מבוא לאלגוריתמים - 234247

סיכומי הרצאות

גלעד קותיאל

11 ביוני 2018

תוכן העניינים

הרצאה 1

הקדמה - חיפוש בגרפים, BFS

הקדמה

> אלגוריתם הוא דרך שיטתית (כלומר כזו שצעדיה מוגדרים היטב) לביצוע של משימה מסוימת, במספר סופי של צעדים.

— ויקיפדיה

המושג אלגוריתם אינו חדש עבורנו, ראינו ומימשנו כבר אלגוריתמים בקורס מבוא למדעי המחשב, מבוא לתכנות מערכות ומבני נתונים.

חשיבות הקורס

בתעשייה - בעיות שמצריכות פתרון אלגוריתמי צצות במגוון תחומים. על פי glassdoor, מפתח אלגוריתמים מרוויח 20 אחוז יותר מאשר מהנדס תוכנה.
במחקר האקדמי - חלק עיקרי של המחקר האקדמי הוא בפיתוח וניתוח אלגוריתמים.

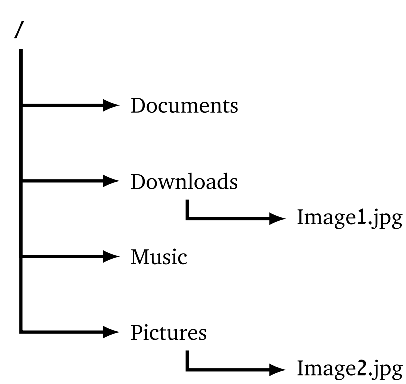
חומר הקורס

בקורס נלמד מגוון אלגוריתמים שעל פי רוב נחשבים לבסיס בתחום האלגוריתמים. הרוב המוחלט של האלגוריתמים שנלמד בקורס הם אלגוריתמים על גרפים וזאת מכיוון שגרפים הוכיחו את עצמם ככלי מאוד חזק בייצוג מגוון רחב של בעיות. לחלק מהאלגוריתמים שימושים ברורים (מסלול קצר ביותר, עצי הופמן), חלק מהאלגוריתמים מהווים פתרון למגוון גדול של בעיות שניתנות לייצוג בצורה מסוימת (זרימה), וחלק מהאלגוריתמים מהווים בסיס לפיתוח אלגוריתמים מורכבים יותר (BFS, DFS, עץ פורש). מעבר לזה נלמד טכניקות (פשוטות יחסית) כלליות לפיתוח אלגוריתמים.

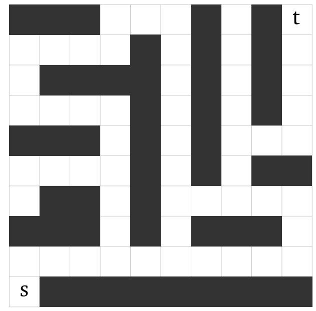
אלגוריתמי חיפוש בגרפים

דוגמה 1 (קבצים)**.** רוצים למצוא (ולהדפיס) את כל קבצי התמונות ששמורות על הכונן הקשיח.

למשל עבור:



דוגמה 2 (מבוך)**.** נתון מבוך, נקודת התחלה ונקודת סוף ורוצים למצוא מסלול מנקודת ההתחלה לנקודת הסיום.



דוגמה 3 (פאזל הזזה). נתון לוח משחק בגודל $n \times m$ על הלוח $nm-1$ חלקים ממוספרים מ-1 עד $nm-1$ ומשבצת ריקה. נתון סידור ראשוני של החלקים ואנו רוצים לסדר את החלקים לפי הסדר כך שבכל שלב מותר לנו להזיז את אחד החלקים ששכנים למשבצת הריקה אל המשבצת הריקה.

למשל עבור $n = m = 3$:

1	3	6
8	4	
5	2	7

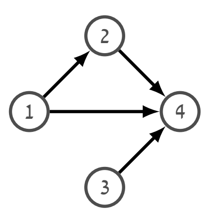
נראה בהמשך שאפשר לייצג כל אחת מהבעיות הנ"ל באמצעות גרף, וסריקה של הגרף המתאים מאפשרת לנו למצוא את הפתרון. כיצד עלינו לסרוק כל אחד מהגרפים המתאימים?

ייצוג גרפים

קיימים שני ייצוגים סטנדרטים של גרפים (מכוונים או לא):

1. על ידי מטריצת שכנויות
2. על ידי רשימת שכנויות

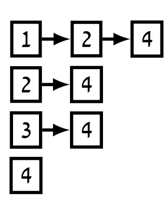
אם לא מצוין אחרת, נניח שהגרף מיוצג על ידי רשימת שכנויות.
למשל את הגרף:



ניתן לייצג על ידי המטריצה

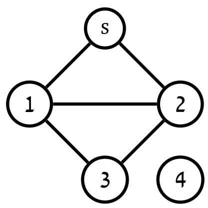
$$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

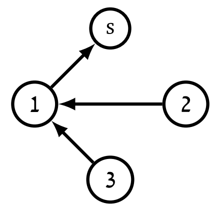
וגם על ידי רשימת שכנויות:



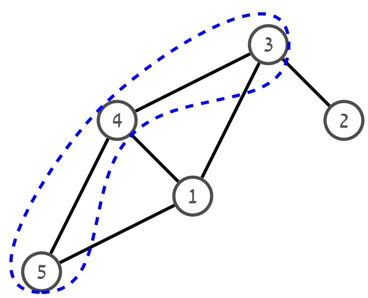
אלגוריתם סריקה כללי

קלט: גרף $G=(V,E)$ (מכוון או לא) וצומת מקור s.
פלט: עץ עם שורש s, $T=(U,F)$, $U \subseteq V$, $F \subseteq E$ כך ש-U היא קבוצת הצמתים שישיגים מ-s. בנוסף, לכל צומת $u \in U$ נרצה ש-$p(u)$ יצביע לאבא של u.
למשל:





הגדרה 1 (חתך)**.** חתך בגרף, $G=(V,E)$, הוא תת קבוצה של צמתים. $S \subseteq V$ נאמר שקשת $uv \in E$ חוצה את החתך S אם $|\{u,v\} \cap S| = 1$.



אלגוריתם:

1. אתחול: $F \leftarrow \emptyset$, $U \leftarrow \{s\}$ ולכל $v \in V$ מציבים $p(v) \leftarrow \text{nil}$
2. כל עוד יש קשת uv שחוצה את U ($u \in U$)
   (א) $U \leftarrow U \cup \{v\}$, $F \leftarrow F \cup \{uv\}$, $p(v) \leftarrow u$
3. החזר $T=(U,F)$

טענה 1. בסיום ריצת האלגוריתם U מכילה את כל הצמתים הישיגים מ-s

הוכחה. בשלילה, בוחרים מסלול מ-s לצומת v שלא נכנס ל-U ומסתכלים על הצומת הראשון במסלול שלא נכנס. □

טענה 2. בכל שלב בריצת האלגוריתם T עץ קשיר. בנוסף המסלול מצומת u ל-s הוא שרשור של הקשת $(u,p(u))$ והמסלול מ-$p(u)$ ל-s

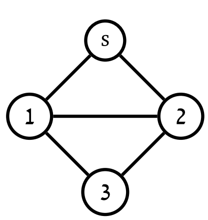
הוכחה. באינדוקציה על צעד האלגוריתם. □

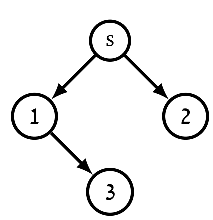
חיפוש לרוחב - Breadth First Search (BFS)

הגדרה 2 (מרחק)**.** בהינתן גרף $G=(V,E)$, נגדיר את המרחק בין שני צמתים $u,v \in V$, ונסמנו $dist_G(u,v)$, כמספר הקשתות המינימלי במסלול מ-u ל-v.

הערה: כאשר ברור על איזה גרף מדובר נסתפק בסימון $dist(u,v)$.

קלט: גרף G (מכוון או לא) וצומת מקור s.
פלט: עץ עם שורש s, $T=(U,F)$, $U \subseteq V$, $F \subseteq E$ כך ש-U היא קבוצת הצמתים שישיגים מ-s. בנוסף, לכל צומת $u \in U$ מתקיים $d(u) = dist_T(s,u) = dist_G(s,u)$ ו-$p(u)$ יצביע לאבא של u.
למשל:





אלגוריתם:

1. אתחול: $F \leftarrow \emptyset$, $U \leftarrow \{s\}$, לכל $v \in V$, $p(v) \leftarrow nil$, $d(v) \leftarrow \infty$, $d(s) \leftarrow 0$
2. כל עוד ישנה קשת uv שחוצה את U ($u \in U$) בחר קשת עם $d(u)$ מינימלי
   (א) $U \leftarrow U \cup \{v\}$, $F \leftarrow F \cup \{uv\}$, $p(v) \leftarrow u$
   (ב) $d(v) = d(u) + 1$

BFS הוא מקרה פרטי של האלגוריתם הכללי.

טענה 3. לכל $v \in V$ מתקיים $d(v) \geq dist_G(s, v)$

הוכחה. נניח בשלילה שהטענה לא מתקיימת ונסתכל על הצומת הראשון, v, שנכנס ל-U ומפר את הטענה, כלומר, אם המרחק של v מ-s הוא k אז $d(v) \leq k - 1$. המצב הנ"ל קורה אמ"מ בקשת uv שהאלגוריתם בחר מתקיים ש $d(u) \leq k - 2$, כלומר, המרחק של u מ-s הוא לכל היותר $k - 2$ וזה סתירה למרחק של v מ-s. □

טענה 4. לכל $v \in V$ מתקיים $d(v) \leq dist_G(s, v)$

הוכחה. נניח בשלילה שהטענה לא מתקיימת ונסתכל על הצומת הראשון, v, שנכנס ל-U ומפר את הטענה, כלומר, אם המרחק של v מ-s הוא k אז $d(v) \geq k + 1$. המצב הנ"ל יכול לקרות אמ"מ האלגוריתם בחר את הקשת uv ו-$d(u) \geq k$. כעת נסתכל על הקשת הראשונה שחוצה את החתך, ww', במסלול באורך k מ-s ל-v. אז מתקיים ש-$d(w) \leq k - 1$ וזה בסתירה להגדרת האלגוריתם. □

טענה 5. לכל $v \in V$ מתקיים $d(v) = dist_T(s, v)$

הוכחה. באינדוקציה על צעד האלגוריתם □

משפט 1. לכל $v \in V$ מתקיים $d(v) = dist_T(s, v) = dist_G(s, v)$

מהו זמן הריצה של BFS ? קשה להגיד כי לא הגדרנו כיצד מתבצעת הבדיקה בשלב 2 של האלגוריתם.

חיפוש לרוחב - מימוש באמצעות תור

ניתן לממש BFS על ידי תור באופן הבא:

1. אתחול: $F \leftarrow \emptyset$, $U \leftarrow \{s\}$, לכל $v \in V$ מציבים $d(v) \leftarrow \infty$, $p(v) \leftarrow nil$, $d(s) \leftarrow 0$, $Q \leftarrow (s)$
2. כל עוד התור לא ריק
   (א) $u \leftarrow Q.pop()$
   (ב) כל עוד ישנה קשת uv שחוצה את U ($u \in U$)
      i. $U \leftarrow U \cup \{v\}$, $F \leftarrow F \cup \{uv\}$, $p(v) \leftarrow u$
      ii. $d(v) = d(u) + 1$
      iii. $Q.push(v)$

נראה שזהו אכן מימוש של BFS.

הגדרה 3 (צומת גבולי). בהינתן גרף $G = (V, E)$ וחתך $U \subseteq V$ צומת $u \in U$ יקרא גבולי אם קיימת קשת uv שחוצה את U.

טענה 6. בכל שלב בריצת האלגוריתם התור מכיל את כל הצמתים הגבוליים

הוכחה. באינדוקציה על צעד האלגוריתם □

טענה 7. התור מונוטוני לא יורד בהתייחס לערכי d

הוכחה. נוכיח טענה חזקה יותר באינדוקציה על צעד האלגוריתם: התור מונוטוני לא יורד וגם $|d(u) - d(v)| \leq 1$ לכל שני צמתים שבתור □

מסקנה 1. זהו אכן מימוש של BFS

סיבוכיות: נשים לב שבמימוש הנ"ל כל צומת נכנס ויוצא מהתור לכל היותר פעם אחת ובנוסף כל קשת נבדקת לכל היותר פעמיים ולכן זמן הריצה הוא $O(|V| + |E|)$

הרצאה 2

חיפוש לעומק - Depth First Search (DFS)

תזכורת

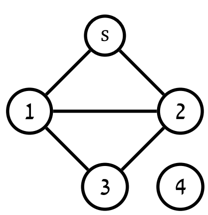
בהינתן גרף G וצומת s רוצים למצוא עץ T שפורש את כל הצמתים שישיגים מ-s.

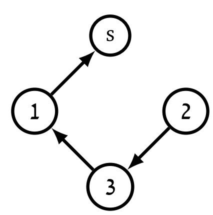
- אלגוריתם כללי
- BFS
- מימוש BFS באמצעות תור

DFS

1. אתחול: $F \leftarrow \emptyset$, $U \leftarrow \{s\}$, לכל $v \in V$ מציבים $d(v) \leftarrow -1$, $p(v) \leftarrow nil$, $d(s) \leftarrow 0$, $i \leftarrow 0$
2. כל עוד ישנה קשת uv שחוצה את U ($u \in U$) בחר קשת עם $d(u)$ מקסימלי
   (א) $U \leftarrow U \cup \{v\}, F \leftarrow F \cup \{uv\}$
   (ב) $p(v) \leftarrow u$
   (ג) $d(v) \leftarrow i$
   (ד) $i \leftarrow i + 1$

דוגמה





מימוש על ידי מחסנית

1. (א) אתחול: $F \leftarrow \emptyset$, $U \leftarrow \{s\}$, לכל $v \in V$ מציבים $d(v) \leftarrow -1$, $p(v) \leftarrow nil$, $d(s) \leftarrow 0$, $i \leftarrow 0$, $S \leftarrow (s)$
2. כל עוד המחסנית לא ריקה
   (א) $u \leftarrow S.top()$
   (ב) אם קיימת קשת uv שחוצה את U ($u \in U$)
      i. $U \leftarrow U \cup \{v\}, F \leftarrow F \cup \{uv\}$
      ii. $p(v) \leftarrow u$
      iii. $d(v) \leftarrow i$
      iv. $S.push(v)$

(ג) אחרת

i. $u \leftarrow S.pop()$

ii. $\beta(u) = i$

(ד) $i \leftarrow i + 1$

טענה 8. בזמן ריצת האלגוריתם, כל הצמתים הגבוליים נמצאים במחסנית

הוכחה. באינדוקציה על צעד האלגוריתם □

טענה 9. המחסנית מונוטונית עולה ביחס ל-d

הוכחה. באינדוקציה על צעד האלגוריתם □

מסקנה 2. זהו מימוש של DFS

הערה: מכיוון שזמני הגילוי של הצמתים הם יחודיים (בשונה מהמרחקים שלהם למשל) אזי המימוש באמצעות מחסנית שקול לכל מימוש אחר של DFS (הדבר אינו נכון לגבי מימוש של BFS באמצעות תור). לכן, כל טענה לגבי המימוש באמצעות מחסנית תקפה עבור DFS באופן כללי.

תכונות

טענה 10. בזמן ריצת DFS, הצמתים במחסנית, $v, \dots, s$ הם המסלול ב-T מ-s ל-v

הוכחה. באינדוקציה על צעד האלגוריתם □

מסקנה 3. עבור שני צמתים u ו-v, v צאצא של u ב-T אם ורק אם u נמצא במחסנית כאשר v מוכנס אליה.

הוכחה. כיוון ראשון מיידי מטענה 10.
כיוון שני גם מטענה 10 כאשר האבחנה היא שבעץ, צומת u הוא אב קדמון של v אם ורק אם הוא נמצא על המסלול מ-s ל-v □

הגדרה 4 (צומת לבן)**.** בזמן ריצת האלגוריתם, נקרא לצמתים ב-U <u>שחורים</u> ולשאר הצמתים <u>לבנים</u>

אבחנה 1. צומת יוצא מהמחסנית רק אחרי שכל שכניו שחורים.

למה 1 (המסלול הלבן)**.** צומת v צאצא של צומת u ב-T אמ"מ כאשר u מוכנס למחסנית קיים ממנו מסלול של צמתים לבנים לצומת v

הוכחה. כיוון 'אם' באינדוקציה על אורך המסלול. הצומת הראשון במסלול שנכנס למחסנית מחלק את המסלול לשני מסלולים קצרים יותר. פרט קטן אך חשוב אחד הצמתים במסלול אכן נכנס למחסנית.
כיוון 'רק אם' נניח בשלילה ש-v צאצא של u אבל כל מסלול ביניהם מכיל צומת שחור אחד לפחות אז בזמן הכנסת v למחסנית תוכן המחסנית מכיל את המסלול מ-u ל-v ולכן הכנסנו למחסנית צומת שחור - סתירה. □

יער DFS

נכליל את אלגוריתם ה-DFS. הקלט הוא גרף (מכוון או לא) הפלט הוא יער של עצים מושרשים כאשר כל עץ מקיים את כל התכונות עליהן דיברנו עד כה.

1. אתחול: $U \leftarrow \emptyset, F \leftarrow \emptyset$, לכל $v \in V$ מציבים $p(v) \leftarrow nil, \alpha(v) \leftarrow -1$, $i \leftarrow 0$
2. כל עוד $U \neq V$

(א) בחר צומת $s \in V \setminus U$

(ב) $U \leftarrow U \cup \{s\}$, $\alpha(s) \leftarrow i$

(ג) כל עוד ישנה קשת uv שחוצה את U ($u \in U$) בחר קשת עם $\alpha(u)$ מקסימלי

i. $U \leftarrow U \cup \{v\}, F \leftarrow F \cup \{uv\}$

ii. $p(v) \leftarrow u$

iii. $\alpha(v) \leftarrow i$

iv. $i \leftarrow i + 1$

נראה בהמשך שזהו בדרך כלל האלגוריתם שנרצה להריץ.

סיכום

DFS משמש כאבן בניין לאלגוריתמים יותר מורכבים (רכיבים בלתי פריקים, רכיבים קשירים היטב, מיון טופולוגי).

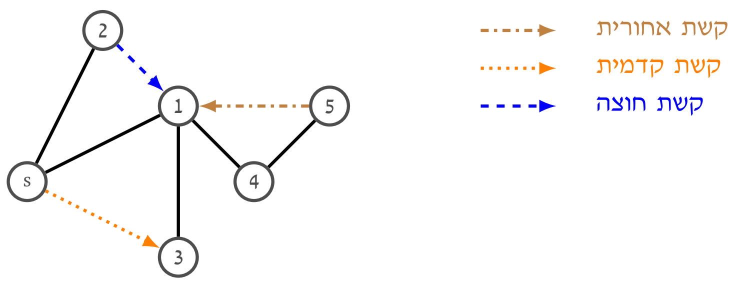
דוגמה 4 (מיון טופולוגי)**.** הרץ DFS, פלוט את הצמתים לפי סדר הוצאתם מהמחסנית.

סיווג קשתות

בהינתן גרף מכוון ותת עץ (מושרש) שלו מסווגים את קשתות הגרף ל-4 סוגים:

1. קשתות עץ
2. קשתות קדמיות
3. קשתות אחוריות
4. קשתות חוצות

הערה: בגרף לא מכוון נתייחס לקשתות קדמיות וקשתות אחוריות כקשתות אחוריות.



טענה 11. בגרף לא מכוון ועץ שהוא פלט של DFS אין קשתות חוצות

הוכחה. באמצעות למת המסלול הלבן ☐

הרצאה 3

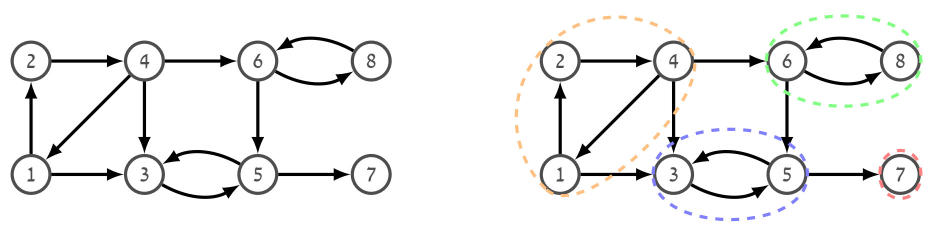
DFS - רכיבים קשירים היטב, צמתי הפרדה, רכיבים אי פריקים

רכיבים קשירים היטב

בהינתן גרף מכוון, $G=(V,E)$ נגדיר את היחס הבא: $R_{\mathcal{C}}=\{(u,v): u \rightsquigarrow v \wedge v \rightsquigarrow u\}$, כלומר צמתים u ו-v ביחס אם קיים מסלול מ-u ל-v וקיים מסלול מ-v ל-u.
תזכורת: יחס, R, הוא יחס שקילות מעל $A \times A$ אם:

1. רפלקסיביות - לכל $a \in A$ מתקיים ש-$(a,a) \in R$.
2. סימטריות - $(a,b) \in R \implies (b,a) \in R$
3. טרנזיטיביות - $(a,b) \in R, (b,c) \in R \implies (a,c) \in R$

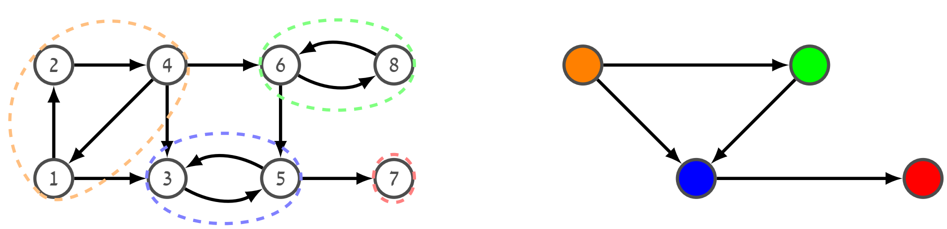
נשים לב ש-$R_{\mathcal{C}}$ הוא יחס שקילות ולכן הוא מגדיר מחלקות שקילות. למחלקות שקילות אלו נקרא קבוצת הרכיבים קשירים היטב (רק"ה) של G.
דוגמה:



דיון: הציעו אלגוריתם (יעיל ככל האפשר) שבהינתן גרף מכוון, $G=(V,E)$, וצומת, $v \in V$, מוצא את רכיב הקשירות של v.

גרף הרכיבים קשירים היטב

נסמן את קבוצת הרק"ה של גרף ב-$\mathcal{C}=\{C_1,\ldots,C_k\}$, אז לכל $i \neq j$ מתקיים $C_i \cap C_j = \emptyset$ וגם $\bigcup_{i=1}^{k} C_i = V$ (יחס שקילות). גרף הרק"ה של G יסומן ב-$G_{scc}=(\mathcal{C},E_{scc})$ כאשר $E_{scc}=\{(C_i,C_j): \exists\,(u,v)\in E, u \in C_i \wedge v \in C_j\}$. ניתן לחשוב על גרף זה כגרף המתקבל על ידי כיווץ הרק"ה של G וביטול קשתות מקבילות.
דוגמה:



אבחנה 2. גרף הרק"ה חסר מעגלים.

הוכחה. אם קיים מעגל אז קיבלנו סתירה להגדרה של הגרף.

□

שאלה: איך נראה גרף הרק"ה של רשת כבישים? כיצד נראה גרף הרק"ה של ויקיפדיה? של דפי האינטרנט?

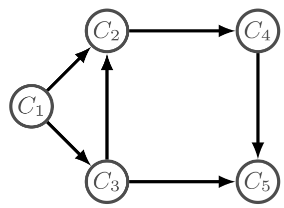
אלגוריתם למציאת רכיבים קשירים היטב

מטרה: בהינתן גרף מכוון נרצה למצוא את גרף הרק"ה שלו. פלט האלגוריתם צריך להיות מיפוי של כל צומת לרכיב קשיר היטב שמכיל אותה (מספר בין 1 ל-k).
נשים לב שבהינתן מיפוי כנ"ל ניתן לבנות את גרף הרק"ה על ידי מעבר בודד על קשתות הגרף המקורי.
עבור קבוצת צמתים $C \subseteq V$ נרחיב את זמן הסיום של אלגוריתם DFS לקבוצת צמתים כך: $f(C) = \max_{v \in C} f(v)$. כלומר זמן הסיום המאוחר ביותר של צומת בקבוצה.
הטענה הבאה מתייחסת לגרף רק"ה.

טענה 12. אם $(C_i, C_j) \in E_{scc}$ אז לכל ריצת DFS על הגרף המקורי, G, מתקיים ש-$f(C_i) > f(C_j)$.

הוכחה. נפריד לשני מקרים:
מקרה ראשון: מבקרים ב-C_i לפני שמבקרים ב-C_j. אז קיים מסלול לבן מהצומת הראשון שמתגלה ב-C_i לכל שאר הצמתים ב-C_i וגם ב-C_j ולפי משפט המסלול הלבן צומת זה יהיה אב קדמון של כל הצמתים הנ"ל ולכן יהיה עם זמן סיום המאוחר ביותר מבניהם.
מקרה שני: מבקרים בצומת מ-C_j לפני שמבקרים בצומת מ-C_i. אז מכיוון שגרף הרק"ה חסר מעגלים אין מסלול מצומת ב-C_j לצומת ב-C_i. ולכן זמן הסיום של הצומת הראשון שמתגלה ב-C_j יהיה מוקדם יותר מזמן הגילוי (ולכן גם הסיום) של כל צומת ב-C_i. מצד שני, בזמן גילוי הצומת הראשון ב-C_j קיים מסלול לבן לכל שאר הצמתים ב-C_j ולכן הצומת הראשון יהיה אב קדמון של כל הצמתים ב-C_j וזמן הסיום שלו יהיה המאוחר ביותר מבניהם. □

דיון: נניח שעבור גרף G גרף הרק"ה, G_{scc}, נראה כך:

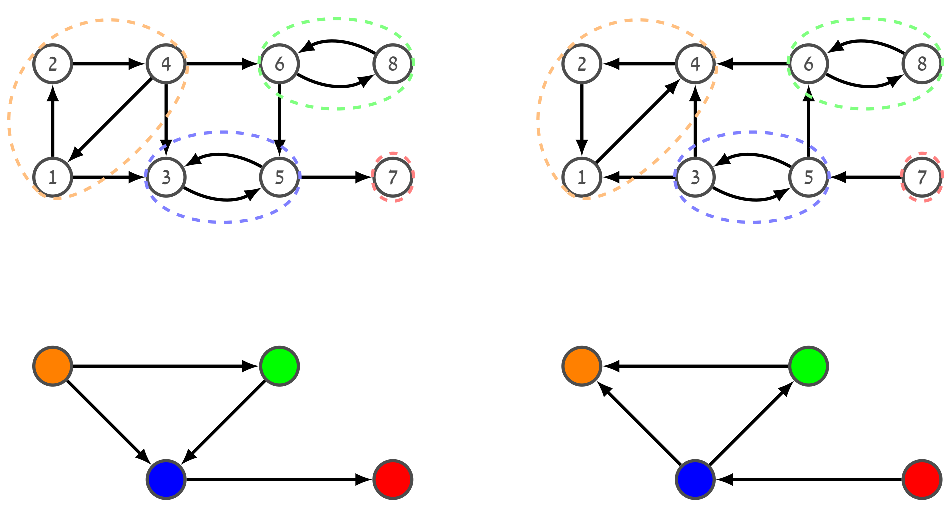


כעת, נניח שהרצנו DFS על G. לאיזה רק"ה שייך הצומת עם זמן הסיום המקסימלי? מה יקרה אם נבחר בו בתור הצומת הראשון בהרצת DFS?

הגדרה 5 (גרף משוחלף). הגרף המשוחלף של גרף מכוון, $G = (V, E)$, הוא הגרף המתקבל על ידי הפיכת כיוון הקשתות, כלומר $G^T = (V, E^T)$ כאשר $E^T = \{(u, v) : (v, u) \in E\}$.

אבחנה 3. $(G_{scc})^T = (G^T)_{scc}$.

דוגמה:



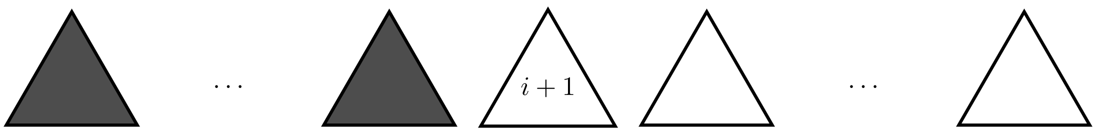
מסקנה 4. אם $(C_i, C_j) \in E^T_{scc}$ אז לכל ריצת DFS על הגרף המקורי, G, מתקיים ש-$f(C_i) < f(C_j)$.

אלגוריתם:

1. הרץ DFS על G.
2. הרץ DFS על G^T, בחר את הצמתים בסדר יורד של זמן הסיום שלהם משלב 1.
3. החזר את יער ה-DFS שהתקבל בשלב 2 (כל עץ הוא רכיב קשיר היטב)

טענה 13. כל עץ ביער ה-DFS שמוחזר בשלב 3 הוא רכיב קשירות.

הוכחה. באינדוקציה על העץ ה-i בריצת ה-DFS.
צעד: נשים לב שעל פי הנחת האינדוקציה, מהשורש ה-$i+1$ קיים מסלול לבן לכל הצמתים ברק"ה שלו. נניח בשלילה שקיים מסלול לבן לצומת ברכיב קשירות אחר ונקבל סתירה על הסדר שבו אנחנו בוחרים את השורשים. □

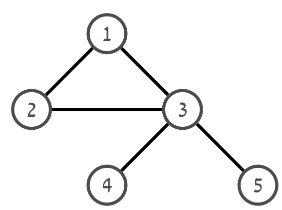


צמתי הפרדה (תלוי סמסטר)

מעתה נעסוק רק בגרפים לא מכוונים וקשירים (בלי הגבלת הכלליות).

הגדרה 6 (צומת הפרדה)**.** צומת v יקרא <u>צומת הפרדה</u> אם $G[V \setminus \{v\}]$ אינו קשיר

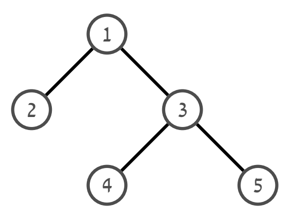
דוגמה: צומת 3 בגרף הבא הוא צומת הפרדה (ורק הוא)



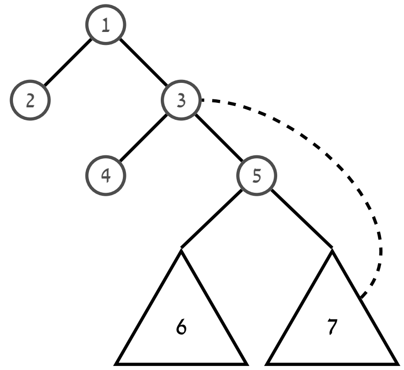
מה המשמעות של צמתי הפרדה ? ברשת כבישים ? רשת תקשורת ? רשת חברתית ?
אלגוריתם טריוויאלי למציאת צמתי הפרדה:

1. עבור כל צומת v
   (א) מחק את v מ-G
   (ב) בדוק אם G קשיר

מה הסיבוכיות ? נרצה לעשות יותר טוב.
שאלה: מי הם צמתי ההפרדה בעצים?



נסתכל על גרף לא מכוון כאיחוד של עץ DFS וקבוצת קשתות אחוריות. מה יכול לקרות כאשר מוציאים קשת שאינו עלה מהגרף ?
למשל מה יקרה אם נסיר את צומת 5 מהגרף הבא ?



קל להשתכנע שתת העץ 7 ישאר מחובר לגרף בעוד שתת העץ 6 יתנתק מהגרף.
בעץ מושרש, T, נסמן ב-T_v את תת העץ ששורשו הוא v. כלומר תת העץ שמכיל את v ואת כל צאצאיו.

הגדרה 7 (קשת עוקפת)**.** נגיד שקשת בגרף מצאצא של u לאב קדמון של u (שניהם לא u עצמו) עוקפת את u

הגדרה 8 (בן מפריד)**.** צומת v עם אבא u יקרא בן מפריד אם לא קיימת ב-T_v קשת שעוקפת את u

טענה 14. צומת u בעץ DFS שאינו עלה הוא מפריד אם ורק אם יש לו בן מפריד. בפרט שורש העץ הוא צומת מפריד אמ"מ הוא אינו עלה (יש לו יותר מבן אחד)

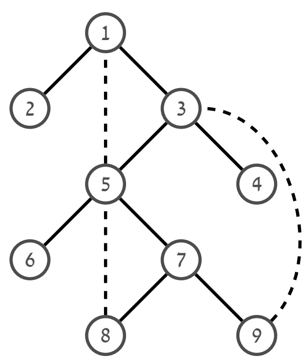
הוכחה. נזכיר שבגרף אין קשתות חוצות, כמו כן נניח ש-u אינו השורש (ניתן להוכיח נכונות לגבי השורש בנפרד)
כיוון ראשון: נניח שמ-T_v אין קשת לאב קדמון של u אזי כל מסלול מהאבא של u ל-T_v חייב לעבור ב-u.
כיוון שני: נניח שלכל בן v של u קיימת קשת עוקפת מ-T_v, wx נשים לב שהוספת הקשת wx סוגרת מעגל שמכיל את הקשת uv ולכן אם נסיר את הקשת uv נקבל שוב עץ, נחזור על הפעולה הזאת לכל בן של u ולבסוף נסיר את u מהעץ. קיבלנו שוב עץ ולכן u אינו צומת הפרדה.

□

הגדרה 9. נגדיר

$$L(u) := \min_{vw \in E : v \in T_u} \alpha(w)$$

כלומר הצומת עם ערך α מינימלי שהוא שכן של T_u.
דוגמה: מה ערכי L בגרף הבא ?



אבחנה 4. צומת v הוא בן מפריד של u אמ"מ $L(v) \geq \alpha(u)$

כלומר, כדי למצוא אלגוריתמית את צמתי ההפרדה כל שעלינו לעשות הוא לחשב את ערכי L. נשים לב שמתקיימת הנוסחה (הרקורסיבית) הבאה:

$$L(u) = \min \begin{cases} \min\limits_{uv \in E} \alpha(v) \\ \min\limits_{v \in C(u)} L(v) \end{cases}$$

נעדכן את המימוש של DFS כך שבריצת האלגוריתם נחשב גם את ערכי L

1. אתחול: ... לכל $v \in V$ מציבים $L(v) \leftarrow \infty$
2. כל עוד המחסנית לא ריקה
   (א) $u \leftarrow S.top()$
   (ב) אם קיימת קשת uv שחוצה את U ($u \in U$)
      i. $U \leftarrow U \cup \{v\}, F \leftarrow F \cup \{uv\}$
      ii. $p(v) \leftarrow u$
      iii. $\alpha(v) \leftarrow i$
      iv. $S.push(v)$
   (ג) אחרת
      i. $L(u) \leftarrow \min\limits_{uv \in E} \alpha(v)$
      ii. $L(p(u)) \leftarrow \min\{L(u), L(p(u))\}$
      iii. $u \leftarrow S.pop()$
      iv. $\beta(u) \leftarrow i$
   (ד) $i \leftarrow i + 1$

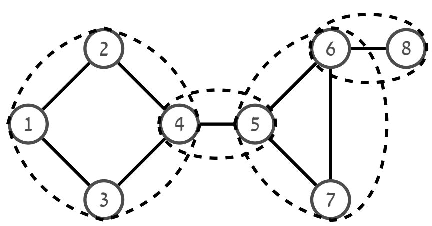
רכיבים אי פריקים

הגדרה 10 (גרף אי פריק). גרף (קשיר) יקרא אי פריק אם אין בו צמתי הפרדה

במילים אחרות, הגרף נשאר קשיר גם אחרי הסרת צומת כלשהו ממנו.

הגדרה 11 (רכיב אי פריק). רכיב אי פריק H של G הוא תת גרף (קשיר) אי פריק מקסימלי של G

דוגמה:



טענה 15. לשני רכיבים אי פריקים H_1 ו-H_2 צומת אחד משותף לכל היותר

הוכחה. נניח בשלילה שישנם שני רכיבים כאלו שחולקים את הצמתים u ו-v. נשים לב שלאחר שהסרה של צומת כלשהו כל רכיב בנפרד נשאר קשיר. מעבר לכך הרכיבים עדיין חולקים צומת משותף ולכן הרכיב שמתקבל מאיחוד שני הרכיבים גם הוא אי פריק. סתירה למקסימליות. □

טענה 16. הרכיבים האי פריקים מהווים חלוקה של קשתות הגרף

הוכחה. קשת היא גרף אי פריק ולכן מוכלת ברכיב אי פריק אחד לפחות. מטענה 15 נובע שגם לכל היותר □

טענה 17. כל מעגל ב-G מוכל ברכיב פריק של G

הוכחה. נובע ישירות מכך שמעגל הוא גרף אי פריק □

נרצה למצוא את הרכיבים האי פריקים של גרף G.

טענה 18. עבור צומת הפרדה u עם בן מפריד v, כל צמתי הרכיב האי פריק שמכיל את uv (פרט ל-u) נמצאים ב-T_v

הוכחה. נשים לב ש-u מפריד את T_v משאר הגרף □

נסמן ב-S את קבוצת הבנים המפרידים אז

מסקנה 5. צמתי הרכיב האי פריק שמכיל את uv הוא $T_v \cup \{v\} \setminus \bigcup_{w \in S: w \neq v} T_w$

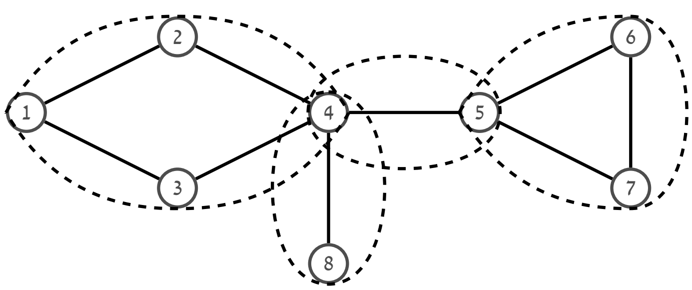
נעדכן את המימוש של DFS כך שבריצת האלגוריתם נחשב גם את הרכיבים האי פריקים (נרצה לשמור לכל צומת את מספר הרכיב אליו הוא שייך)

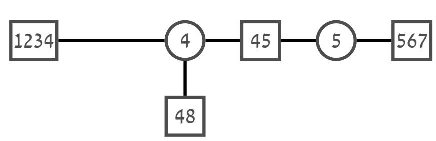
1. אתחול: ... $S' \leftarrow (s)$, $b \leftarrow 0$, לכל צומת $v \in V$ $B(v) \leftarrow -1$
2. כל עוד המחסנית לא ריקה
   (א) $u \leftarrow S.top()$
   (ב) אם קיימת קשת uv שחוצה את U ($u \in U$)
      i. ...
      ii. $S'.push(v)$
   (ג) אחרת
      i. $v \leftarrow S.pop()$
      ii. $\beta(v) \leftarrow i$
      iii. אם v בן מפריד של u
         א'. $w \leftarrow S'.pop()$
         ב'. כל עוד $w \neq v$
            - $B(w) = b$
            - $w \leftarrow S'.pop()$
         ג'. $b \leftarrow b + 1$
   (ד) $i \leftarrow i + 1$

עץ רכיבים אי פריקים

עבור גרף G נגדיר את גרף הרכיבים האי פריקים, $B(G)$, כגרף הדו צדדי על קבוצות הצמתים B ו-S כאשר ב-B צומת עבור כל רכיב אי פריק ב-G וב-S צומת עבור כל צומת הפרדה ב-G. בגרף הנ"ל תהיה קשת bs, $b \in B$, $s \in S$, אמ"מ הרכיב שמתאים ל-b מכיל את הצומת s.
נשים לב שכל מסלול ב-G מתאים למסלול (יחיד) ב-$B(G)$ ולכן $B(G)$ קשיר. כמו כן נשים לב שב-$B(G)$ לא קיימים מעגלים כי זה יגרור שקיים מעגל ב-G שאינו שעובר ביותר מרכיב אי פריק אחד.

מסקנה 6. $B(G)$ הוא עץ





הרצאה 4

עץ פורש מינימלי - פרים, קרוסקל

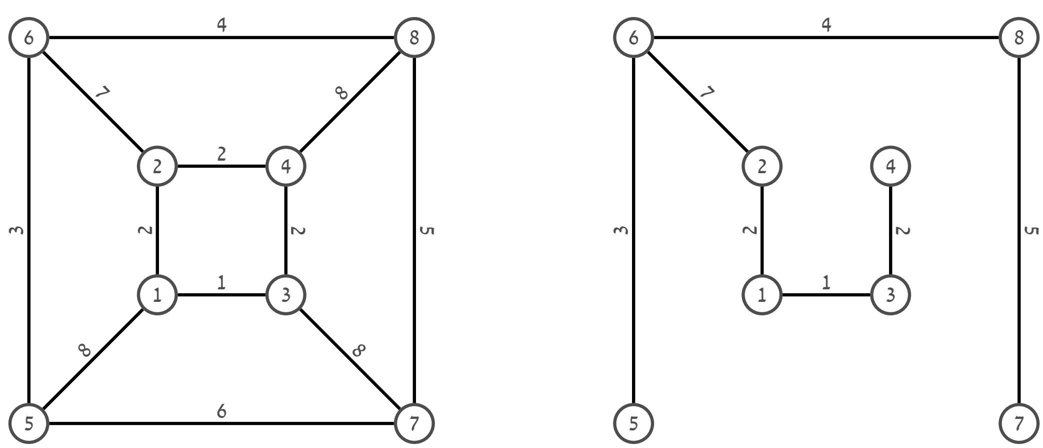
הגדרות ואבחנות

יער - גרף חסר מעגלים

עץ - יער קשיר

הגדרה 12 (עץ פורש מינימלי)**.** בהינתן זוג של גרף לא מכוון $G=(V,E)$ ופונקציית משקל (אי שלילית) $w:E\to\mathbb{R}$ עץ פורש מינימלי הוא כל עץ $T=(V,F)$ שממזער את הערך $\sum\limits_{e\in F} w(e)$.

דוגמה:



חתך - תת קבוצה של צמתים $S\subseteq V$

קשת חוצה - קשת uv חוצה חתך S אם $|\{u,v\}\cap S|=1$

אבחנה 5. גרף קשיר אמ"מ כל חתך לא טריוויאלי נחצה על ידי קשת אחת לפחות.

אבחנה 6. הוספת קשת uv לעץ סוגרת מעגל שמכיל את הקשת + המסלול מ-v ל-u בעץ. כעת ניתן להסיר כל קשת מהמעגל הנ"ל ולקבל בחזרה עץ.

אבחנה 7. אם קשת uv נמצאת על מעגל והיא חוצה חתך S אז את S חוצה קשת אחת נוספת (לפחות) ששייכת למעגל.

אבחנה 8. מחיקת קשת e מעץ יוצרת יער עם שני רכיבי קשירות. נסמן ב-U_e את החתך שהרכיבים הללו מגדירים (אחד הרכיבים באופן שרירותי). אם נוסיף לגרף קשת כלשהי שחוצה את החתך U_e קיבלנו שוב עץ.

הכלל הכחול והאדום

נניח שנתון לנו גרף לא מכוון וקשיר, $G=(V,E)$, כך שהקשתות שלו צבועות בכחול, אדום ולבן. נסמן ב-R,B ו-W את החלוקה של E בהתאם. בנוסף, נתונה לנו פונקציית משקל $w:E\to\mathbb{R}$
נגדיר שני כללים:

- הכלל האדום: בחר מעגל שלא מכיל קשתות אדומות, מבין הקשתות הלבנות על המעגל בחר אחת עם משקל מקסימלי וצבע אותה באדום.
- הכלל הכחול: בחר חתך שלא נחצה על ידי קשתות כחולות, מבין הקשתות הלבנות שחוצות את החתך בחר אחת עם משקל מינימלי וצבע אותה בכחול.

נניח בנוסף שקיים עפ"מ, $T \subseteq E$, שמכיל את כל הקשתות הכחולות ולא מכיל קשתות אדומות, כלומר: $B \subseteq T$ וגם $T \cap R = \emptyset$. נוכיח את שתי הטענות הבאות:

טענה 19. אם ניתן להפעיל את הכלל האדום אז קיים עפ"מ שמקיים את התנאים הנ"ל גם אחרי הפעלת הכלל האדום.

הוכחה. נניח שהפעלנו את הכלל האדום וצבענו את הקשת e באדום. אם $e \notin T$ סיימנו. אחרת, נסתכל על החתך שנוצר על ידי הסרה של e מ-T. מכיוון ש-e קשת על מעגל אז קיימת קשת נוספת, e' שחוצה את החתך (ולא שייכת ל-T). מכיוון של-e משקל מקסימלי, העץ $T \setminus \{e\} \cup \{e'\}$ הוא עפ"מ. □

טענה 20. אם ניתן להפעיל את הכלל הכחול אז קיים עפ"מ שמקיים את התנאים הנ"ל גם אחרי הפעלת הכלל הכחול.

הוכחה. נניח שהפעלנו את הכלל הכחול על חתך S וצבענו את הקשת e בכחול. אם $e \in T$ סיימנו. אחרת, נוסיף את e ל-T, אז e נמצאת על מעגל שאינו מכיל קשתות אדומות ולכן את S חוצה קשת לבנה, e'. מכיוון של-e משקל מינימלי בחתך אז $T \setminus \{e'\} \cup \{e\}$ עפ"מ. □

הטענות הבאות מראות שניתן להפעיל את הכללים לפי הצורך:

טענה 21. ניתן להפעיל את הכלל הכחול כל עוד הקשתות הכחולות אינן עץ.

הוכחה. נניח שהקשתות הכחולות לא מהוות עץ, אז ביחס לקשתות הכחולות קיימים ב-G שני רכיבי קשירות (לפחות) כל רכיב קשירות כזה מהווה חתך מתאים. □

טענה 22. ניתן להפעיל את הכלל האדום כל עוד הקשתות **הלא** אדומות אינן עץ.

הוכחה. אם הקשתות הלא אדומות אינן עץ אז קיים לפחות מעגל אחד לא אדום. □

מסקנה 7. ניתן להפעיל את הכלל הכחול והאדום בסדר כלשהו כדי לקבל עפ"מ.

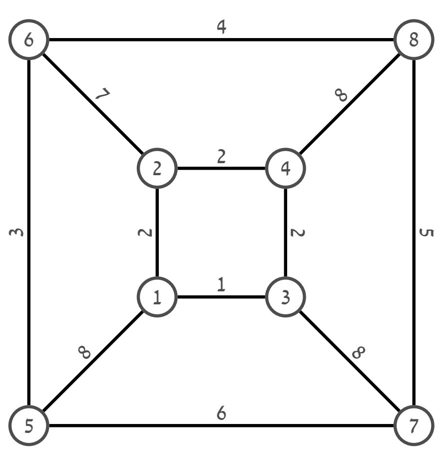
אלגוריתם פרים (Prim)

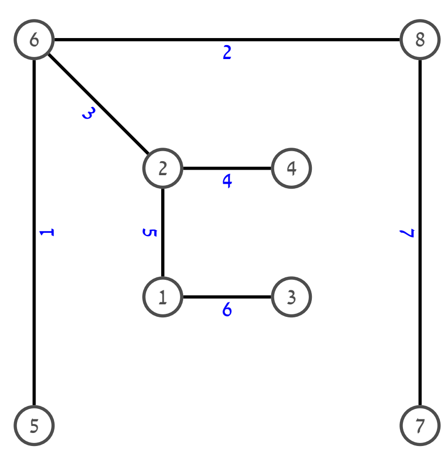
אלגוריתם פרים מתחיל מעץ שמכיל צומת אחד ובכל איטרציה מוסיף לעץ את הקשת הכי זולה שקצה אחד שלה בדיוק נוגע בעץ. פורמלית:

1. אתחול: $U \leftarrow \{u\}, B \leftarrow \emptyset$, כאשר u צומת שרירותי.
2. כל עוד $(U \neq V)$ הפעל את הכלל הכחול:
   (א) בחר את החתך U, וצבע בכחול את הקשת עם המשקל המינימלי שחוצה אותו, uv.
   (ב) $B \leftarrow B \cup \{uv\}$, $U \leftarrow U \cup \{v,u\}$

אבחנה 9. אלגוריתם פרים הוא מימוש של האלגוריתם הכללי שמפעיל את הכלל הכחול.

דוגמה:





הערות:

- כדי לממש את האלגוריתם צריך לדעת בכל שלב אילו קשתות חוצות את החתך ולבחור מהן את הקלה ביותר
- כאשר מוסיפים צומת לחתך יתכנו השינויים הבאים:
  - קשת שחצתה את החתך עכשיו היא פנימית לחתך
  - קשת חיצונית לחתך עכשיו חוצה את החתך
- השינויים היחידים הם עבור קשתות שנוגעות בצומת שהוסף לחתך ולכן בכל פעם שמוסיפים צומת לחתך מעדכנים רק את הקשתות שנוגעות בו, סך הכל מעדכנים כל קשת פעמיים לכל היותר.
- נשים לב שאם שומרים את הקשתות שחוצות את החתך בערימת מינימום אז אנחנו מבצעים $|E|$ הכנסות ו-$|V|$ הוצאות. הוצאה והכנסה של כל קשת לוקחת $O(\log|E|)$ לכל היותר.
- סך הכל זמן הריצה של האלגוריתם הוא $O(|E|\log|E|)=O(|E|\log|V|)$
- קיימים מבני נתונים יותר יעילים עם פונקציונליות של ערימת מינימום שתומכים בהכנסה בזמן ממוצע $O(1)$ והוצאה בזמן $O(\log|E|)$ ולכן ניתן לממש את האלגוריתם בזמן $O(|E|+|V|\log|V|)$

אלגוריתם קרוסקל (Kruskal)

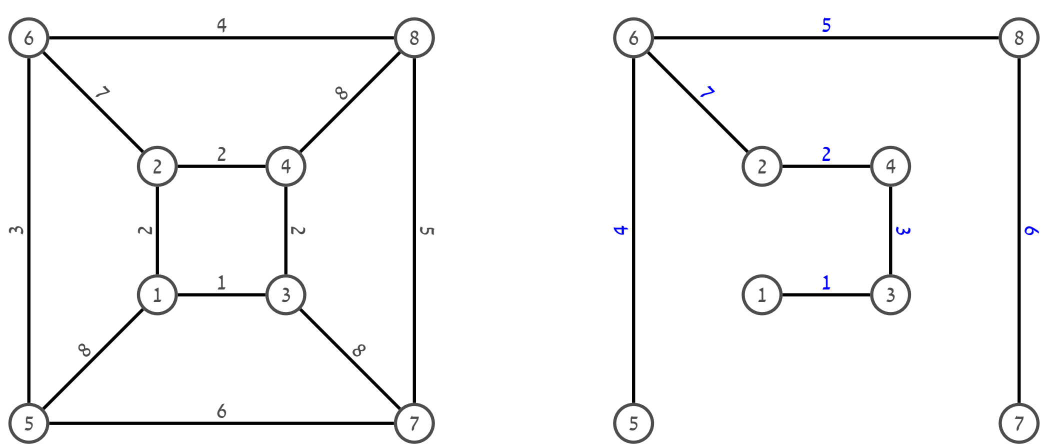
אלגוריתם קרוסקל מתחיל מיער ללא קשתות, בכל שלב באלגוריתם נמזג שני רכיבי קשירות באמצעות הקשת הקלה ביותר שמחברת שני רכיבי קשירות. פורמלית:

1. אתחול: $\mathcal{C}\leftarrow\{\{v\}:v\in V\}$,
2. כל עוד $T=(V,B)$ אינו קשיר הפעל את הכלל הכחול:

   (א) תהי e הקשת הקלה ביותר שמחברת שני רכיבי קשירות, C_i,C_j

   (ב) עדכן $B\leftarrow B\cup\{e\}$, $\mathcal{C}\leftarrow\mathcal{C}\setminus\{C_i,C_j\}\cup\{C_i\cup C_j\}$

דוגמה:



הערות:

- ניתן לממש את האלגוריתם באופן הבא:
  - מיין את הקשתות בסדר לא יורד של משקלן
  - עבור כל קשת לפי הסדר, אם היא מחברת שני רכיבי קשירות הפעל עליה את הכלל הכחול ועדכן את C
- זמן הריצה של מימוש כזה הוא $O(|E|\log|E|)$ עבור המיון ובנוסף לכל קשת צריך לבדוק אם היא מחברת שני רכיבים שונים ואם כן לעדכן את מבנה הרכיבים. כזכור מקורס מבני נתונים קיים מימוש פשוט שעושה זאת בזמן ממוצע של $O(\log|V|)$ ולכן הזמן הריצה הכולל הוא $O(|E|\log|E|)=O(|E|\log|V|)$.

משפט האפיון של עצים פורשים מינימלים

הגדרה 13. בהינתן עץ $T=(V,F)$ וקשת $e\notin F$ נסמן ב-C_e את המעגל שמכיל את e בגרף $(V,F\cup\{e\})$

הגדרה 14. בהינתן עץ $T=(V,F)$ וקשת $e=uv\in F$ נסמן ב-U_e את החתך שמכיל את כל הצמתים שישיגים מ-u בגרף $(V,F\setminus\{e\})$

משפט 2. עץ פורש $T=(V,F)$ של גרף $G=(V,E)$ הוא מינימלי אמ"מ לכל $e\in F$ מתקיים ש-e קשת במשקל מינימלי שחוצה את U_e. באופן שקול, T מינימלי אמ"מ לכל $e\in E\setminus F$ מתקיים ש-e קשת במשקל מקסימלי במעגל C_e

הוכחה. נוכיח כיוון אחד. נניח שמתקיים שלכל $e\in F$ מתקיים ש-e קשת במשקל מינימלי שחוצה את U_e. אז עץ כזה מתקבל על ידי הפעלה של הכלל הכחול על אוסף החתכים $\{U_e : e\in F\}$. באופן דומה, נניח שלכל $e\in E\setminus F$ מתקיים ש-e קשת במשקל מקסימלי במעגל C_e אז עץ כזה מתקבל על ידי הפעלת הכלל האדום על אוסף המעגלים $\{C_e : e\in E\setminus F\}$ $\square$

הרצאה 5

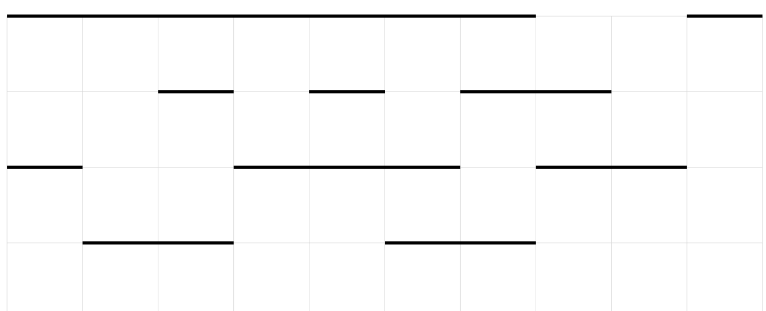
אלגוריתמים חמדניים - שיבוץ אינטרוולים, שיבוץ משימות

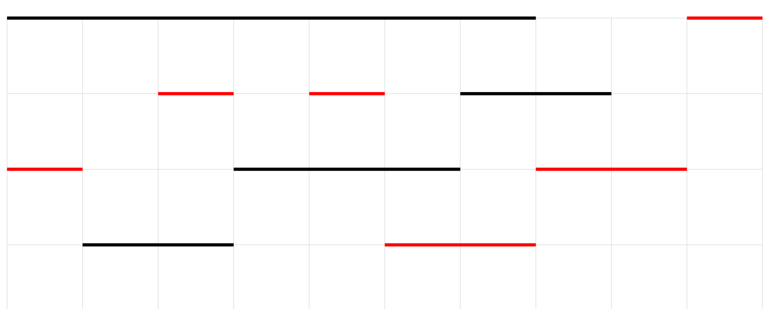
הקדמה

לעיתים קרובות אפשר לייצג בעיות אופטימזציה כקבוצה של אלמנטים כאשר פתרון חוקי הוא תת קבוצה של אלמנטים שמקיימת תכונות מסויימות. למשל, עץ פורש מינימלי. בדרך כלל יש פונקציית מחיר / רווח לכל תת קבוצה והמטרה שלנו היא למזער / למקסם את הערך הזה.
אלגוריתם חמדן, באופן לא פורמלי, הוא כזה שבונה פתרון (תת קבוצה של אלמנטים) באופן איטרטיבי ובכל שלב מוסיף / מסיר מהקבוצה

קבוצת אינטרוולים בלתי תלויה בגודל מקסימלי

נתונים n אינטרוולים $A = \{a_1, \ldots, a_n\}$, נסמן ב-$s(a_i)$ את זמן ההתחלה של האינטרוול a_i וב-$e(a_i)$ את זמן הסיום שלו. לכל אינטרוול מתקיים ש-$s(a_i), e(a_i) \in \mathbb{R}_+$ וכן $s(a_i) < e(a_i)$ רוצים למצוא תת קבוצה בגודל מקסימלי $I \subseteq A$ כך שהאינטרוולים ב-I זרים בזוגות, כלומר לכל $a, b \in I$, אחד התנאים מתקיים: $e(a) < s(b)$ או ש-$e(b) < s(a)$.
דוגמה:



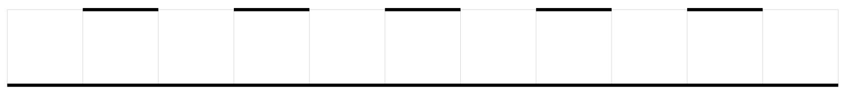


אלגוריתם חמדן:

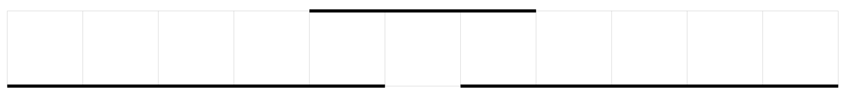
1. אתחול: $I \leftarrow \emptyset$, $\bar{e} \leftarrow 0$

2. עבור כל אינטרוול a בסדר לא יורד של ערכי $e(a)$:
   (א) אם $s(a) \geq \bar{e}$
      i. $I \leftarrow I \cup \{a\}$
      ii. $\bar{e} \leftarrow e(a)$

לפני שנוכיח נכונות נראה דוגמאות לגישות חמדניות שלא עובדות:
לבחור את האינטרוול עם זמן התחלה הכי מוקדם



לבחור את האינטרוול הכי קצר



לבחור את האינטרוול שנחתך עם הכי מעט אינטרוולים

הוכחת נכונות: נוכיח את הטענה הבאה, בכל צעד של האלגוריתם קיימת קבוצה בגודל מקסימלי, I' כך ש-I רישא שלה ביחס למיון ע"פ ערכי e.
בסיס: באתחול טריוויאלי
צעד: נבחן את הקבוצות I ו-I' בצעד ה-$i+1$. לפי הנחת האינדוקציה הקבוצות, ממוינות על פי ערכי e נראות כך:

$$\begin{aligned} I &= \{\alpha_1, \ldots, \alpha_i, \alpha_{i+1}\} \\ I' &= \{\alpha_1, \ldots, \alpha_i, \beta_1, \ldots, \beta_k\} \end{aligned}$$

נסתכל על הפתרון

$$I'' = \{\alpha_1, \ldots, \alpha_i, \boldsymbol{\alpha_{i+1}}, \ldots, \beta_k\}$$

מכיוון ש-I' פתרון חוקי האינטרוולים שם זרים בזוגות ולכן גם האינטרוולים ב-I'' למעט אולי α_{i+1}. מכיוון שהאלגוריתם בונה פתרון חוקי אז אנחנו יודעים ש-α_{i+1} זר ל-$\alpha_1, \ldots, \alpha_i$ ובגלל האופי החדמני של האלגוריתם מתקיים ש-$e(\alpha_{i+1}) \leq$ $e(\beta_1) \leq s(beta_2) \leq, \ldots, \leq s(\beta_k)$ ולכן I'' פתרון בגודל מקסימלי כך ש-I רישא שלו.

שיבוץ משימות

נתונות n משימות $A = \{a_1, \ldots, a_n\}$ נסמן ב-$t(a_i)$ את הזמן הנדרש לביצוע משימה a_i וב-$d(a_i)$ את זמן הסיום הרצוי של המשימה. בהינתן סדר ביצוע המשימות (פרמוטציה) $\pi : A \rightarrow [n]$. נסמן ב-$\delta(a_i)$ את זמן הסיום של המשימה a_i, כלומר

$$\delta(a_i) = \sum_{i \leq \pi(a_i)} t(\pi^{-1}(i))$$

נסמן ב-$l(a_i) := \delta(a_i) - d(a_i)$ את האיחור בביצוע משימה a_i. רוצים למצוא סדר שממזער את האיחור המקסימלי, כלומר

$$\arg\min_{\pi}\{\max_{i} l(a_i)\}$$

דוגמה: בהינתן שלוש המשימות הבאות:

A	t	d
a_1	2	7
a_2	3	10
a_3	5	5

שני שיבוצים אפשריים, אחד ללא איחור כלל והשני עם איחור של 5.

האלגוריתם החמדן יבצע את המשימות בסדר לא יורד של זמני הסיום הרצויים.

הוכחת נכונות

נוכיח באינדוקציה את הטענה הבאה:

לכל i קיים פתרון אופטימלי שמבצע את i המשימות הראשונות לפי זמני הסיום שלהן.

בסיס: טריוויאלי

צעד: נסתכל על המשימה, a, שזמן הסיום שלה הוא ה-$i+1$ לפי סדר לא יורד. אם הפתרון האופטימלי מבצע את המשימה הזאת בזמן $i+1$ סיימנו, אחרת הוא מבצע אותה בזמן $j > i+1$ נסתכל על סדר ביצוע המשימות מזמן $i+1$ עד זמן j:

$$b_{i+1}, \ldots, b_{j-1}, a$$

נבחן פתרון שמבצע את המשימות הללו בסדר הבא:

$$a, b_{i+1}, \ldots, b_{j-1}$$

נבדוק את האיחור המקסימלי של משימות אלו (האיחור המקסימלי של יתר המשימות לא השתנה) ונניח בשלילה שהוא גדל (אחרת סיימנו). אם זמן הסיום גדל זה חייב להיות בגלל אחת מהמשימות $b_{i+1}, \ldots, b_{j-1}$, נסמן אותה ב-b. נסמן את זמן הסיום שלה לפי הסדר החדש ב-$\delta'(b)$ אנחנו יודעים אבל ש-$\delta'(b) \leq \delta(a)$ וגם ש-$d(a) \leq d(b)$ ולכן $\delta'(b) - d(b) \leq \delta(a) - d(a)$.

הרצאה 6

קוד האפמן

הקדמה

רוצים לשמור קובץ טקסט על הדיסק בצורה חסכונית. אפשרות אחת היא לקודד כל תו בטקסט במספר סיביות קבוע. מספר הסיביות שנזדקק לכל תו הוא $\lceil \log |\Sigma| \rceil$. אפשרות נוספת היא לקודד כל תו במספר סיביות שונה. נשים לב שקידוד כזה יכול להיות חסכוני יותר כאשר יש שוני בין שכיחויות התווים בטקסט.

דוגמה:

עבור הא"ב $\{A, B, C, D\}$ והמחרוזת הבאה: AAABCD קידוד באורך קבוע יהיה באורך $6 \times 2 = 12$.

לעומת זאת, אם נבחר את הקידוד הבא

A	1
B	01
C	001
D	000

אז אורך הקידוד יהיה 11 בלבד.

הגדרה 15 (קוד בינרי)**.** בהינתן א"ב סופי Σ קידוד הוא פונקציה שממפה כל תו בא"ב למחרוזת בינרית $c : \Sigma \to \{0, 1\}*$

הגדרה 16 (הרחבה של קוד)**.** הרחבה של קוד היא פונקציה $c : \Sigma^* \to \{0, 1\}$ שמוגדרת להיות $c(t_1 \dots t_k) = c(t_1) \dots c(t_k)$

תכונות

נבחן שלושה קידודים שונים לא"ב $\{A, B, C, D\}$

$c_1 =$

A	1
B	01
C	001
D	000

$c_2 =$

A	0
B	01
C	011
D	111

$c_3 =$

A	1
B	01
C	011
D	111

באופן טבעי נדרוש שהקוד יהיה ניתן לפענוח (חד פענח), כלומר נרצה שההרחבה תהיה פונקציה חד חד ערכית.

דוגמה: ניתן לפענח את c_1, c_2-ו, אבל לא את c_3.

תכונה רצויה היא שנוכל לפענח כל תו ברגע שקראנו את המילה שמקודדת אותו (פענוח מידי).

דוגמה: התכונה מתקיימת עבור c_1, אבל לא מתקיימת עבור c_2, ו-c_3.

קודים חסרי רישות

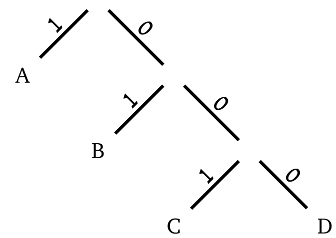
קוד c יקרא חסר רישות אם לא קיימים $a, b \in \Sigma$ כך ש-$c(a)$ רישא של $c(b)$

קל לראות שקודים חסרי רישות ניתנים לפענוח וכן לפענוח מידי. מעבר לכך המשפט הבא (ללא הוכחה) מראה שלמטרתנו מספיק להתמקד בקודים חסרי רישות.

משפט 3. לכל קוד חד פענח c קיים קוד חסר רישות c' כך שלכל $a \in \Sigma$ מתקיים $|c(a)| = |c'(a)|$.

קוד חסר רישות כעץ בינרי

ניתן לייצג כל קוד חסר רישות כעץ בינרי, למשל את הקוד c_1 ניתן לייצג על ידי העץ הבא:



נשים לב שבתיאור כזה ישנה התאמה חד חד ערכית בין עלי העץ למילות קוד.

קוד האפמן

נניח שנתון לנו קובץ טקסט מעל א"ב Σ, וכן נתונה לנו פונקציה שמתארת את מספר המופעים של כל תו בקובץ $f:\Sigma\to\mathbb{N}$ נרצה למצוא קוד חסר רישות (עץ בינרי) שיקודד את הקובץ במינימום סיביות, כלומר:

$$\min_c \sum_{a\in\Sigma} |c(a)|\cdot f(a)$$

במונחים של עצים נרצה למצוא עץ שממזער את הערך

$$\min_c \sum_{a\in\Sigma} d(a)\cdot f(a)$$

כאשר $d(a)$ הוא עומק העלה שמתאים לתו a בעץ.
לעץ שממזער את הערך הנ"ל נקרא עץ האפמן

טענה 23. כל עץ האפמן הוא עץ מלא (לכל צומת פנימי יש שני בנים)

הוכחה. נסתכל על עץ האפמן שממזער את מספר הצמתים הפנימיים עם בן אחד, נניח בשלילה שיש בן כזה אז אפשר להחליף צומת כזה עם הבן שלו ולהקטין את ערך העץ □

טענה 24. אם $a,b\in\Sigma$ שני איברים בעלי ערך f מינימלי, אז קיים עץ האפמן שבו a ו-b הם אחים ובעלי עומק מקסימלי

הוכחה. אם לא, נבחר שני עלים אחים בעלי עומק מקסימלי ונחליף אותם עם a ו-b. □

למה 2. אם $a,b\in\Sigma$ שני איברים בעלי ערך f מינימלי, נגדיר $\Sigma'=\Sigma\setminus\{a,b\}\cup\{z\}$ כאשר $z\notin\Sigma$. כמו כן נגדיר $f(z)=f(a)+f(b)$.
אם T' עץ האפמן של Σ' אז העץ T שמתקבל מ-T' על ידי החלפה של העלה z בצומת פנימי עם שני בנים a ו-b הוא עץ האפמן של Σ.

הוכחה. ניקח עץ האפמן $\hat{T}$ על Σ שבו a ו-b אחים. ממנו נייצר עץ $\hat{T}'$ על Σ' על ידי איחוד העלים a ו-b לעלה z. נראה שמתקיים

$$w(T)=w(T')+f(a)+f(b)\leq w(\hat{T}')+f(a)+f(b)=w(\hat{T})$$

□

אלגוריתם לבניית עץ האפמן

1. אם $|\Sigma|=2$ מחזירים עץ בינארי עם 3 צמתים
2. יהיו $a,b\in\Sigma$ שני האיברים עם ערכי f מינימליים
   (א) מגדירים $\Sigma'\leftarrow\Sigma\setminus\{a,b\}\cup\{z\}$
   (ב) קובעים $f(z)=f(a)+f(b)$
   (ג) קוראים לאלגוריתם באופן רקורסיבי על Σ' ומקבלים T' מוסיפים לעלה z ב-T' את הבנים a ו-b לקבלת T
   (ד) מחזירים T

טענה 25. האלגוריתם מחזיר עץ האפמן

הוכחה. באינדוקציה על גודל הא"ב ובעזרת למה 2 □

דוגמת הרצה: גנן גידל דגן בגן

הרצאה 7

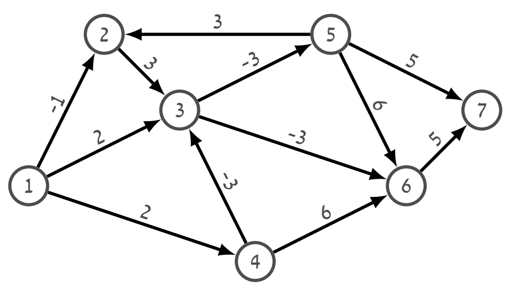
מסלולים קלים ביותר - אלגוריתם גנרי

הקדמה

נתון לנו גרף (מכוון או לא) $G=(V,E)$ וכן פונקציית משקל על הקשתות $w:E\to\mathbb{R}$. נסמן ב-$P_{st}=(s=v_0,\dots,v_k=t)$ מסלול מצומת s לצומת t וב-$\delta(s,t)$ את משקל המסלול הקל ביותר בין שני צמתים s ו-t. כלומר:

$$\delta(s,t)=\inf_{P_{st}} w(P_{st})$$

דוגמה: למה שווה $\delta(1,3)$ בגרף הבא ? למה שווה $\delta(1,7)$?



הערות:

- לאלגוריתמים למציאת מסלול קל ביותר שימושים רבים, אולי המידי שבהם הוא חישוב מסלול קצר ביותר בין שתי נקודות במפה.
- יתכנו משקלים שלילים על הקשתות, למשל אם אנחנו מעוניינים לתכנן מסלול לרכב חשמלי והמטרה שלנו היא לחסוך בסוללה.
- כאשר צומת t לא ישיג מצומת s נגדיר $\delta(s,t)=\infty$
- כאשר יש מעגל שלילי ישיג מצומת s, נגדיר $\delta(s,v)=-\infty$ לכל v שישיג מ-s (בדרך כלל במקרה כזה רק נרצה לזהות שזהו אכן המצב).

תכונות

טענה 26. אם אין בגרף מעגלים שלילים אז קיים מסלול פשוט קל ביותר

הוכחה. נסתכל על המסלול הקל ביותר עם הכי מעט מעגלים, נוריד מעגל אחד. $\square$

טענה 27. אם $p=(v_0,\dots,v_k)$ מסלול קל ביותר מ-v_0 ל-v_k אז לכל $0\le i\le j\le k$ מתקיים ש-$(v_i,\dots,v_j)$ מסלול קל ביותר בין v_i ל-v_j.

הוכחה. אם לא, נחליף את המסלול הקל ביותר בתת מסלול הקיים ונקבל מסלול קל יותר. $\square$

טענה 28 (אי שוויון המשולש)**.** לכל $u,v\in V, uv\in E$ מתקיים ש-$\delta(s,v)\le\delta(s,u)+w(uv)$

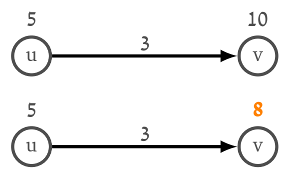
הוכחה. משקל המסלול הקל ביותר מ-s ל-u ומשם ל-v הוא $\delta(s,u)+w(uv)$ (אבל יתכנו מסלולים קלים יותר ממסלול זה). $\square$

מקור בודד

בהינתן גרף $G=(V,E)$ וצומת מקור s, נרצה לחשב את הערך $\delta(s,v)$ לכל $v\in V$.

הגדרה 17 (פונקציית חסם עליון). בהינתן גרף $G=(V,E)$, פונקציה $d:V\to\mathbb{R}$ תקרא פונקציית חסם עליון אם לכל $v\in V$ מתקיים ש-$d(v)\geq\delta(s,v)$

ניסיון שיפור: בהינתן גרף $G=(V,E)$ ופונקציית חסם עליון $d:V\to\mathbb{R}$ ניסיון שיפור של $d(v)$ לפי קשת uv מוגדר להיות $d(v)\leftarrow\min\{d(v),d(u)+w(uv)\}$
דוגמה:



טענה 29. אם d היא פונקציית חסם עליון לפני ניסיון שיפור אז d היא פונקציית חסם עליון אחרי ניסיון השיפור.

הוכחה. אם אחרי ניסיון השיפור מתקיים ש-$d(v)<\delta(v)$ אז מתקיים ש:

$$d(v)<\delta(s,v)\leq\delta(s,u)+w(uv)\leq d(u)+w(uv)=d(v)$$

□

הגדרה 18. (קשת משפרת) קשת uv תקרא משפרת אם $w(uv)<d(v)-d(u)$

אלגוריתם גנרי לחישוב ערך המסלול הקל ביותר ממקור בודד

1. אתחול: לכל $v\in V$ הצב $d(v)\leftarrow\infty$, הצב $d(s)\leftarrow 0$
2. כל עוד קיימת קשת משפרת uv
   (א) $d(v)\leftarrow d(u)+w(uv)$

טענה 30. אם האלגוריתם עוצר וצומת v ישיג מ-s אז $d(v)<\infty$

הוכחה. נניח בשלילה שלא ונסתכל על קשת uv במסלול מ-s ל-v כך ש-$d(u)<\infty$ ו-$d(v)=\infty$ - סתירה. □

טענה 31. אם קיים בגרף מעגל שלילי ישיג מ-s אז האלגוריתם לא עוצר.

הוכחה. נשים לב שקשת אינה משפרת אמ"מ $w(uv)\geq d(v)-d(u)$. נסתכל על מעגל שלילי $v_1,\ldots,v_k,v_1$ נשים לב ש:

$$0=d(v_1)-d(v_k)+\sum_{i=1}^{k-1}d(v_{i+1})-d(v_i)\leq w(v_1v_k)+\sum_{i=1}^{k-1}w(v_{i+1}v_i)<0$$

□

טענה 32. אם האלגוריתם עוצר אז $d(v)=\delta(v)$ לכל $v\in V$

הוכחה. נשים לב ש-d היא פונקציית חסם עליון והפעולה היחידה שהאלגוריתם מבצע היא ניסיון שיפור ולכן בסיום האלגוריתם d היא עדיין פונקציית חסם עליון, כלומר $d(v)\geq\delta(v)$ לכל $v\in V$.
נראה שמתקיים $d(v)\leq\delta(v)$. נניח בשלילה שקיים צומת ישיג מ-s, w, כך שהטענה לא מתקיימת עבורו. נסתכל על קשת uv במסלול קל ביותר מ-s ל-w כך ש-$d(u)=\delta(s,u)$ ו-$d(v)>\delta(s,v)$. מכיוון שזהו מסלול קל ביותר אז מתקיים ש-

$$w(uv)=\delta(v)-\delta(u)<d(v)-d(u)$$

ומכאן ש-uv קשת משפרת.

□

מציאת עץ המסלולים הקלים ביותר נעדכן את האלגוריתם כך שלכל צומת יהיה מצביע לצומת הקודם אליו במסלול הקל ביותר אליו מ-s.

1. אתחול: לכל $v\in V$ הצב $d(v)\leftarrow\infty,p(v)\leftarrow nil$, הצב $d(s)\leftarrow 0$

2. כל עוד קיימת קשת משפרת uv

(א) $d(v) \leftarrow d(u) + w(uv)$

(ב) $p(v) \leftarrow u$

נגדיר $V' = \{v : p(v) \neq nil\} \cup \{s\}$ ו-$E' = \{uv : p(v) = u\}$

טענה 33. בכל שלב בזמן ריצת האלגוריתם הגרף $T = (V', E')$ הוא עץ ומשקל המסלול מ-s ל-v שווה ל-$d(v)$ לכל $v \in V'$

הוכחה. באינדוקציה.
בסיס: נכון
צעד: אם ניסיון שיפור לפי uv סגר מעגל משקל המעגל הוא $d(u) - d(v) + w(uv) < d(u) - d(v) + d(v) - d(u) = 0$ קל לוודא שאם ניסיון שיפור לא סוגר מעגל אז הטענה עדיין מתקיימת $\square$

הרצאה 8

מסלולים קלים ביותר - בלמן פורד, דייקסטרה

אלגוריתם בלמן-פורד

1. אתחול: לכל $v \in V$ מציבים $d(v) \leftarrow \infty$, $p(v) \leftarrow nil$. מציבים $d(s) \leftarrow 0$
2. מבצעים $|V|-1$ פעמים:
   (א) לכל קשת $e \in E$ בצע ניסיון שיפור לפי e
3. אם עדיין יש קשתות משפרות קבע כי יש מעגל שלילי, אחרת החזר את d ו-p

זמן הריצה של האלגוריתם הוא $O(|V||E|)$.

טענה 34. אם אין מעגלים שליליים אז בסיום האלגוריתם $d(v) = \delta(s,v)$ לכל צומת $v \in V$.

הוכחה. באינדוקציה על עומק הצומת בעץ לפי p. $\square$

טענה 35. אם קיים מעגל שלילי האלגוריתם קובע שקיים כזה.

הוכחה. נובע מהגדרת האלגוריתם והטענות על האלגוריתם הגנרי. $\square$

משפט 4. אלגוריתם בלמן פורד פולט עץ מסלולים קלים ביותר אם בגרף אין מעגלים שליליים, אחרת הוא מודיע כי קיים כזה.

הוכחה. מיידי מטענות 35 ו-34. $\square$

אלגוריתם דייקסטרה

אלגוריתם דייקסטרה מניח שבגרף אין משקלים שליליים.

1. אתחול: לכל $v \in V$ מציבים $d(v) \leftarrow \infty$, $p(v) \leftarrow nil$. מציבים $d(s) \leftarrow 0$, וכן $Q \leftarrow V$
2. כל עוד Q לא ריק
   (א) יהי $u \in Q$ צומת עם ערך d מינימלי
   (ב) הוצא את u מ-Q ולכל $uv \in E$ בצע ניסיון שיפור לפי uv

אם ממשים את Q על ידי ערימת מינימום אז זמן הריצה של האלגוריתם הוא $O(|V| + |E|\log|V|)$.

טענה 36. ערכי d של הצמתים לפי סדר הוצאתם מ-Q הם פונקציה מונוטונית לא יורדת.

הוכחה. באינדוקציה על צעד האלגוריתם + שימוש בנתון שלא קיימים משקלים שליליים. $\square$

מסקנה 8. ברגע שצומת יצא מ-Q, ערך d שלו לא משתנה.

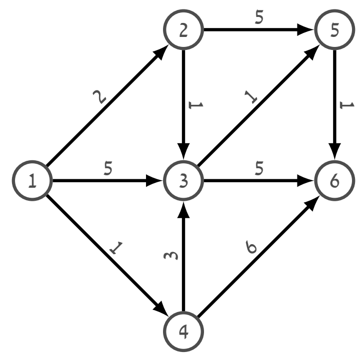
טענה 37. בסיום ריצת האלגוריתם אין בגרף קשתות משפרות.

הוכחה. הוכחה באינדוקציה על צעד האלגוריתם שאין קשתות משפרות בין צמתים מחוץ ל-Q. $\square$

משפט 5. אלגוריתם דייקסטרה פולט את עץ המסלולים הקלים ביותר.

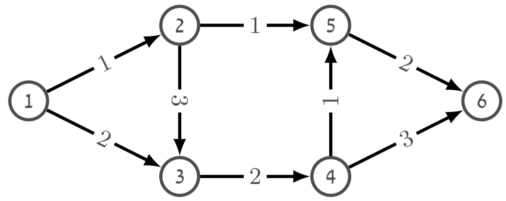
הוכחה. לפי טענה 37 והטענות על האלגוריתם הגנרי. $\square$

דוגמה

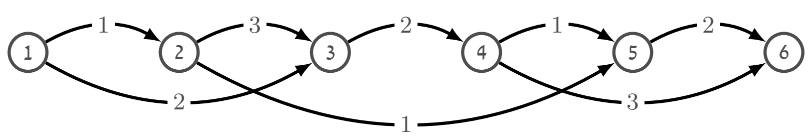


מסלולים קלים ביותר בגרף חסר מעגלים

כאשר הגרף חסר מעגלים ניתן למצוא את משקל המסלול הקל ביותר על ידי נוסחה רקורסיבית פשוטה.
דוגמה: נתון גרף חסר מעגלים.



ונתון מיון טופולוגי שלו:



טענה 38.

$$\delta(j) = \min_{ij \in E} \delta(i) + w(ij)$$

הוכחה. באינדוקציה על j. $\square$

נשים לב שאם מחשבים את הערך של δ לפי סדר המיון הטופולוגי אז סיבוכיות החישובי היא $O(|E| + |V|)$.
הערה: ניתן גם לשחזר את המסלולים הקלים ביותר על ידי שמירת מצביע לאבא של כל צומת.
שאלה: מדוע אי אפשר להשתמש באותה טכניקה גם עבור גרפים שמכילים מעגלים?

הרצאה 9

תכנון דינאמי - שיבוץ אינטרוולים, מסלולים קלים ביותר

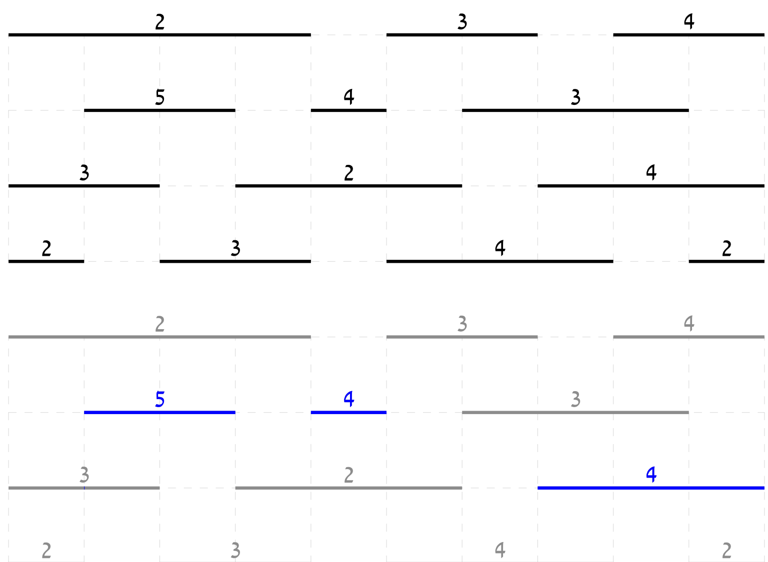
קבוצה בלתי תלויה של אינטרוולים עם משקל מקסימלי

נתונים n אינטרוולים (נניח שכבר אחרי מיון לפי זמן סיום) $A = (a_1, \ldots, a_n)$ לכל אינטרוול זמן התחלה $s(a_i)$, זמן סיום $e(a_i) > s(a_i)$ ומשקל $w(a_i)$. תת קבוצה $I \subseteq A$ של אינטרוולים נקראת בלתי תלויה אם לכל $a_i, a_j \in I$ אחד מהשניים מתקיים:

1. $s(a_j) > e(a_i)$
2. $s(a_i) > e(a_j)$

רוצים למצוא קבוצה בלתי תלויה של אינטרוולים עם משקל מקסימלי.

דוגמה: קלט לבעיה וקבוצה בלתי תלויה במשקל 13.



נסמן $A_i = (a_1, \ldots, a_i)$. נגדיר

$$p(i) = \max \begin{cases} \max\{j : e(a_j) < s(a_i)\} \\ 0 \end{cases}$$

כלומר $j = p(i)$ הוא האינדקס המקסימלי כך ש-a_j מסתיים לפני ש-a_i מתחיל או 0 אם לא קיים כזה.
נגדיר את $\alpha(i)$ להיות משקל פתרון אופטימלי עבור A_i אז $\alpha(n)$ הוא הערך אותו אנחנו מחפשים.

טענה 39.

$$\alpha(i) = \max \begin{cases} w(i) + \alpha(p(i)) \\ 0 + \alpha(i-1) \end{cases}$$

כמו כן מתקיים ש:

$$\alpha(0) = 0$$

הוכחה. באינדוקציה על i.

בסיס: עבור $i=0$ טריוויאלי.

עבור i כלשהו נקבע פתרון אופטימלי OPT ונסמן $OPT_i = OPT \cap A_i$.

אם $a_{i+1} \in OPT$ אז OPT לא יכול להכיל אף אינטרוול $a_{p(i)+1}, \ldots, a_{i+1}$ לפי הנחת האינדוקציה

$$\alpha(p(i)+1) \geq w(OPT_{p(i)})$$

ולכן הטענה מתקיימת כי

$$\alpha(i) \geq \alpha(p(i)) + w(a_i) \geq w(OPT_{p(i)}) + w(a_i)$$

מצד שני, אם $a_{i+1} \notin OPT$ אז לפי ההנחה

$$\alpha(i-1) \geq OPT_{i-1}$$

והטענה מתקיימת. □

חישוב יעיל של O

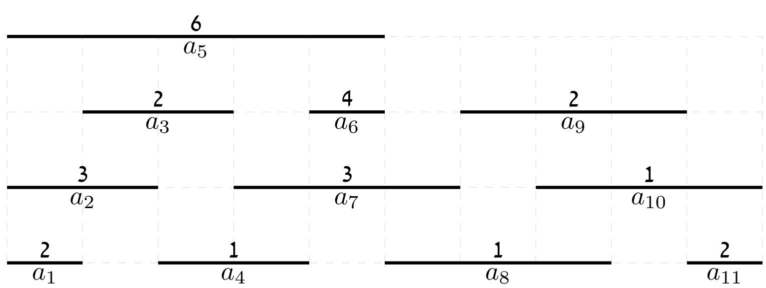
כיצד נחשב את O ביעילות ? נשים לב שאם מחשבים את ערכי O מ-1 עד n ושומרים את הערכים (למשל במערך) אז חישוב של כל ערך לוקח $O(1)$ זמן.

זמן הריצה של האלגוריתם:

1. מיון - $O(n \log n)$
2. חישוב p - $O(n \log n)$ (חיפוש בינארי לכל i)
3. חישוב O - $O(n)$

סך הכל $O(n \log n)$

דוגמת הרצה:



נחשב:

i	0	1	2	3	4	5	6	7	8	9	10	11
p	0	0	0	1	2	0	4	3	6	7	7	9
α	0	2	3	4	4	6	8	8	9	10	10	12

נמצא את הקבוצה עצמה:

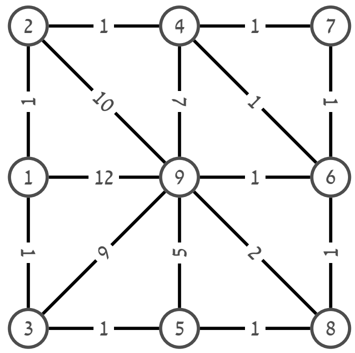
i	0	1	2	3	4	5	6	7	8	9	10	11
p	0	0	0	1	2	0	4	3	6	7	7	9
α	0	2	3	4	4	6	8	8	9	10	10	12

נקודות חשובות:

- מה יקרה אם במקום לחשב את ערכי α בסדר עולה ושמירת הערכים במערך נחשב את ערכי α באופן רקורסיבי על המחסנית ?
- מה יקרה אם לכל תא במערך נזכור גם את הקבוצה שמתאימה לערך התא ?

מסלולים קלים ביותר בין כל הזוגות

בהינתן גרף (מכוון או לא) עם n צמתים נרצה להדפיס טבלה בגודל $n \times n$ שבכניסה ה-ij שלה נמצא ערך מסלול קל ביותר מצומת i לצומת j.
ניתן, כמובן, לעשות זאת על ידי n הרצות של אלגוריתם בלמן פורד או דייקסטרה ולמצוא את התשובה בסיבוכיות זמן של $O(n^2m)$ ו- $O(nm\log n)$ בהתאמה. נראה שאפשר גם יותר טוב.
בהינתן גרף שצמתיו ממוספרים מ-1 עד n נגדיר את d_{ij}^k להיות משקל מסלול קל ביותר מצומת i לצומת j שיכול לעבור רק בצמתי ביניים עם אינדקסים ב-$[k]$.
דוגמה:



למה שווים הערכים הבאים d_{19}^0, d_{19}^1, d_{19}^2, d_{19}^6, d_{19}^7, d_{19}^9?

אבחנה 10. משקל מסלול קל ביותר מצומת i לצומת j שווה ל-d_{ij}^n.

אבחנה 11. $d_{ij}^0 = w(ij)$

נניח עכשיו שלכל i, j ו-k אנחנו מקבעים מסלול קל ביותר עבור d_{ij}^k.

אבחנה 12. אם הצומת k לא שייך למסלול שמתאים ל-d_{ij}^k אז $d_{ij}^k = d_{ij}^{k-1}$

אבחנה 13. אם הצומת k שייך למסלול שמתאים ל-d_{ij}^k אז $d_{ij}^k = d_{ik}^{k-1} + d_{kj}^{k-1}$.

מסקנה 9.

$$d_{ij}^k = \begin{cases} w(ij) & k = 0 \\ \min\{d_{ij}^{k-1}, d_{ik}^{k-1} + d_{kj}^{k-1}\} & k > 0 \end{cases}$$

חישוב: נגדיר $n+1$ מטריצות בגודל $n \times n$, $A^0, \ldots, A^n$, כאשר $A_{ij}^k = d_{ij}^k$. נמלא את ערכי המטריצות מ-A^0 ועד A^n כאשר נשים לב כי בשביל למלא את מטריצה A^k אנו צריכים לדעת אך ורק את ערכי המטריצה A^{k-1}.
סיבוכיות: אנחנו מחשבים n^3 ערכים וחישוב של ערך בודד לוקח $O(1)$ פעולות.

מסלולים קלים ביותר

כעת נפתור את בעיית המסלול הקל ביותר.
תזכורת: בהינתן גרף (מכוון או לא) $G = (E, V)$, פונקציית משקל $w : E \to \mathcal{R}$, צומת מקור $s \in V$, וצומת יעד $t \in V$ רוצים למצוא מסלול מ-s ל-t במשקל מינימלי.

ניסיון ראשון

נגדיר את $a(v)$ להיות המסלול הקל ביותר מ-s ל-v, אז מתקיים ש:

$$a(v) = \min_{uv \in E} a(u) + w(uv))$$

מה הבעיה ?

ניסיון שני

נגדיר את $a(v, U)$ להיות המסלול הקל ביותר מ-s ל-v בגרף $G[U]$, אז מתקיים ש:

$$a(v, U) = \min_{uv \in E} a(u, U \setminus \{v\}) + w(uv)$$

מה הבעיה ?

פתרון

נגדיר את $a(v,k)$ להיות מסלול קל ביותר מ-s ל-v עם k קשתות לכל היותר, ונחשב:

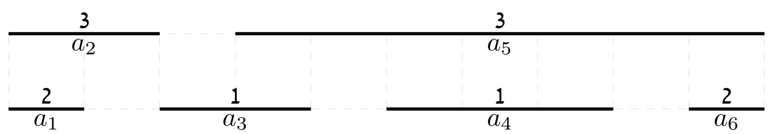
$$\begin{aligned}
&\forall\, v \neq s, 1 \leq k \leq n-1 && a(v,k) = \min_{uv \in E} a(u,k-1) + w(uv) \\
&\forall\, u \neq s && a(u,0) = \infty \\
& && a(s,0) = 0
\end{aligned}$$

טענה 40. אם ב-G אין מעגלים שליליים אז לכל v, $a(v,n-1)$ הוא משקל המסלול הקל ביותר מ-s ל-v.

הוכחת נכונות: כתרגיל.

גרף החישוב

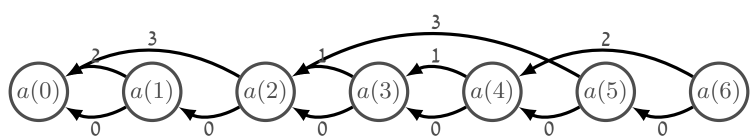
בהינתן נוסחת נסיגה, f, נסתכל על גרף החישוב שלה, G_f זהו גרף מכוון שבו כל צומת מתאימה למצב (ערך פרמטרים מסוים לנוסחה) וקיימת קשת ממצב s_i למצב s_j אמ"מ לצורך חישוב מצב s_i יש צורך לחשב את מצב s_j.
למשל עבור הקלט הבא לבעיית האינטרוולים:



ונוסחת הנסיגה

$$\alpha(i) = \max \begin{cases} w(i) + \alpha(p(i)) \\ 0 + \alpha(i-1) \end{cases}$$

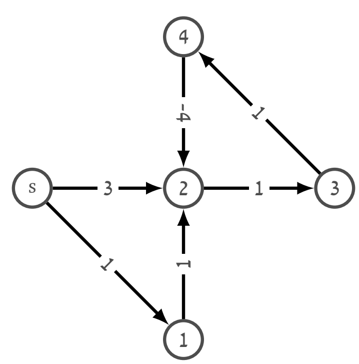
גרף החישוב יראה כך (ניתן אף לשים משקלים מתאימים על הקשתות):

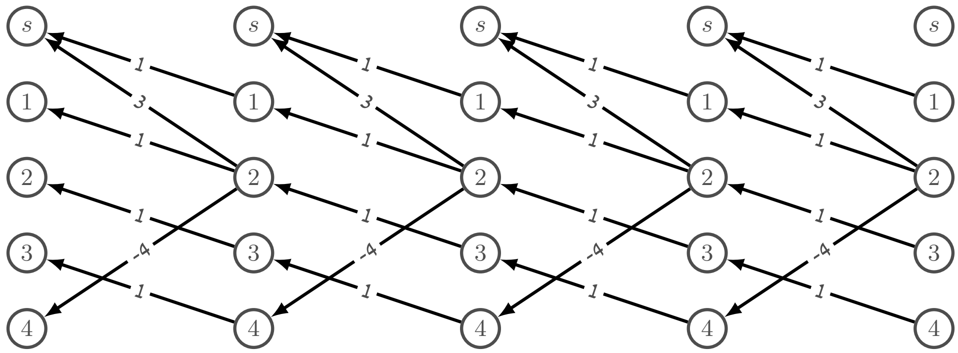


מה נדרוש מגרף החישוב ?

1. חסר מעגלים
2. לא גדול מדי
3. ניתן לחשב את הערכים של הבורות

דוגמה נוספת, כיצד יראה גרף החישוב עבור נוסחת הנסיגה של מסלולים קלים ביותר והקלט הבא:





1.9 מימוש

נסתכל על שלוש פונקציות שמחשבות את מספר פיבונצ'י ה-i. מה הסיבוכיות של כל פונקציה?

```
function fib(i: number): number {
    if (i <= 1) return 1
    return fib(i - 1) + fib(i - 2)
}

function dp(i: number): number {
    const fib = [1, 1]
    for (let j = 2; j <= i; j++)
        fib[j] = fib[j - 1] + fib[j - 2]
    return fib[i]
}

const cache = [1, 1]
function dp2(i: number): number {
    if (cache[i]) return cache[i]
    const n = dp2(i - 1) + dp2(i - 2)
    cache[i] = n
    return n
}

console.time('no dp')
console.log(fib(40))
console.timeEnd('no dp')
// no dp: 1501.243ms

console.time('dp')
console.log(dp(40))
console.timeEnd('dp')
// dp: 0.129ms

console.time('dp2')
console.log(dp2(40))
console.timeEnd('dp2')
// dp2: 0.112ms
```

הרצאה 10

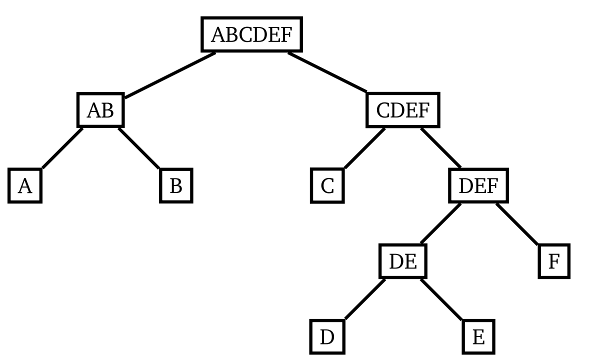
תכנון דינאמי - כפל מטריצות, התאמת מחרוזות

אופטימיזציה של כפל מטריצות

תזכורת: כפל נאיבי של מטריצה בגודל $a \times b$ עם מטריצה בגודל $b \times c$ לוקח $O(a \cdot b \cdot c)$ פעולות. התוצאה של המכפלה היא מטריצה מגודל $a \times c$.
כאשר כופלים n מטריצות, $A_1, \ldots, A_n$ מגדלים $x_i \times y_i$ בהתאמה, אז תוצאת המכפלה תהיה מטריצה מגודל $x_1 \times y_n$. מספר הפעולות שיש לבצע תלוי בסדר בו נבחר לבצע את המכפלה.
דוגמה: כמה פעולות נבצע כדי לבצע את המכפלה ABC ?

$$\begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{100} \end{pmatrix} \begin{pmatrix} b_1 & b_2 & \ldots & b_{100} \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{100} \end{pmatrix}$$

אם נבצע את המכפלה לפי הסדר משמאל לימין אז נזדקק ל-$10,000 = 100 \cdot 1 \cdot 100$ פעולות עבור הכפל של AB ו-$10,000 = 100 \cdot 1 \cdot 100$ פעולות נוספות עבור הכפל של $(AB)C$. אם נחשב את המכפלה $A(BC)$ אז נזדקק לסדר גודל של 200 פעולות בלבד !!!
בעיה: בהינתן n מטריצות, $A_1, \ldots, A_n$ מגדלים $x_i \times y_i$ בהתאמה, רוצים לחשב סדר מכפלות שדורש מינימום פעולות.
ייצוג סדר מכפלות ייצוג טבעי לסדר הפעולות הוא בעזרת עץ, למשל העץ הבא מתאים לחישוב $(AB)(C((DE)F))$:



נתייחס לעץ כזה כעץ ביטוי, עץ ביטוי הוא עץ בינרי מלא שבו העלים הם המטריצות מהקלט וכל צומת מייצג מכפלה של המטריצות המתאימות לעלים של תת העץ שלו.
אלגוריתם: עבור כל $1 \leq i \leq j \leq n$ נגדיר את $\alpha(i,j)$ להיות מספר הפעולות המינימלי שצריך כדי לבצע את המכפלה $A_i \cdot A_{i+1} \cdot \ldots \cdot A_j$
אז מתקיים ש:

$$\alpha(i,j) = \min_{i \leq k < j} \alpha(i,k) + \alpha(k+1,j) + x_i \cdot y_k \cdot y_j$$

בנוסף מתקיים ש:

$$\forall\ 1 \leq i \leq n\ \alpha(i,i) = 0$$

סיבוכיות: אם מחשבים את ערכי נוסחת הנסיגה על ידי שימוש בטבלה, למשל, אז נדרש לחשב $O(n^2)$ ערכים. זמן החישוב של כל ערך הוא $O(n)$ ולכן בסך הכל זמן ריצת האלגוריתם הוא: $O(n^3)$.

דוגמת הרצה:

$$A_1^{9\times 2}A_2^{2\times 10}A_3^{10\times 4}A_4^{4\times 3}$$

התאמת מחרוזות

רוצים לבצע תיקון של שגיאות איות, למשל:

סבון
סוויבון
סביון
סביבון

כדי לדעת אילו תיקונים להציע רוצים למדוד את המרחק בין המחרוזת שהוקלדה לבין המילה המוצעת.
בהינתן א"ב Σ נגדיר $\Sigma' = \Sigma \cup \{\_\}$. ונגדיר:

הגדרה 19 (הרחבה)**.** מחרוזת $s' \in \Sigma'^*$ היא הרחבה של $s \in \Sigma^*$ אם לאחר מחיקת כל תווי ה-_ מ-s' מקבלים את s.

בהינתן פונקציית משקל $w : \Sigma' \times \Sigma' \to \mathcal{R}$ המרחק בין שתי הרחבות בעלות אורך זהה, l, הוא:

$$\sum_{i=1}^{l} w(s'_1[i], s'_2[i])$$

דוגמה 5.

ס	ו	ו	י	ב	ו	ו
ס	ב	_	י	ב	ו	ו

דוגמה 6.

ס	ו	ו	י	ב	_	ו	ו
ס	_	_	_	ב	י	ו	ו

עבור פונקציית המשקל

$$w(\alpha,\beta) = \begin{cases} 0 & \text{if } \alpha = \beta \\ 1 & \text{otherwise} \end{cases}$$

אז המרחק בדוגמה 5 הוא 2 ובדוגמה 6 הוא 4.

הגדרה 20 (מרחק)**.** המרחק בין שתי מחרוזות (לאו דווקא באורך זהה) מעל Σ הוא המרחק המינימלי האפשרי בין כל שתי הרחבות שלהן מאורך זהה.

הערה: אם מניחים שפונקציית המשקל אי שלילית אז מספר ההרחבות הרלוונטיות הוא סופי.
מטרה: בהינתן שתי מחרוזות רוצים לחשב את המרחק ביניהן.
פתרון: נסמן $|s| = m$, $|r| = n$ ו-$\alpha(i,j)$ להיות המרחק בין $s[i \dots m-1]$ ל-$r[j \dots n-1]$.
ונחשב

$$\alpha(i,j) = \min \begin{cases} w(s[i], r[j]) + \alpha(i+1, j+1]) \\ w(\_, r[j]) + \alpha(i, j+1) \\ w(s[i], \_) + \alpha(i+1, j) \end{cases}$$

כמו כן מתקיים ש:

$$\alpha(m,n) = 0$$
$$\alpha(m,k) = w(\_, r[k]) + \alpha(m, k+1]) \quad \forall\, 0 \le k < n$$
$$\alpha(k,n) = w(s[k], \_) + \alpha(k+1, n]) \quad \forall\, 0 \le k < m$$

סיבוכיות: אם מחשבים את ערכי נוסחת הנסיגה על ידי שימוש בטבלה, למשל, אז נדרש לחשב mn ערכים וחישוב של כל ערך לוקח $O(1)$ פעולות. בסך הכל מקבלים $O(mn)$ פעולות.

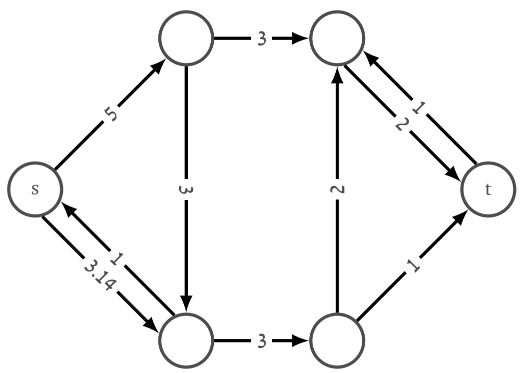
הרצאה 11

רשתות זרימה - אלגוריתם פורד פלקרסון

הקדמה

הגדרה 21 (רשת זרימה)**.** רשת זרימה היא גרף מכוון, $G=(V,E)$ עם קיבולים על הקשתות, $c:E\to\mathbb{R}_+$ צומת מקור, $s\in V$ וצומת בור, $t\in V$.

דוגמה:



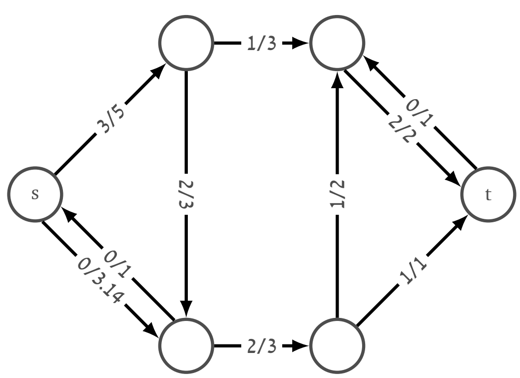
נסמן ב-$\delta(u):=\{uv:uv\in E\}$ את אוסף הקשתות שיוצאות מצומת u וב-$\rho(v):=\{uv:uv\in E\}$ את אוסף הקשתות שנכנסות לצומת v.

בהינתן פונקציה, $f:E\to\mathbb{R}_+$ נגדיר: $\forall v\in V\ f(v):=\sum_{e\in\delta(v)}f(e)-\sum_{e\in\rho(v)}f(e)$

הגדרה 22 (זרימה)**.** בהינתן רשת זרימה, (G,s,t,c) זרימה היא פונקציה, $f:E\to\mathbb{R}_+$ אשר מקיימת

1. **חוק הקשת** $\forall\ e\in E\quad 0\le f(e)\le c(e)$
2. **חוק הצומת** $\forall\ v\in V\setminus s,t\quad f(v)=0$

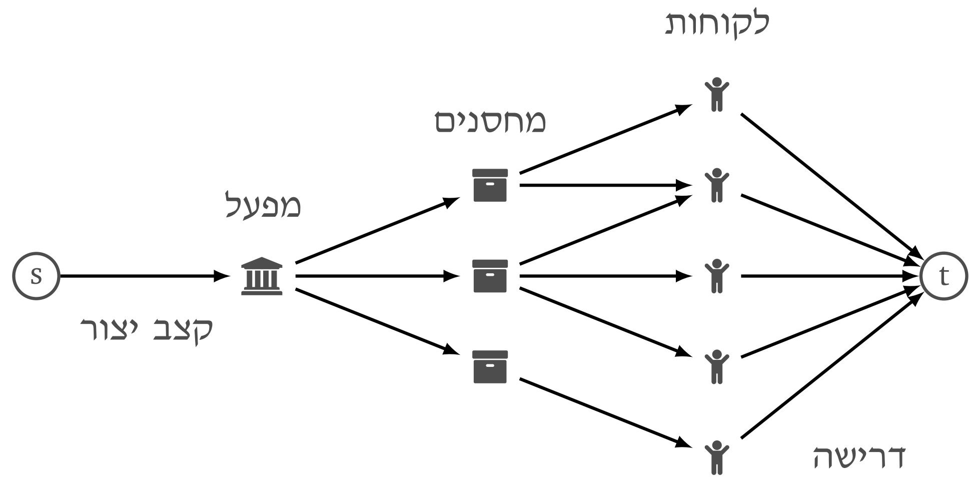
דוגמה:



נסמן ב-$|f|:=f(s)$ את ערך הזרימה. בדוגמה הקודמת מתקיים ש-$|f|=3$

מטרה: למצוא פונקציית זרימה חוקית עם ערך מקסימלי.

לבעיית זרימת המקסימום יישומים רבים בבעיות אופטימיזציה, למשל, רוצים לדעת איך להפיץ סחורה מהמפעל למחסנים ומשם ללקוחות.



חתך-st

נרחיב את הסימונים δ, ρ, f ו-c עבור קבוצת צמתים, כלומר

$$\delta(S) := \{uv \in E : u \in S \wedge v \notin S\}$$

$$\rho(S) := \{uv \in E : u \notin S \wedge v \in S\}$$

$$f(S) := \sum_{e\in\delta(S)} f(e) - \sum_{e\in\rho(S)} f(e)$$

$$c(S) := \sum_{e\in\delta(S)} c(e)$$

ונשים לב שלכל $S \subseteq V$ מתקיים

אבחנה 14. $f(S) = \sum_{v\in S} f(v)$

הגדרה 23 (חתך-st)**.** חתך-st הוא תת קבוצה של צמתים שמכילה את s ואינה מכילה את t.

למה 3. לכל חתך-st, S, מתקיים $|f| = f(S)$.

הוכחה. נשים לב שלכל צומת, $v \in S$, שאינו s מתקיים $f(v) = 0$ ובנוסף, לפי ההגדרה, מתקיים ש-$|f| = f(s)$. □

בפרט מתקיים ש:

$$|f| = f(V \setminus \{t\}) = -f(t)$$

טענה 41. לכל חתך-st, S, מתקיים $|f| \leq c(S)$

הוכחה. לפי למה 3 ולפי הגדרת פונקציית זרימה מתקיים ש-$|f| = f(S) \leq c(S)$. □

הטענה האחרונה מאפשרת לנו למצוא חסם עליון על זרימת המקסימום. בהמשך נראה כי זהו חסם עליון הדוק.

רשת שיורית

הנחה: נניח בלי הגבלת הכלליות (למה?) שברשת הזרימה (המקורית) אין קשתות אנטי-מקביליות.

הגדרה 24 (רשת שיורית)**.** בהינתן רשת זרימה, $N = (G, s, t, c)$ וזרימה, f, הרשת השיורית היא $N_f = (G_f, s, t, c_f)$ כאשר אם $G = (V, E)$ אז

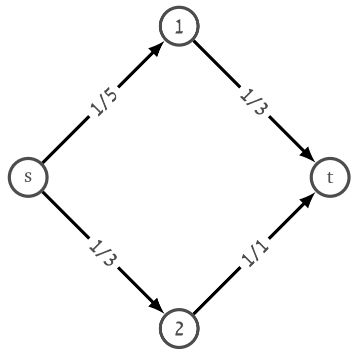
$$G_f = (V, E_f = E \cup \overline{E})$$

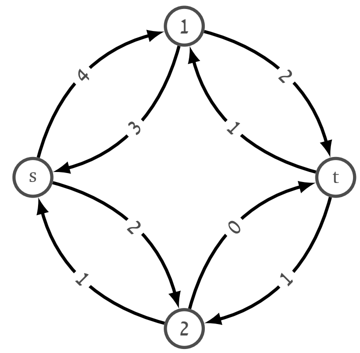
$$\overline{E} = \{\overline{e} = vu : e = uv \in E\}$$

$$c_f(\overline{e}) := f(e)$$

$$c_f(e) := c(e) - f(e)$$

דוגמה:





נשים לב שאם g זרימה ברשת שיורית N_f אז מתקיים:

$$
\begin{aligned}
g(v) = & \\
& \sum_{vu\in E_f} g(vu) - \sum_{uv\in E_f} g(uv) = \\
& \sum_{vu\in E} g(vu) + \sum_{uv\in E} g(vu) - \sum_{uv\in E} g(uv) - \sum_{vu\in E} g(uv) = \\
& \sum_{vu\in E} (g(vu) - g(uv)) - \sum_{uv\in E} (g(uv) - g(vu))
\end{aligned}
$$

הגדרה 25 (חיבור זרימות)**.** אם f זרימה ב- N ו-g זרימה ב- N_f נגדיר את הסכום שלהן להיות:

$$\forall e \in E: \; h(e) = f(e) + g(e) - g(\overline{e})$$

למה 4. h היא פונקציית זרימה ב-N ומתקיים $|h| = |f| + |g|$.

הוכחה.
חוק הקשת:

$$c(e) \geq f(e) + c(e) - f(e) - g(\overline{e}) \geq f(e) + g(e) - g(\overline{e}) \geq f(e) + g(e) - f(e) \geq 0$$

חוק הצומת:

$$
\begin{aligned}
h(v) = \sum_{vu\in E} h(vu) - \sum_{uv\in E} h(uv) = & \\
& \sum_{vu\in E} f(vu) + g(vu) - g(uv) - \sum_{uv\in E} f(uv) + g(uv) - g(vu) = \\
& \sum_{vu\in E} f(vu) - \sum_{uv\in E} f(uv) + \\
& \sum_{vu\in E} (g(vu) - g(uv)) - \sum_{uv\in E} (g(uv) - g(vu)) = \\
& 0 + 0
\end{aligned}
$$

ערך הזרימה:

$$
\begin{aligned}
h(s) = \sum_{su\in E} h(su) - \sum_{us\in E} h(us) = & \\
& \sum_{su\in E} f(su) + g(su) - g(us) - \sum_{us\in E} f(us) + g(us) - g(su) = \\
& \sum_{su\in E} f(su) - \sum_{us\in E} f(us) + \\
& \sum_{su\in E} (g(su) - g(us)) - \sum_{us\in E} (g(us) - g(su)) = \\
& f(s) + g(s)
\end{aligned}
$$

□

למה 5. אם P מסלול (פשוט) מ-s ל-t ברשת זרימה (G,s,t,c) ו-ε הקיבול המינימלי של קשת ב-P אז הפונקציה:

$$f_P(e) = \begin{cases} \varepsilon & \text{if } e \in P \\ 0 & \text{otherwise} \end{cases}$$

היא פונקציית זרימה.

הוכחה. חוק הקשת מתקיים לפי ההגדרה של f ושל ε. עבור צומת פנימי במסלול, v, מתקיים ש-$f(v) = f(vw) - f(uv) = 0$ כאשר u ו-w הצמתים לפני ואחרי v במסלול בהתאמה. □

נסמן ב- $G_f^+ = (V, E_f^+)$ כאשר $E_f^+ = \{e \in E_f : c(e) > 0\}$.

הגדרה 26 (מסלול שיפור)**.** בהינתן רשת זרימה, (G,s,t,c), וזרימה, f, מסלול שיפור הוא מסלול (פשוט) מ-s ל-t ב-G_f^+.

למה 6. אם P הוא מסלול שיפור ביחס לרשת זרימה (G,s,t,c) וזרימה f אז $h = f + f_P$ זרימה חוקית ו-$|h| = |f| + \varepsilon$.

הוכחה. נובע ישירות מלמות 4 ו-5. □

חתך מינימום זרימת מקסימום

המשפט המרכזי על רשתות זרימה קובע ש:

משפט 6. תהי f פונקציית זרימה ברשת (G,s,t,c). התנאים הבאים שקולים:

1. f היא זרימת מקסימום.
2. לא קיים מסלול שיפור (ב-G_f^+).
3. קיים חתך-st ברשת שהקיבול שלו שווה ל-$|f|$.

הוכחה.
$\mathbf{1 \Rightarrow 2}$
נניח בשלילה שקיים ונקבל סתירה לפי למה 6.
$\mathbf{2 \Rightarrow 3}$
נסמן ב-S את קבוצת הצמתים הישיגים מ-s ב-G_f^+. מלמה 3 נובע כי $f(S) = |f|$ ולפי ההגדרה של רשת שיורית נובע כי ב-G מתקיים:

$$\forall e \in \delta(S) \quad f(e) = c(e)$$
$$\forall e \in \rho(S) \quad f(e) = 0$$

$\mathbf{3 \Rightarrow 1}$
מיידי מטענה 41. □

אלגוריתם פורד פלקרסון

אלגוריתם גנרי למציאת זרימת מקסימום:

1. אתחול: מציבים $f(e) \leftarrow 0$ לכל $e \in E$
2. כל עוד יש מסלול שיפור ברשת השיורית (G_f, s, t, c_f)
   (א) מציבים ב-f את הזרימה המשופרת לפי למת שיפור הזרימה
3. פולטים את f

טענה 42. אם וכאשר האלגוריתם עוצר, פלט האלגוריתם הוא זרימת מקסימום.

הוכחה. מיידית ממשפט 6. □

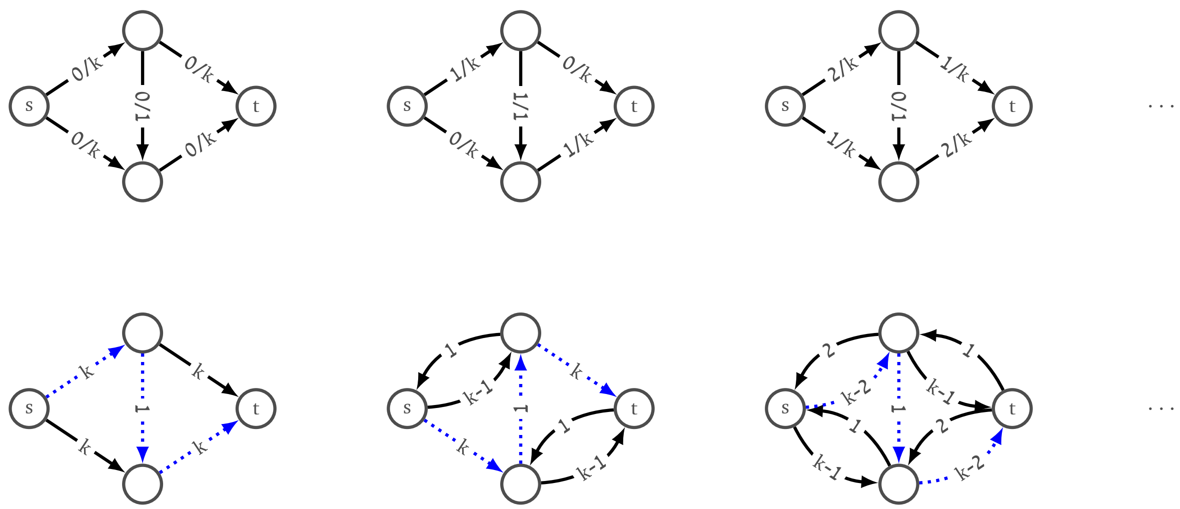
מסקנה 10. ברשת זרימה בה כל הקיבולים שלמים, קיימת זרימת מקסימום עם ערכים שלמים.

סיבוכיות: בשביל לנתח את הסיבוכיות של האלגוריתם יש לדעת:

1. לאחר כמה איטרציות האלגוריתם עוצר (אם בכלל)

2. מה הסיבוכיות של כל איטרציה

נתמקד בסעיף 1. אם כל הערכים הם שלמים אז ברור שמספר האיטרציות חסום על ידי ערך זרימת המקסימום. אם הערכים רציונליים אפשר להכפיל אותם כך שכל הערכים יהיו שלמים. הדוגמה הבאה מראה שבמקרה הגרוע זהו חסם עליון הדוק.



הרצאה 12

רשתות זרימה - אלגוריתם אדמונדס קרפ, שידוך בגרף דו צדדי, משפט הול

תזכורת

האלגוריתם הגנרי של פורד פלקרסון:

1. אתחול: מציבים $f(e) \leftarrow 0$ לכל $e \in E$
2. כל עוד יש מסלול שיפור ברשת השיורית (G_f, s, t, c_f)
   (א) מציבים ב-f את הזרימה המשופרת לפי למת שיפור הזרימה
3. פולטים את f

ראינו שאלגוריתם זה אינו פולינומי אפילו כאשר כל הקיבולים שלמים (ואף במקרה הכללי הוא אינו עוצר כלל).

אלגוריתם אדמונדס קרפ

מקרה פרטי של אלגוריתם זה הוא האלגוריתם של אדמונדס וקרפ:

1. אתחול: מציבים $f(e) \leftarrow 0$ לכל $e \in E$
2. כל עוד יש מסלול שיפור ברשת השיורית (G_f, s, t, c_f)
   (א) יהי P מסלול קצר ביותר מ-s ל-t.
   (ב) שפר לפי P והצב ב-f את הזרימה המשופרת לפי למת שיפור הזרימה
3. פולטים את f

כעת נראה שאלגוריתם זה הוא פולינומי ללא תלות בפונקציית הקיבול.
נסמן ב-$f_1, f_2 \ldots$ את פונקציית הזרימה שמחשב האלגוריתם בכל איטרציה, וב-$d_{f_i}(v)$ את המרחק של הצומת v מהצומת s ברשת השיורית.

טענה 43. לכל i ולכל v מתקיים ש-$d_{f_i}(v) \leq d_{f_{i+1}}(v)$.

הוכחה. עבור i נתון, נוכיח באינדוקציה על k - המרחק של v מ-s ב-$G_{f_{i+1}}$.
בסיס: עבור $k=0$ טריוויאלי.
צעד: עבור צומת v במרחק $k+1$ מ-s ומסלול $s = v_0, \ldots, v_k, v_{k+1} = v$ מתקיים (לפי הנחת האינדוקציה) ש:

$$d_{f_i}(v_k) \leq d_{f_{i+1}}(v_k)$$

אם הקשת $v_k v_{k+1}$ קיימת ב-G_{f_i} אז סיימנו.
אחרת במסלול השיפור ב-G_{f_i} קיימת הקשת $v_{k+1} v_k$ ומכאן:

$$d_{f_i}(v_{k+1}) = d_{f_i}(v_k) - 1 \leq d_{f_{i+1}}(v_k) - 1 = d_{f_{i+1}}(v_{k+1}) - 2$$

□

מסקנה 11. לכל v ולכל $i < j$ מתקיים $f_i(v) \leq f_j(v)$

מסקנה 12. אם $uv \in E_{f_{i+1}}$ ו-$uv \notin E_{f_i}$ אז $d_{f_{i+1}}(v) \geq d(f_i(v)) + 2$.

הגדרה 27 (קשת קריטית)**.** בהינתן מסלול P נגיד שקשת e במסלול היא קריטית אם הקיבול שלה הוא המינימלי מבין כל הקשתות במסלול.

טענה 44. אם P מסלול שיפור ב-G_{f_i} ו-e קשת קריטית במסלול, אז $e \notin E_{fi+1}$.

הוכחה. נובע ישירות מהגדרת הרשת השיורית. □

מסקנה 13. במהלך ריצת האלגוריתם, קשת uv יכולה להיות קריטית $\frac{|V|}{2}$ פעמים לכל היותר.

סיבוכיות ריצה: נשים לב שבכל איטרציה קיימת קשת קריטית אחת לפחות ולכן מספר האיטרציות חסום על ידי $|E| \cdot \frac{|V|}{2}$ ניתן לממש כל איטרציה על ידי BFS ולקבל זמן כולל של $O(|E|^2|V|)$.

שידוך

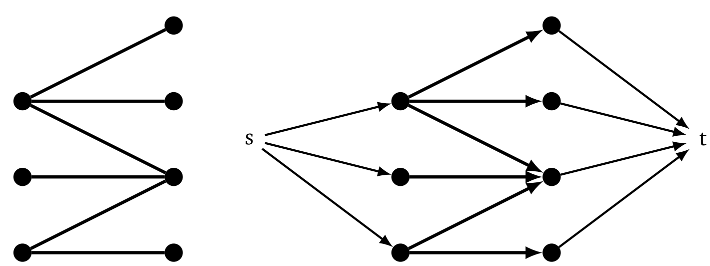
שידוך בגרף לא מכוון $G = (V, E)$ הוא תת קבוצה בלתי תלויה של קשתות $M \subseteq E$. כלומר, לכל שתי קשתות $e_1, e_2 \in M$ מתקיים ש-$e_1 \cap e_2 = \emptyset$.

הגדרה 28 (שידוך מקסימום)**.** שידוך M יקרא שידוך מקסימום אם לכל שידוך אחר, M', מתקיים $|M| \leq |M'|$.

שידוך בגרף דו צדדי:
בהינתן גרף דו צדדי, $G = (L, R, E)$ נגדיר את רשת הזרימה, $N = (G', c)$ כאשר

$$G' = (L \cup R \cup \{s, t\}, E')$$
$$E' = \{uv : uv \in E, u \in L, v \in R\} \cup \{s\} \times L \cup R \times \{t\}$$
$$c(e) = 1 \qquad \forall e \in E'$$

דוגמה:



טענה 45. אם M שידוך ב-G אז קיימת זרימה f ב-N כך ש-$|f| = |M|$.

הוכחה. לכל $uv \in M$ נציב $f(su) = f(uv) = f(vt) = 1$ ולכל שאר הקשתות נציב זרימה 0. קל לוודא שזו אכן פונקציית זרימה עם ערך $|M|$. □

טענה 46. אם f זרימה בשלמים ב-N אז קיים שידוך, M, ב-G כך ש-$|f| = |M|$

הוכחה. נגדיר $M = \{uv : u \in L, v \in R, f(uv) = 1\}$. קל לוודא שזהו אכן שידוך וש-$|f| = |M|$. □

מסקנה 14. קיים אלגוריתם פולינומי שמוצא שידוך מקסימום.

שידוך מושלם

הגדרה 29 (שידוך מושלם)**.** שידוך יקרא מושלם אם $|M| = \frac{|V|}{2}$.

הגדרה 30 (גרף רגולרי)**.** גרף לא מכוון יקרא d-רגולרי אם הדרגה של כל צומת היא d.

טענה 47. לכל $d \geq 1$ בגרף דו צדדי d-רגולרי קיים שידוך מושלם.

הוכחה. ראשית נשים לב שבגרף דו צדדי רגולרי מתקיים ש-$|R| = |L|$. נסמן $|L| = n$ ומספיק להראות שקיימת זרימה (לאו דווקא שלמה) שערכה n.
נגדיר:

$$f(sv) = 1 \quad \forall v \in L$$
$$f(vt) = 1 \quad \forall v \in R$$
$$f(uv) = \frac{1}{d} \quad \forall uv \in E : u \in L, v \in R$$

אפשר לוודא שזו אכן זרימה עם ערך n. □

משפט הול

עבור תת קבוצה של צמתים $U \subseteq V$ בגרף $G = (V, E)$ נסמן ב-$N(U)$ את קבוצת השכנים של U, כלומר:

$$N(U) := \{v : uv \in E, u \in U, v \notin U\}$$

משפט 7 (משפט הול)**.** בגרף דו צדדי $G = (L \cup R, E)$ שמקיים $|L| = |R| = n$ קיים שידוך מושלם אם ורק אם לכל $U \subseteq L$ מתקיים ש-$|U| \leq N(U)$.

נוכיח את המשפט באמצעות טיעונים על זרימה.
בהינתן גרף דו צדדי $G = (L \cup R, E)$ רשת זרימה מתאימה $N = (G', c)$ ו-חתך-st S נסמן ב-$U := S \cap L$ וב-$W := S \cap R$.

טענה 48. אם S חתך-st מינימום אז לא קיים צומת $v \in W \setminus N(U)$.

הוכחה. אם קיים אז $S \setminus \{v\}$ חתך קטן יותר. □

טענה 49. קיים חתך-st מינימום כך ש-$W = N(U)$.

הוכחה. נסתכל על חתך-st מינימום, S שממזער את $|N(U) \setminus W|$ (השכנים של U שלא ב-S) נניח בשלילה שקיים צומת $v \in N(U) \setminus W$ אז $S \cup \{v\}$ חתך עם ערך לא גדול משל S - סתירה. □

הוכחת משפט הול. אם S חתך-st מינימום כך ש-$W = N(U)$ ומתקיים $|U| \geq |W|$ אז ערך החתך הוא לפחות n. □

דוגמה: בדוגמה הבאה החתך הכתום מקווקו הוא חתך מינימום אבל אינו מכיל את כל השכנים של U, אם מוסיפים את השכן של U שמחוץ לחתך נקבל שוב חתך מינימום. החתך המנוקד הכחול אינו חתך מינימום ומכיל צומת שאינו שכן של U, אם נוציא את הצומת מהחתך נקבל חתך מינימום.