול שיורי (residual capacity):

- קשת קדמית (u,v) עם קיבול שיורי $c(e)-f(e)$.
- קשת אחורית (v,u) עם קיבול שיורי $f(e)$.

(הקיבולים השיוריים מיצגים את הזרימה שניתן להוסיף או להחסיר על הקשת (u,v).)

זרימה וקיבול	קיבול שיורי
$u \xrightarrow{f(e)\ ,\ c(e)} v$	$u \xrightarrow{c(e)-f(e)} v$ $u \xleftarrow{f(e)} v$

שימו לב שהקיבול השיורי הוא תמיד אי שלילי.

- <u>הגרף השיורי</u> G_f הוא תת הגרף של G המכיל את הקשתות עם קיבול שיורי חיובי ביחס לזרימה f, ואת צמתי הקצה של קשתות אלו.
- <u>מסלול שיפור</u> (ביחס ל G ו f) הוא מסלול מכוון מ- s ל- t בגרף השיורי G_f.

למה 11.3 יהי p מסלול שיפור ו Δ הקיבול השיורי הקטן ביותר לאורך p. אז אם לכל קשת קדמית ב p נגדיל בגרף G את הזרימה בקשת המתאימה ב- Δ, ולכל קשת אחורית ב- p

נקטין את הזרימה בקשת המתאימה ב- Δ, נקבל פונקציית זרימה חוקית שמגדילה את ערך הזרימה ב- Δ.

הוכחה

חוק הקשת: תהי e קשת במסלול p. מהגדרת הקיבול השיורי נובע כי אם e קשת אחורית מתקיים $\Delta \le f(e)$, ואם e קשת קדמית מתקיים $\Delta \le c(e) - f(e)$. בשני המקרים ערך הזרימה המתקבל לאחר השינוי נמצא בתחום $[0, c(e)]$, ולכן חוק הקשת נשמר.

כעת נוכיח שגם חוק הצומת נשמר. מספיק להראות זאת עבור הצמתים הפנימיים ב- p (למה?):

$p = s \to \cdots \xrightarrow{e_1} v \xrightarrow{e_2} \cdots \to t$

נטפל בארבע האפשרויות ש e_1, e_2 מייצגות קשתות קדמיות או אחוריות בגרף המקורי.

- אם e_1, e_2 קדמיות, אז אנו מבצעים $\xrightarrow{+\Delta} v \xrightarrow{+\Delta}$ ומתקיים שימור הזרימה עבור v.
- אם e_1 אחורית ו- e_2 קדמית אז אנו מבצעים $\xleftarrow{-\Delta} v \xrightarrow{+\Delta}$ ושוב מתקיים שימור הזרימה עבור v.
- אם e_1 קדמית ו- e_2 אחורית אז אנו מבצעים $\xrightarrow{+\Delta} v \xleftarrow{-\Delta}$ ומתקיים שימור הזרימה עבור v.
- אם e_1, e_2 אחוריות, אז אנו מבצעים $\xleftarrow{-\Delta} v \xleftarrow{-\Delta}$ וגם במקרה זה מתקיים שימור הזרימה עבור v.

נראה כי אכן הגדלנו את F ב- Δ. נסתכל על המסלול p, הצומת האחרון במסלול זה הוא t.

$p = s \to \cdots \xrightarrow{e} t$

- אם e קשת קדמית אז אנו במצב בו $p = s \to \cdots \xrightarrow{+\Delta} t$.
- אם e קשת אחורית נקבל $p = s \to \cdots \xleftarrow{-\Delta} t$.

בשני המקרים הגדלנו את חוזק הזרימה ב Δ. ■

משפט 11.4: חתך מינימום – זרימת מקסימום (Min Cut – Max Flow)

תהי f פונקצית זרימה ברשת זרימה $N=(G,s,t,c)$. הטענות הבאות שקולות:

1. f זרימת מקסימום
2. אין מסלול שיפור מ- s ל- t בגרף השיורי G_f.
3. קיים חתך $s-t$ $(S,\bar{S})$ עבורו $F=c(S,\bar{S})$.

הוכחה

נראה $1\Rightarrow 2\Rightarrow 3\Rightarrow 1$.

א. $1\Rightarrow 2$: אם קיים מסלול שיפור ב G_f אז ניתן להגדיל את f בΔ (הקיבול השיורי המינימאלי במסלול), ולכן מלמה 11.3 f אינה זרימת מקסימום.

ב. $2\Rightarrow 3$: נגדיר קבוצה של צמתים S:

$$S=\{v: there\ is\ a\ path\ in\ G_f\ from\ s\ to\ v\}$$

ברור כי $s\in S$. מההנחה שאין מסלול שיפור מ s ל t, נובע גם ש $t\notin S$. לכן $(S,\bar{S})$ הוא חתך $s-t$.

מהגדרת S מתקבל ש $f(e)=0$ עבור $e\in(\bar{S}\to S)$, ו $f(e)=c(e)$ עבור $e\in(S\to\bar{S})$. מכך מתקבל:

$$|f|=\sum_{e\in(S\to\bar{S})} f(e)-\sum_{e\in(\bar{S}\to S)} f(e)=\sum_{e\in(S\to\bar{S})} c(e)-\sum_{e\in(\bar{S}\to S)} 0=\sum_{e\in(S\to\bar{S})} c(e)=c(S,\bar{S})$$

למה 11.1

הגדרת S

ג. $3\Rightarrow 1$: נובע ישירות מהמסקנה שאחרי למה 2.11.■

כפי שהזכרנו, חתך המקיים את השוויון במשפט 11.4(3) נקרא "חתך מינימום", שכן הוא בעל קיבול מינימלי אפשרי (על סמך למה 11.1).

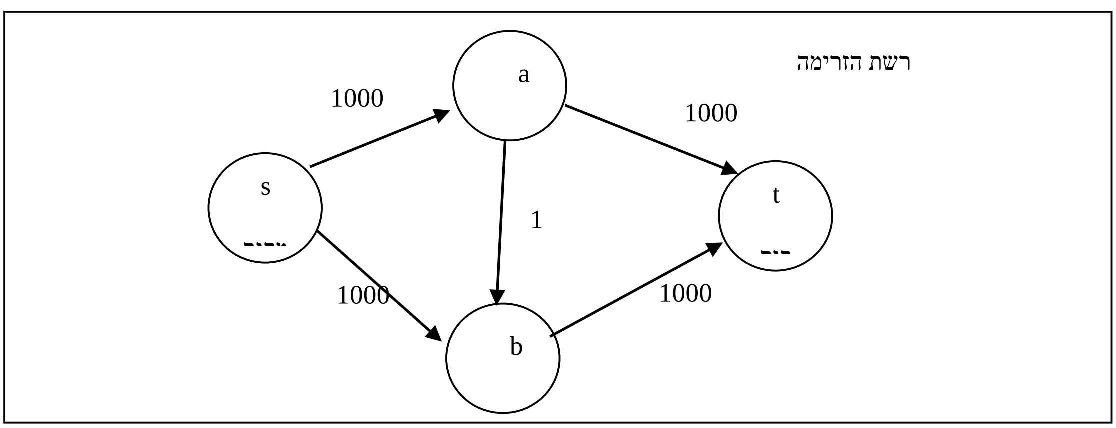
האלגוריתם הגנרי של פורד-פלקרסון לזרימת מקסימום

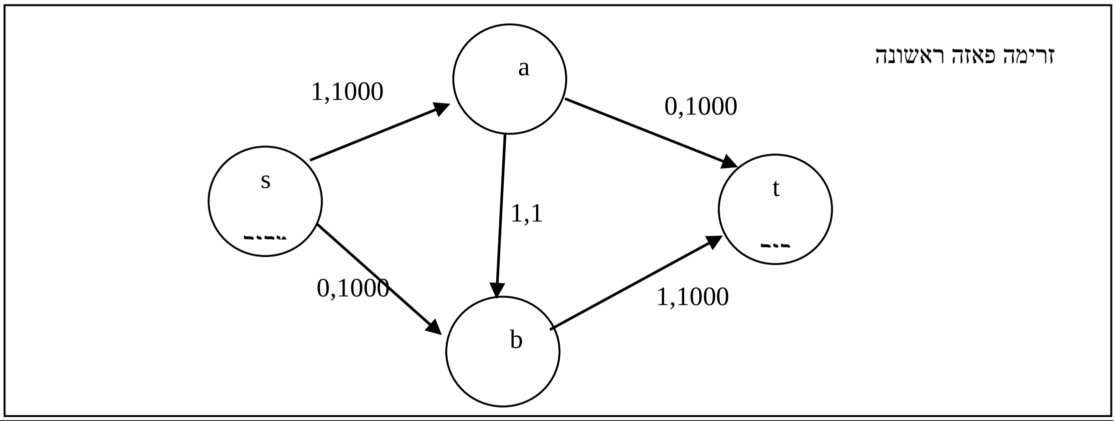
- אתחול: קבע $f(e)=0$ לכל קשת e.
- כל עוד קיים מסלול שיפור p מ s אל t ב G_f, שפר את הזרימה לאורך p.

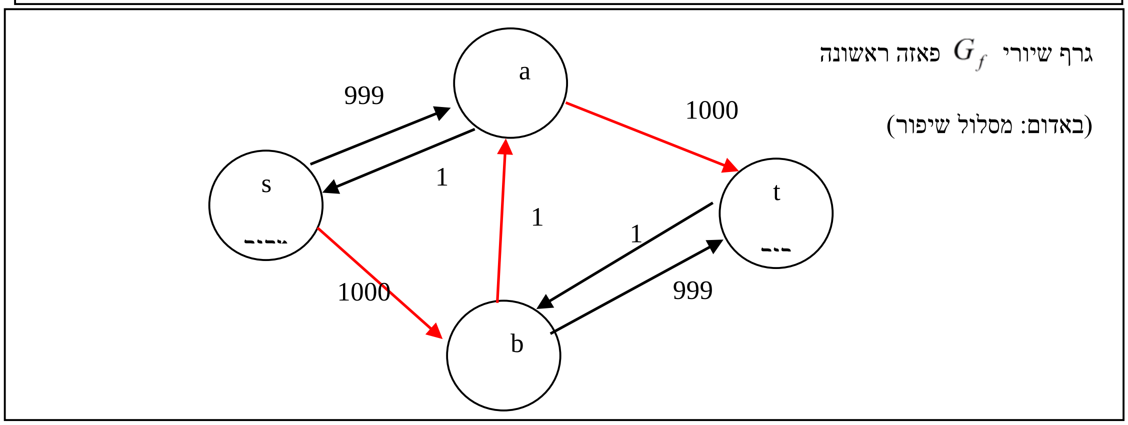
הערות

- זהו "אלגוריתם גנרי" – הוא לא קובע את האופן בו נבחר מסלול שיפור בכל איטרציה.
- אם האלגוריתם עוצר אז אין מסלול שיפור ולכן (ממשפט 11.4) f היא זרימת מקסימום.
- יש דוגמא של רשת זרימה בה הקיבולים על הקשתות אינם רציונאליים, והאלגוריתם מייצר סדרה אין סופית של מסלולי שיפור ולא עוצר. במקרה כזה ערך פונקציות הזרימה תמיד מתכנס (תרגיל בחדו"א 1...), אך לא קשה להראות שהוא עלול להתכנס לזרימה שאינה זרימת מקסימום.
- אם לכל קשת e, הזרימה $f(e)$ והקיבול $c(e)$ הם <u>מספרים שלמים</u>, אז $\Delta(p)$ הוא מספר שלם חיובי לכל מסלול שיפור p, ולכן ערך הזרימה גדל כל איטרציה בלפחות 1. מכאן נובע שמספר האיטרציות הוא לכל היותר חוזק הזרימה של זרימת מקסימום. לכן כשהקיבולים שלמים האלגוריתם תמיד יעצור וימצא זרימת מקסימום בשלמים. (אם הקיבולים רציונאליים אז אפשר לעשות רדוקציה לזרימה במספר שלמים, ולכן גם אז האלגוריתם תמיד עוצר).

להלן דוגמה בה מספר האיטרציות הוא בדיוק הערך של זרימת מקסימום $|f|=2000$:





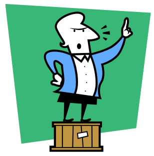


2000 איטרציות עבור בחירה גרועה של מסלולי שיפור.

בהמשך נציג שני אלגוריתמים יעילים יותר המבוססים על מציאת מסלולי שפור קצרים :

- אלגוריתם Edmonds Karp בסיבוכיות $O\left(|V||E|^2\right)$
- אלגוריתם של דיניץ בסיבוכיות $O\left(|V|^2|E|\right)$

(הסיבוכיות מחושבת בהנחה שגרף התשתית קשיר, כלומר $E \geq V-1$).



הרצאה מס' 12

החומר בפרק זה מספיק ליותר מהרצאה אחת. מומלץ להעביר רק אחד משני האלגוריתמים: אדמונדס-קרפ או דיניץ , ואם נשאר זמן גם את "מסלולי שיפור מקסימאליים".

אלגוריתם אדמונדס-קרפ

האלגוריתם של אדמונדס-קרפ הוא המימוש הבא של האלגוריתם הגנרי של פורד ופלקרסון: בכל איטרציה נבחר מסלול שיפור <u>קצר ביותר</u> מהמקור לבור.

<u>הגדרה:</u> <u>קשת קריטית</u> במסלול שיפור היא קשת בעלת קיבול שיורי מינימאלי במסלול.

נוכיח שהסבוכיות של האלגוריתם של אדמונדס-קרפ היא $O(VE^2)$ על ידי כך שנראה שכאשר מבצעים שיפורים לארך מסלולי שיפור קצרים ביותר, כל קשת יכולה להיות קשת קריטית במסלול שיפור שהאלגוריתם שיפר באמצעותו את הזרימה $O(V)$ פעמים.

בהינתן זרימה f, נסמן ב $d_f(v) = dist_f(s,v)$ את המרחק (אורך המסלול הקצר ביותר) מ s אל v ב G_f. נשים לב שאם קשת (u,v) נמצאת במסלול שיפור קצר ביותר, אז חייב להתקיים $d_f(v) = d_f(u) + 1$.

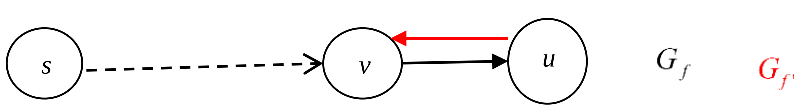
הטענה הבאה מאפיינת קשתות שנוספות או מסולקות מהגרף השיורי כתוצאה מפעולת השיפור. נכונותה נובעת ישירות מההגדרה של הגרף השיורי ושל מסלולי שיפור:

<u>למה 12.1:</u> תהי f הזרימה לפני פעולת שיפור לארך מסלול שיפור קצר ביותר. אזי:

א. אם (u,v) היא קשת קריטית במסלול השיפור אז לאחר ביצוע השיפור (u,v) מסולקת מהגרף השיורי.

ב. קשת (u,v) נוספת לגרף השיורי לאחר ביצוע השיפור רק אם הקשת האנטי מקבילה (v,u) נמצאת על מסלול השיפור, ולכן $d_f(v) = d_f(u) - 1$ (ראה ציור).

<u>הוכחה:</u> יהי p מסלול שיפור עם קיבול מינימאלי Δ. לכל קשת (u,v) במסלול, שיפור זרימה ב p מקטין את הקיבול השיורי ב (u,v) ב Δ, ומגדיל את הקיבול השיורי בקשת הנגדית (v,u) ב Δ. מכאן שהקיבול השיורי בכל קשת קריטית מתאפס ולכן קשת כזו מסולקת מהמסלול. זה מוכיח את א. ב. נובע מכך שקשת (v,u) נוספת לגרף השיורי רק אם הזרימה השיורית ב (v,u) גדלה, וע"ס הנכתב לעיל זה יתכן רק אם הקשת הנגדית (u,v) נמצאת במסלול.



ג.

למה 12.2: תהי $\hat{f}$ הזרימה לאחר ביצוע השיפור. לכל קשת $(u,v)\in G_{\hat{f}}$ מתקיים:
$d_f(v)-d_f(u)\le 1$.

הוכחה: אם $(u,v)\in G_f$ הטענה נובעת מכך ש d_f היא פונקציית מרחק על G_f. אחרת מקבלים מלמה 12.1 ב. תנאי חזק יותר: $d_f(v)-d_f(u)=-1$. ■

למה 12.3: תהי $\hat{f}$ כמו בלמה 12.2. לכל צומת v מתקיים: $d_{\hat{f}}(v)\ge d_f(v)$.

הוכחה : נניח ש $d_{\hat{f}}(v)=k$, ויהי $s=v_0\to v_1\ldots\to v_k=v$ מסלול קצר ביותר ב $G_{\hat{f}}$ מ s ל v.
צריך להוכיח ש $d_f(v)\le k$. זה נובע מהשוויונות ואי השוויון הבאים

$$d_f(v)=d_f(v_k)-d_f(v_0)=\sum_{i=1}^{k}[d_f(v_i)-d_f(v_{i-1})]\le\sum_{i=1}^{k}1=k$$

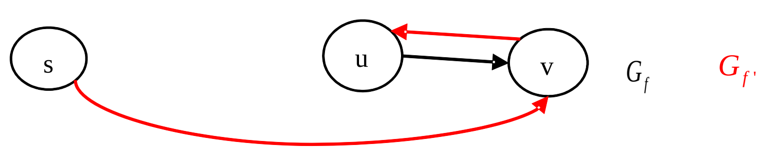
השוויון השמאלי נובע מכך ש $v_0=s,v_k=v$ ולכן $d_f(v_0)=0,\ d_f(v_k)=d_f(v)$, ואי השוויון נובע מלמה 12.2.■

למה 12.4: מספר הפעמים שקשת (u,v) היא קשת קריטית במסלול שיפור ששימש לשיפור הזרימה במהלך האלגוריתם חסום על ידי $V/2$.

הוכחה: תהי f הזרימה בפעם מסוימת בה (u,v) היא קשת קריטית במסלול שיפור כנ"ל, ויהי $d_f(u)=k$. מלמה 12.1 א. נובע שלאחר פעולת השיפור, הקשת (u,v) נעלמת מהגרף השיורי. כך שהפעם הבאה ש (u,v) תהיה קריטית (אם בכלל) תקרה לאחר ש (u,v) תשוב ותופיע בגרף השיורי. ע"ס למה 12.1 ב., (u,v) תוחזר לגרף השיורי כאשר תהיה זרימה $\tilde{f}$ כך שמסלול שיפור קצר ביותר ב $G_{\tilde{f}}$ מכיל את (v,u). במקרה זה מתקיים:

$$d_{\tilde{f}}(u)=d_{\tilde{f}}(v)+1\ge d_f(v)+1=d_f(u)+2$$

השוויון הימני נובע מכך ש (u,v) נמצאת על מסלול שיפור ב G_f, השוויון השמאלי נובע מכך ש (v,u) נמצאת על מסלול השיפור ב $G_{\tilde{f}}$, ואי השוויון מכך ש $d_{\tilde{f}}(v)\ge d_f(v)$ (ע"ס למה 12.3). ראו ציור.



במיוחד, מתקבל שבין שתי פעמים בהן (u,v) קשת קריטית, המרחק בגרף השיורי בין s ל u גדל ב 2 לפחות. הלמה נובעת מכך שבכל גרף שיורי, המרחק מ s ל u קטן מ V.■

משפט 12.5: האלגוריתם של אדמונדס וקרפ עוצר לאחר לכל היותר VE איטרציות.

הוכחה: על סמך למה 12.4, קשת (u,v) יכולה להיות קריטית במסלול שיפור ששימש לשיפור הזרימה לכל היותר $\frac{V}{2}$ פעמים. בכל איטרציה יש לפחות קשת אחת כזו, ויש סה"כ $2E$ קשתות שיכולות להופיע במסלולי שיפור (כל קשת בגרף יכולה להופיע כקשת קדמית או קשת אחורית). לכן מספר האיטרציות הוא לכל היותר $\frac{V}{2}\cdot 2E = VE$ ■

מסקנה: הסיבוכיות של האלגוריתם היא $O(E^2V)$: בכל איטרציה מוצאים מסלול שיפור בזמן $O(E)$, ויש $O(VE)$ איטרציות.

האלגוריתם של דיניץ

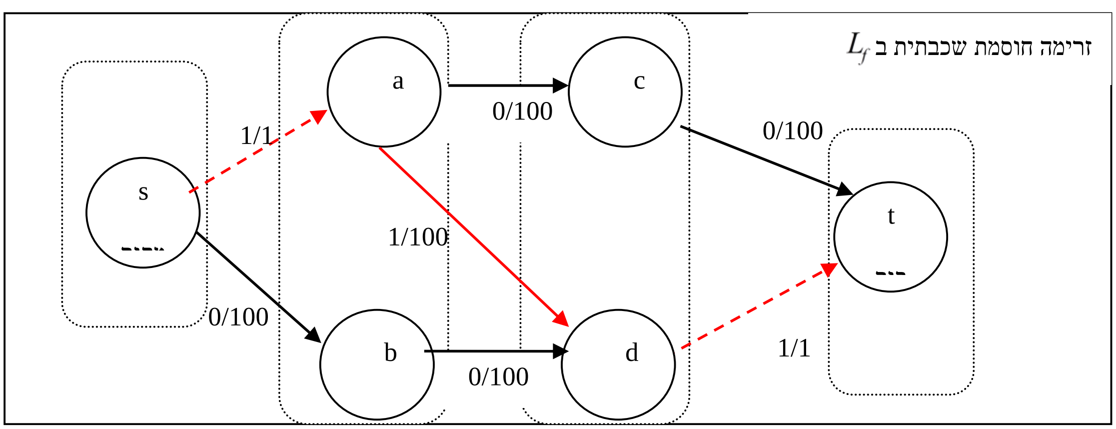
זהו אלגוריתם יעיל יותר המסתמך על רעיון דומה, שסיבוכיות הזמן שלו היא $O\left(V^2E\right)$.

הרעיון: לשפר זרימה בשלבים (פאזות), כך ש $\delta_f(t)$ (ארך מסלול השיפור הקצר ביותר בגרף השיורי) יגדל בכל פאזה. לצורך זה משתמשים ב"גרף שכבות":

גרף השכבות L_f: תת גרף של G_f המכיל רק קשתות של מסלולים קצרים ביותר מ s. כלומר $(u,v)\in L_f$ רק אם $\delta_f(u)+1=\delta_f(v)$. שכבה k ב L_f מכילה את כל הצמתים שמרחקם מ s ב G_f הוא k, והשכבה האחרונה מכילה את t (כלומר הגרף מכיל רק צמתים v כך ש $\delta_f(v)\le\delta_f(t)$).

זרימה חוסמת שכבתית: q היא זרימה חוסמת שכבתית ב L_f אם היא פונקצית זרימה על L_f המקיימת: בכל מסלול מכוון מ s ל t ב L_f יש לפחות קשת רוויה אחת (קשת רוויה: $q(e)=c(e)$).

דוגמה לזרימה חוסמת שכבתית:



כפי שהדוגמה מראה, זרימה חוסמת שכבתית אינה בהכרח זרימת מקסימום בגרף השכבות.

<u>האלגוריתם של דיניץ לזרימת מקסימום</u>

<u>אתחול:</u>

$i \leftarrow 0$ (i הוא מספר הפאזה).

לכל קשת e : $f_0(e) \leftarrow 0$.

<u>פעולת האלגוריתם בכל פאזה</u> i:

בנה את גרף השכבות L_{f_i} .

אם $t \notin L_{f_i}$: עצור - f_i היא זרימת מקסימום.

אחרת - חשב זרימה חוסמת שכבתית q ב- L_{f_i} , ושפר באמצעותה את f_i . הזרימה המתקבלת היא f_{i+1} . עבור לפאזה $i+1$

אם האלגוריתם עוצר אז אין ב- L_{f_i} מסלול מכוון מ- s ל- t ולכן אין ב- G_{f_i} מסלול מכוון מ- s ל- t. כלומר – אין מסלול שיפור, וממילא f_i זרימת מקסימום.

כעת נראה אלגוריתם לחישוב זרימה חוסמת שכבתית בזמן $O(VE)$:

אלגוריתם למציאת זרימה חוסמת שכבתית

כל עוד יש קשתות יוצאות מ- s בצע:

1) הרץ DFS מ- s עד שתגיע לצומת v עם דרגת יציאה אפס.
2) אם $v=t$ שפר את הזרימה לאורך המסלול מ- s ל- t וסלק מ- L_f קשתות שהקיבול השיורי שלהן התאפס.
3) אחרת $(v \neq t)$ סלק מ- L_f את כל הקשתות שנכנסות ל- v .

כאשר האלגוריתם לזרימה חוסמת שכבתית עוצר – אין יותר מסלולי שיפור בגרף השכבות מ- s ל- t וזאת משום שמחקנו רק קשתות שלא יכולות להופיע על מסלולי שיפור כאלו, ומחקנו את כל הקשתות שיוצאות מ- s .

סיבוכיות: כל הרצה של ה DFS מגיעה לאחר פחות מ V צעדים לצומת v ללא קשתות יוצאות, ובכל פעם כזאת מוחקים לפחות קשת אחת בהשקעה של עוד $O(V)$ זמן. לכן כל $O(V)$ צעדים נמחקת לפחות קשת אחת, וחוזרים על זה $O(E)$ פעמים כי בכל שלב מורידים קשת. לכן מציאת זרימה חוסמת שכבתית דורשת $O(VE)$ זמן.

הוכחנו שכל פאזה באלגוריתם של דיניץ דורשת $O(VE)$ זמן. נותר להוכיח שמספר הפאזות חסום ע"י V. לצורך זה יספיק להוכיח שלכל i, $\delta_{f_{i+1}}(t) \geq \delta_{f_i}(t)+1$, שכן כל פונקצית זרימה f מקיימת ש $1 \leq \delta_f(t) \leq V-1$.

נסמן:

$f = f_i$ - פונקציית הזרימה בתחילת פאזה i.

$f' = f_{i+1}$ - פונקציית הזרימה בתחילת פאזה $i+1$. f' מתקבלת על ידי שיפור הזרימה f על ידי הזרימה החוסמת השכבתית q שנמצאה ב L_f.

נוכיח ש $\delta_{f'}(t) \geq \delta_f(t)+1$, כלומר שמספר השכבות גדל כל פאזה.

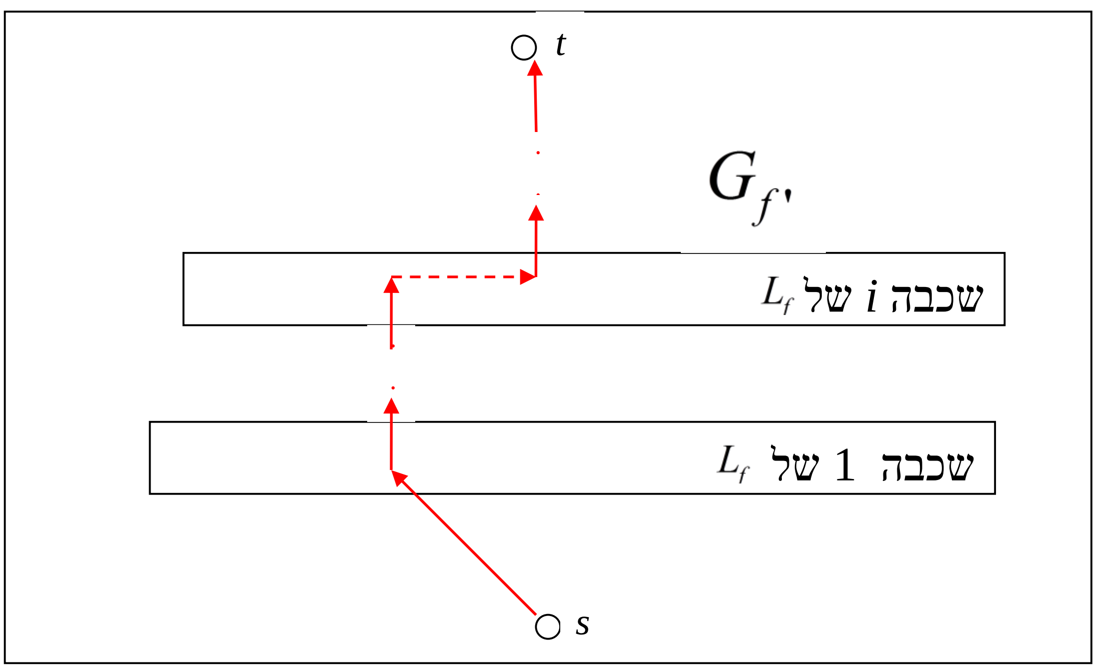
למה 12.6: עבור כל קשת $u \to v$ ב- $G_{f'}$ מתקיים $\delta_f(v) \leq \delta_f(u)+1$.

הוכחה: אם e גם קשת ב- G_f הדבר נובע מהגדרת δ_f. אם $e \notin G_f$, אז היא נוספה ל $G_{f'}$ כתוצאה משיפור בזרימה לאורך קשת $v \to u$ בגרף השכבות, כאשר v בשכבה i ו u בשכבה $i+1$. מכאן ש $\delta_f(v) = \delta_f(u)-1$.

למה 12.7: $\delta_f(s,t) < \delta_{f'}(s,t)$.

הוכחה: יספיק להוכיח שלכל מסלול שיפור p ב $G_{f'}$ מתקיים $\delta_f(s,t) < length(p)$, כאשר $length(p)$ הוא האורך של p. להלן ההוכחה (ראו ציור):

המסלול p מתחיל בצומת s בשכבה 0 של L_f ומסתיים בצומת t בשכבה $\delta_f(s,t)$ של L_f, ועל סמך למה 12.6, כל קשת ב p עוברת מצומת בשכבה k לצומת בשכבה $k+1$ לכל היותר, ולכן $\delta_f(s,t) \leq length(p)$, ושוויון קיים רק אם כל קשת ב p עוברת מצומת בשכבה k לצומת בשכבה $k+1$, כלומר אם p הוא מסלול שיפור ב L_f. אבל ב $G_{f'}$ יש זרימה חוסמת שכבתית ולכן לא קיים מסלול שיפור ב L_f. מכאן ש $\delta_f(s,t) < length(p)$. ■



הוכחת למה 12.7: כל מסלול שיפור p (באדום) ב $L_{f'}$ מכיל לפחות קשת אחת שלא עוברת לשכבה הבאה ב L_f (הקשת המקווקות)

מסלולי שיפור מקסימאליים

כעת נציג מימוש נוסף של האלגוריתם של פורד פלקרסון.

הקיבול השיורי של מסלול שיפור p, שיסומן $c(p)$, הוא הקיבול השיורי המינימלי של קשת בp. מסלול שיפור בעל קיבול שיורי גדול ככל האפשר ברשת נתונה נקרא "מסלול שיפור מקסימאלי". המימוש שנציג עכשיו מבצע בכל שלב שיפור על מסלול שיפור מקסימלי.

כדי לקבוע אם קיים מסלול שיפור p כך ש $c(p) \geq \Delta$, מסלקים מהגרף את כל הקשתות עם קיבולת שיורית קטנה מ Δ, ובודקים אם בגרף שנותר יש מסלול מ s ל t. באמצעות טכניקה זאת ניתן למצא מסלול שיפור מקסימלי בסיבוכיות זמן $O(E \log V)$, על ידי (1) מיון הקשתות לפי קיבוליהן ו(2) ביצוע חיפוש בינארי על הקשתות הממוינות, ומציאת הקשת עם הקיבול המקסימלי Δ כך שיש מסלול שיפור עם קיבול שיורי לפחות Δ.

<u>למה 12.8</u>: אם הקיבול השיורי של מסלול שיפור מקסימאלי $\leq \Delta$, אז קיים בגרף חתך בעל קיבול $\leq \Delta E$ ולכן ערך זרימת מקסימום F^* מקיים $F^* \leq \Delta E$.

הוכחה: נגדיר $S_\Delta = \{v \in V:$ קיים מסלול מ s ל v עם קיבול שיורי גדול מ $\Delta\}$.

ע"ס ההנחה $s \in S_\Delta,\ t \notin S_\Delta$, וכל קשת ב $(S_\Delta \to \bar{S}_\Delta)$ היא בעלת קיבול לכל היותר Δ. ממשפט זרימת מקסימום-חתך מינימום מקבלים: $F^* \leq c(S_\Delta, \bar{S}_\Delta) = \sum_{e \in (S_\Delta \to \bar{S}_\Delta)} c(e) \leq \Delta E$. ■

<u>למה 12.9</u>: יהי F ערך זרימת המקסימום בגרף השיורי G_f, תהי $\bar{f}$ זרימה הנוצרת ע"י שיפור על מסלול שיפור מקסימלי ב G_f, ויהי $\bar{F}$ ערך זרימת המקסימום בגרף השיורי $G_{\bar{f}}$.

אזי $\bar{F} \leq F(1-\frac{1}{E})$.

<u>הוכחה:</u> יהי Δ הקיבול של מסלול שיפור מקסימלי p ב G_f. שיפור באמצעות p מקטין את הקיבול של כל חתך בגרף השיורי ב Δ (הוכיחו..), ולכן $F = F^* - \Delta$. מלמה 12.8 מקבלים ש $\bar{F} \leq \Delta E$. מכאן מתקבל $\bar{F} = F - \Delta \leq F - \frac{F}{E}$. ■

מלמה 12.9 (וחדו"א 1) מתקבל שזרימת המקסימום בגרף השיורי לאחר E איטרציות, היא לכל היותר $F(1-\frac{1}{E})^E < \frac{F}{e}$ (כאשר e הוא הבסיס של הלוגריתמים הטבעיים). לכן לאחר $E \ln F$ איטרציות, ערך זרימת המקסימום בגרף השיורי שנוצר, שתסומן $\tilde{F}$, מקיים

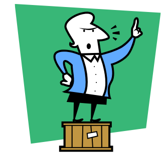
$$\tilde{F} \leq F(1-\tfrac{1}{E})^{E \ln F} < \frac{F}{e^{\ln F}} = 1.$$

כאשר קיבולי הקשתות הם מספרים שלמים, ערך הזרימה בגרף השיורי הוא מספר שלם, ולכן אם $\tilde{F} < 1$ אז $\tilde{F} = 0$ והאלגוריתם עוצר. מכאן שלמה 12.9 מבטיחה שהאלגוריתם יעצור לאחר לכל היותר $E \ln F$ איטרציות.

מאחר והסיבוכיות של כל איטרציה היא $O(E \log V)$, סה"כ סיבוכיות האלגוריתם היא $O(E^2 \log V \ln F^*)$, כאשר F^* היא זרימת המקסימום בגרף.

<u>הערה 1:</u> ניתן לשפר את סיבוכיות האלגוריתם על ידי כך שבמקום למצא בכל איטרציה מסלול שיפור מקסימלי, מבצעים שיפורים על מסלולי שיפור "כמעט מקסימליים" – שהקיבול השיורי שלהם גדול מחצי קיבול שיורי של מסלול מקסימלי. שינוי זה מקטין את הסיבוכיות של איטרציה בודדת ל $O(E)$ ואת הסיבוכיות הכוללת ל $O(E^2 \ln F^*)$ (ראו למשל פרק 7.3 בספר של Kleinberg&Tardos).

<u>הערה 2:</u> הסיבוכיות של האלגוריתם היא פולינומיאלית בגודל הקלט, אך אינה פולינומיאלית במספר הקשתות והצמתים בגרף. אלגוריתם לזרימת מקסימום נקרא "פולינומיאלי חזק" אם הסיבוכיות שלו פולינומיאלית במספר הקשתות והצמתים בגרף הקלט, ולכן אלגוריתם זה הוא לא פולינומיאלי חזק. שימו לב שהאלגוריתם של אדמונדס וקרפ הוא פולינומיאלי חזק.



הרצאה מס' 13

<u>הגדרה:</u> שידוך בגרף לא מכוון $G = (V,E)$ הוא קבוצת קשתות $M \subseteq E$ כך שבכל צומת $v \in V$ נוגעת לכל היותר קשת אחת ב-M. צומת נקרא משודך אם נוגעת בו קשת.

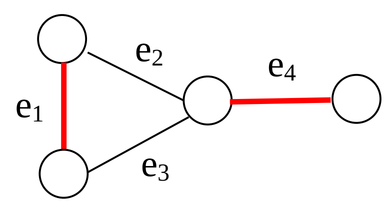
- M נקרא שידוך מושלם (perfect matching) ב-G, אם כל צומת ב-G משודך (מכאן ששידוך מושלם ייתכן רק בגרף עם מספר זוגי של צמתים).

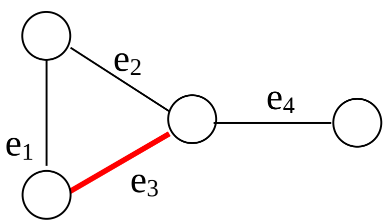
- M הוא שידוך <u>מקסימלי</u> אם הוא לא מוכל בשידוך גדול יותר. M הוא שידוך <u>מקסימום</u> אם לכל שידוך אחר 'M מתקיים $|M| \le |M'|$.

<u>דוגמה</u>

$\{e_1, e_4\}$ הוא שידוך מקסימום (וגם מושלם)

$\{e_3\}$ הוא שידוך מקסימלי





<u>תזכורת</u>

גרף לא מכוון G=(V,E) הוא דו-צדדי אם $V = (L \cup R)$ כך ש-$L \cap R = \emptyset$ ולכל קשת $(u,v) \in E$ מתקיים $u \in L, v \in R$ (או להפך).

<u>בעיית שידוך מקסימום בגרף דו-צדדי:</u> בהנתן גרף דו-צדדי $G = (L \cup R, E)$ יש למצוא ב-G שידוך מקסימום.

<u>דוגמא:</u> במסיבה יש n בנים ו-n בנות, כל בת מדווחת ל-DJ עם מי מהבנים היא מוכנה לרקוד.
הנחה: כל בן מוכן לרקוד עם כל בת שתזמין אותו.
מטרת ה-DJ – כמה שיותר זוגות על רחבת הריקודים.

<u>פתרון הדוגמא בעזרת שידוך:</u> נבנה גרף L={בנות}, R={בנים}. $(u,v) \in E$ אמ"ם u מוכנה לרקוד עם v. שידוך מושלם הוא שידוך בו כל הבנים והבנות רוקדים: |L|=|R| וגם |M|=|L|.

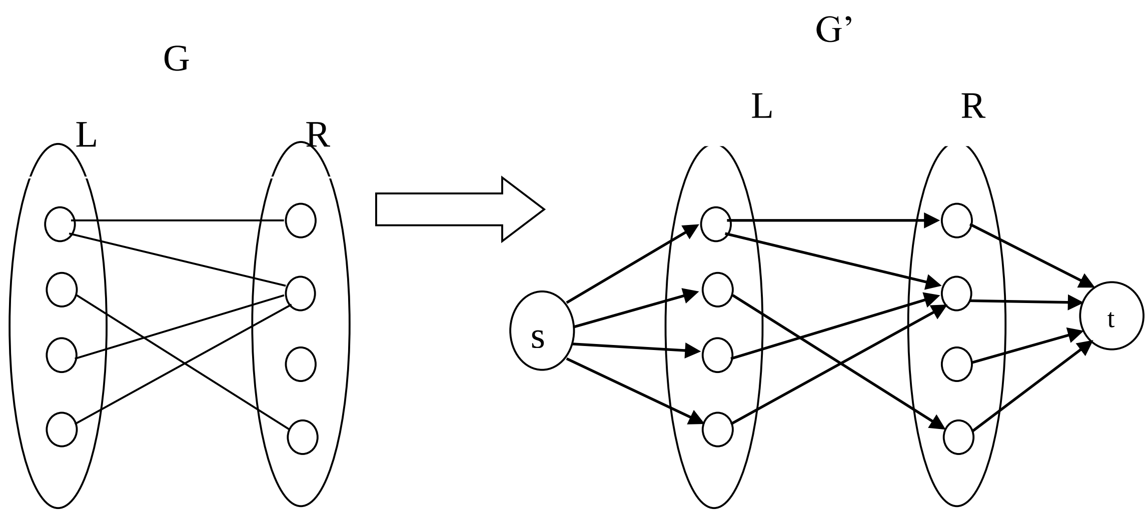
<u>אלגוריתם לשידוך מקסימום בגרף דו צדדי על ידי זרימת מקסימום</u>

עבור הגרף הדו-צדדי $G=(L\cup R,E)$ נגדיר את רשת הזרימה N=(G',s,t,c).

$$V'=L\cup R\cup\{s,t\}$$

$$E'=\{(s,v)\mid v\in L\}\cup\{(u,t)\mid u\in R\}\cup\underbrace{\{(u,v)\in E\mid u\in L,v\in R\}}_{\text{הקשתות המקוריות ב-G, מכוונות מ L ל R}}$$

הקיבול של כל קשת $e'\in E'$ יהיה $c(e')=1$. הרדוקציה מתוארת בציור:



<u>למה 13.1:</u> יהיו G ו-N גרף דו-צדדי ורשת זרימה מתאימה (כפי שבנינו). אז אם M שידוך ב-G, קיימת ב-N זרימה בשלמים f כך ש- $|M|=|f|$. ולהיפך: אם f זרימה בשלמים ב-N, אז קיים ב-G שידוך M כך ש- $|M|=|f|$.

<u>הוכחה:</u>

א) נניח ש-M שידוך נתון ב-G. נגדיר פונק' זרימה בשלמים באופן הבא:

אם $(u,v)\in M$ כך ש-$u\in L, v\in R$, אז: $f((s,u)) = f((u,v)) = f((v,t)) = 1$.
הזרימה על שאר הקשתות היא 0.

- נשמרים אילוצי הקיבול כי $c(e')=1$ לכל $e'\in E'$.

- מתקיים שימור הזרימה, כי המסלולים שהגדרנו זרים בצמתים למעט s ו-t. זאת משום ש-M שידוך, ולכן לכל צומת משודך ערך הזרימה הנכנסת = ערך הזרימה היוצאת = 1, ולכל צומת שונה מ s,t שאינו משודך נכנסת ויצאת זרימה בחוזק 0.
קיבלנו שהשידוך מגדיר M מסלולים זרים מ s ל t, שעל כל אחד מהם ערך הזרימה הוא 1, ולכן $|M| = |f|$.

ב) בהנתן פונ' זרימה בשלמים f ב-N, נגדיר שידוך ב-G באופן הבא:

$$M = \{(u, v) \mid u \in L, v \in R, f(u, v) > 0\}$$

תהי $(u,v) \in M$, כלומר $f(u,v) > 0$. נבדוק מהו ערך הזרימה בקשת זו.
מאחר ולכל צומת $u \in L$ יש בדיוק קשת נכנסת אחת שקיבולה 1, סך כל הזרימה הנכנסת ל-u היא לכל היותר 1, והיא שווה לסך כל הזרימה היוצאת מ-u. היות ו-f היא זרימה בשלמים, ערך הזרימה הוא בדיוק 1 ולכן קיים צומת יחיד $v \in R$ כך שהזרימה על (u,v) גדולה מ 0.
מכאן ש u משודך רק לצומת v ב R. משיקולים דומים, v משודך רק לצומת u בL. לכן M הוא שידוך ב-G. נסתכל על החתך $(S, \bar{S})$ כאשר $S = \{s\} \cup L$. הזרימה העוברת דרך חתך זה היא בדיוק גודל השידוך. לכן $|M|=|f|$.■

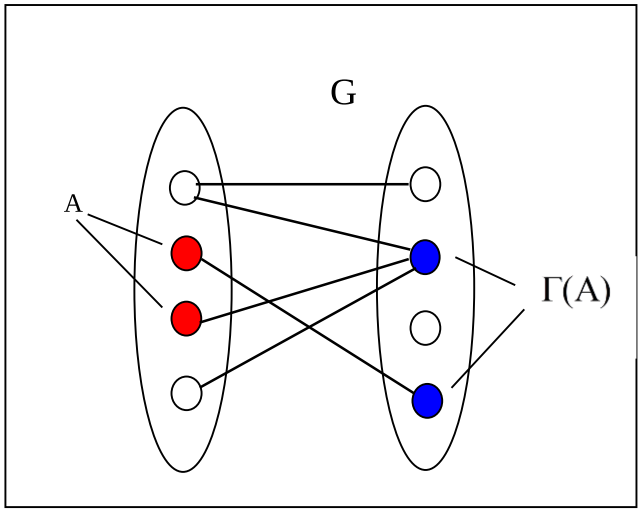
ראינו בשעור קודם כי אם כל הקיבולים שלמים, אז כל אלגוריתם לזרימת מקסימום שמגדיל זרימה באמצעות מסלולי שיפור מוצא זרימת מקסימום שהיא זרימה בשלמים.

<u>מסקנה:</u> ניתן לחשב שידוך מקסימום ב-G ע"י מציאת זרימת מקסימום ברשת N.

<u>סיבוכיות:</u> גודל שידוך מקסימום הוא לכל היותר $\min\{|L|,|R|\}=O(V)$.

מאחר וכל הקיבולים שלמים, כל מימוש של האלגוריתם הגנרי של FF יסתיים לאחר$O(V)$ איטרציות (בכל איטרציה משפרים את הזרימה ב-1 לפחות). סיבוכיות כל איטרציה היא $O(E)$. ולכן הסיבוכיות הכוללת - $O(VE)$.

כעת נוכיח משפט בסיסי בקומבינטוריקה – משפט Hall.
בהנתן גרף דו-צדדי $G = (L \cup R, E)$ וקב' צמתים $A \subseteq L$, נסמן $\Gamma(A)$ את קב' הצמתים ב-R שיש להם שכן אחד או יותר ב-A (ראו ציור).



<u>משפט הול</u>: בגרף דו-צדדי $G=(L\cup R,E)$ שבו $|L|=|R|$ יש שידוך מושלם אמ"ם לכל $A\subseteq L$, $|A|\le|\Gamma(A)|$.

משפט זה נקרא גם "משפט החתונה". קיים שידוך מושלם בו כל בת בוחרת בן המוכר לה אמ"ם כל קבוצת בנות אינה גדולה מקבוצת הבנים שהן מכירות.

<u>הוכחה</u>

(i) נניח שב-G שידוך מושלם, דהיינו לכל צומת $u\in L$ קיים בן זוג $v\in R$ (והזוגות זרים). לכן לכל קב' $A\subseteq L$ מתקיים:

$$|A|=|\{v\in R\mid v\ is\ matching\ to\ u\in L\}|\le|\Gamma(A)|$$

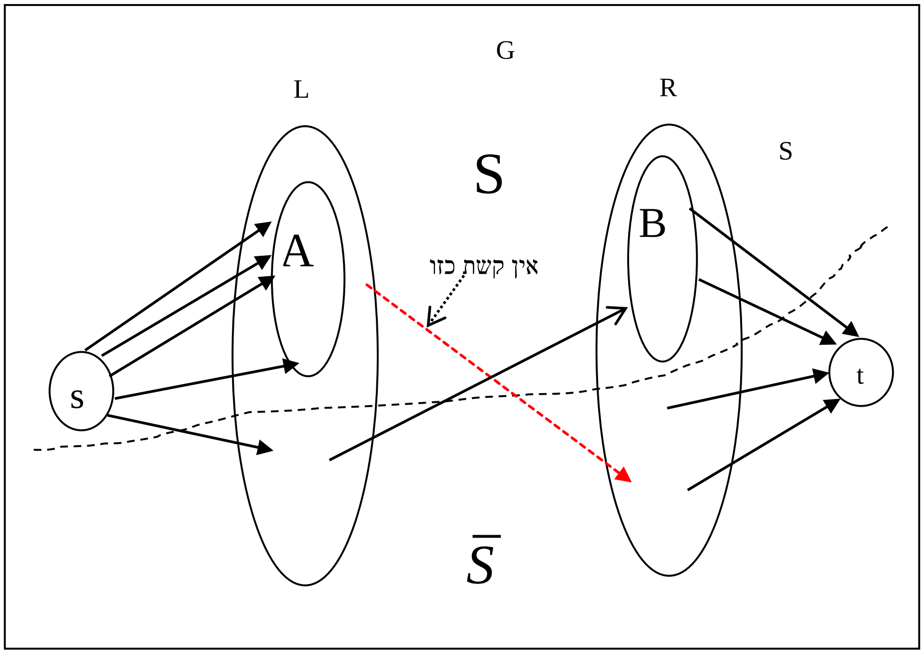
(ii) נתון כי לכל קב' $A\subseteq L$ מתקיים $|A|\le|\Gamma(A)|$. נראה שקיים שידוך מושלם ב-G. נבנה רשת N בדומה לרשת למציאת שידוך מקסימום למעט ההבדל הבא: עבור קשתות e'=(u,v) כך ש-$u\in L, v\in R$ נגדיר $c(e')=\infty$.

- ניתן לראות כי הלמה והמשפט שהוכחנו ביחס לשידוך מקסימום מתקיימים גם תחת הגדרה זו (תרגיל). ובפרט גודל שידוך המקסימום ב-G שווה לערך זרימת מקסימום ברשת N.

$\Leftarrow$ מספיק להראות שערך זרימת המקסימום הוא |L|.

מאחר וברור ש $F^*\le|L|$, יספיק להוכיח כי $F^*\ge L$.

יהי $(S,\bar{S})$ חתך מינימום ב-N. לכן (ממשפט חתך מינימום – זרימת מקסימום) $F^*=C(S,\bar{S})$.
נגדיר $A=L\cap S$, $B=R\cap S$. אם $A=\Phi$ אז גודל החתך $\ge|L|$, לכן ניתן להניח $A\ne\Phi$.



- אם $v \in \Gamma(A)$ אז $v \in B$, אחרת קיימת קשת מ-L ל-R שחוצה את החתך, ולכן $C(S,\bar{S}) = \infty$ בסתירה להיותו חתך מינימום, כי יש אחר: $(\{s\},V\setminus\{s\})$ שקיבולו הוא סופי - $|L|$.

- מכאן נובע ש- $\Gamma(A) \subseteq B$, לכן $|A| \le |\Gamma(A)| \le |B|$, וסה"כ:

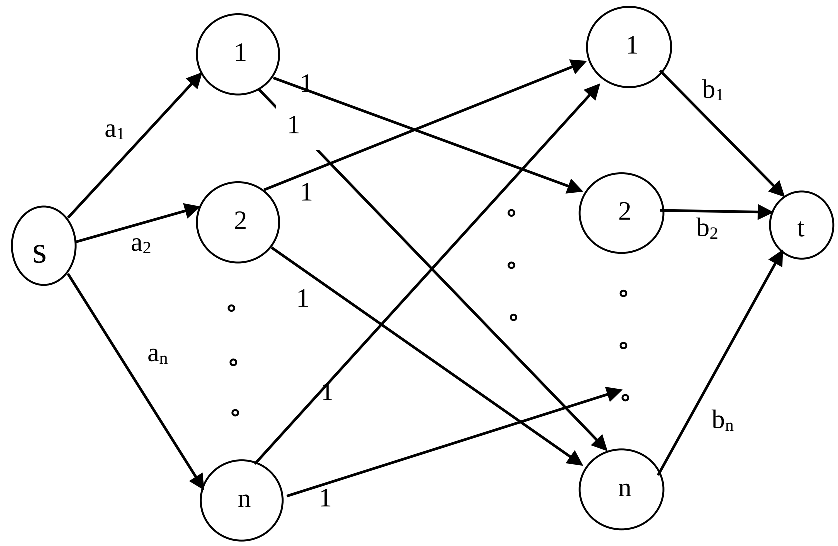
$$F^* = C(S,\bar{S}) = |L| - |A| + |B| \ge |L| - |A| + |A| = |L| \quad \blacksquare$$

<u>דוגמה נוספת: קיום גרף מכוון עם דרגות נתונות</u>

- נתונים שני וקטורים של שלמים חיוביים $(a_1,\dots,a_n)$, $(b_1,\dots,b_n)$ כך ש- $\sum_{i=1}^{n} a_i = \sum_{i=1}^{n} b_i = m$

הצע אלגוריתם הבודק האם קיים גרף מכוון פשוט בעל n צמתים ללא לולאות עצמיות שבו דרגות היציאה הם $(a_1,\dots,a_n)$, ודרגות כניסה הם $(b_1,\dots,b_n)$. a_i ו b_i דרגות כניסה ויציאה של צומת i.

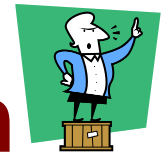
נגדיר את רשת הזרימה הבאה (יש קשת מכוונת $(i_L \to j_R)$ עם קיבולת 1 אםם $i \neq j$):



נמצא את זרימת המקסימום f. בכל קשת $(i_L \to j_R)$ הזרימה היא 0 או 1.

<u>טענה:</u> קיים גרף G המקיים התנאים אםם חוזק הזרימה הוא m.

רעיון ההוכחה: חוזק הזרימה הוא m אםם כל הקשתות שיוצאות מ s (וכל הקשתות שנכנסות ל t) הן רוויות. קשת $(i \to j)$ נמצאת בגרף G אםם $f(i_L \to j_R) > 0$.



הרצאה מס' 14

זרימה עם חסמים עליונים ותחתונים

קלט: $N=(G,s,t,c,b)$, כאשר ל G,s,t,c המשמעות הרגילה, ו $b:E\to R^{+}$ קובעת לכל קשת חסם תחתון על הזרימה שיכול לעבור בה (מכאן ש $0\le b(e)\le c(e)$).

זרימה חוקית:

- חוק הצומת לא השתנה
- חוק הקיבול הוא $b(e)\le f(e)\le c(e)$ לכל $e\in E$.

לא תמיד קיימת זרימה חוקית, בניגוד לזרימה ללא חסמים תחתונים בה תמיד יש זרימה אפס שהיא חוקית. בהרצאה זו נגדיר אלגוריתם הקובע אם קיימת זרימה חוקית.

נפתור את הבעיה ע"י בניית רשת $N'=(G',s',t',c')$, כך שקיימת זרימה חוקית ב- N אם"ם קיימת זרימת מקסימום בעלת ערך נתון ב- N' . (ערך זה שווה בפועל ל- $\sum_{e\in E} b(e)$).

נגדיר את הרדוקציה באופן פורמאלי:

$$E'=E\cup\{t\to s\}\cup\{s'\to u,\ u\to t':u\in V\}$$

עבור כל $e\in E$,

$$c'(e)=c(e)-b(e)$$

$$c'(t,s)=\infty$$

$$c'(s',u)=\sum_{e\in in(u)} b(e)$$

$$c'(u,t')=\sum_{e\in out(u)} b(e)$$

משפט 14.1

קיימת זרימה חוקית ב- N אם"ם ערך זרימת מקסימום ב N' הוא $\sum_{e\in E} b(e)$

הוכחה

א. נניח שקיימת ב- N זרימה חוקית f . נגדיר ב- N' זרימה f' באופן הבא:

$(\forall e\in E)\ f'(e)=f(e)-b(e)$

$(\forall u\in V)\ f'(s',u)=c'(s,u)$

$(\forall u\in V)\ f'(u,t')=c'(u,t')$

$f'(t,s)=|f|$

חוק הקיבול מתקיים שהרי

$(\forall e\in E)\ 0\le f'(e)=f(e)-b(e)\le c(e)-b(e)=c'(e)$

לכל הקשתות שיוצאות מ-s' או נכנסות ל-t':

$f'(e)=c'(e)$

$0\le f(t,s)<\infty$

חוק הצומת מתקיים עבור כל צומת שאינו s',t'.

כמות הזרימה הנכנסת לכל צומת u בפונקציה f' שווה לכמות הנכנסת אליה בפונקציה f.

s,t מקיימים את חוק הצומת מפני שהקשת החדשה מ-t ל-s משווה את הזרימה הנכנסת והזרימה היוצאת, בכל אחד מהצמתים.

ב. מההנחה נובע שקיימת זרימת מקסימום ב-N' המרווה את כל הקשתות שיוצאות מ-s' או נכנסות אל t'. נגדיר זרימה f ב-N באופן הבא: לכל קשת e

$f(e)=f'(e)+b(e)$

חוק הקיבול מתקיים מהגדרות c,c'.

חוק הצומת מתקיים מאותו נימוק שהוזכר קודם. הזרימה שהופחתה ע"י סילוק הקשתות החדשות נוספה ע"י הגבלת הזרימה בקשתות השונות.■

בהצלחה במבחן !

אלגוריתמים 1 – דף נוסחאות

אלגוריתמים

מיון טופולוגי

קלט: גרף $G=(V,E)$ שהוא DAG.

פלט: מיון טופולוגי של הגרף.

1. חשב את קבוצת כל המקורות בגרף, נסמנה ב- S .
2. אתחל $l \leftarrow 1$
3. כל עוד $V \neq \varnothing$
   1. בחר $v \in S$.
   2. קבע $L(v) \leftarrow l$
   3. קבע $l \leftarrow l+1$.
   4. הסר את v מהגרף, יחד עם כל הקשתות היוצאות ממנו.
   5. קבע $S \leftarrow S \setminus \{v\}$.
   6. הוסף ל- S את כל הצמתים מבין $\{u : vu \in E\}$ שהם מקורות.
4. החזר את L.

BFS(G,s):

1. for any u in V do
   a. d[u] := ∞
   b. p[u] := nil
2. Q := {s}; d(s):=0
3. while Q is not empty do
   a. u := dequeue(Q)
   b. for each v in adj(u) do
      i. if d[v] = ∞ then
         1. d[v] := d[u] + 1
         2. p[v] := u
         3. enqueue(Q,v)

DFS(G,s):

1. for all v in V
   a. k[v] = 0
   b. p[v] = nil
   c. i:=0
2. while there is a vertex s with k(s)=0
   a. STACK:={s}, i:=i+1, k[s]:=i
3. While STACK ≠ ∅
4. u:= head(STACK)
5. if there is v∈adj(u) s.t. k[v]=0
   a. i:=i+1
   b. k[v]:=i
   c. p[v]:=u
   d. push(STACK,v)
6. else
   a. pop(STACK)

Strongly Connected Components(G):

1. Call DFS(G) to compute f[u] for all u in V
2. Compute G^T
3. Call DFS(G^T) on the vertices ordered in decreasing order of f[u] (computed in (1))
4. output the vertices in each DFS tree generated in (3) as separate component

הכלל האדום (לקשתות כבדות):

תנאי: קיים ב G מעגל C חסר קשתות אדומות, וe1 היא קשת לא צבועה בעלת משקל מקסימאלי (מבין הקשתות הלא צבועות) ב C.

פעולה: צבע את e באדום.

הכלל הכחול (לקשתות קלות):

תנאי: קיים ב G חתך D חסר קשתות כחולות, וe1 היא קשת לא צבועה בעלת משקל מינימאלי (מבין הקשתות הלא צבועות) ב D.

פעולה: צבע את e בכחול.

האלגוריתם הגנרי:

בחר קשת כלשהי עליה ניתן להפעיל את הכלל האדום או הכלל הכחול, והפעל כלל זה. עצור כאשר אין בG קשת כזו.

האלגוריתם של Prim

1. אתחול – כל הקשתות אינן צבועות; $U:=\{r\}$.
2. כל עוד $U \neq V$ בצע:
   א. הפעל את הכלל הכחול על חתך $(U,V\setminus U)$ וצבע בכחול קשת $e=(u,v)$ מינימאלית בחתך זה כך ש- $u\in U, v\in V\setminus U$; בצע $U:=U\cup\{v\}$.

האלגוריתם של Kruskal

1. מיין את הקשתות בסדר לא יורד לפי משקל.
2. עבור על הרשימה הממוינת, ולכל קשת $e=(u,v)$ בצע:
   א. אם יש מסלול כחול מ u ל v, צבע את e באדום,
   ב. אחרת צבע את e בכחול.

אלגוריתם לעפ"מ צהוב ביותר

נגדיר פונקצית משקל חדשה

$$w'(e)=\begin{cases} w(e)-1/n^2 & e \text{ is yellow} \\ w(e) & \text{otherwise} \end{cases}$$

קלט: גרף לא מכוון וקשיר $G=(V,E)$ ופונקצית משקל $w:E\to N$.

פלט: עפ"מ צהוב ביותר של G.

1. חשב את פונקצית המשקל w'.
2. מצא עפ"מ T לפי פונקצית המשקל w', והחזר את T כפלט.

אלגוריתם גנרי למסלולים קלים ביותר ממקור יחיד

פונקציית חסם עליון על מרחקים (בקיצור פח"ע): זוהי פונקציה $d:V\to\mathbb{R}$ המקיימת: $d(s)\le 0$, $d(v)\ge dist(s,v)$.

שיפור מקומי על קשת (u,v) – relaxation: בהינתן פח"ע d, שיפור מקומי על קשת (u,v) הוא הכלל: אם $d(v)>d(u)+w(u,v)$ אז $d(v)\leftarrow d(u)+w(u,v)$.

האלגוריתם הגנרי:

1. קבע פח"ע
2. כל זמן שקיימת קשת (u,v) כך ש $d(v)>d(u)+w(u,v)$ בצע שיפור (u,v).

אלגוריתם בלמן-פורד:

קלט: גרף מכוון $G=(V,E)$ ממושקל, ללא מעגלים שליליים, וצומת $s\in V$.

פלט: לכל צומת v - $dist(s,v)=d(v)$.

האלגוריתם:

1. אתחול: לכל $v\in V\setminus\{s\}$ אתחל $d(v)\leftarrow\infty$, $parent(v)\leftarrow nil$
   $d(s)\leftarrow 0$;
2. עבור $i=1$ עד $V-1$ בצע
   a. לכל קשת (u,v) בצע שיפור (u,v).

<u>אלגוריתם דייקסטרה:</u>

<u>קלט</u>: גרף ממושקל מכוון ללא משקלים שליליים, וצומת s .

<u>פלט</u>: לכל צומת v - $d(v)=dist(s,v)$

<u>האלגוריתם</u>:

1. לכל $v\in V$ אתחל $d(v)\leftarrow\infty$, ואתחל $d(s)\leftarrow 0$
2. בנוסף אתחל $Q\leftarrow V$ (Q מכיל את כל הצמתים עבורם $d(v)\neq dist(s,v)$).
3. כל זמן ש $Q\neq\emptyset$ בצע
   a. מצא ב- Q צומת u כך ש $d(u)$ מינימאלי.
   b. הוצא את u מ- Q ולכל קשת (u,v) בצע שיפור (u,v).

<u>אלגוריתם ג'ונסון:</u>

נגדיר גרף חדש $G'=(V',E')$ כך ש- $V'=V\cup\{s\}$ ו- $E'=E\cup\{sv:v\in V\}$. כמו כן נגדיר $w':E\to R$ כך ש- $w'(sv)=0$ לכל $v\in V$ ו- $w'(e)=w(e)$ לכל קשת אחרת. נסמן ב- $\delta_s(v)$ את משקל המסלול הקל ביותר מ- s ל- v ב- G' . נגדיר גם $w_{\delta_s}(uv)=\delta_s(u)+w(uv)-\delta_s(v)$.

<u>קלט</u>: גרף מכוון $G=(V,E)$ עם פונקציית משקל על הקשתות $w:E\to R$ ללא מעגלים שליליים.

<u>פלט</u>: משקל מסלול קל ביותר מכל צומת לכל צומת.

1. בנה את G' כפי שהוגדר לעיל.
2. חשב את δ_s כפי שהוגדר לעיל, בעזרת האלגוריתם של Bellman Ford.
3. חשב את פונקצית המשקל w_{δ_s} .
4. מכל צומת $u\in V$ הרץ דייקסטרה לחישוב משקל מסלולים קלים ביותר מ- u לכל הצמתים, ביחס לפונקציית המשקל w_{δ_s} . נסמן את הערך המתקבל ב- $d'(u,v)$.
5. עבור כל זוג צמתים $u,v\in V$ החזר כפלט $d'(u,v)+\delta_s(v)-\delta_s(u)$.

<u>אלגוריתם חמדן למציאת מספר מקסימלי של קטעים זרים:</u>

<u>קלט</u>: אוסף משימות $S=\{a_1,...,a_n\}$. לכל משימה זמן התחלה s_i וזמן סיום f .

<u>פלט</u>: תת קבוצה מקסימלית $A\subset S$ של משימות (או קטעים) הזרים בזוגות.

<u>האלגוריתם</u>:

1. מיין את המשימות לפי סדר עולה של זמני סיום. (מעכשיו נניח כי המשימות מסודרות כך ש $f_1\leq f_2\leq\cdots\leq f_n$).
2. אתחל $A\leftarrow\emptyset$, $i\leftarrow 0$ ו- $f_0\leftarrow-\infty$. (i הוא האינדקס של המשימה האחרונה שנכללת בפתרון, ו- f_i נק' הסיום שלה).
3. עבור $m=1$ עד n בצע:
   א. אם $f_i\leq s_m$ אז $A\leftarrow A\cup\{a_m\}$ ו- $i\leftarrow m$, (אם המשימה האחרונה ב- A הסתיימה לפני ש a_m מתחילה, הוסף את a_m לA)

<u>אלגוריתם חמדן לצביעת גרף:</u>

<u>קלט</u>: גרף לא מכוון $G=(V,E)$, וסדר כלשהו על הצמתים.

<u>פלט</u>: צביעה חוקית c של G המשתמשת לכל היותר ב - d_1+1 צבעים.

1. עבור צומת צומת ע"פ הסדר הנתון של הצמתים. יהא v הצומת הנוכחי.
   א. קבע את $c(v)$ להיות הצבע המינימאלי שאינו בשימוש על ידי שכניו של v .

<u>אלגוריתם Huffman לקוד פרפיקסי אופטימלי:</u>

:Recursive_Huffman($[\Sigma, f]$)

<u>קלט</u>: א"ב עם תדירות לכל אות $[\Sigma, f]$

<u>פלט</u>: עץ האפמן של $[\Sigma, f]$

1. אם $|\Sigma| = 2$ החזר עץ בעל שני עלים שהם שני איברי Σ.
2. אם $|\Sigma| > 2$:
   א. יהי $[\Sigma', f']$ הקלט הנוצר על ידי החלפת שתי אותיות x, y בעלי תדירות מינימאלית ב Σ באות חדשה z שתדירותה $f'(z) = f(x) + f(y)$.
      i. T' := Recursive_Huffman($[\Sigma', f']$).
   ב. הוסף לעלה המתאים ל- z ב-T' את x ו- y כבנים, והחזר את העץ שהתקבל T.

<u>אלגוריתם פלויד וורשל למציאת כל המסלולים הקלים ביותר</u>: (גרסה שקולה לזו שהועברה בכיתה)

שיפור מקומי באמצעות צמת ביניים (שיפור מקומי מוכלל) עבור שלשה (i, k, j) הוא הכלל:

אם $d(i,j) > d(i,k) + d(k,j)$ אז $d(i,j) \leftarrow d(i,k) + d(k,j)$

<u>קלט</u>: $G = (V, E)$ עם פונקציית משקל $w: E \to \mathbb{R}$, ללא מעגלים שליליים.

בה"כ נניח כי $V = \{1, \ldots, n\}$, ונגדיר את פונקציית המשקל המוכללת $\bar{w}: V \times V \to \mathbb{R}$ השווה ל – 0 אם $i = j$, ל - $w(i,j)$ אם $(i,j) \in E$ ול - ∞ אחרת.

<u>פלט</u>: לכל זוג צמתים $i, j \in V$ מוחזר המרחק מ- i ל- j - $dist(i, j)$.

<u>אתחול</u>: לכל זוג צמתים i, j בצע $d(i, j) \leftarrow \bar{w}(i, j)$.

עבור $k = 1$ עד n בצע

לכל זוג צמתים i, j בצע שיפור (i, k, j).

<u>האלגוריתם קבוצה בלתי תלויה של קטעים בעלת משקל מקסימלי:</u>

<u>אתחול</u>:

1. מיין את המטלות לפי סדר עולה של זמני סיום. לאחר המיון $f_1 \le f_2 \le \cdots \le f_n$.
2. אתחל $A(0) \leftarrow \varnothing$, ו- $opt(0) \leftarrow 0$.
3. לכל i מצא את $pred(i)$.

<u>גוף האלגוריתם</u>:

1. עבור $i = 1$ עד n בצע
2. אם $opt(i-1) < opt(pred(i)) + w_i$ אז
   א. $A(i) \leftarrow A(pred(i)) \cup \{a_i\}$.
   ב. $opt(i) \leftarrow opt(pred(i)) + w_i$.
3. אחרת
   א. $A(i) \leftarrow A(i-1)$.
   ב. $opt(i) \leftarrow opt(i-1)$.
4. החזר $A(n)$.

<u>אלגוריתם תכנות דינמי לבעיית ה-Knapsack:</u>

<u>קלט</u>: n פריטים $a_1, a_2, \dots, a_n$ כאשר ערכיהם $p_1, p_2, \dots, p_n$ ומשקליהם $w_1, w_2, \dots, w_n$ וכן קיבולת שק W.

<u>פלט</u>: הערך המקסימלי של קבוצת פריטים העומדת בקיבולת השק.

1. צור מטריצה F מגודל $(n+1)\times(W+1)$
2. אתחל $F(0,w)=0$ לכל $0 \le w \le W$
3. עבור מהשורה המתאימה ל- $k=0$ עד לשורה המתאימה ל- $k=n$
   א. חשב את כל כניסות השורה לפי הנוסחא:

$$F(k,w)=\begin{cases} F(k-1,w) & w_k > w \\ \max\left(F(k-1,w),\, p_k + F(k-1,w-w_k)\right) & otherwise \end{cases}$$

4. החזר את $F(n,W)$

<u>האלגוריתם הגנרי של פורד-פלקרסון לזרימת מקסימום:</u>

1. קבע $f(e)=0$ לכל קשת e.
2. כל עוד קיים מסלול שיפור p מ s אל t ב G_f, שפר את הזרימה לאורך p.

<u>האלגוריתם של אדמונדס-קרפ:</u>

1. קבע $f(e)=0$ לכל קשת e.
2. כל עוד קיים מסלול שיפור מ- s אל t ב- G_f, יהא p מסלול השיפור הקצר ביותר, שפר את הזרימה לאורך p.

<u>אלגוריתם שידוך מקסימום בגרף דו-צדדי:</u>

נגדיר את הגרף המכוון $G'=(V',E')$ באופן הבא

$$V'=V\cup\{s,t\}$$

$$E'=\{(s,u):u\in L\}\cup\{(u,v):uv\in E, u\in L, v\in R\}\cup\{(v,t):v\in R\}$$

ונצייד כל קשת עם קיבול יחידה.

<u>קלט</u>: גרף דו-צדדי $G=(V,E)$.

<u>פלט</u>: שידוך מקסימום M עבור G.

1. בנה את רשת הזרימה $N'=(G',s,t,c)$ עם G' כפי שהוגדר לעיל ו- $c\equiv 1$.
2. הרץ את אלגוריתם פורד-פולקרסון לחישוב זרימת מקסימום ב- N'.
3. החזר $M=\{uv: f(u,v)=1\}$.

<u>התמרת פורייה המהירה:</u>

בהינתן שני ווקטורים $a=(a_0,a_1,\dots,a_{n-1})$, $b=(b_0,b_1,\dots,b_{n-1})$, ניתן לחשב את $a*b=(a_0b_0, a_0b_1+a_1b_0, a_0b_2+a_1b_1+a_2b_0, \dots, a_{n-1}b_{n-1})$ (ה*קונבולוציה* ביניהם) בסיבוכיות O(nlogn) באמצעות התמרת פורייה המהירה:

<u>קלט</u>: שני ווקטורים $a=(a_0,a_1,\dots,a_{n-1})$, $b=(b_0,b_1,\dots,b_{n-1})$.

<u>פלט</u>: $a*b$.

1. חשב את ערכי הפולינומים $a(x)=\sum_{i=0}^{n-1}a_ix^i, b(x)=\sum_{i=0}^{n-1}b_ix^i$. ב - $2n$ שורשי היחידה המרוכבים מסדר $2n$, נסמנם $\{w_{s,2n}\}_{s=0}^{2n-1}$. החישובים נעשים ע"י התמרת פורייה המהירה בזמן O(nlogn).
2. חשב את ערך $c(x)=a(x)\cdot b(x)$ ב - $\{w_{s,2n}\}_{s=0}^{2n-1}$.
3. הגדר $\mathrm{D}(x)=\sum_{s=0}^{2n-1}c(w_{s,2n})x^s$.
4. החזר את הווקטור c כך ש - $c_s=\frac{1}{2n}D(w_{2n-s,2n})$. c מחושב ע"י התמרת פורייה ההפוכה בזמן O(nlogn).

טענות ומשפטים שימושיים

<u>סיווגי קשתות ביער DFS:</u>

<u>קשתות עץ</u>: (u,v) קשת עץ אםם $u=p(v)$.

<u>קשתות אחוריות</u>: קשת (u,v) שמחברת את u לאב קדמון של u בעץ DFS (לולאה עצמית תחשב כקשת אחורית).

<u>קשתות קדמיות</u>: קשתות (u,v) מ- u לצאצא של u בעץ DFS.

<u>קשותות חוצות</u>: כל הקשתות האחרות ב– G בין צמתים באותו עץ DFS ללא יחס אב קדמון – צאצא, או בין עצי DFS שונים.

הקשר בין סוגי הקשתות לזמני הגילוי והנסיגה הוא הקשר הבא:

קשת uv היא קשת עץ או קשת קדמית אם"ם $d(u)<d(v)<f(v)<f(u)$.

קשת uv היא קשת אחורית אם"ם $d(v)<d(u)<f(u)<f(v)$.

קשת uv היא קשת חוצה אם"ם $d(v)<f(v)<d(u)<f(u)$.

<u>משפט המסלול הלבן:</u>

ביער ה-DFS של גרף (מכוון או לא מכוון), צומת v הוא צאצא של צומת u אם"ם בזמן גילוי u (דהיינו בנקודת הזמן $d(u)$) קיים מסלול מ- u ל- v דרך צמתים שטרם התגלו.

<u>גרף הרכיבים קשירים היטב:</u>

נניח שהרק"ה ב- G הם $C_1,...,C_k$.

גרף הרכיבים קשירים היטב של G, שיסומן $G^*=(V^*,E^*)$, מוגדר באופן הבא.

$V^*=\{v_1,...,v_k\}$

$E^*=\{(v_i,v_j):$ יש ב-E קשת שיוצאת מצומת ב- C_i אל צומת ב- C_j $\}$

<u>למה:</u> גרף הרק"ה G* הוא אציקלי.

<u>למה:</u> בכל ריצת אלגוריתם DFS, כל רכיב קשיר היטב מוכל במלואו באחד מעצי ה-DFS שהאלגוריתם מחזיר.

שמורת הצבע: גרף G שחלק מקשתותיו כחולות וחלק אחר אדומות מקיים את ***שמורת הצבע*** אםם קיים בG עפ"מ שמכיל את כל הקשתות הכחולות ואף אחת מהקשתות האדומות.

<u>למה:</u> בכל ביצוע של האלגוריתם הגנרי על גרף לא מכוון משוקלל וקשיר G (שקשתותיו מלכתחילה אינן צבועות), הגרף G מקיים את שמורת הצבע.

<u>למה:</u> לכל עפ"מ קיימת ריצה של קרוסקל המוצאת אותו וריצה של פרים המוצאת אותו.

<u>למה:</u> יהא $G=(V,E)$ גרף לא מכוון וקשיר. יהיו $w:E\to R,\ w':E\to R$ שתי פונקציות משקל על הקשתות המקיימות $w(e_1)\le w(e_2)$ אם"ם $w'(e_1)\le w'(e_2)$ לכל זוג קשתות $e_1,e_2\in E$. T הוא עפ"מ לפי w_1 אם"ם T הוא עפ"מ לפי w_2.

<u>למה:</u> יהא $G=(V,E)$ גרף לא מכוון וקשיר עם פונקצית משקל $w:E\to R$. לכל משקל נתון, כל עפ"מ של G מכיל את אותו מספר של קשתות ממשקל זה.

<u>למה:</u> קשת e נמצאת באיזשהו עפ"מ של G אם"ם כל מעגל המכיל את e מכיל קשת $e'\neq e$ כך ש- $w(e')\ge w(e)$.

<u>למה:</u> קשת e נמצאת בכל עפ"מ של G אם"ם כל מעגל המכיל את e מכיל קשת e' כך ש- $w(e')>w(e)$.

<u>למה:</u> אם אין ב- $G=(V,E)$ ***מעגלים שליליים*** (שסכום משקלי קשתותיהם שלילי) שניתנים להגעה מ- s אזי לכל v שניתן להגעה מ- s קיים מסלול קל ביותר מ- s ל- v

<u>מבנה אופטימאלי של מסלולים קלים ביותר</u>: אם P הוא מסלול קל ביותר מ- u ל- v, אז כל תת-מסלול של P הוא מסלול קל ביותר בין זוג הצמתים המתאימים.

<u>אי-שוויון המשולש</u>: לכל u,v,w מתקיים $dist(u,v)\le dist(u,w)+dist(w,v)$.

למה: יהי $G=(V,E)$ גרף משוקלל, שאין בו מעגל בעל משקל שלילי המכיל את s. תהי $d: V\to \mathbb{R}\cup\{\infty\}$ פח"ע. אז 2 הטענות הבאות נכונות:

1. d נשארת פח"ע גם אחרי ביצוע שיפור (u,v).
2. אם לכל קשת (u,v) מתקיים $d(v)\le d(u)+w(u,v)$ (כלומר לא ניתן לבצע שיפור מקומי) אז לכל צומת v מתקיים $d(v)=dist(s,v)$.

קידוד של א"ב, קוד: נתון א"ב סופי $\Sigma=\{s_1,\dots,s_n\}$. "קידוד של Σ" הוא מיפוי $C:\Sigma\to\{0,1\}^+$ המתאים לכל אות ב Σ מילה בינארית $C(s_i)=w_i$. קבוצת המלים $\{w_1,\dots,w_n\}$ היא "קוד", שגם הוא נקרא C. לכל מילה u ב Σ^+, $C(u)$ היא המלה הבינארית המתקבלת משרשור הקידודים של האותיות ב u.

קוד חד פענח: לכל זוג מלים שונות $u_1,u_2\in\Sigma^*$, צריך ש $C(u_1)\ne C(u_2)$.

קוד פרפיקסי (חסר רישות): אף מילת קוד אינה פרפיקס (רישא) של מילת קוד אחרת.

רשת זרימה: רביעיה $N=(G,s,t,c)$, כאשר $G=(V,E)$ הוא גרף מכוון ללא לולאות או קשתות מקבילות. $s,t\in V$ כאשר s הוא צומת "מקור" ו- t הוא צומת "בור". $c:E\to R^+$ היא פונקציית קיבול המגדירה לכל קשת $e\in E$ קיבול אי שלילי $c(e)$. לכל צומת v, $in(v)$ היא קבוצת הקשתות הנכנסות לצומת ו $out(v)$ היא קבוצת הקשתות היוצאות ממנו.

פונקצית זרימה: $f:E\to R^+$, היא פונקציה המגדירה לכל קשת e את כמות הזרימה $f(e)$ העוברת בה. פונקצית זרימה חייבת לקיים שני אילוצים:

1. <u>אילוץ הקיבול</u> (נקרא גם <u>חוק הקשת</u>) לכל קשת מתקיים $0\le f(e)\le c(e)$, כלומר הזרימה דרך קשת היא אי שלילית ואינה יכולה לחרוג מקיבול הקשת.
2. <u>שימור הזרימה</u> (נקרא גם <u>חוק הצומת</u>). לכל צומת $v\in V\setminus\{s,t\}$ (כלומר שאינו מקור או בור) מתקיים $\sum_{e\in out(v)} f(e)=\sum_{e\in in(v)} f(e)$, כלומר סה"כ הזרימה הנכנסת לצומת שווה לסה"כ הזרימה היוצאת מממנו.

ערך (או חוזק) פונקצית הזרימה: הזרימה נטו הנכנסת לבור. מסומן ב F או $|f|$:

$$F=\sum_{e\in in(t)} f(e)-\sum_{e\in out(t)} f(e)$$

חתך s-t ברשת זרימה:

חתך $(S,\bar S)$ הוא חלוקה של V: $S\subset V,\ \bar S=V-S$. חתך *s-t* מקיים $s\in S,\ t\in\bar S$.

יהי $(S,\bar S)$ חתך נתון. $(S\to\bar S)$ היא קבוצת הקשתות החוצות את החתך מ S ל $\bar S$ (ב"כיוון הקדמי"): $(S\to\bar S)=\{(u,v)\in E: u\in S, v\in\bar S\}$.

$(\bar S\to S)=\{(v,u)\in E: u\in S, v\in\bar S\}$ היא קבוצת הקשתות החוצות אותו ב"כיוון האחורי".

למה: לכל חתך $(S,\bar S)$ ולכל פונקצית זרימה f מתקיים

$$F=\sum_{e\in(S\to\bar S)} f(e)-\sum_{e\in(\bar S\to S)} f(e)$$

קיבול של חתך $(S,\bar S)$: סכום קיבולי הקשתות החוצות אותו בכיוון הקדמי:

$$c(S,\bar S)=\sum_{e\in(S\to\bar S)} c(e)$$

למה: לכל חתך $(S,\bar S)$ ולכל פונ' זרימה f מתקיים $F\le c(S,\bar S)$

הגרף השיורי ומסלולי שיפור:
רשת זרימה G ופונקציית זרימה f המוגדרת עליה מגדירים לכל קשת $e=(u,v)$ עם קיבול $c(e)$ וזרימה $f(e)$ שתי קשתות אנטי מקבילות עם קיבול שיורי (residual capacity):

קשת קדמית (u,v) עם קיבול שיורי $c(e)-f(e)$.

קשת אחורית (v,u) עם קיבול שיורי $f(e)$.

הגרף השיורי G_f הוא אוסף כל הקשתות עם קיבול שיורי חיובי.

מסלול שיפור (ביחס ל-G ו-f): מסלול מ- s ל- t בגרף השיורי G_f.

<u>משפט חתך מינימום – זרימת מקסימום (Min Cut – Max Flow):</u>

תהי f פונקצית זרימה ברשת זרימה $N=(G,s,t,c)$. הטענות הבאות שקולות:

1. f זרימת מקסימום.
2. אין מסלול שיפור מ- s ל- t בגרף השיורי G_f.
3. קיים חתך $s-t$ $(S,\bar{S})$ עבורו $F=c(S,\bar{S})$.

<u>למה:</u> אם לכל קשת e, הקיבול $c(e)$ הוא מספר שלם אז קיימת ברשת זרימת מקסימום בשלמים, ושיטת פורד פלקרסון תמיד תעצור ותמצא זרימת מקסימום בשלמים.

זרימה במספר בורות ומספר מקורות: על מנת למצוא זרימת מקסימום כאשר ישנם מספר בורות ומספר מקורות נוסיף צומת חדש s שיהיה המקור היחיד, ונמתח ממנו קשתות בקיבול ∞ לכל אחד מהמקורות הישנים. באופן דומה, נוסיף צומת חדש t שיהיה הבור היחיד, ונמתח אליו קשתות בקיבול ∞ מכל אחד מהבורות הישנים.

זרימה עם אילוצי קיבול על הצמתים: נפצל כל צומת $v \in V$ לשני צמתים v_{in} ו-v_{out}. את כל הקשתות שנכנסו ל-v נכוון ל-v_{in} ואת כל הקשתות שיצאו מ-v נוציא מ-v_{out}. כמו כן נוסיף קשת חדשה $(v_{in} \to v_{out})$ שקיבולה $c(v)$.

סיבוכיות של אלגוריתמים

בעיה	אלגוריתם	סיבוכיות	הערות
מיון טופולוגי	מבוסס מחיקת מקורות	$O(V+E)$	DAG בלבד
מסלולים קצרים ביותר	BFS	$O(V+E)$	
	DFS	$O(V+E)$	
גרף הרכיבים קשירים היטב	מבוסס DFS	$O(V+E)$	
עץ פורש מינימום	פרים	מבוסס מערך $O(V^2)$ מבוסס ערימה $O(ElogV)$	
עץ פורש מינימום	קרוסקל	מבוסס UF $O(ElogV)$	
מסלולים קלים ביותר ממקור יחיד	בלמן פורד	$O(VE)$	מוצא מעגלים שליליים אם יש
מסלולים קלים ביותר ממקור יחיד	דיקסטרה	$O(ElogV)$	קשתות במשקל אי שלילי בלבד
מסלולים קלים ביותר בין כל זוג	ג'ונסון	$O(VElogV)$	מוצא מעגלים שליליים אם יש
מסלולים קלים ביותר בין כל זוג	פלויד-וורשל	$O(V^3)$	
מספר מקסימלי של קטעים זרים	חמדן	$O(nlogn)$	
צביעת גרף ב-d_1+1 צבעים	חמדן	$O(V+E)$	
בניית עץ Huffman	רקורסיבי	$O(nlogn)$	
קבוצה בלתי תלויה של קטעים בעלת משקל מקסימלי	מבוסס תכנות דינמי	$O(nlogn)$	
Knapsack	מבוסס תכנות דינמי	$O(nW)$	
זרימת מקסימום	כל אלגוריתם המתאים לאלגוריתם הגנרי של פורד פלקרסון	$O(Ef^*)$	f^* הוא ערך הזרימה המקסימלית ברשת. התכנסות מובטחת רק במקרה של זרימה בשלמים.
זרימת מקסימום	אדמונדס-קרפ	$O(VE^2)$	
שידוך מקסימום בגרף דו-צדדי	מבוסס זרימה	$O(VE)$	
חישוב קונבולוציה	טרנספורם פורייה המהיר	$O(nlogn)$	