<u>תרגול מס' 10 – אלגוריתם דיניץ</u>

<u>הגדרה:</u>
רשת שכבות : תהי N_f רשת שיורית אשר קיים בה מסלול קצר ביותר מ-s אל t באורך k.
שכבה i של רשת השכבות Λ_f עבור $0 \le i \le k$ מכילה את הקודקודים $V_i = \{u \in V \mid d_f(u) = i\}$,
כאשר $d_f(u)$ הוא המרחק המינימאלי בצלעות מ- s ל- u ב (V, E_f).
E_f מכילה את כל הצלעות בין השכבות. $E_i = \{(u,v) \in E_f \mid u \in V_{i-1}, v \in V_i\}$.
גרף השכבות השיורי יהיה $L_f = (V_0 \cup V_1 \cup .... \cup V_k, E_1 \cup .... \cup E_k)$.
רשת השכבות תסומן : $\mathsf{L}_f = (L_f, c_f, s, t)$.

<u>תיאור סכמטי של האלגוריתם :</u>

א. אתחול: זרימה $f \leftarrow 0$, ורשת שיורית N_f המושרית מ-f.
ב. כל עוד יש מסלול מ-s ל-t ברשת השיורית בצע:
   a. בנה רשת שכבות L_f עבור N_f.
   b. מצא זרימה חוסמת b לרשת השכבות L_f.
   c. עדכן $f \leftarrow f + b$, ועדכן את N_f.

<u>תיאור בניית רשת השכבות, באמצעות BFS מורחב:</u>

הערה: *השכבה* V_i מכילה את הקדקודים שמרחקם מ-s הוא i.

א. הכנס את s ל- V_0, $i \leftarrow 0$.
ב. כל עוד V_i אינו ריק ו-t לא שייך ל- V_i בצע:
   a. עבור כל קדקוד u ב- V_i בצע:
      עבור כל שכן v של u, אם v לא נמצא באף V_j, $j \le i$, הוסף את v ל- V_{i+1}
      (אם הוא עדיין לא שם) והוסף (u,v) ל- E_f.
   b. $i \leftarrow i + 1$
ג. אם t לא שייך ל V_i, אין מסלול בין s ל t.
   אחרת (t שייך ל V_i) : הפלט הוא רשת השכבות.

<u>תיאור אלטרנטיבי לבניית רשת שכבות:</u>

א. הרץ BFS מקודקוד s, ולכל קודקוד v, יהי $d(s,v)$ המרחק מ-s ל-v כפי שנמצא ע"י הרצת ה-BFS.
ב. אם $d(s,t) = \infty$: אין מסלול בין s ל t.
ג. אחרת:
   a. לכל $(u,v) \in E$ בצע:
      i. אם $d(s,u) + 1 = d(s,v) \le d(s,t)$ הוסף את (u,v) ל-L_f.
   b. החזר את L_f.

Algorithm 1 Dinitz *(G, c, s, t)*

$f \leftarrow 0$; construct the residual network $N_f = (G_f, c_f, s, t)$;
while there is a path from s to t in G_f **do**
    ≤ *invariant assertion: f is a flow in N ;*
    construct the layered network $\mathsf{L}_f = (L_f, c_f, s, t)$;
    find a blocking flow b for L_f ; ≤ *assertion : $d_{f+b}(t) > d_f(t)$* ;
    $f \leftarrow f + b$;
    construct the residual network N_f ;
≤ *Post-condition of while loop: f is a maximum flow in N* ;

Algorithm 2 Layered_Network_Construction *(N_f)*

$V_0 \leftarrow \{s\}$; $i \leftarrow 0$;
while $(V_i \neq \phi)$ and $(t \notin V_i)$ **do**
    $V_{i+1} \leftarrow \phi$ **;** $E_{i+1} \leftarrow \phi$ **;**
    for each $u \in V_i$ **do**
        for each $v \in V$ such that $(\langle u,v \rangle \in E_f)$ and $(v \notin V_j, \forall j \leq i)$ **do**
            if $(v \notin V_{i+1})$ **then**
                add v to V_{i+1} ;
            add $\langle u,v \rangle$ to E_i ;
    $i \leftarrow i + 1$;
if $(V_i = \phi)$ **then**
    return $\mathsf{L}_f = (\phi, c_f, s, t)$; ≤ *assertion: there is no path from s to t in N_f*
$L_f \leftarrow (V_0 \cup \ldots \cup V_i, E_1 \cup \ldots \cup E_i)$;
return $\mathsf{L}_f = (L_f, c_f, s, t)$;
Post-condition: if t is reachable from s in N_f , then L_f is the layered network for N_f

---

Algorithm 3 Blocking_Flow *(L_f)*
<u>*Using a greedy algorithm*</u>

---

1. $b \leftarrow 0$; $M = (V_M, E_M) \leftarrow L_f$; $c \leftarrow c_f$;
2. **repeat**
3. ◉ *assertion : s is the only vertex in M with zero in-degree ;*
4. find a path p from s to t in M; let its bottleneck capacity be $c_M(p)$;
5. **for each** edge $\langle u,v \rangle \in p$ **do**
6. $b(u,v) \leftarrow b(u,v) + c_M(p)$; $b(v,u) \leftarrow -b(u,v)$;
7. $c_M(u,v) \leftarrow c_M(u,v) - c_M(p)$;
8. **if** $c_M(u,v) = 0$ **then**
9. remove $\langle u,v \rangle$ from M;
10. *cleanup of M*;
11. if *indegree (v) = 0* then
12. CleanForward (v, M) ;
13. **until** *indegree (t) = 0* ;

Algorithm 4 CleanForward *(u, M)*

Pre-condition: indegree (u) = 0
1. **for each** v such that $< u, v > \in M$ **do**
2. remove $\langle u,v \rangle$ from M;
3. **if** *indegree (v) = 0* **then**
4. CleanForward *(v, M);*

<u>*Comment:*</u> *Notice that the paths from s to t are not affected, and all of them remain in M.*

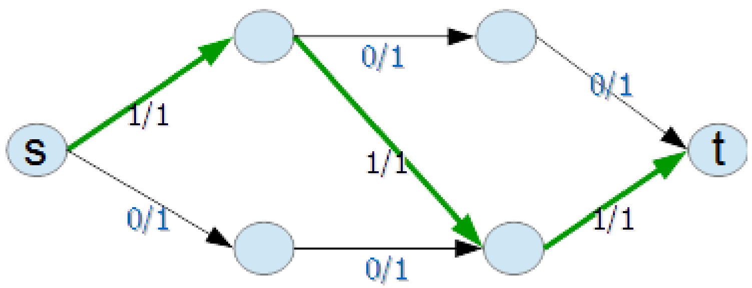
<u>הערות חשובות על אלגוריתם דיניץ:</u>

המושג "<u>שלב</u>" באלגוריתם יציין איטרציה בודדת של הלולאה באלגוריתם 1 (עדכון זרימה על סמך זרימה חוסמת ברשת השכבות).

<u>שורה 4 באלגוריתם 3:</u> מציאת מסלול ברשת השכבות מתבצעת על ידי סריקה אחורה מ-t עד ל-s. כיוון שדואגים לנקות את הרשת בכל שלב (CleanForward), מובטח כי סדרת הצעדים בהכרח לא תעצור

לפני הגעה ל-*s*, כאשר כל קשת היא בין שכבות עוקבות. כלומר P יהיה מסלול קצר ביותר מ-s ל-t. הזמן הדרוש למציאת מסלול P ברשת השכבות הוא לינארי במספר השכבות, O(|P|).

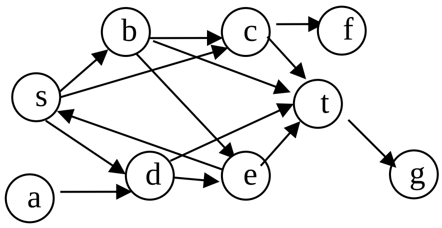
המושג "זרימה חוסמת" ברשת השכבות: שימו לב כי זרימה זו אינה בהכרח זרימה מקסימאלית ברשת השכבות. בשלב זה **לא מעדכנים** את קיבולי הקשתות ההפוכות לקשתות בהן עברה זרימה על ידי קשתות חרטה (אלו הן קשתות משכבה i+1 לשכבה i, ולכן הן לא חלק מרשת השכבות). בדוגמא הבאה ניתן לראות זרימה חוסמת (בירוק) ברשת השכבות שאיננה מקסימאלית בה:



נשים לב: באלגוריתם Ford-Fulkerson לא שומרים שום מידע מתהליך מציאת מסלול שיפור למציאת המסלול הבא בו "נעביר" זרימה. בניגוד לזה, האלגוריתם של דיניץ מנצל את מבנה הנתונים "רשת השכבות" (הנבנה ע"י סריקה אחת ב-G_f) למציאת מספר מסלולים קצרים ביותר, מבחינת מספר קשתות.

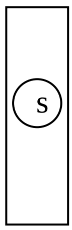
דוגמא לבניית רשת שכבות:

נתון הגרף הבא :



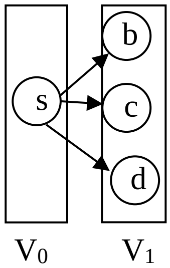
קודקוד המקור הוא s, קודקוד הבור הוא t.

שלבי בניית רשת השכבות יהיו כדלקמן :
בשלב הראשון, שכבה בודדת אחת:

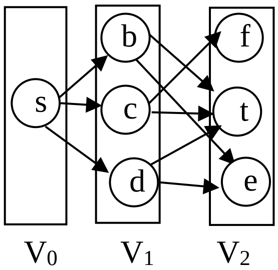


V_0

בשלב השני, שתי שכבות:



בשלב השלישי, שלוש שכבות:



כעת t מצוי בשכבה האחרונה, כלומר קיים מסלול מ-s ל-t ברשת השיורית, ואין לנו צורך בשכבות נוספות לעת עתה.

ניתוח זמן ריצה עבור שלב אחד:

1. אתחול – בניית הרשת השיורית: $O(|E|)$.
2. בניית רשת שכבות: (על ידי BFS מורחב מ-s עד t): $O(|E|)$.
3. נגדיר את זמן מציאת זרמה חוסמת לרשת השכבות ע"י $t(|E|,|V|)$.
4. עדכון הזרימה: $O(|E|)$.

סה"כ זמן ריצה עבור שלב: $O(|E|+t(|E|,|V|))$

ננסה כעת לספק חסם עבור מציאת הזרימה החוסמת (t (|E| + |V| :

תיאור מציאת זרימה חוסמת לרשת שכבות L_f:

"נשכפל" את רשת השכבות L_f ונבצע עידכונים על העותק ה"משוכפל", M.

א. אתחל זרימה חוסמת $b \leftarrow 0$.

ב. כל עוד דרגת הכניסה של t ב-M גדולה מ-0 (שקול: קיים מסלול P מ-s ל-t ב-M) בצע:

a. עדכן את b,M לפי מסלול P (בדומה ל-FF ללא הוספת קשתות חרטה נגדיות ב-M).

b. עבור כל קשת (u,v) שהתאפסה בעקבות שינוי הזרימה במסלול P

i. מחק את (u,v) מ- M.

ii. אם דרגת הכניסה של v היא אפס וגם $v \neq t$. בצע ניקוי-קדימה (v,M).

ג. החזר b.

תיאור ניקוי-קדימה $(v,(V',E'))$:

לכל קשת (v,x) (קשת היוצאת מ-v) בצע:

a. מחק את (v,x) מ-E'.

b. אם דרגת הכניסה של הקודקוד x היא 0 וגם $x \neq t$ בצע: ניקוי-קדימה$(x,(V',E'))$.

<u>ניתוח זמן ריצה עבור מציאת זרימה חוסמת לרשת שכבות</u> $t(|E|,|V|)$:
הערה: בכל הקשור לרשתות זרימה, $O(|V|+|E|)=O(|E|)$. הסיבה לכך נעוצה בהנחה שגרף התשתית של רשת הזרימה קשיר ולכן $|E|\leq|V|-1$. (במידה והוא אינו קשיר, ניתן למחוק מרכיבי הקשירות שאינם מכילים את הקדקודים s,t בשלב העיבוד המוקדם.)

1. ***בניה:*** יצירת רשת השכבות והעתקתה ל-M מתבצעת באמצעות הפעלת BFS מורחב, ולכן עלותה הינה $O(|V|+|E|)=O(|E|)$.
2. ***עדכון זרימה:*** מציאת מסלול שיפור P ברשת השכבות עולה $O(|P|)=O(|V|)$: כיוון שאנו מבצעים "ניקוי", לכל קודקוד שאינו s ישנה דרגת כניסה גדולה מ-0; לכן, לכל קודקוד שמגיעים אליו בסריקה הפוכה מ-t ישנה צלע נכנסת, ולכן סריקה זו לא תעצור לפני הגעה ל-s. בנוסף, לאחר מציאת המסלול P, הקיבולת של P מתווספת לזרימה בכל אחת מצלעותיו ויורדת מהקיבולת השיורית שלה; עלות - $O(|P|)=O(|V|)$ גם כן.
איטרציות מסלולים כאלה ברשת השכבות מתבצעות לכל היותר E פעמים, כיוון שבכל פעם נמחקת קשת אחת, לפחות, מרשת השכבות. סה"כ הזמן הדרוש לעבודה כנ"ל עם כל המסלולים אם כן הוא $O(|E|\cdot|V|)$.
3. ***מחיקת צלעות:*** כדי לחשב את הזמן הכללי של ניקוי רשת השכבות, נשתמש בשיטת פיזור החיובים (ניתוח פחת):
   a. כל קשת וקדקוד יכולים להימחק לכל היותר פעם אחת $\Leftarrow O(|V|+|E|)=O(|E|)$ עבור כל הקשתות והקדקודים.
   b. בדיקת דרגת הכניסה עבור קודקוד מתבצעת בכל פעם שאנו מוחקים קשת (ניתן לבדוק דרגת קודקוד בזמן קבוע ע"י שמירת מוני דרגה) $\Leftarrow O(|E|)$ עבור כל בדיקות הקודקודים.

   <u>כלומר</u>, סך עלות כל העדכונים של M הינה $O(|E|)$ (כמו עלות בניית M).

<u>החסם המתקבל עבור מציאת זרימה חוסמת:</u>
$t(|E|,|V|)=O(|V|\cdot|E|)+O(|E|)=O(|V|\cdot|E|)$.

אם כן, <u>זמן ריצה עבור שלב אחד באלגוריתם</u>: $O(|E|)+t(|E|,|V|)=O(|V|\cdot|E|)$.

ישנם לכל היותר $|V|$ שלבים (בשיעור ראיתם כי בכל שלב מספר השכבות ברשת השכבות גדל, כאשר הוא חסום ע"י $|V|-1$), לכן זמן הריצה לאלגוריתם כולו הוא $O(|V|^2\cdot|E|)$.

ניתוח זמן הריצה עבור אלגוריתם דיניץ המפורט לעיל נכון עבור גרף כללי עם משקולות כלשהן. אך ישנן רשתות זרימה עבורן ניתן למצוא חסמים טובים יותר. <u>האלגוריתם עצמו לא ישתנה</u>, אך הניתוח יהיה עדין יותר ויתבסס על תכונות הרשת.

גרפים בעלי קשתות בקיבול 1 בלבד:

נראה שבגרפים כאלה ניתן להגיע לחסם זמן ריצה של $O(|E|\sqrt{|E|})$.

כדי להראות זאת, נראה שבגרפים בעלי קיבול 1 ניתן לחסום בצורה הדוקה יותר הן את עלותו של כל שלב והן את מספר השלבים.

א. נראה קודם כי עלותו של כל שלב באלגוריתם אורכת $O(|E|)$ בלבד (במקום החסם לא הדוק $t(|E|,|V|)=O(|E|\cdot|V|)$).

הוכחת א: נתבונן ברשת השכבות בתחילת שלב כלשהו, ונסמן את מספר השכבות ברשת זו כ-m+1 ($0,1,\ldots,m$). ברשת זו, אורך כל מסלול שיפור הינו m. כמה מסלולי שיפור יכול האלגוריתם למצוא? בהינתן זרימה b ברשת השכבות, נסתכל על סכום ערכי הזרימה בצלעות שנמצאות ברשת השכבות בלבד, אשר יסומן ב-$sum_m(b)$. (הערה: כמובן שערך זה גדול מ-|b|, מפני שאנו סופרים זרימה לאורך כל השכבות ולא רק הצלעות שיוצאות מ-s.)

תחילה, $b=0$ ולכן $sum_m(\mathrm{b})=0$. בנוסף, $sum_m(\mathrm{b})$ תמיד חסום ע"י סך קיבולי הצלעות שנבחרו לרשת השכבות. לכל צלע ברשת השכבות הקיבול שווה ל-1 או 2, יש לכל היותר $|E|$ צלעות לכן תמיד יתקיים : $sum_m(\mathrm{b})\le 2|E|$.

כעת, בכל מסלול ישנן m צלעות וצוואר הבקבוק הוא 1 או 2. לפיכך, לאחר שינוי הזרימה לאורך מסלול שיפור $sum_m(\mathrm{b})$ גדל ב-m או $2m$. מכיוון שאנו בונים זרימה חוסמת באמצעות אלגוריתם חמדן (אשר אינו "מתחרט"), נסיק כי הזרימה בקשתות הרשת יכולה רק לגדול.

התוצאה היא שניתן למצוא לכל היותר $\frac{2|\mathrm{E}|}{m}$ מסלולי שיפור.

כעת נראה ש-$t(|E|,|V|)=O(|E|)$, כלומר שכל שלב בודד עולה לכל היותר $O(|E|)$:

עלות מציאת מסלול ברשת השכבות ועדכון הזרימה והקיבולות השיוריות לאורך מסלול זה מבוצעות בעלות $O(m)$. כמו כן, תהליך מציאת מסלול ועדכונו מתבצע לכל היותר $\frac{2|\mathrm{E}|}{m}$ פעמים. לכן סך עלות מציאת המסלולים ועדכון הזרימה והקיבולות השיוריות ברשת השכבות הינה $\frac{2|E|}{m}\cdot O(m)=O(|E|)$.

מכיוון שעלות יתר הפעולות (בנית הרשת וניקוי הקשתות) הינה $O(|E|)$ בניתוח זמן הריצה עבור כל רשת זרימה כללית, נקבל שעלות כל שלב הינה $O(|E|)$.

בשלב זה של ההוכחה, ניתן לומר כי זמן הריצה של כל האלגוריתם חסום ע"י $O(|V|\cdot|E|)$ (ולא $O(|V|^2\cdot|E|)$ לפי ניתוח זמן הריצה של המקרה הכללי).

הערה למיטיבי לכת:

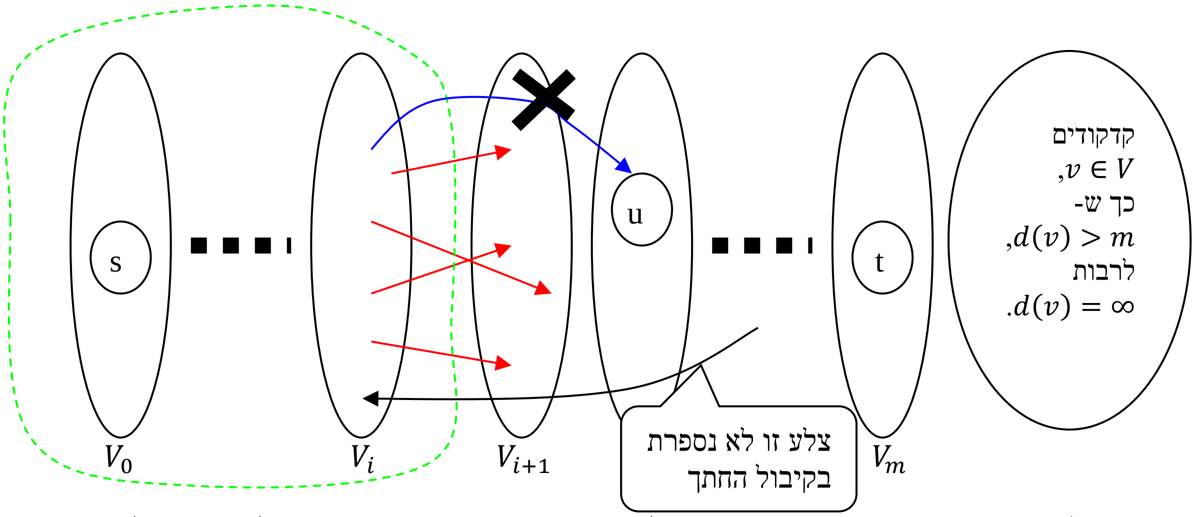
בפועל, סך קיבולי הצלעות ברשת שיורית חסום ע"י $|E|$ בדיוק:

- אם בין הקדקודים u,v ישנה קשת בכיוון אחת בלבד, אזי לאחר "הזרמת" יחידה אחת על קשת זו, נקבל הופעה של קשת בכיוון הנגדי במקום הקשת המקורית. כלומר קיבולת הצלעות שבין שני הקדקודים u,v לא גדלה.
- אם בין u,v ישנן קשתות בשני הכיוונים, אזי לאחר הזרמת יחידה באחת מן הקשתות, קשת זו תעלם מן הרשת השיורית, ואילו הקיבולת השיורית של הצלע הנגדית תהיה 2, ולכן סף הקיבולות נותר כפי שהיה במקור – 2.

ב. עתה נראה כי ניתן להקטין גם את החסם עבור מספר השלבים שמתבצעים ע"י האלגוריתם.

תזכורת: עבור חתך $(S,V\backslash S)$, צלע חוצה היא צלע שעוברת מ S ל $V\backslash S$, וקיבולת החתך היא סכום קיבולות הצלעות החוצות אותו.

נתבונן בשלב בו המרחק מ-s ל-t שווה m. נסמן ב f את הזרימה בתחילת השלב. נגדיר חתכים $(S_i, V\setminus S_i)$ ברשת השיורית N_f (לא ברשת השכבות!), כאשר S_i היא קבוצת כל הקודקודים במרחק עד i מ-s. ניזכר בסימון $V_i = \{\, v \in V \mid d_f(s,v) = i\}$; נשתמש עבור V_i במונח "השכבה ה i" של G_f.



אבחנה: לפי תכונות מרחקים (או תכונות BFS) **לא ייתכן שישנן קשתות משכבה i לשכבה גדולה מ-$i + 1$**. בדוגמא הקשת הכחולה (זו עם ה-**X** למדפיסים בשחור-לבן) לא תיתכן ברשת השיורית הנ"ל. כלומר הקשתות האדומות מ V_i ל V_{i+1} הן היחידות החוצות את החתך הירוק (המסומן בקו מקווקו) "קדימה" מ-S_i ל-$V\setminus S_i$. מסקנה: *החתכים $(S_i, V\setminus S_i)$ ברשת השיורית זרים בצלעות החוצות.*

סה"כ ישנם m חתכים כאלה, וכן ישנן סה"כ לכל היותר $|E|$ קשתות בהם. לכן מספר הקשתות הממוצע לחתך הוא לכל היותר $\frac{|E|}{m}$. לפיכך, קיים ביניהם חתך אשר מספר הקשתות בו הוא לכל היותר $\frac{|E|}{m}$. כיוון שקיבולי כל הקשתות הם לכל היותר 2 (מכיוון שבגרף המקורי כל הצלעות הם בקיבול 1), קיבולו של חתך זה חסום על ידי $\frac{2|E|}{m}$. לסיכום: קיים חתך ברשת השיורית שקיבולו הוא לכל היותר $\frac{2|E|}{m}$.

ניזכר בטענה: "גודל זרימת מקסימום שווה לקיבולת חתך מינימום". על פיה, גודל זרימת מקסימום ברשת השיורית הנ"ל חסום ע"י $\frac{2|E|}{m}$. בגלל שכל מסלול שיפור מגדיל את הזרימה ב-1 לכל הפחות, לכן $\frac{2|E|}{m}$ הינו חסם על מספר מסלולי השיפור שהאלגוריתם ימצא מרגע זה ועד סוף ביצועו!

טענה (ללא הוכחה): *אם g' זרימת מקסימום ברשת השיורית N_f אזי $f + g'$ היא זרימת מקסימום ב N, וגודלה $|f| + |g'|$*. (טענה זו נובעת מטענה אחרת אשר הוכחה בהרצאות, לפיה עבור כל זרימה f ברשת זרימה N וכל זרימה g ברשת השיורית N_f, מתקיים ש-$(f + g)$ הינה זרימה חוקית ברשת N.)

נתבונן בשלב בביצוע האלגוריתם שבו לראשונה המרחק m מ-s ל-t הינו $\geq \sqrt{2|E|}$. מאופן בחירת השלב, נסיק כי לפניו היו לכל היותר $\sqrt{2|E|}$ שלבים (לפי טענה אשר הוכחה בהרצאות, בכל שלב s מתרחק מ-t לפחות באחד). כפי שהוכח לעיל, מן השלב הנוכחי ועד תום ריצת האלגוריתם יתבצעו לכל היותר $\frac{2|E|}{m}$ איטרציות, ולכן לכל היותר $\frac{2|E|}{m}$ שלבים (כי בכל שלב מתבצעת לפחות איטרציה אחת). מכיוון ש- $m \geq \sqrt{2|E|}$, נסיק כי יישארו לכל היותר $\frac{2|E|}{\sqrt{2|E|}} \leq \sqrt{2|E|}$ שלבים. התקבל שסה"כ מספר השלבים שמבצע האלגוריתם במהלך ריצתו הינו לכל היותר: $\leq \sqrt{2|E|}$ שלבים שבוצעו פלוס $\leq \sqrt{2|E|}$ שלבים שיבוצעו ← לכל היותר $2\sqrt{2|E|}$ שלבים.

מסקנה: *סה"כ זמן ריצת אלגוריתם דיניץ עבור רשת עם כל קיבולות הצלעות 1 הוא* $2\sqrt{2|E|} \cdot O(|E|) = O\left(|E|^{\frac{3}{2}}\right)$.