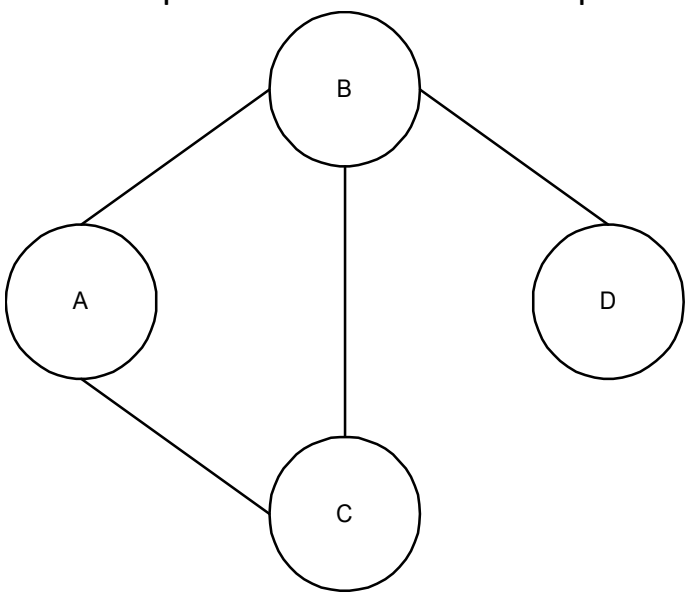
6.5 סריקה לרוחב – (BFS)

פעולה חשובה ושכיחה בגרף היא סריקה של קודקודי גרף. בעת סריקה נעבור על פני הגרף כך שנבקר בכל קודקוד פעם אחת בלבד ונבצע בו עיבוד בדרך כלשהי. קיימות שתי שיטות סריקה שונות הנבדלות זו מזו בסדר שבו מתבצעת סריקת קודקודי הגרף. השיטות הן :

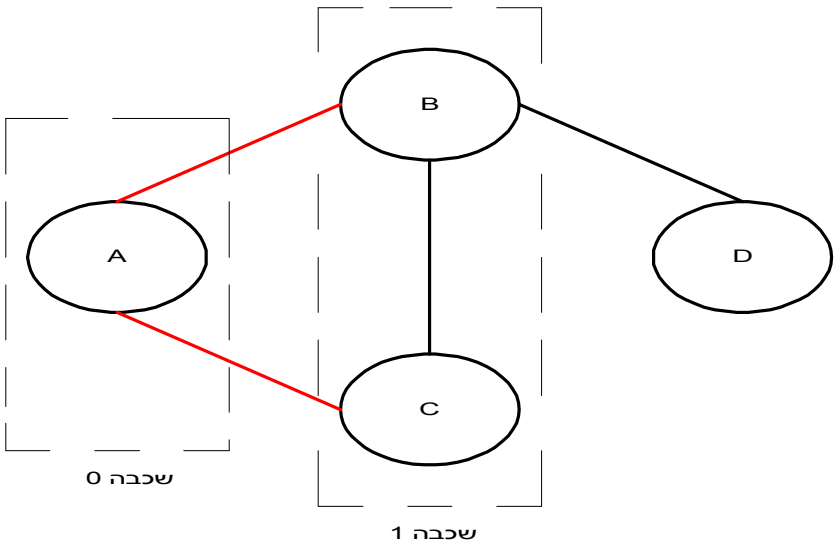
סריקה לרוחב – (BFS) Breadth-First Search
סריקה לעומק – (DFS) Depth-First Search

בפרק זה נכיר את השיטה – סריקה לרוחב ובפרק הבא – סריקה לעומק.
להלן תיאור השיטה – BFS.
נתון גרף G=(V,E) . בשיטה זו מתחילים את הסריקה מאחד הקודקודים בגרף הנקרא קודקוד מקור ונסמנו ב- S.
המטרה לטייל על פני הקשתות מקודקוד לקודקוד של גרף כך שנבקר בכל קודקוד של גרף ונבצע בו עיבוד כלשהו פעם אחת בלבד.
להשגת המטרה שיטת סריקה זו תתבצע כדלהלן:
מחלקים את קודקודי הגרף לשכבות באופן הבא:
בשכבה ראשונה יהיו קודקודים שניתן להגיע אליהם, מקודקוד מקור S ,בעזרת מסלול שאורכו אחד.
בשכבה שניה יהיו קודקודים שניתן להגיע אליהם, מקודקוד מקור S ,בעזרת מסלול שאורכו 2.
ובאופן כללי : בשכבה i , לכל $i \geq 1$,יהיו קודקודים כך שניתן להגיע אליהם, מקודקוד מקור S ,בעזרת מסלול שאורכו i .
בשלב הראשון של האלגוריתם הנדון מבקרים בכל הקודקודים השייכים לשכבה הראשונה.
בשלב השני של האלגוריתם מבקרים בכל הקודקודים השייכים לשכבה השניה וכך נמשך תהליך הביקור בקודקודי הגרף עד שנבקר בכל קודקודי הגרף פעם אחת בלבד.

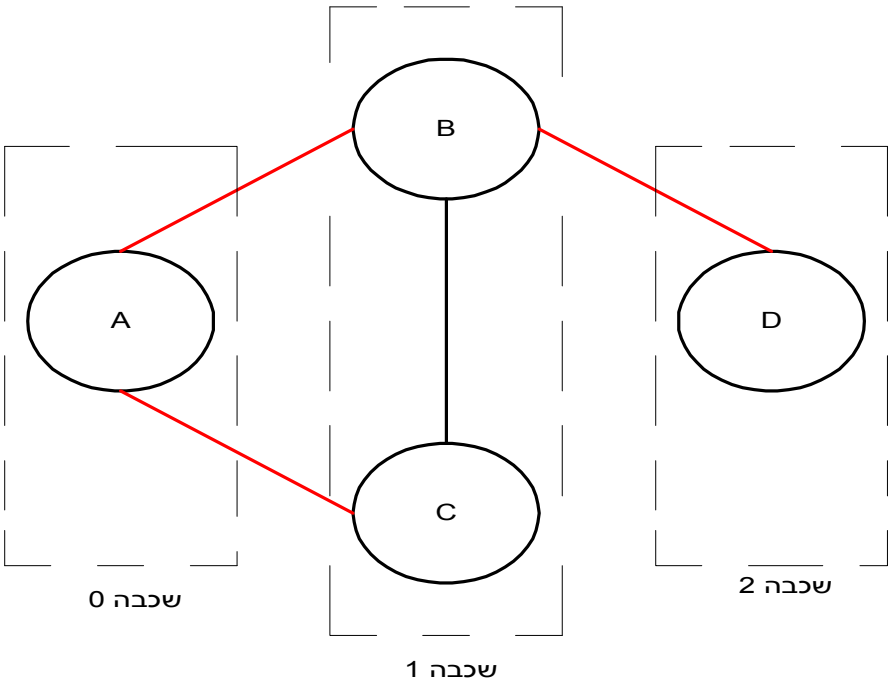
נדגיש כי האלגוריתם הנדון מבקר בכל קודקודי הגרף השייכים לשכבה K ,לפני שמבקרים(מגלים) קודקוד כלשהו השייך לשכבה K+1 .
נדגים את התהליך שתואר לעיל על הגרף הבא:



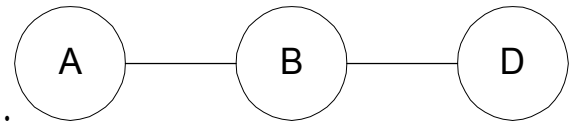
נניח שהקודקוד ממנו מתחילים את הסריקה הנו קודקוד A .
קודקוד A שייך לשכבה 0 כיוון שאורך המסלול מצומת A לעצמו הינו 0. עתה נבחר בכל הקשתות הנוגעות לקודקוד A ומובילות לקודקודים שעדיין לא ביקרנו בהם.בדוגמא שלנו הקשת (A,B) מובילה לקודקוד B שעדיין לא ביקרנו בו והקשת (A,C) מובילה לקודקוד B שעדיין לא ביקרנו בו.
לכן נבחר בשתי הקשתות ותמונת המצב הנה:



הקודקודים C ו B שייכים לשכבה 1 מאחר שניתן להגיע אליהם , מקודקוד מקור A ,בעזרת מסלול שאורכו 1.
שוב נצטרך לבחור בכל הקשתות הנוגעות לכל הקודקודים השייכים לשכבה מספר 1 ומובילות לקודקודים שעדיין לא ביקרנו בהם.
בדוגמא שלנו הקשת (B,D) מובילה מקודקוד B , ששייך לשכבה מספר 1, לקודקוד D שעדיין לא ביקרנו בו.
לעומת זאת הקשת (B,C) מובילה מקודקוד B , ששייך לשכבה מספר 1 ,לקודקוד C שכבר ביקרנו בו.לכן בשלב זה נבחר רק בקשת אחת - (B,D) ותמונת המצב הנה:



הקודקוד D שייך לשכבה 2 מאחר שניתן להגיע אליו מקודקוד מקור A דרך מסלול שאורכו 2 שהינו:



.

שוב נצטרך לבחור בכל הקשתות הנוגעות לכל הקודקודים השייכים לשכבה 2 ומובילות לקודקודים שעדיין לא ביקרנו בהם.
בדוגמא שלנו לשכבה מספר 2 שייך רק קודקוד אחד D ודרך הקשת היחידה הנוגעת בו (B,D) לא ניתן להגיע לקודקוד שעדיין לא ביקרנו בו(מכיוון שבקודקוד B כבר ביקרנו!).
בשלב זה נעצור את הסריקה כיוון שסיימנו את הביקור בכל קודקודי הגרף.
בסוף התהליך, כאשר מסיימים לבקר בכל קודקודי הגרף, השכבה שאליה שייך הקודקוד תציין את אורך המסלול מקודקוד מקור עד אליו ומסלול זה יהיה בעל אורך מינימלי (מספר הקשתות מינימלי).
הערה:
האלגוריתם פועל על גרפים מכוונים ולא מכוונים כאחד.
מאחר שברצוננו לבקר בכל קודקוד אך ורק פעם אחת , נשתמש במערך בוליאני used כך ש- used[v] יציין אם ביקרנו בקודקוד v או לאו.בתחילת האלגוריתם עדיין לא גילינו (ביקרנו) את קודקודי הגרף,לכן עבור כל קודקוד v בגרף, נציב ל-used[v] את הערך false .
כמו כן נרצה עבור כל קודקוד לשמור מידע על זהות הקודקוד הקודם לו בעת הסריקה.לכן נשתמש במערך P , כך ש- P[v] יציין קודקוד בגרף שממנו הגענו ל- v בעת הסריקה.
הערה: הסימן P ,נבע מהסיבה ש- P[v] ייצג "הורה" (Parent) של קודקוד v בעת הסריקה.
בנוסף נרצה לשמור עבור כל קודקוד u מידע על המרחק מקודקוד S לקודקוד u.לכן נשתמש במערך dist כך ש- dist[u] יציין את המרחק מקודקוד המקור S לקודקוד u .
בנוסף נשתמש במבנה נתונים תור (Queue) - Q אשר ינהל את השכבות.כמו כן נשתמש בפעולות הבסיסיות המוגדרות על תור. כגון:

Insert(Q,w)	המוסיפה את האיבר w לתור Q .
Delete(Q,v)	המסירה את האיבר שבחזית התור Q ושמה אותו ב - v .

Is_empty(Q) המחזירה ערך "אמת" (TRUE) אם התור Q ריק,אחרת היא מחזירה ערך "שקר" (FALSE).

Head(Q) מחזירה את האיבר שבחזית התור Q.

<u>להלן אלגוריתם לשיטת סריקה לרוחב</u>

<u>צעד 1.</u> לכל קודקוד $v \in V$, פרט לקודקוד מקור (S), בצע:

1.1 $dist[v] \leftarrow \infty$ (בתחילת האלגוריתם אורך המסלול מקודקוד מקור S לקודקוד כלשהו v הנו ∞).

1.2 $P[v] \leftarrow NIL$ (בתחילת האלגוריתם לקודקוד v אין אבות).

1.3 $used[v] \leftarrow FALSE$

<u>צעד 2.</u> 2.1 $dist[S] \leftarrow 0$ (אורך המסלול מהמקור S ל-S עצמו הוא 0).

2.2 $P[S] \leftarrow NIL$ (לקודקוד מקור S אין אב).

2.3 $used[S] \leftarrow TRUE$ (בתחילת האלגוריתם מבקרים רק בצומת המקור S).

<u>צעד 3.</u> $Q \leftarrow \{S\}$ (אתחול התור).

<u>צעד 4.</u> כל עוד התור Q לא ריק , בצע:

4.1 $v \leftarrow Head(Q)$

4.2 לכל קודקוד w בגרף,שהינו שכן של v, בצע:

4.2.1 אם $used[w] = FALSE$?

אז בצע:

4.2.1.1 $used[w] \leftarrow TRUE$

4.2.1.2 $dist[w] = dist[v] + 1$

4.2.1.3 $P[w] \leftarrow v$

4.2.1.4 $Insert(Q, w)$

4.3 Delete(Q,v)

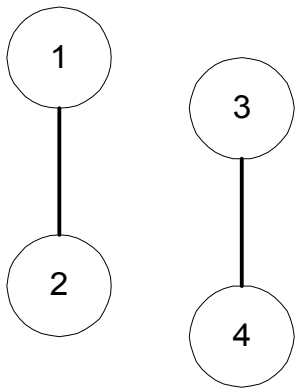
<u>צעד 5.</u> סוף האלגוריתם.

<u>הסבר על צעדי האלגוריתם</u>

<u>צעד 1:</u>

א.נתבונן על הגרף הבא:

ונניח שקודקוד מקור הנו S=1.



קל לראות שמקודקוד מקור S לא ניתן להגיע לקודקודים שמספרם 3 ו 4 .לכן ברור כי הערך של dist[3] הוא ∞ ,מכיוון שלא קיים מסלול מקודקוד 1 לקודקוד 3.

לאור זאת נחליט שבתחילת האלגוריתם לכל קודקוד v ,פרט למקור, נבצע את ההצבה הבאה:

$dist[v] \leftarrow \infty$,מאחר שלא ידוע מראש אם קיים מסלול באורך כלשהו מקודקוד מקור S לקודקוד כלשהו v .

במהלך האלגוריתם נרצה לשפר את אורך המסלול מקודקוד מקור – S לקודקוד v , לכל $v \in V$.

ב. אם לא קיים מסלול באורך כלשהו מקודקוד מקור S לקודקוד

כלשהו v , אז לקודקוד זה אין הורה (או P[v] = nil?)

.

לכן, בתחילת האלגוריתם לאף קודקוד אין הורה , מכיוון שתהליך הסריקה עדיין לא התחיל .

לכן נבצע את ההצבה הבאה :

לכל קודקוד $v \in V$ -{S} P[v]← NIL .

ג. ברור כי לכל קודקוד $v \in V$ -{S} נבצע את ההצבה :
Used[v] ← FALSE כיוון שבתחילת האלגוריתם עדין לא גילינו(ביקרנו) את צמתי הגרף.

צעד 2 :

א. מיותר להסביר מדוע אורך המסלול מקודקוד מקור S לעצמו
הינו 0 .

ב. לקודקוד מקור בגרף אין אב (בגלל זה הוא מקור) ולכן נבצע
השמה P[s]← NIL .

מאחר שהסריקה מתחילה מקודקוד מקור S אז הקודוקד S מתגלה מיד ,ולכן יש לבצע את ההשמה הבאה:
Used[S] ← TRUE .

צעד 3 :

כאמור תפקידו של תור (Q) לנהל את השכבות ולקבוע בכל שלב i ,
לכל $i \geq 0$, מיהם הקודקודים ששייכים לשכבה ה- i ית.
ברור כי לשכבה 0 שייך אך ורק צומת המקור S .
לכן נבצע את ההשמה : Q ← {S}.
בשלב זה התור Q מכיל קבוצת קודקודים השייכים לשכבה 0.
הערה :
בתור Q נמצאים קודקודים אשר נתגלו (ביקרנו בהם תוך כדי סריקה),
אך רשימת הסמיכות (שכנים) שלהם טרם נתגלו.

צעד 4:

א. נסתכל על הקודקוד שבחזית התור Q ונכנה אותו בשם v.

ב. עתה נבחן את כל השכנים של הקודקוד v.

ג. לכל קודקוד w ,שהינו שכן של הקודקוד v , ושעדיין לא נתגלה נבצע את הצעדים הבאים:

1.נבקר בו כעת (Used[w] ← TRUE) .

2.אורך המסלול מ-S ל-w שווה לאורך המסלול מ-S ל- v ועוד אחת(עקב המעבר על הקשת (v,w)).

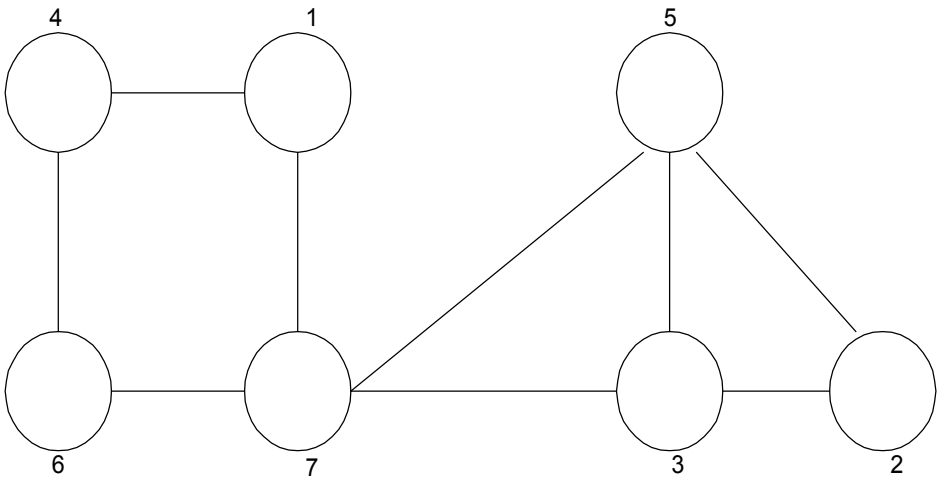
3.יש לזכור שהקודקוד v הינו "הורה" של הקודקוד w

4.צרף את הקודקוד w לתור Q .

ד. לאחר שנבחנו כל הקודקודים השכנים של v , נסיר מהתור Q את הקודקוד v כיוון שהטיפול בו הסתיים .

כך חוזר התהליך על עצמו עד שמטפלים בכל הקודקודים שניתן להגיע אליהם מ-S (קודקוד מקור).
ברגע שהתור Q יתרוקן אלגוריתם יסתיים, מכיוון שכל הקודקודים שנשיגים מ-S נתגלו(טופלו).

נדגים את צעדי האלגוריתם שתואר לעיל על הגרף הבא:



בגרף הנתון 7 קודקודים הממוספרים באופן אקראי מ-1 עד 7 והם מצויינים בסמוך לצומת .

הערה: לצורך ההדגמה נצייג את הערך הבוליאני FALSE ("שקר") בעזרת 0 ואת הערך הבוליאני TRUE ("אמת") בעזרת 1 .

כאמור בתחילת האלגוריתם תמונת המערך Used הינה:

	1	2	3	4	5	6	7
Used	0	0	0	0	0	0	0

במהלך התיאור של האלגוריתם בכל קודקוד של גרף מופיעים
2 מספרים : השמאלי מייצג את השכבה אליה שייך הקודקוד , והימני מייצג את "ההורה"של הקודקוד כפי שנקבע בעת הסריקה.

נניח שקודקוד מקור הינו קודקוד S=1 .

צעד (1) – (3):
לאחר ביצוע שלושת הצעדים הראשונים תמונת המצב מוצגת בטבלה הבאה:

קודקוד	1	2	3	4	5	6	7
Dist	**0**	∞	∞	∞	∞	∞	∞
P	**nil**	nil	nil	nil	nil	nil	nil
Used	**1**	0	0	0	0	0	0

Q={1}

צעד 4:

4.1 v ← 1 .

4.2 השכנים של v הם הקודקדים 4 ו 7 .

(בעזרת המערך Used רואים כי עדיין לא נתגלו הקודקודים 4 ו 7).

אורך המסלול מקודקוד 1 לעצמו הוא 0

ומקודקוד 1 לקודקוד 4 הוא 1.

לכן אורך המסלול מקודקוד 1 לקודקוד 4 הוא 1 .

באופן אנלוגי אורך המסלול מקודקוד 1 לקודקוד 7 הוא 1 .

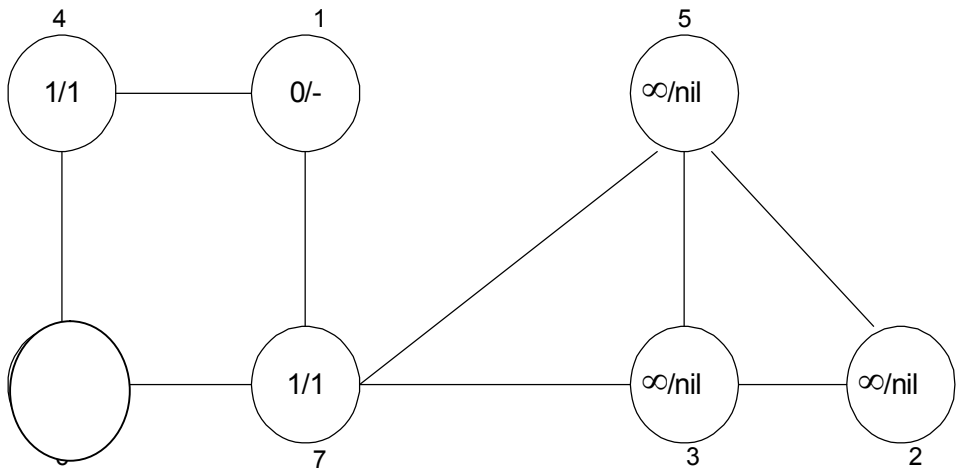
בנוסף ברור כי ה"הורה" של הקודקודים 4 ו 7 הוא הקודקוד 1.

וכמו כן Used[4]=Used[7]=TRUE , כי כעת הם נתגלו .

לכן תמונת המצב מוצגת בטבלה הבאה:

קודקוד	1	2	3	4	5	6	7
Dist	0	∞	∞	**1**	∞	∞	**1**
P	nil	nil	nil	**1**	nil	nil	**1**
Used	1	0	0	**1**	0	0	**1**

תמונת המצב אשר מוצגת בטבלה שנתקבלה עד כה, ניתן לתארה
גם בתרשים הבא:



הקודקודים 4 ו 7 מתווספים לתור Q , לכן $Q=\{1,4,7\}$.
מאחר שהטיפול בקודקוד 1 היסתיים נוציא אותו מהתור Q
ולכן $Q=\{4,7\}$.
התהליך חוזר חלילה וחוזרים לצעד 4.

צעד 4:

4.1 4 ← v .

4.2 השכנים של v הם הקודקדים 1 ו 6 .

(בעזרת המערך Used רואים כי בקודקוד 1 כבר ביקרנו, אך בקודקוד 6 עדין לא ביקרנו, לכן נבקר אך ורק בקודקוד 6) .

אורך המסלול מקודקוד 1 ל- 4 הוא 1 ומקודקוד 4 לקודקוד 6 הוא 1,לכן אורך המסלול מקודקוד1 לקודקוד 6 הוא 2.

ניתן לתאר את המסלול מקודקוד 1 לקודקוד 6

באופן הבא :

1 1 2

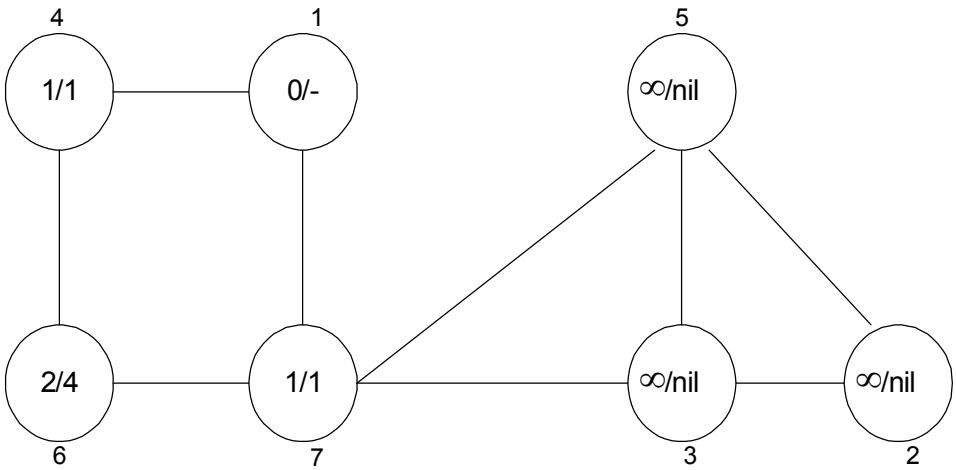
1~~~~~~4 ~~~ 6 1 ⟶ 6

ברור כי ה"הורה" של הקודקוד 6 הוא קודקוד 4 ובנוסף ברור כי Used[6]=True כי כעת מבקרים בקודקוד 6.

לכן תמונת המצב מוצגת בטבלה הבאה:

קודקוד	1	2	3	4	5	6	7
Dist	0	∞	∞	1	∞	**2**	1
P	nil	nil	nil	1	nil	**4**	1
Used	1	0	0	1	0	**1**	1

כעת קודקוד 6 מתווסף לתור Q , לכן $Q=\{4,7,6\}$.
שים לב שבתור Q קודקודים השייכים לשכבה 1
(קודקודים 4 ו 7)
וקודקודים השייכים לשכבה 2 (קודקוד 6).
מאחר שהטיפול בקודקוד 4 היסתיים נוציא אותו מהתור Q
ולכן $Q=\{7,6\}$.
התהליך חוזר חלילה וחוזרים לצעד 4 .
תמונת המצב אשר מוצגת בטבלה שנתקבלה עד כה, ניתן לתארה
גם בתרשים הבא:



כאמור בכל קודקוד של גרף מופיעים 2 מספרים.
השמאלי מייצג את השכבה אליה שייך הקודקוד , והימני את ה"הורה" של הקודקוד כפי שנקבע בעת הסריקה.

תמונת התור הינה: 6 7 Q
2 1 שכבה

עתה שוב חוזרים לביצוע צעד 4.

צעד 4:

4.1 7 ← v .

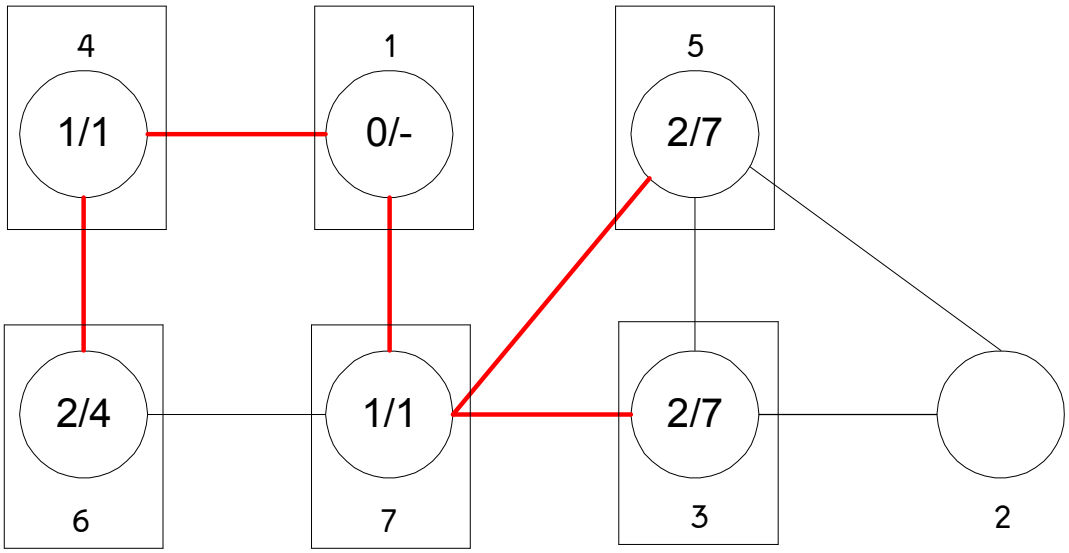
4.2 השכנים של הקודקוד 7, שעדיין לא ביקרנו בהם,
הם הקודקדים 3 ו 5 .
אורך המסלול מקודקוד 1 לקודקוד 7 הוא 1
ומקודקוד 7 לקודקוד 3 הוא 1 , לכן אורך המסלול מקודקוד 1 לקודקוד 3 הוא 2 .
ניתן לתאר את אורך המסלול מקודקוד 1 לקודקוד 3
באופן הבא:

1 1 ~~~~~~ 7 7 ⟶ 3

באופן אנאלוגי, אורך המסלול מקודקוד 1 לקודקוד 5 הוא 2 .ברור כי ה"הורה" של הקודקודים 3 ו 5 הינו קודקוד 7
ובנוסף ברור כי Used[3]=Used[5]=TRUE
כיוון שכעת מבקרים בהם .
מאחר שהשכבה של הקודקוד 7 היא 1, אז ברור ששכבתם של הקודקודים 3 ו 5 הינה 2.

כן תמונת המצב מוצגת בטבלה ובתרשים הבא :

קודקוד	1	2	3	4	5	6	7
dist	0	∞	**2**	1	**2**	2	1
p	nil	nil	**7**	1	**7**	4	1
used	1	0	**1**	1	**1**	1	1



כעת הקודקודים 3 ו 5 מתווספים לתור Q, לכן $Q=\{7,6,3,5\}$.
שים לב שבתור Q קודקודים השייכים לשכבה 1 (קודקוד 7)
וקודקודים השייכים לשכבה 2 (קודקודים 6,3,5)
מאחר שהטיפול בקודקוד 7 הסתיים נוציא אותו מהתור
ותמונת התור הינה:

Q	6	3	5
מספר שכבה	2	2	2

התהליך חוזר חלילה וחוזרים לצעד מספר 4.

<u>צעד 4</u>

4.1 $v \leftarrow 6$

4.2 קבוצת הקודקודים שהם שכנים של הקודקוד 6 ושעדין לא ביקרנו בהם היא קבוצה ריקה.
לכן לא מטפלים באף קודקוד.
מאחר שהטיפול בקודקוד 6 הסתיים נוציא אותו מהתור Q ותמונת התור הינה :

Q	3	5
מספר השכבה	2	2

התהליך חוזר חלילה וחוזרים לצעד מספר 4.

<u>צעד 4</u>

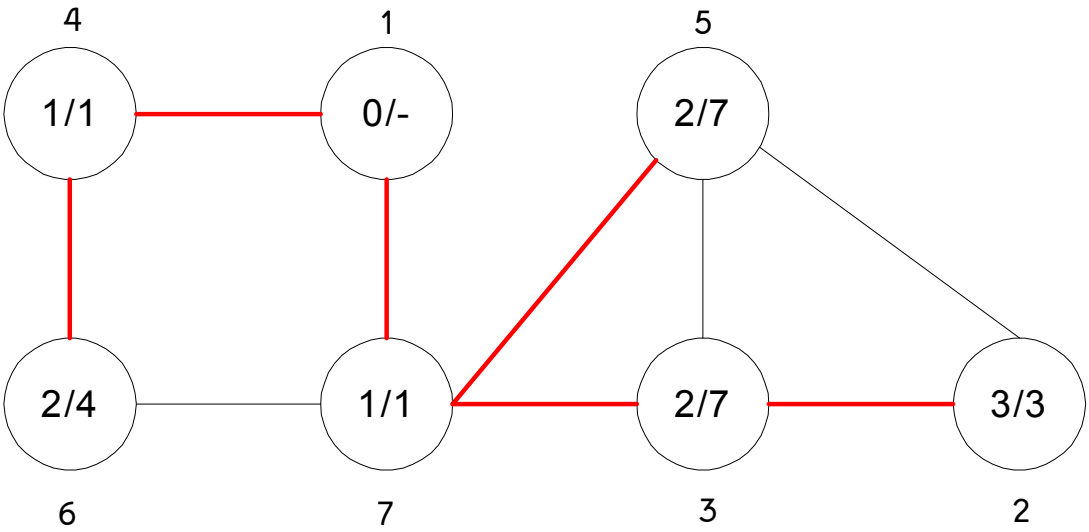
4.1 $v \leftarrow 3$

4.2 השכנים של קודקוד 3, שעדין לא ביקרנו בהם היא קודקוד 2.
ניתן לתאר את אורך המסלול מקודקוד 1 לקודקוד 2 באופן הבא :

$$1 \overset{2}{\text{- - - -}} 3 \xrightarrow{1} 2$$

לכן אורך המסלול מקודקוד 1 לקודקוד 2 הוא 3.
ברור כי ה"הורה" של הקודקוד 2 הוא קודקוד 3 ובנוסף ברור כי Used[2]=TRUE , מכיוון שהקודקוד 2,נתגלה זה עתה.
השכבה של קודקוד 3 (קודקוד האב) היא 2, לכן השכבה של הקודקוד 2 היא 3.

עתה תמונת המצב מוצגת בטבלה ובתרשים הבא:



קודקוד	1	2	3	4	5	6	7
dist	0	**3**	2	1	2	2	1
p	nil	**3**	7	1	7	4	1
used	1	**1**	1	1	1	1	1

כעת הקודקוד 2 מתווסף לתור Q לכן Q={3,5,2}.
שים לב שבתור Q הקודקודים השייכים לשכבה 2 (קודקוד 3 ו 5) וקודקודים השייכים לשכבה 3 (קודקוד 3).
מאחר שהטיפול בקודקוד 3 הסתיים נוציא אותו מהתור ותמונת התור הינה :

Q	5	2
מספר שכבה	2	3

התהליך חוזר חלילה לצעד מספר 4.

<u>צעד 4</u>

4.1 $v \leftarrow 5$

4.2 קבוצת הקודקודים שהם שכניו של קודקוד 5 ושעדין לא ביקרנו בהם היא קבוצה ריקה.
לכן לא מטפלים באף קודקוד.
מאחר שהטיפול בקודקוד 5 הסתיים נוציא אותו מהתור Q ותמונת התור הינה : $Q = \{2\}$

התהליך חוזר חלילה וחוזרים לצעד מספר 4.

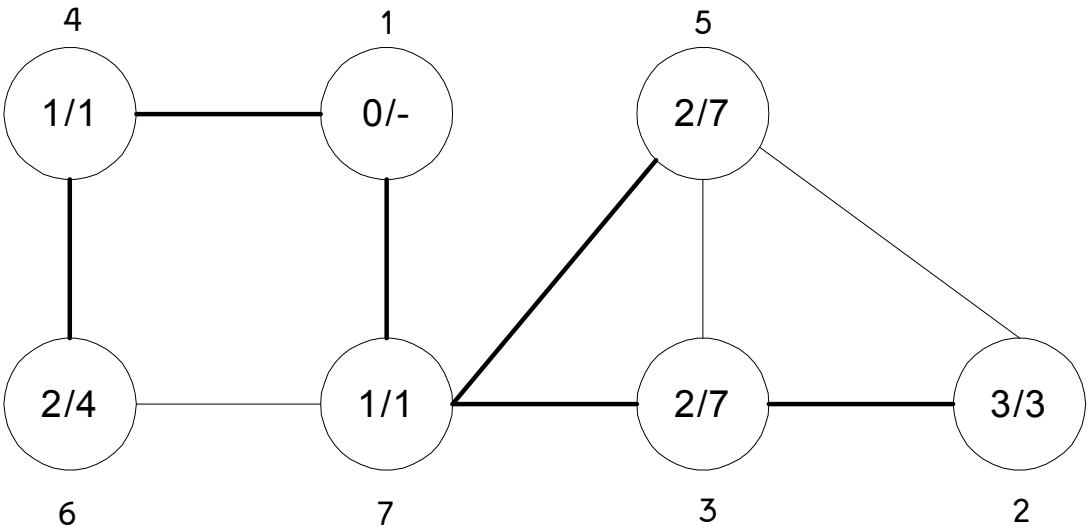
צעד 4

4.1 $v \leftarrow 2$

4.2 קבוצת הקודקודים שהם שכניו של קודקוד 2 ושעדין לא ביקרנו בהם היא קבוצה ריקה.
לכן לא מטפלים באף קודקוד.
מאחר שהטיפול בקודקוד 2 הסתיים, נוציא אותו מהתור Q והתור Q יישאר ריק.
התהליך חוזר חלילה וחוזרים לצעד מספר 4.

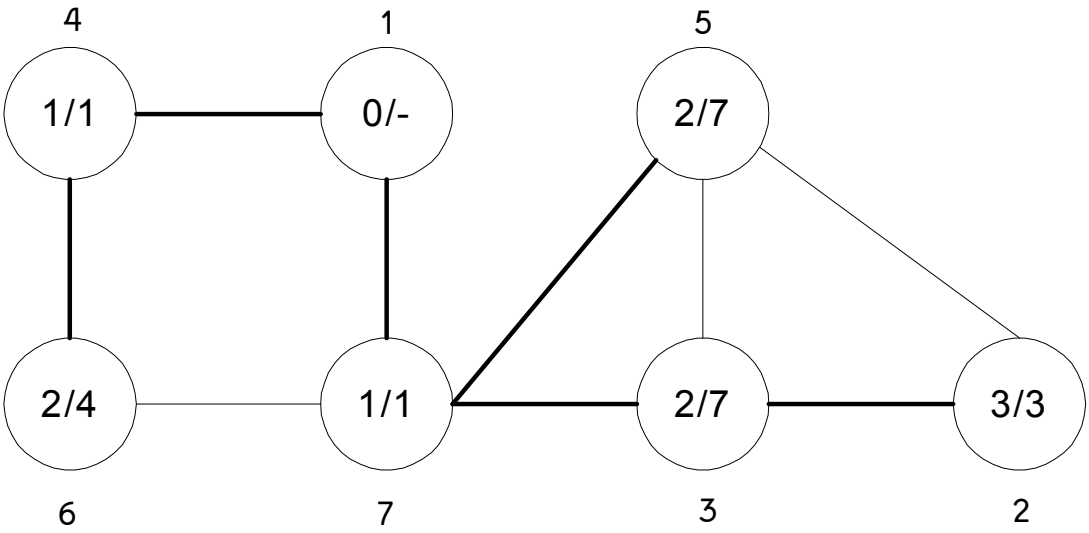
צעד 4

מאחר שהתור ריק האלגוריתם הסתיים וסופית תמונת התרשים הינה:



עץ פורש BFS

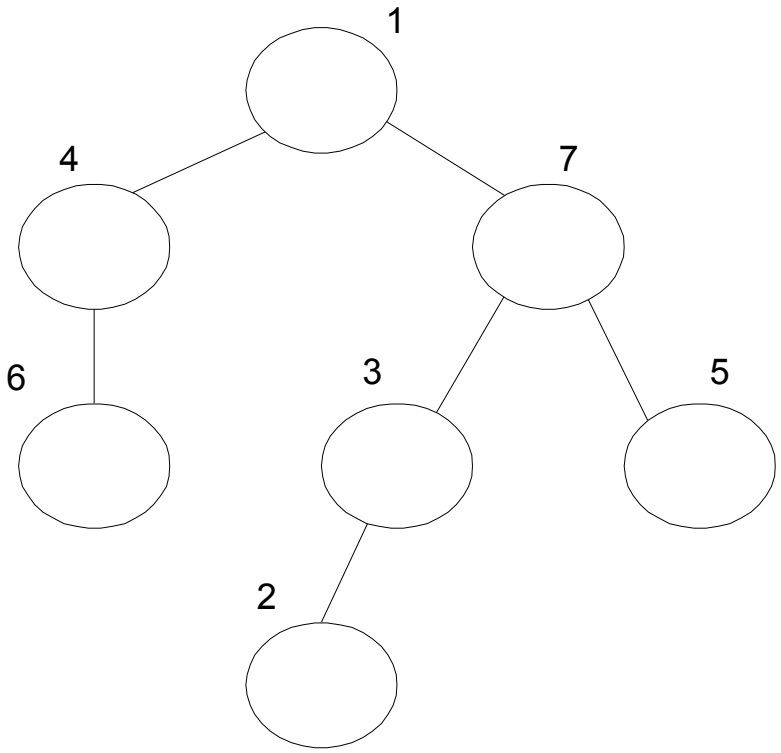
בתום ביצוע האלגוריתם תמונת המצב שנתקבלה הינה:



תוך כדי סריקת קודקודי הגרף $G=(V,E)$ נוצר תת גרף $G'=(V,E')$ כאשר E' זוהי קבוצת הקשתות שעברנו דרכן בכדי לסרוק את קודקודי הגרף בשיטת BFS. קשתות אלו מסומנות ומודגשות.

$G^{/}$ הוא עץ המכיל את כל הקודקודים הנשיגים מקודקוד מקור(S) ונראה בהמשך שלכל צומת v שנשיג מ- S, אורך המסלול הקצר ביותר מ-S ל-v נמצא בעץ הזה.

עץ פורש BFS של הגרף שראינו בהרחבה בדוגמא, הינו:



<u>טענה</u>:

תת גרף $G^{/}=\left(V^{/},E^{/}\right)$ מגדיר עץ T אשר יקרא עץ פורש של שיטת סריקה לרוחב (בקיצור-עץ פורש BFS).
זו טענה טריביאלית, כיוון שקל לראות כי $\left|E^{/}\right|=\left|V^{/}\right|-1$ והגרף $G^{/}$ הוא גרף קשיר, לכן הגרף $G^{/}$ הינו עץ.

ניתוח יעילות של האלגוריתם – סריקה לרוחב (BFS)

טרם נחקור את היעילות של האלגוריתם הנדון, ניזכר כי אם G גרף מכוון והוא מיוצג באמצעות רשימות סמיכות אזי הסכום של אורכי כל רשימות הסמיכות הוא $|E|$, ואם הגרף G לא מכוון והוא מיוצג באמצעות רשימות סמיכות אזי הסכום של אורכי כל רשימות הסמיכות הוא $2|E|$. לכן סופית סכום האורכים של רשימת הסמיכות הוא $O(|E|)$.

<u>צעד 1</u> דורש זמן $O(|V|)$, מכיוון שסורקים את כל קודקודי הגרף.

<u>צעד 2</u> דורש זמן $O(1)$, מכיוון שיש גישה ישירה לאינדקס במערך.

<u>צעד 3</u> דורש זמן $O(1)$, מיכוון שזוהי פעולה של הכנסת איבר לתור .

<u>צעד 4</u> הפעולות על התור (התייחסות לאיבר שבחזית התור והסרת איבר מהתור) דורשות זמן $O(1)$.

כל קודקוד נכנס לתור פעם אחת לכל היותר ולכן יוצא גם ממנו פעם אחת לכל היותר ולכן הזמן הנדרש לפעולות על התור הוא $O(|V|)$.

רשימת הסמיכות של כל קודקוד v נסרקת רק כאשר מסירים את הקודקוד v מהתור וברור מתוך האלגוריתם שרשימת סמיכות של כל קודקוד v נסרקת פעם אחת לכל היותר.

אך סכום האורכים של רשימות הסמיכות הוא $O(|E|)$, לכן הזמן הכולל הנדרש עבור סריקת רשימות הסמיכות הוא לכל היותר $O(|E|)$. לכן סופית,

סיבוכיות זמן הריצה של האלגוריתם הנדון היא $O(\|V\|+\|E\|)$

מסלולים קצרים

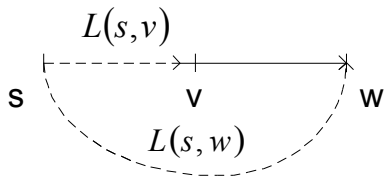
הגדרה: נתון גרף $G=(V,E)$. אורך המסלול הקצר מקודקוד מקור S לקודקוד כלשהו v בגרף מסומן כ-L(S,v) ומוגדר כמספר הקשתות המינימלי שדרכן עובר המסלול בכדי להגיע מקודקוד מקור S לקודקוד v.

הערה: אם אין מסלול מקודקוד מקור S לקודקוד כלשהו v בגרף אזי $L(S,v) \leftarrow \infty$

טענה 1

נתון גרף $G=(V,E)$ (יכול להיות מכוון או לא מכוון).
לכל קשת $(v,w) \in E$ מתקיים $L(S,w) \le L(S,v)+1$

הוכחה:
נתבונן באיור הבא:



$$L(S,w)=\min\{L(S,w),L(S,v)+1\}$$

אם אורך המסלול מקודקוד S לקודקוד w דרך הקשת (v,w) גדול יותר מאשר מבלי לעבור דרך קשת זו אזי ברור ש- $L(S,w)<L(S,v)+1$ אחרת מתקיים השוויון.

<u>טענה2</u>

נתון גרף $G=(V,E)$ ויהי S צומת מקור .נריץ את האלגוריתם $BFS(G)$.לכל קודקוד v בגרף מתקיים $dist[v]\geq L(S,v)$.

הוכחה:

נוכיח את הטענה -באינדוקציה על אורך המסלול.

<u>בסיס האינדוקציה</u> –טריביאלי כי -

ולכל קודקוד v השונה מ-s

מתקיים:

$$dist[S]=0\geq 0=L(S,S)$$
$$dist[v]=\infty\geq L(S,v)$$

<u>הנחת האינדוקציה</u>

נניח את נכונות הטענה עבור כל קודקוד w ,הסמוך לקודקוד v , כך ש$dist[w]<dist[v]$.כלומר מתקיים :

לכן מתקיים:

$$dist[w]\geq L(S,w)$$
$$dist[v]=dist[w]+1\underset{\text{לפי הנחת האינדוקציה}}{\geq} L(S,w)+1\underset{\text{לפי טענה 1}}{\geq} L(S,v)$$

הערה : יש לשים לב שבכל שלב של האלגוריתם נמצאים בתור Q לכל היותר שתי שכבות סמוכות של קודקודים.

<u>משפט</u> (ללא הוכחה)

נתון גרף $G=(V,E)$ ויהי S צומת מקור .נריץ את האלגוריתם $BFS(G)$.לכל קודקוד v בגרף מתקיים : $dist[v]=L(S,v)$.

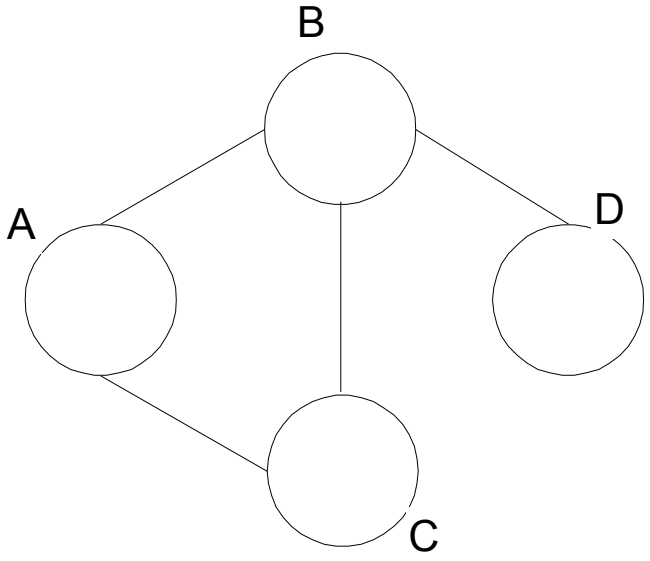
6.6 סריקה לעומק (DFS)

נתון גרף $G=(V,E)$. בשיטה זו מתחילים את הסריקה מאחד הקודקודים בגרף ונטייל על פני הקשתות מקודקוד לקודקוד של גרף , כך שנבקר בכל קודקוד של הגרף ונבצע בו עיבוד כלשהו פעם אחת בלבד.
להשגת המטרה שיטת סריקה זו תתבצע כדלהלן:
שיטת סריקה זו ,כשמו הוא , מבצעת סריקה לעומק בגרף ככל שניתן .

כזכור בשיטת סריקה BFS , אם קודקוד כלשהו בגרף $v \in V$ נשיג מקודקוד מקור S ,אז מבקרים מקודקוד v בכל קודקודי הגרף שהינם שכנים של v שעדיין לא נתגלו (לא נסרקו).

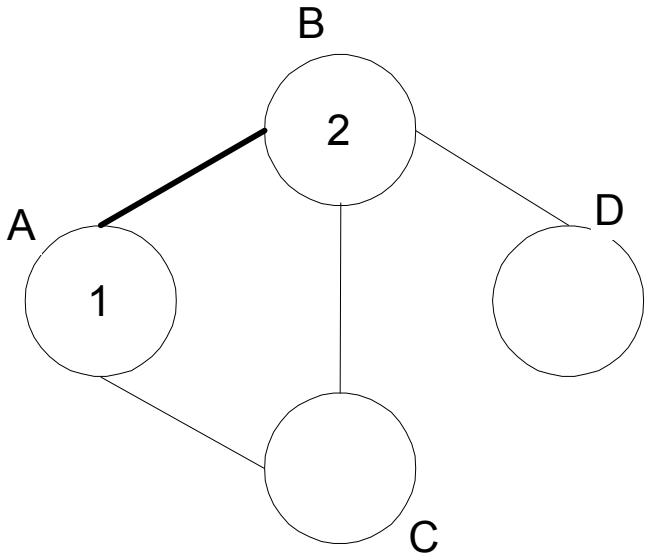
בשיטת סריקה DFS , אם קודקוד כלשהו בגרף $v \in V$ אשר נשיג מקודקוד מקור S , לא מבקרים מקודקוד v בכל קודקודי הגרף שהינם שכנים של v ושעדיין לא נתגלו (לא נסרקו).בשיטה זו בוחרים באופן שרירותי אחד הקודקודים שהוא שכן של v שעדיין לא סומן , כלומר עד כה עדיין לא ביקרו בו בעת הסריקה וממנו ממשיך תהליך הסריקה לעומק. בשלב מסוים של הסריקה כאשר מגיעים לקודקוד כלשהו v בגרף , שכל שכניו מסומנים כלומר עד כה סרקנו את כל השכנים של v , אז נסוגים ל"הורה" של הקודקוד v וממנו נמשך תהליך הסריקה. תהליך הסריקה נמשך עד שנתגלו (סרקנו)כל הקודקודים שהינם נשיגים מ-S.

נדגים את תהליך שתואר לעיל על הגרף הבא:



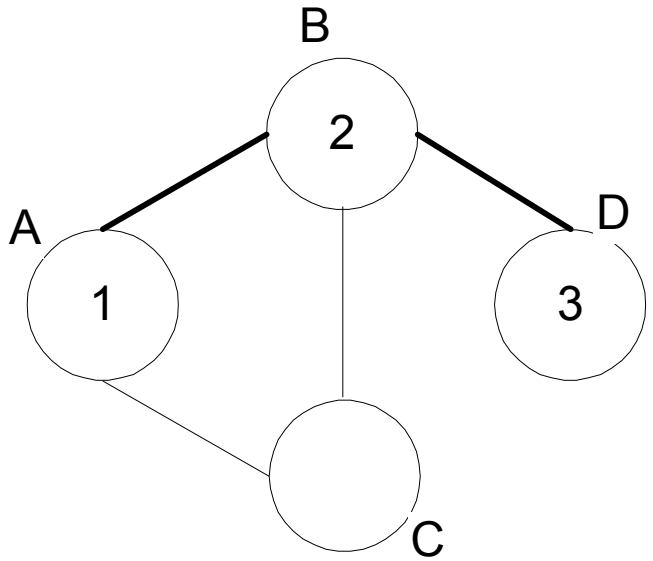
נניח שהסריקה תחל מהקודקוד A. נסמן בקודקוד A שהוא הראשון שביקרנו בו. עתה , נבחר קשת , (באופן אקראי לחלוטין) היוצאת מהקודוקד A , מבין הקשתות המובילות לקודקוד שעדיין לא ביקרנו בו.

נניח שבוחרים בקשת (A , B) אשר מובילה לקודקוד B ,שעדיין לא ביקרנו בו. נסמן בקודקוד B שהוא הקודקוד השני שביקרנו בו . תמונת המצב הינה :



עתה הקודקוד B הופך להיות מרכז הפעילות וממנו נמשיך את תהליך הסריקה . עתה נבחר קשת , (באופן אקראי לחלוטין) היוצאת מקודקוד B , מבין הקשתות המובילות לקודקוד שעדיין לא ביקרנו בו.

נניח שבחרנו בקשת (B , D) אשר מובילה לקודקוד D שעדיין לא ביקרנו בו. נסמן בקודקוד D שהוא הקודקוד השלישי שביקרנו בו . תמונת המצב הינה :



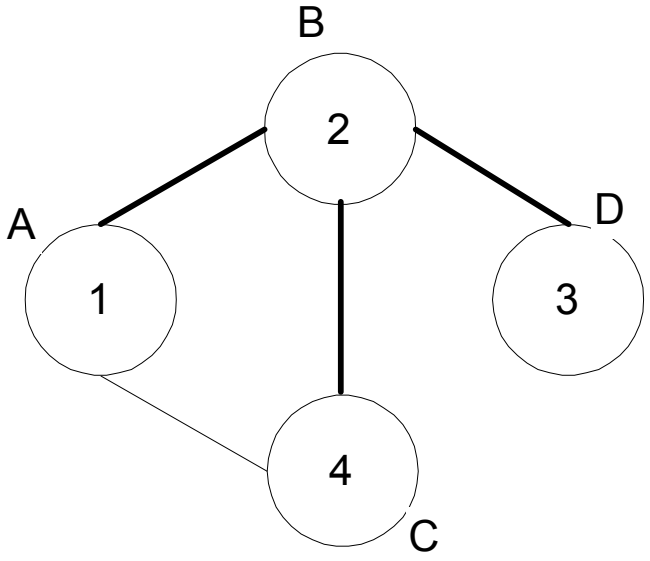
עתה ננסה לבחור קשת (כאמור באופן אקראי לחלוטין)היוצאת מהקודקוד D , מבין הקשתות המובילות לקודקוד שעדיין לא ביקרנו בו.

קל לראות שבשלב זה אין קשת כזו.לכן נסוגים מיד לקודקוד B שהינו "הורה" של קודקוד D .

מרכז הפעילות הועבר שוב לקודקוד B.

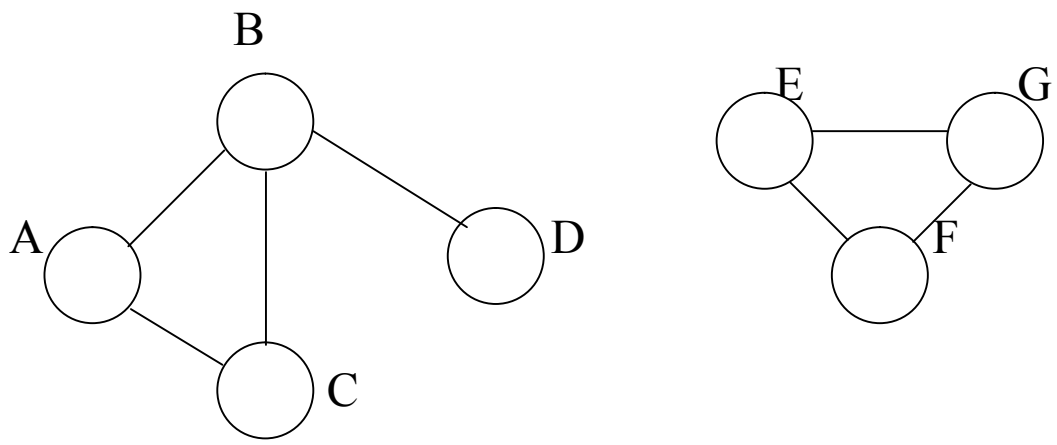
עתה נבחר קשת (באופן אקראי לחלוטין) ,היוצאת מהקודקוד B , מבין הקשתות המובילות לקודקוד שעדיין לא ביקרנו בו.

ישנה רק קשת אחת כזו והיא (B , C) , אשר מובילה לקודקוד C שעדיין לא ביקרנו בו. נסמן בקודקוד C שהוא הקודקוד הרביעי שביקרנו בו . תמונת המצב הינה :

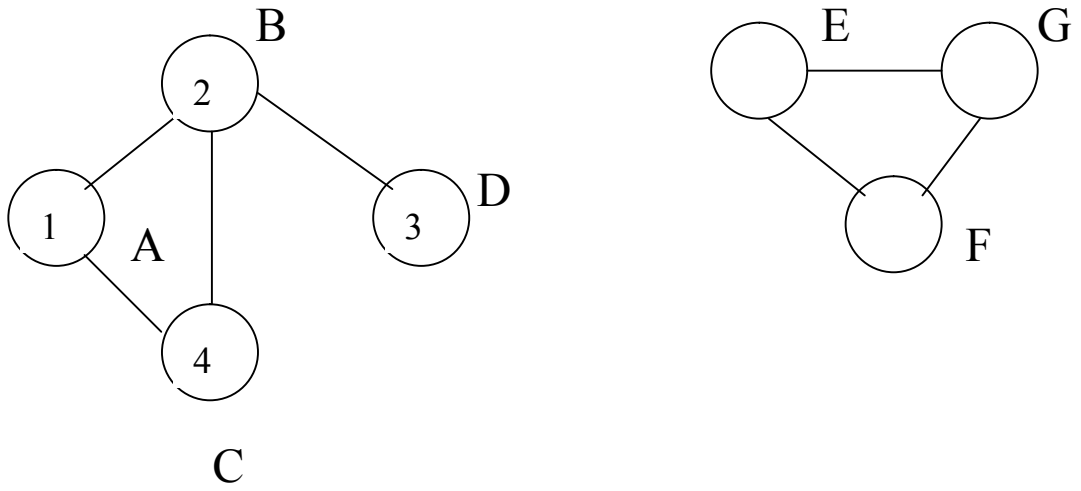


מרכז הפעילות הועבר לקודקוד C. מקודקוד C לא יוצאת אף קשת אשר תוביל לקודקוד שעדיין לא ביקרנו בו,לכן נחזור להורה של הקודקוד C שהינו קודקוד B. מקודקוד B לא יוצאת אף קשת אשר תוביל לקודקוד שעדיין לא ביקרנו בו,לכן נחזור להורה של קודקוד B שהינו קודקוד A- קודקוד ממנו התחלנו את הסריקה .מקודקוד A לא יוצאת אף קשת אשר תוביל לקודקוד שעדין לא ביקרנו בו, לכן נרצה לחזור להורה של הקודקוד A. אולם לקודקוד A אין "הורה" (אב) ואין לאן לחזור ולכן נעצור את הסריקה ובזה סיימנו את הביקור בכל קודקודי הגרף.

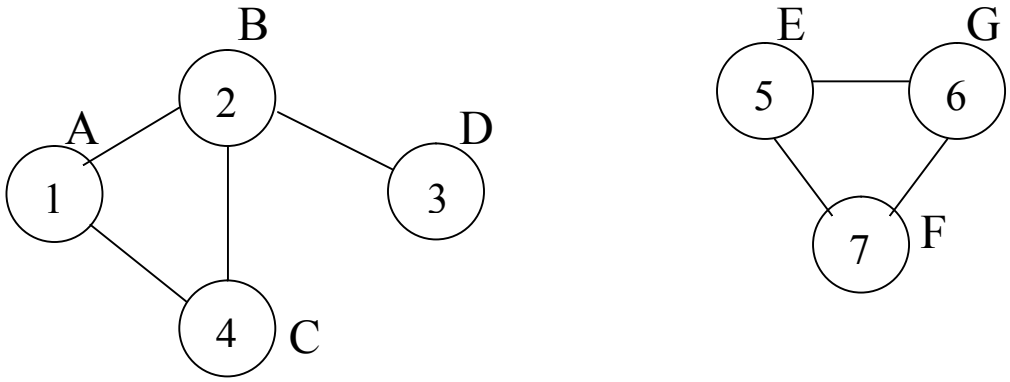
אם בתום הסריקה עדיין נותרו קודקודים שאינם נשיגים מקודקוד מקור - s , אזי קובעים מבינהם קודקוד מקור חדש ומפעילים את שיטת הסריקה DFS עד שנבקר בכל קודקודי הגרף. לדוגמא, נתבונן בגרף הבא (שהינו גרף יותר מורחב מזה שראינו קודם):



שים לב! זהו גרף אחד.
נניח שקודקוד מקור הינו קודקוד A. ראינו קודם שבתום הסריקה התמונה שנתקבלה הינה:



רואים כי עדיין לא ביקרנו בכל קודקודי הגרף. הקודקודים שעדיין לא ביקרנו בהם הם: {E, F, G}, לכן נקבע באופן שרירותי כי, מבין הקודקודים E, F, G קודקוד מקור חדש – יהיה קודקוד E. עתה מרכז הפעילות הועבר לקודקוד E.
מקודקוד E מפעילים את התהליך שתואר לעיל וקל לראות שבתום הסריקה תמונת המצב תהיה:



ומרכז הפעילות נמצא ב- E, קודקוד ממנו התחלנו את הסריקה. לקודקוד E אין "הורה" (אב) ואין לאן לחזור ולכן נעצור את הסריקה ובזה סיימנו את הביקור בכל קודקודי הגרף.

גם בשיטה זו, בדומה לשיטת סריקה BFS, נשתמש במערך בוליאני Used כך ש- Used[v] יציין האם ביקרנו בקודקוד v או לאו.

ברור שבתחילת האלגוריתם ל- Used[v], לכל קודקוד v בגרף, נציב את הערך FALSE, מכיוון שבתחילת האלגוריתם עדיין לא ביקרנו בקודקודי הגרף.

לגבי כל קודקוד חייב האלגוריתם לשמור מידע על זהות הקודקוד הקודם לו בעת הסריקה. לכן נשתמש במערך P, כך ש- P[v] ייצג קודקוד שממנו הגענו ל- v בעת הסריקה.

<u>הערה</u>: הסימון P, נבע מהסיבה ש- P[v] מייצג "הורה" (Parent) של קודקוד v בעת הסריקה.

כמו כן, לגבי כל קודקוד נרצה, אם כי הדבר לא הכרחי, למספר את קודקודי הגרף לפי סדר הביקורים (התיוג) בקודקודי הגרף.
לקודקוד הראשון בו נבקר נצמיד "תג 1",
לקודקוד השני בו נבקר נצמיד "תג 2" וכן הלאה.

לאור זאת, נשתמש באלגוריתם במערך בשם C (לכבוד המילה באנגלית (Count), כך ש- C[v] יציין את המספור שנקנה לקודקוד v בגרף בעת הסריקה.

<u>להלן האלגוריתם לשיטת סריקה לעומק, אשר מתאים לגרף לא מכוון:</u>

<u>צעד 0:</u> אפס את המשתנה i
v ← s (v מכיל קודקוד מקור)

<u>צעד 1:</u> לכל קודקוד v בגרף פרט לקודקוד המקור
Used[v] ← FALSE, (* בתחילת האלגוריתם עדיין לא ביקרנו בקודקודי הגרף *)
Used[1] ← TRUE, (* מאחר שהסריקה מתחילה מקודקוד מקור ובתחילת האלגוריתם מבקרים בו *).

<u>צעד 2:</u> i ← i+1 ו C[v] ← i (* הקניית מספור לקודקוד v *).

<u>צעד 3:</u> אם לקודקוד v אין קודקודים שכנים שעדיין לא ביקרנו בהם אז לך לצעד 5.

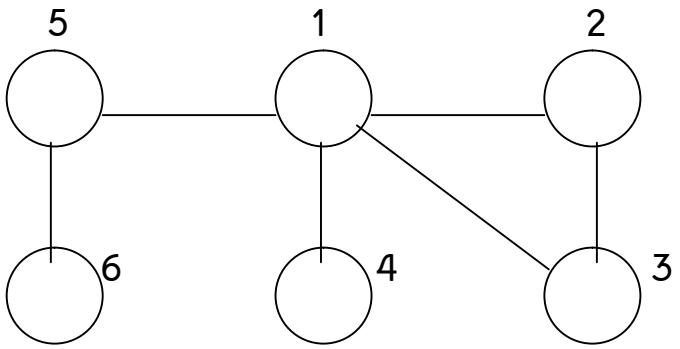
<u>צעד 4:</u> (* בנקודה זו יש לפחות קודקוד אחד u בגרף שהינו "שכן" של v ושעדיין לא ביקרנו בו. לכן נבצע את הצעדים הבאים:
4.1 P[u] ← v ו v ← u ו Used[u] ← TRUE
4.2 לך לצעד 2.

<u>צעד 5:</u> (* אם v קודקוד מקור ואין אף קודקוד שכן שלו בגרף שעדיין לא ביקרנו בו אז צריך להפסיק את האלגוריתם *).
אם C[v]=1 אזי עצור!

<u>צעד 6:</u> (* אם הקודקוד v אינו קודקוד מקור ואין אף שכן של הקודקוד v שעדיין לא ביקרנו בו, אז יש מקום לחזור ל"הורה" של v ומה"הורה" של v יש להמשיך את הסריקה *).
$v \leftarrow P[v]$ ולך לצעד 3.

<u>סוף</u>

נדגים את התהליך שתואר לעיל על הגרף הבא:



בגרף הנתון 6 קודקודים הממוספרים באופן אקראי מ- 1 עד 6, והם מצויינים בסמוך לצומת.

<u>הערה:</u> לצורך ההדגמה נייצג את הערך הבוליאני FALSE ("שקר") בעזרת 0 ואת הערך הבוליאני TRUE ("אמת") בעזרת 1.

כאמור, בתחילת האלגוריתם תמונת המערך Used הינה:

	1	2	3	4	5	6
Used	0	0	0	0	0	0

במהלך התיאור של האלגוריתם בכל קודקוד v של גרף מופיעים שני מספרים, השמאלי מייצג את המספור של הקודקוד לפי הסדר בעת הסריקה והימני מייצג את ה"הורה" של v כפי שנקבע בעת הסריקה.

נניח שקודקוד מקור הינו קודקוד 1.

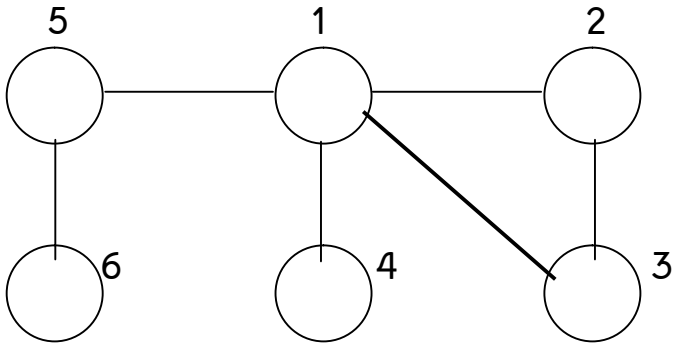
בתחילת האלגוריתם:

- (מתחילים את הסריקה מקודקוד 1) $v \longleftarrow 1$

Used

1	2	3	4	5	6
1	0	0	0	0	0

- (לא ביקרנו עדיין באף קודקוד של הגרף)
- $i \longleftarrow 0$
- הגרף הינו:



מתחילים להפעיל את האלגוריתם החל מצעד 2.

<u>צעד 2:</u> $v \longleftarrow 1$ (המספר התורן לצורך מספור הקודקודים בהתאם לסדר סריקת קודקודי הגרף).
$C[1] \longleftarrow 1$.

<u>צעד 3:</u> לצומת v, שהינו 1, <u>ישנם</u> קודקודים שכנים 2,3,4,5 שעדיין לא ביקרנו בהם.

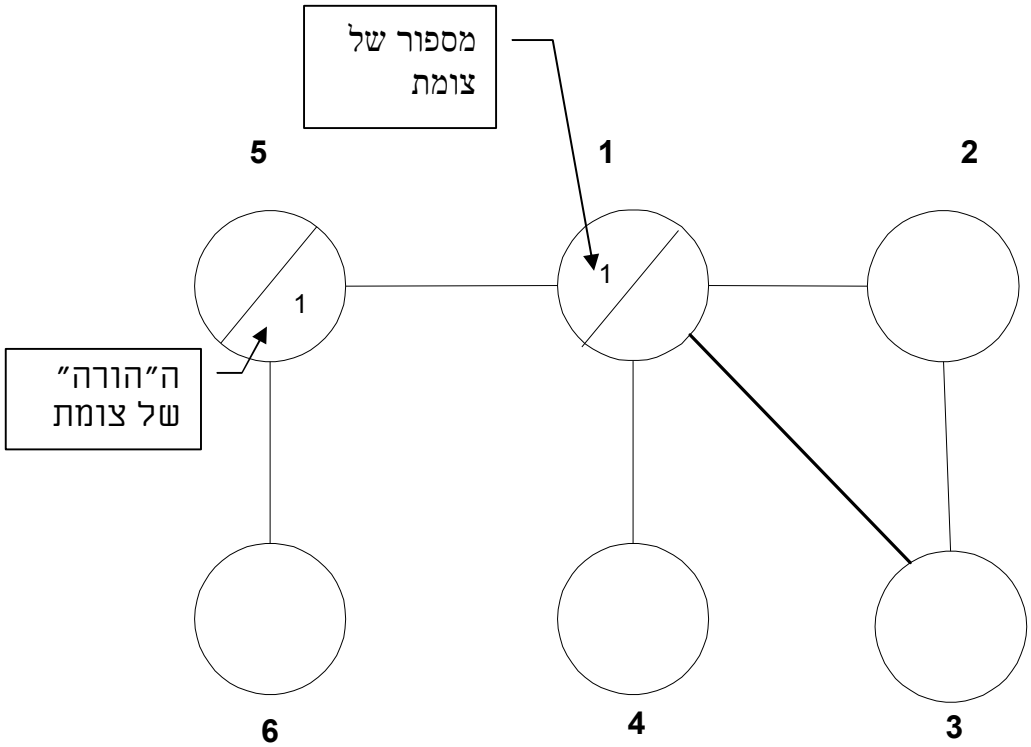
<u>צעד 4:</u> נבחר מבין הקודקודים 2,3,4,5 , כפי שצויין בצעד 3, <u>באופן שרירותי</u> לחלוטין קודקוד u=5
ונבצע את הצעדים הבאים:

4.1 P[5] ⟵ 1 (* ה"הורה" של קודקוד 5 על פי סדר הסריקה הינו קודקוד 1 *).

v ⟵ 5 (* הסריקה תימשך החל מקודקוד שמספרו 5 *).

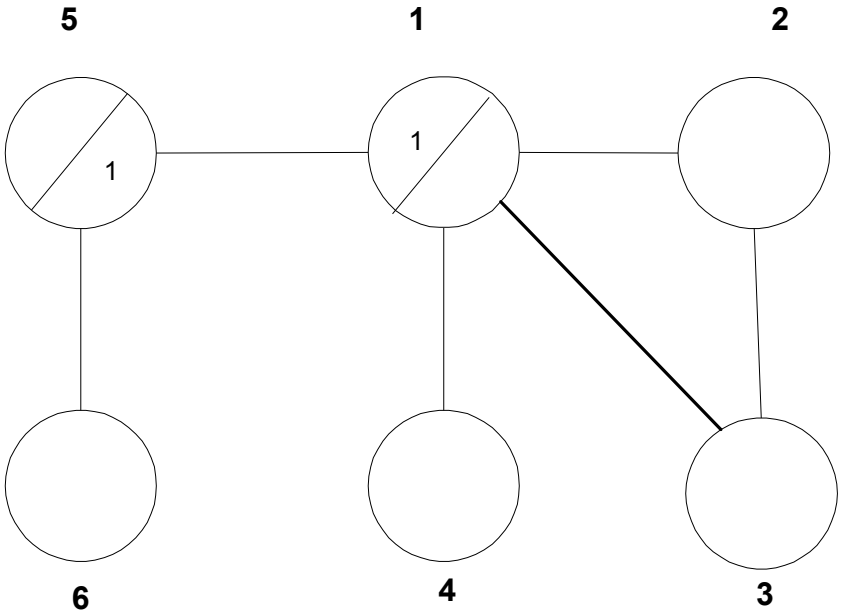
Used[5] ⟵ 1 (* כעת נתגלה 5 וסרקנו אותו *)

תוצאת המצב מוצגת בתרשים הבא:



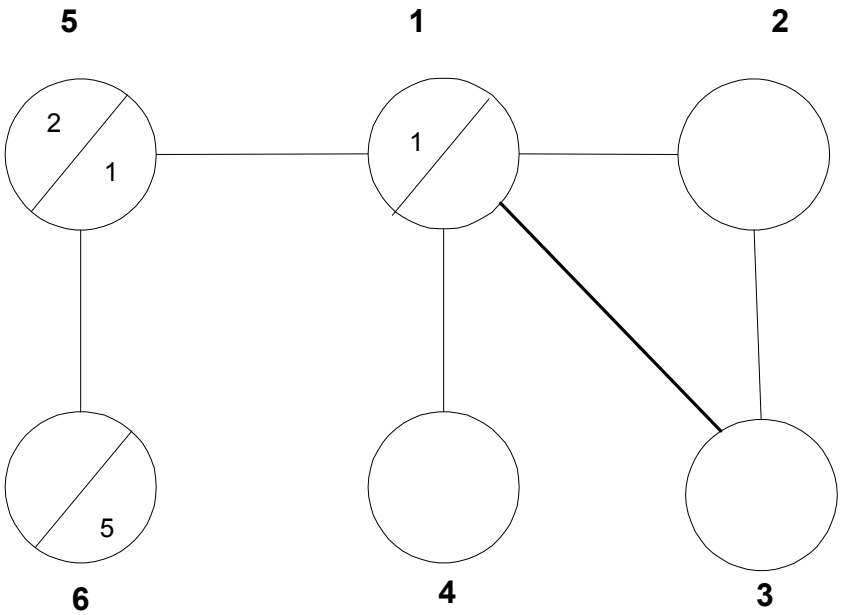
תהליך זה חוזר חלילה וחוזרים לצעד 2.

האיורים הבאים מתארים את התקדמות אלגוריתם הסריקה לעומק (DFS) של הגרף שקיבלנו עד כה



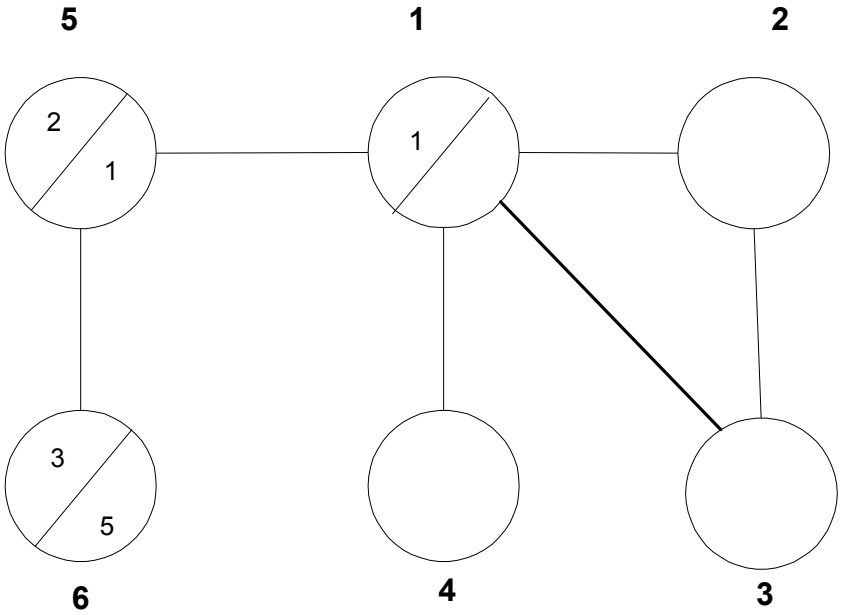
Used	1	2	3	4	5	6
	1	0	0	0	1	0

עתה i=2 ואז נקבל:



Used	1	2	3	4	5	6
	1	0	0	0	1	1

עתה i=3, נקבל:

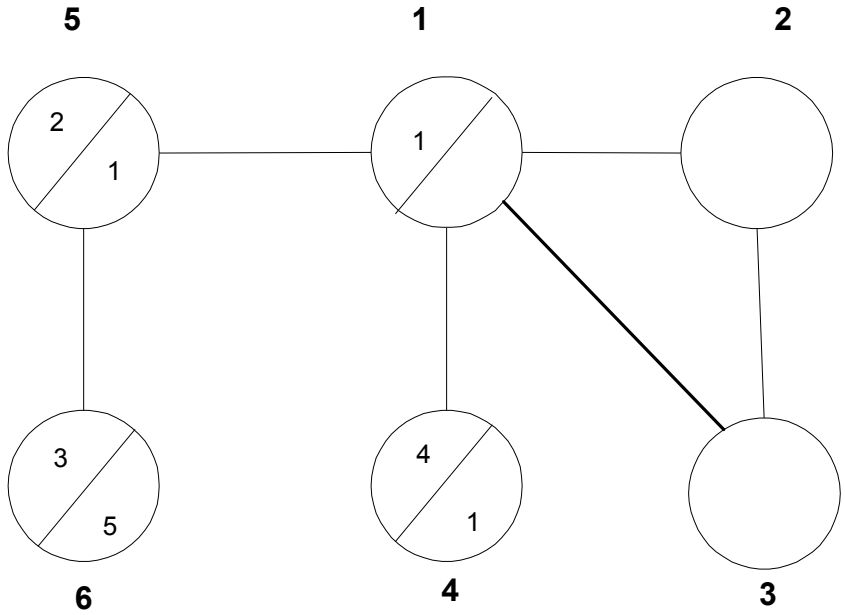


Used	1	2	3	4	5	6
	1	0	0	0	1	1

מאחר שהקודקוד v=6 אינו קודקוד מקור ואין אף שכן שלו שעדיין לא ביקרנו בו אז עלינו לסגת ל"הורה" של v=6 שהינו 5 ומהקודקוד v=5 נמשיך את תהליך הסריקה.

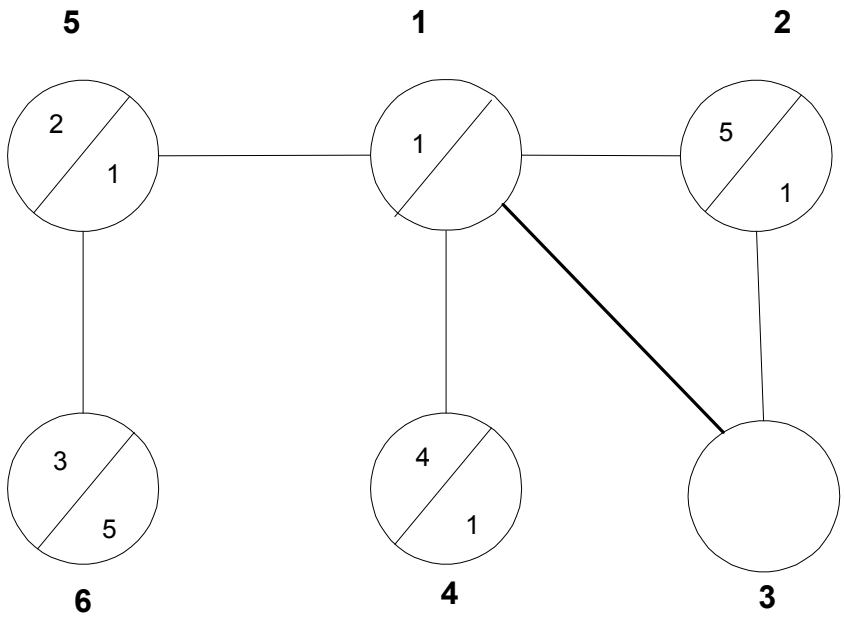
אך הקודקוד v=5 אינו קודקוד מקור ואין אף שכן שלו שעדיין לא ביקרנו בו, לכן עלינו לסגת ל"הורה" של v=5 שהינו קודקוד 1 ומקודקוד v=1 נמשיך את תהליך הסריקה.

עתה נקבל:



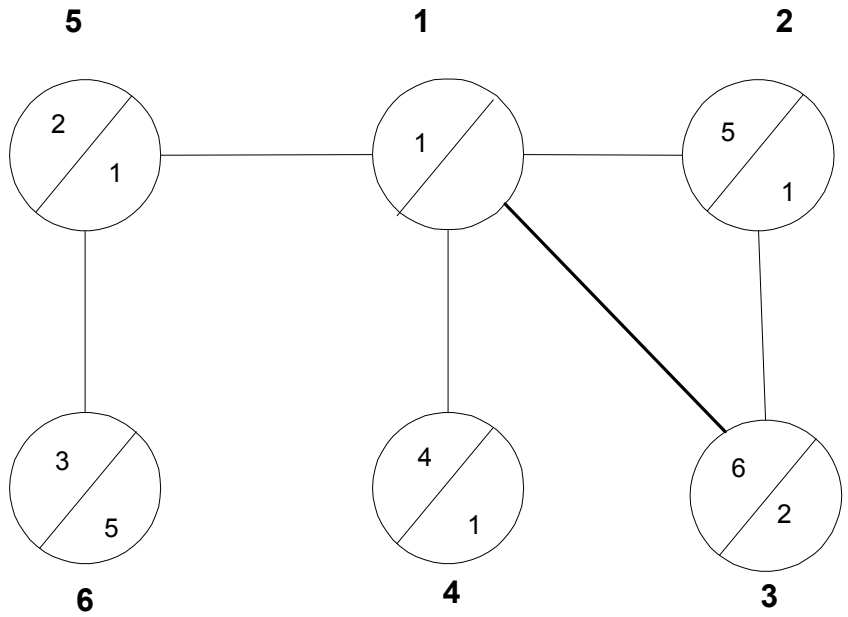
Used	1	2	3	4	5	6
	1	0	0	1	1	1

מאחר שהקודקוד v=4 אינו קודקוד מקור ואין אף שכן שלו עדיין לא ביקרנו בו, אז עלינו לחזור ל"הורה" של v=4 שהינו 1 ומהקודקוד v=1 נמשיך את תהליך הסריקה.
עתה נקבל:



Used	1	2	3	4	5	6
	1	1	0	1	1	1

ובהמשך נקבל:



Used	1	2	3	4	5	6
	1	1	1	1	1	1

עתה ברור כי תתבצע נסיגה מהקודקוד v=3 לקודקוד v=2 ומהקודקוד v=2 לקודקוד v=1. מאחר שהקודקוד v=1 הינו קודקוד מקור וכל השכנים שלו נבדקו, אז האלגוריתם לסריקה מסתיים.

עץ פורש DFS

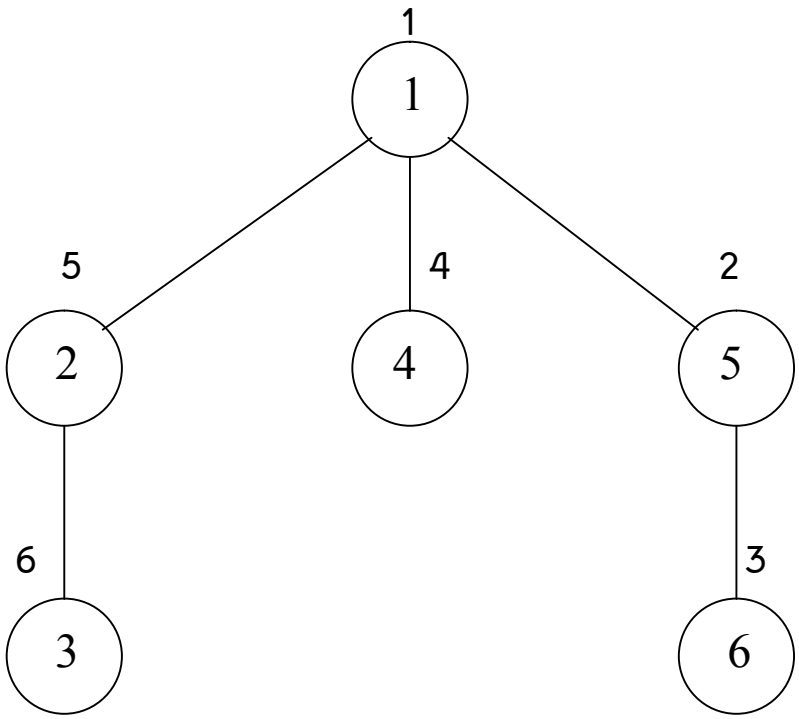
בתום סריקת קודקודי הגרף G=(V,E) נוצר תת גרף G'=(V,E') כאשר E' זוהי קבוצת הקשתות שעברנו דרכן בכדי לסרוק את קודקודי הגרף בשיטת DFS. ניתן להגדיר את E' כדלקמן:

E`= {(P(v),v) | v ε V וגם P(v) ≠ Nil }

הערה:

תת גרף G'=(V,E') עבור גרף לא מכוון G מגדיר עץ T אשר יקרא עץ פורש של שיטת סריקה לעומק (בקיצור עץ פורש DFS).

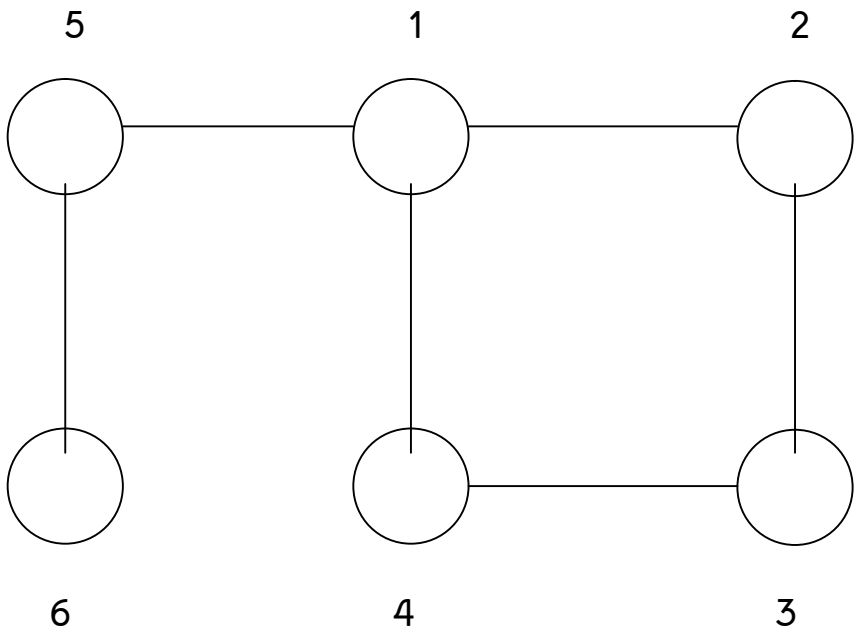
העץ הפורש של שיטת סריקה לעומק(בקיצור: עץ פורש DFS) של הגרף שראינו בהרחבה בדוגמה, הינו:



בסמוך לכל צומת מופיע מספרו של הקודקוד, ובכל קודקוד רשומים מספרים המציינים את סדר התיוג של הקודקודים בשיטת DFS.

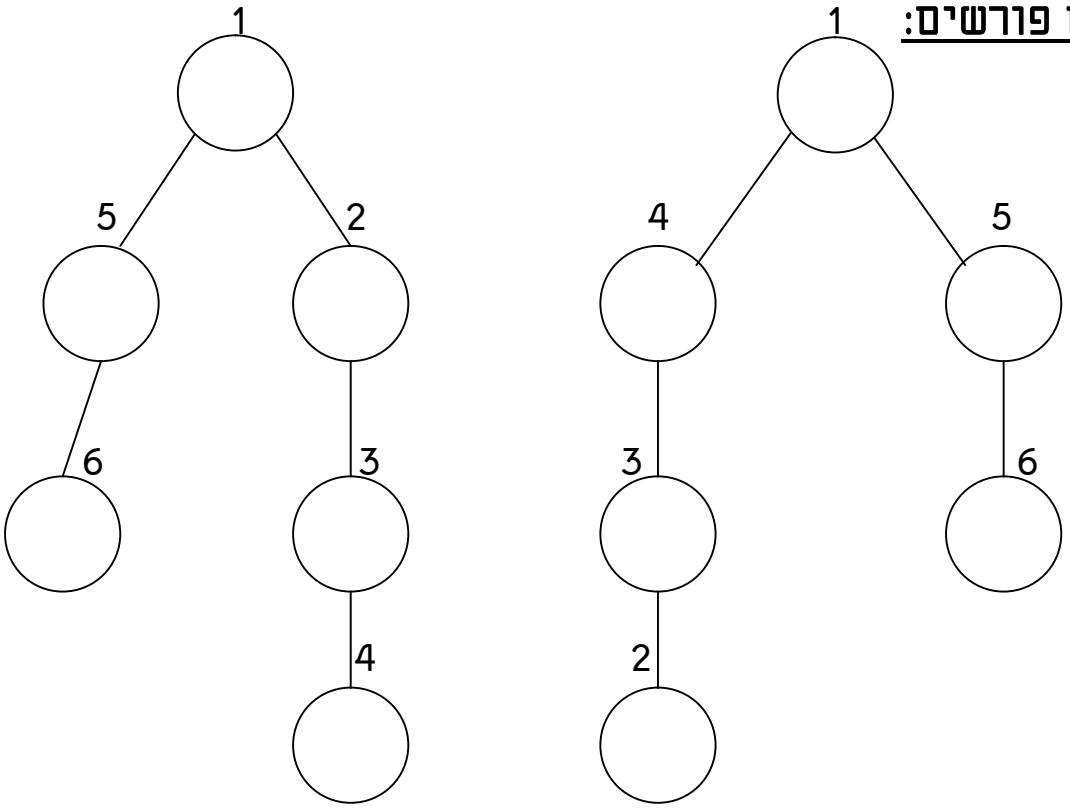
<u>הערות חשובות:</u>

א. שים לב לכך שבצעד 4 יכולנו לבחור כל קודקוד u, מבין השכנים של קודקוד v, שעדיין לא ביקרנו בו. הבחירה היא שרירותית לחלוטין, לכן סדר התיוג (או הסריקה) אינו בהכרח יחיד. לדוגמא, עבור הגרף הבא, אנו יכולים לקבל עצי פורש DFS שונים כמתואר באיור:



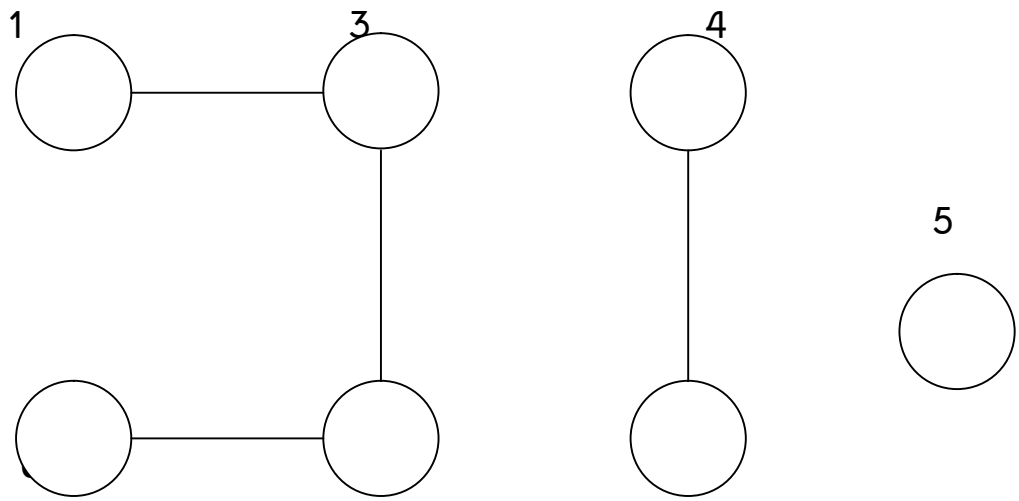
<u>גרף</u>

עצים פורשים:



וישנן עוד אפשרויות נוספות.

ב. נתבונן על הגרף הלא_קשיר הבא:



נניח שקודקוד מקור הוא v=1. קל לראות שבעזרת האלגוריתם שתואר לעיל לא נגיע לעולם לקודקודים שמספרם 5,4 ו 7.

ג. לסיכום, להלן אלגוריתם לשיטת סריקה(DFS) של קודקודי הגרף G,כאשר הגרף G הוא לא בהכרח קשיר ויכול להיות גרף מכוון או גרף לא מכוון.

להלן אלגוריתם המנסה למצוא עץ פורש DFS (נסמנו ב-T) תוך כדי סריקת קודקודי הגרף וקביעת סדר התיוג של הקודקודים:

DFS(G)

T=0

1. (*קבוצה ריקה*)
   (*בהתחלה עץ פורש – ריק*)
2. $i \leftarrow 1$
3. לכל קודקוד v בצע: $Used[v] \leftarrow FALSE$
4. סוף הלולאה.
5. כל עוד קיים קודקוד $v \in V$, שעבורו $Used[v]=FALSE$

   אזי בצע: קרא לשיגרה Search_DFS(v).
6. סוף הלולאה.

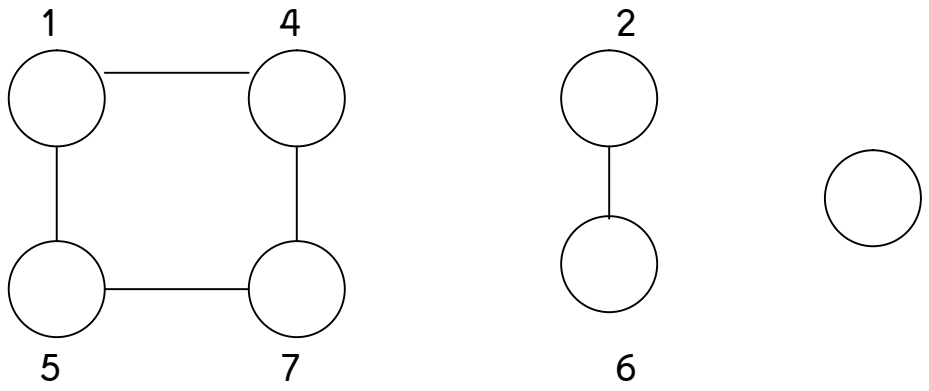
ולהלן השיגרה הרקורסיבית שכותרתה:
Search_DFS(v).

1. $Used[v] \leftarrow TRUE$
2. $C[v] \leftarrow i$
3. $i \leftarrow i+1$

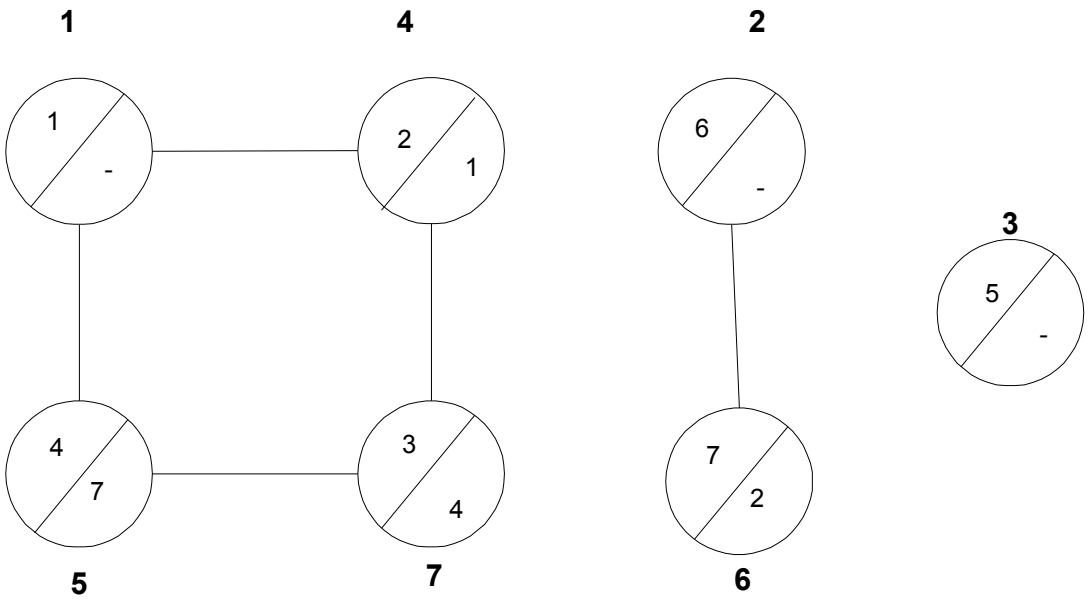
4. לכל קודקוד w שהינו שכן של קודקוד v
בצע:
4.1 אם Used(w)=FALSE?

אז בצע:
4.1.1 הוסף את הקשת (v,w) לקבוצת
הקשתות המהוות עץ פורש T.
4.1.2 P[w] ← v
4.1.3 Search_DFS(w)
5. סוף הלולאה

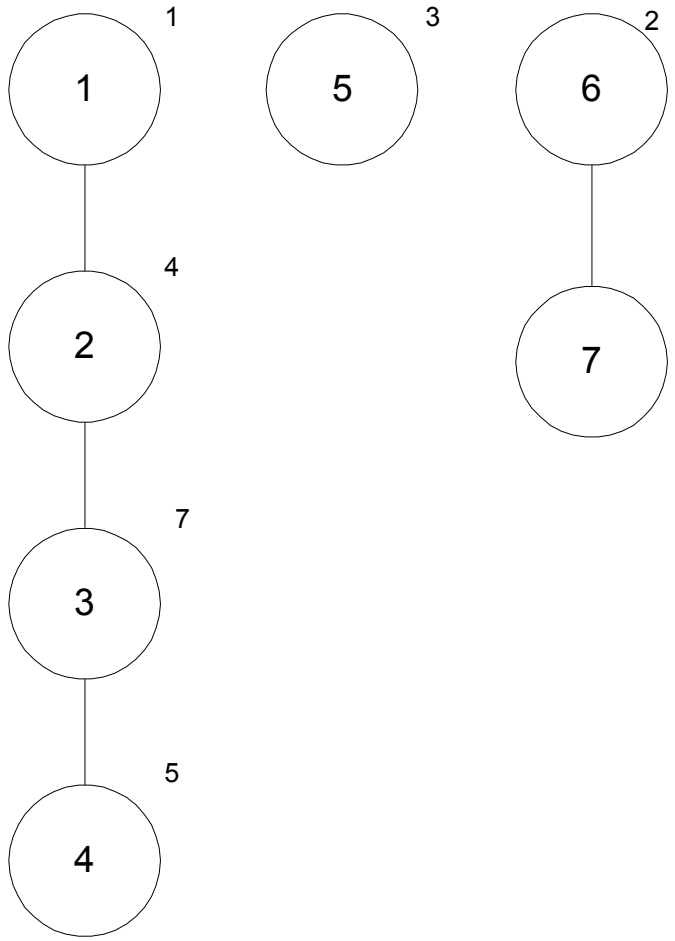
<u>דוגמא :</u>
נתון גרף G לא מכוון הבא :



נפעיל על הגרף הנתון את האלגוריתם הרקורסיבי שתואר לעיל – בשם DFS(G). קל לבדוק (בדוק!) שתמונת המצב לאחר הפעלת האלגוריתם מבוטא בתרשים הבא :



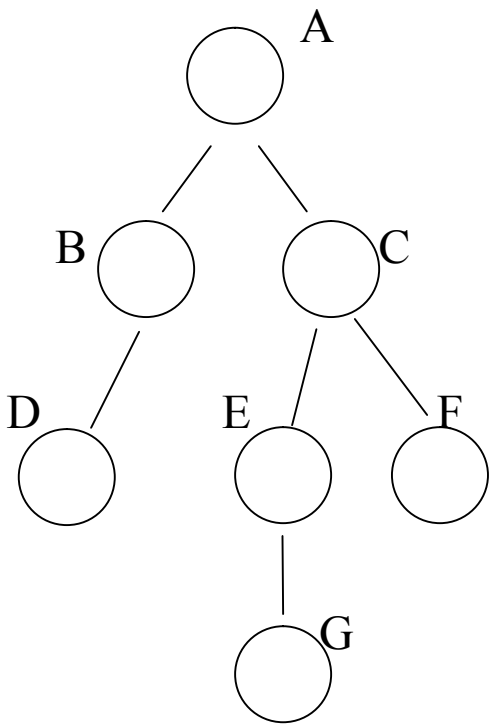
והיער הפורש, המתאים לסדר התיוג הינו :



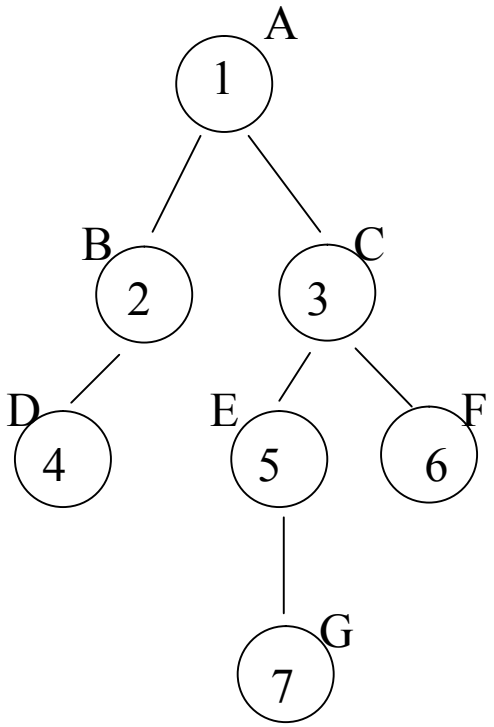
קיבלנו יער.
כאמור סדר התיוג אינו בהכרח יחיד ולכן גם היער שנקבל אינו בהכרח יחיד.

ניתוח יעילות של האלגוריתם סריקה לעומק (DFS)

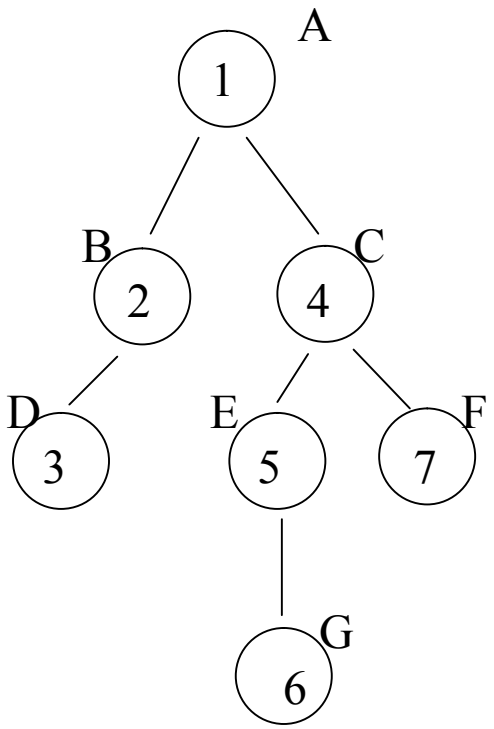
כידוע סכום האורכים של רשימות הסמיכות, המייצגת את הגרף $G=(V,E)$, הוא $O(|E|)$.
הלולאות שבשורות 3 ו 5 של השגרה DFS מתבצעות בזמן $O(|V|)$, לא כולל הזמן הנדרש לביצוע הקריאה לשגרה Search_DFS(v).
השגרה Search_DFS נקראת בדיוק פעם אחת עבור כל קודקוד $v \in V$. הלולאה המרכזית של Search_DFS(v) מתבצעת כסכום האורכים של רשימות הסמיכות ולכן השגרה Search_DFS(v) מתבצעת בזמן $O(|E|)$, ולכן סופית סיבוכיות זמן הריצה של האלגוריתם DFS הוא :
$O(|V| + |E|)$.
הערה : נתון עץ, שהינו מקרה פרטי של גרף, הבא :



לאחר הרצת BFS על העץ הנתון נקבל עץ פורש BFS הבא :



ולאחר הרצת DFS על העץ הנתון נקבל עץ פורש הבא :



קל לראות שכאשר מפעילים DFS או BFS על עץ מקבלים אותו עץ פורש, פרט לסדר הביקור בקודקודי העץ.

אלגוריתם נוסף לחיפוש לעומק

באלגוריתם זה הקודקודים נצבעים במטרה לציין את מצבם.
כל קודקוד נצבע באחד מבין הצבעים הבאים : לבן, שחור ואפור.
בתחילת האלגוריתם כל קודקוד יהיה צבוע בצבע לבן.
כאשר קודקוד מתגלה, כלומר מגיעים אליו תוך כדי הסריקה הוא נצבע בצבע אפור.
כאשר אנו מסיימים את הטיפול בקודקוד (עוזבים אותו ולא חוזרים אליו לטיפול) אז הקודקוד יצבע בצבע שחור.
נגדיר :
d[u] מציין את מועד הגילוי של הקודקוד u בעת הסריקה.
f[u] מציין את מועד סיום הטיפול בקודקוד u בעת הסריקה.
ברור כי $d(u) < f(u)$
כמו כן ברור כי הקודקוד u, לכל קודקוד $u \in V$, בצבע לבן לפני הזמן d[u], בצבע אפור בין הזמן d[u] לבין f[u] ובצבע שחור אחרי הזמן f[u].
להלן האלגוריתם :

DFS(G)

1. עבור כל קודקוד $u \in V$

בצע : 1.1 color[u] ← white
1.2 p[u] ← NULL
/* בהתחלה כל הקודקודים צבועים בצבע לבן ולאף */
/* אחד מהם אין "הורה". */
2. time ← 0
/* השעון מתחיל לתקתק */
3. עבור כל קודקוד $u \in V$ בצע :
אם color[u] הוא צבע לבן
אז בצע : DFS_Visit(u)

<u>DFS_Visit(u)</u>

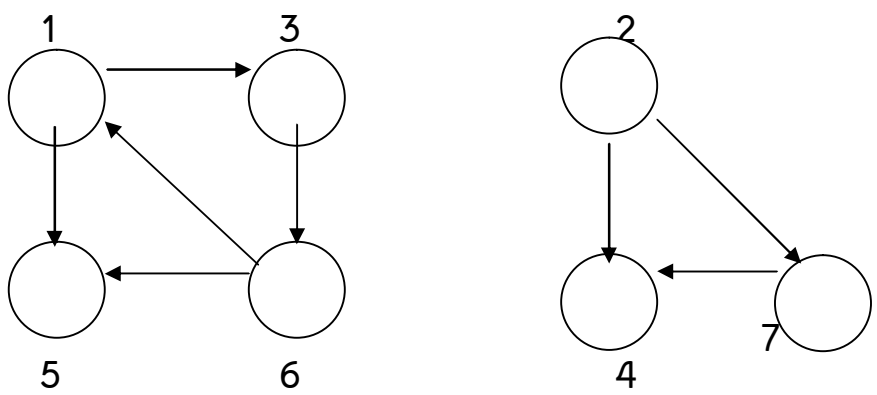
1. אפור ⟵ color[u]
2. time+1 ⟵ time
3. time ⟵ d[u]
4. עבור כל קודקוד v שהינו קודקוד סמוך ל- u בצע :

   אם color[v] הוא צבע לבן

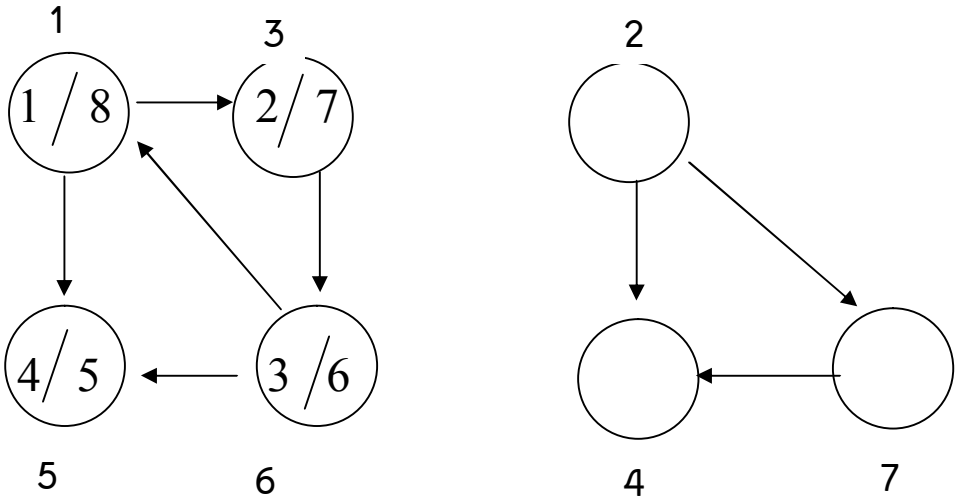
   אז בצע : 4.1 u ⟵ p[v]

   4.2 DFS_Visit(v)
5. שחור ⟵ color[u]
6. time+1 ⟵ time
7. time ⟵ f[u]

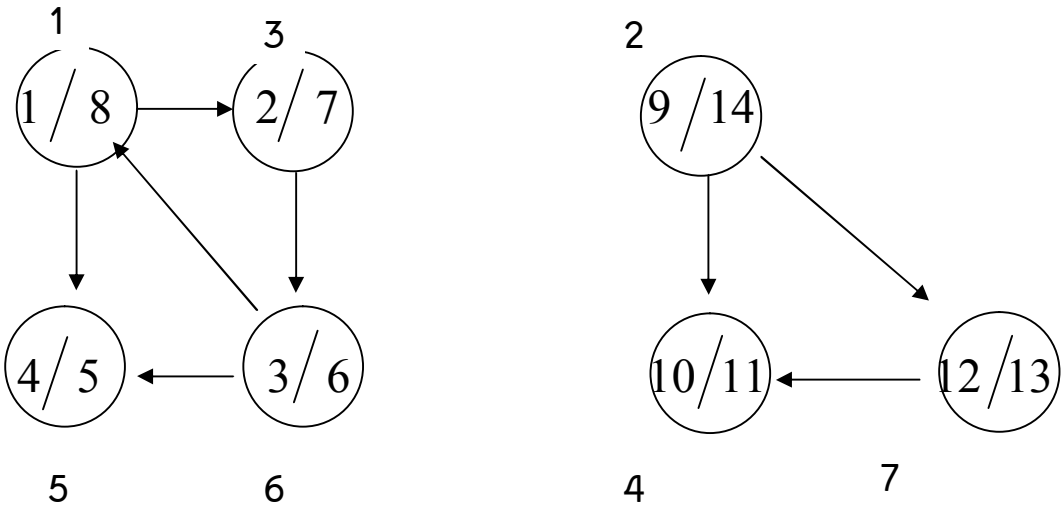
נראה את תוצאת ההרצה של חיפוש לעומק על הגרף הבא :



התוצאה היא : (מתחילים את הסריקה בקודקוד 1)



עתה האלגוריתם DFS יפעיל את הסריקה החל בקודקוד 2. נקבל :



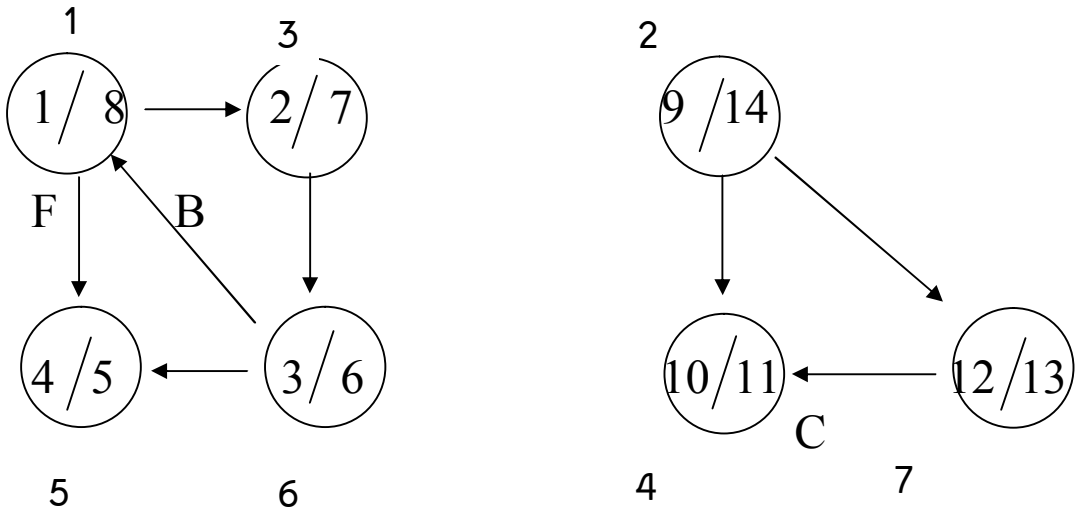
בסוף הפעלת האלגוריתם כל קודקודי הגרף צבועים בצבע שחור.

ניתן לשכתב את האלגוריתם DFS כך שהוא יוכל לסווג את הקשתות שהוא נתקל בהן.

כל קשת שנבדקת לראשונה מסווגת, על פי צבעו של הקודקוד שמגיעים אליו באופן הבא :

1. לבן מצביע על קשת העץ (עץ פורש DFS).
2. אפור מצביע על קשת אחורה.
3. שחור מצביע על קשת קדימה או חוצה.

בגרף שראינו קודם, נסווג את הקשתות ונקבל :



B מציין קשתות אחורה
F מציין קשתות קדימה
C מציין קשתות חוצות
וכל היתר הינן קשתות שמהוות עץ פורש DFS.

<u>טענה :</u> עבור גרף בלתי מכוון G בתום האלגוריתם DFS, כל קשת היא קשת עץ פורש (DFS) או קשת אחורה.
נשאיר את ההוכחה כתרגיל לקורא.

להלן השגרה המשוכתבת DFS_Visit(u)

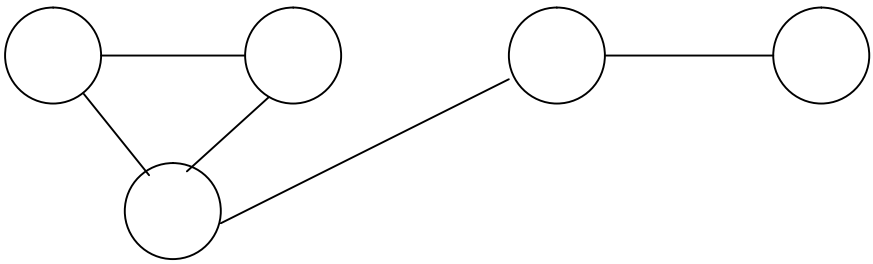
1. אפור ⟵ color[u]
2. time+1 ⟵ time
3. time ⟵ d[u]
4. עבור כל קודקוד v שהינו קודקוד סמוך ל- u בצע :
   בדוק את הצבע של הקודקוד v :
   4.1 אם color[v] הוא צבע לבן
   אז בצע : 4.1.1 סמן את הקשת (u,v) כ"קשת עץ" (tree_edge).
   4.1.2 u ⟵ p[v]
   4.1.3 DFS_Visit(v)
   4.2 אם color[v] הוא צבע אפור
   אז בצע : סמן את הקשת (u,v) כ"קשת אחורית" (back_edge).
   4.3 אם color[v] הוא צבע שחור
   אז בצע :
   אם d[v] > d[u]
   אז בצע :סמן את הקשת (u,v) כ"קשת קדימה"
   אחרת בצע :סמן את הקשת (u,v) כ"קשת חוצה"
5. שחור ⟵ color[u]
6. time+1 ⟵ time
7. time ⟵ f[u]

6.6.2 רכיבי קשירות

הגדרה : גרף לא מכוון נקרא קשיר אם בין כל שני קודקודים קיים מסלול.

הגדרה : נתון גרף לא מכוון $G = (V,E)$.
רכיב קשיר זוהי קבוצה מקסימלית של קודקודים בגרף G, כך שבין כל שני קודקודים בקבוצה זו קיים מסלול.
קודקוד בודד ללא שכנים יקרא גם כן רכיב קשיר.

דוגמא 1 : בתרשים הבא מוצג גרף קשיר.



דוגמא 2 : בתרשים הבא מוצג גרף שאינו קשיר.

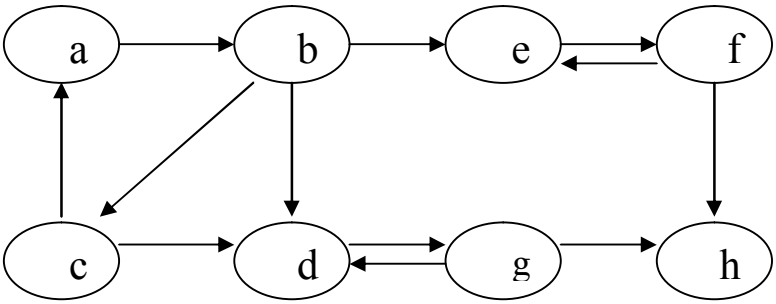
בדוגמא מס' 2 נתון גרף שבו שלושה רכיבים קשירים והם :
{ a,b,c,d } { e,f } { g } .

הגדרה - גרף מכוון יקרא "גרף קשיר חזק" אם קיים מסלול מכוון מכל קודקוד לכל קודקוד אחר .

הגדרה – נתון גרף מכוון $G=(V,E)$.

"רכיב קשיר חזק" (בקיצור רק"ח) זוהי קבוצה מקסימלית של קודקודים בגרף G, כך שבין כל שני קודקודים בקבוצה זו קיים מסלול מכוון.

<u>דוגמא</u> בתרשים הבא המתאר גרף מכוון ישנם 4 רכיבי קשירות חזקה.



הרכיבים הם:

{a,b,c}

{d,g}

{e,f}

{h}

בגרף לא מכוון G , DFS(G) מוצא את רכיבי הקשירות
ו- BFS(G,a) מוצא את רכיב הקשירות ש- a נמצא בו.
אפשר לשכתב בקלות את האלגוריתם BFS(G) כך שנוכל למצוא את כל רכיבי הקשירות.

אלגוריתם למציאת רכיבים קשירים בגרף לא מכוון

בעזרת BFS

בתהליך של סריקת צמתי הגרף בשיטת BFS נשתמש במערך בוליאני Used, כך ש Used[v] מציין האם ביקרנו בקודקוד v או לא. בנוסף נשתמש במשתנה count אשר מונה את מספר הרכיבים הקשירים.
להלן אלגוריתם המבוקש:

1. count ← 0

2. לכל קודקוד v בגרף בצע:

2.1 אם ל- Used[v] ערך FALSE אז בצע:

2.1.1 קרא לשגרה BFS[G] כאשר

קודקוד מקור הינו v.

2.1.2 count ← count + 1

הערות:

1. קל לראות שסיבוכיות זמן הריצה של האלגוריתם הינה $O(|E| + |V|)$.

2. אם ברצונינו להדפיס את כל הקודקודים שנמצאים באותו

רכיב קשיר של V, נוכל לעשות זאת בצורה רקורסיבית בעזרת פרישת עץ בשיטת inorder, או preorder, בזמן

. O(| V|)

3. נוכל לבדוק את השתייכותם של שני קודקודים כלשהם

u,v בגרף G=(V , E) לאותו רכיב קשיר בעזרת הרעיון הרקורסיבי הבא:

קודם נריץ את האלגוריתם BFS(G). וכעת אנו מוכנים למימוש הרעיון הרקורסיבי.

<u>שיגרה:</u> **קיים_מסלול**(G,v,w)?

אם v=w אזי החזר ערך "אמת" (TRUE)

אחרת

אם P[w]= 0 אזי החזר ערך "שקר" (FALSE)

אחרת **קיים_מסלול**(G,v,P[w])?

4. ברצונינו לבדוק האם קיים מסלול בין 2 צמתי גרף.
אם התשובה שלילית נרצה להדפיס הודעה שאין מסלול כזה , אחרת נדפיס את המסלול עצמו.
קודם נריץ את האלגוריתם BFS(G) ואחרי כן נפעיל את

השגרה הרקורסיבית הבאה מדפיסה את הקודקודים לאורכו של מסלול קצר ביותר מ-s ל-v:

<u>הדפס_מסלול (G,s,v)</u>

אם v==s אז הדפס את הקודקוד s
אחרת אם P[v]=nil
אז הדפס שלא קיים מסלול ועצור!
אחרת בצע:
א. **הדפס_מסלול**(G,s,P[v]) (*קריאה רקורסיבית*)
ב. הדפס את הקודקוד v .

5. באלגוריתם , למציאת רכיבים קשירים בגרף לא מכוון,

במקום המשפט "קרא לשגרה BFS(G) כאשר קודקוד מקור הינו v". ניתן לרשום את המשפט הבא :

"קרא לשגרה DFS(G) , כאשר קודקוד מקור הינו v"

בסוף האלגוריתם תוחזר אותה תוצאה.

דוגמא: נתון גרף G=(V,E) . נגדיר גרף חדש אשר יקרא גרף הפוך ומוגדר כדלקמן:

$$G^T=(V ,E^T)$$

כאשר $E^T= \{ (U,V) \mid (V,U) \in E\}$

כלומר הופכים את כיווני הקשתות . קל לכתוב אלגוריתם בעל זמן ריצה $O(|E| +|V|)$ אשר קולט את הגרף G ויחזיר את הגרף G^T. (עשה זאת !)

6.6.3

אלגוריתם למציאת רכיבי קשירות חזקה בגרפים מכוונים

Strong Connected Component (SCC)

צעד 1. מריצים DFS(G) ויוצרים רשימת קודקודים (L) אשר ממויינת בסדר יורד לפי זמני סיום הטיפול בהם.

צעד 2. הופכים את הגרף G=(V,E) ומקבלים

$$G^T=(V,E^T)$$

כאשר $E^T = \{ (U,V) \mid (V,U) \in E\}$

,כלומר הופכים את קשתות הגרף.

צעד 3. מריצים $DFS(G^T)$, כך שהלולאה המרכזית של DFS עוברת על קודקודי הגרף לפי הסדר, כפי שנקבע בצעד 1 ברשימה L.

יעילות האלגוריתם

צעד 1 דורשת זמן $O(|E|+|V|)$

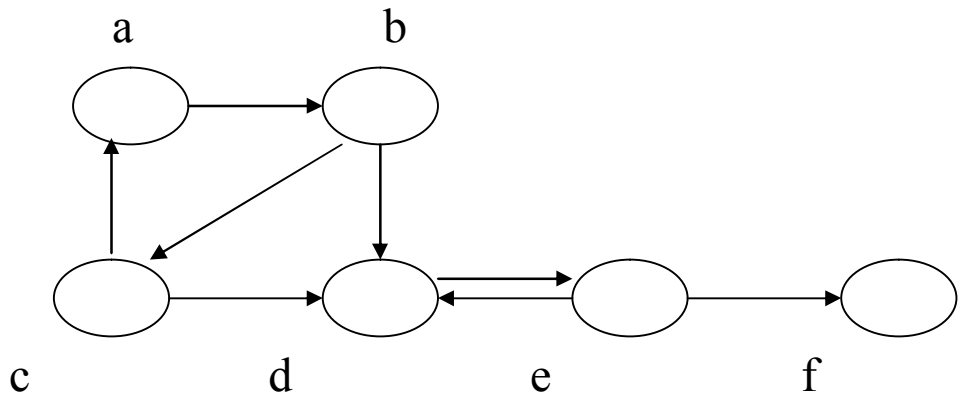
צעד 2 דורשת זמן $O(|E|+|V|)$

צעד 3 דורשת זמן $O(|E|+|V|)$

סופית: סיבוכיות זמן הריצה של האלגוריתם הנידון היא:

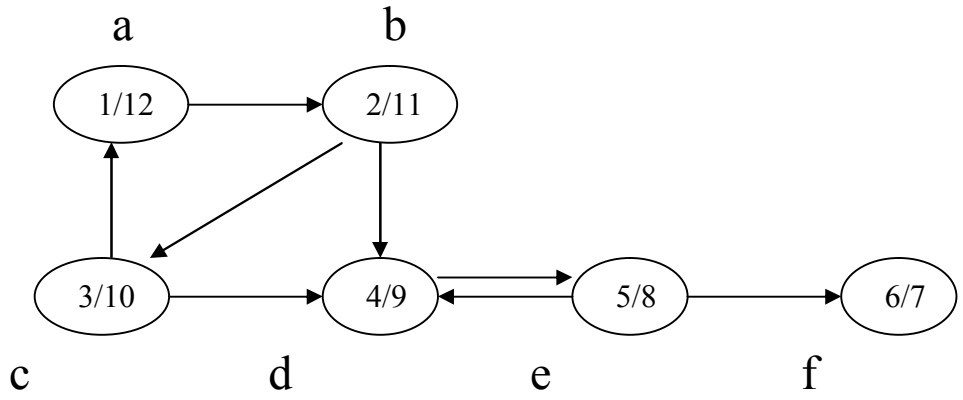
$O(|E| + |V|)$.

נדגים את אופן הפעולה של האלגוריתם (SCC) שתואר לעיל על הגרף הבא:



צעד 1

נריץ את האלגוריתם DFS(G) כאשר קודקוד מקור הינו a.
בשלב זה תמונת המצב מוצגת בתרשים הבא:

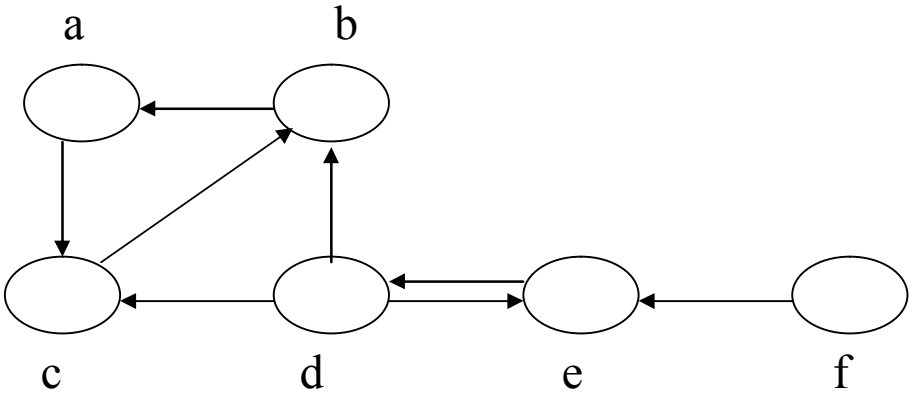


(בדוק !)

L={ a,b,c,d,e,f }

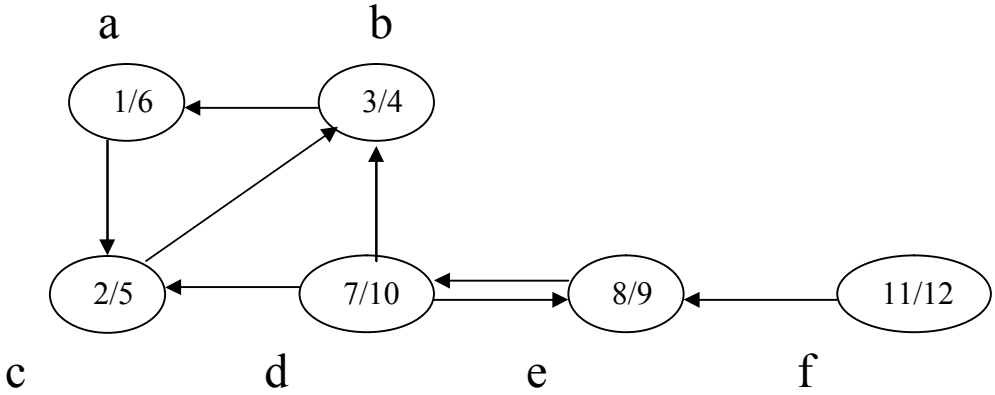
צעד 2

ניצור גרף G^T (בו קשתות הפוכות)
תמונת המצב מוצעת בתרשים הבא:

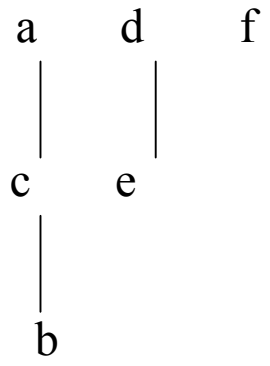


צעד 3

כעת נריץ את האלגוריתם DFS על הגרף G^T לפי הסדר
כפי שמופיע ברשימה L, אשר התקבלה בצעד 1 .
בשלב זה תמונת המצב מוצגת ברישום הבא:



עצי פרישה DFS שמתקבלים הינם:



לכן בגרף הנתון 3 רכיבים קשירים והם (כצפוי)

{a,c,b}

{d,e}

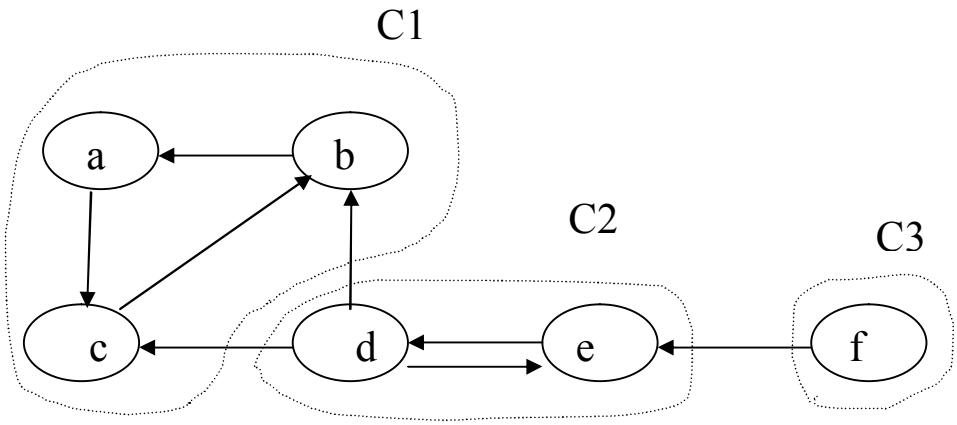
{f}

בספר זה אנו לא נעסוק בנכונות האלגוריתם המודע כיוון שהוכחת נכונותו של השיטה חורגת מן הדרישות שבספר.
בהמשך לדוגמא קודמת קיבלנו 3 רכיבי קשירות חזקה (רק"ח) והם:

$C_1=\{a,b,c\}$

$C_2=\{d,e\}$

$C_3=\{f\}$

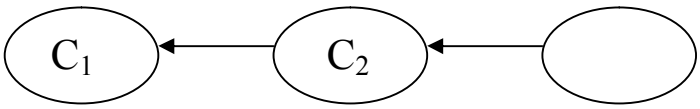


<u>הגדרה</u>: גרף העל של רק"ח הוא $G=(V,E)$ כאשר

$$V=\left\{ C_1,C_2,\ldots\ldots C_m \quad m \geq 1 \text{ ו-} \quad C_i \text{ רק"ח} \right\}$$

$$E=\left\{ (C_i,C_j) \,\middle|\, (x,y) \text{ קיימת קשת וקיימת } y\epsilon C_j \text{ ו } x\epsilon C_i \text{ קיימים בגרף } G \right\}$$

בהמשך לדוגמה האחרונה גרף העל של הגרף הנתון הוא הגרף הבא :



קל לראות כי גרף העל הוא תמיד גרף מכוון ללא מעגלים.(מדוע ?).

דוגמא 1

נתון גרף מכוון $G=(V,E)$ המיוצג על ידי רשימות שכנות.
נתאר אלגוריתם המוצא את כל הצמתים ב-G אם יש כאלה בכלל, אליהם אפשר להגיע במסלול מכוון מכל צמתי הגרף

פתרון
1- נמצא רכיבי קשירות חזקה .
2- עתה נעבור על "גרף העל" ונספור את מספר הצמתים להם דרגת יציאה 0. (חייב להיות לפחות אחד כזה).
3- אם קיים צומת אחד ,בגרף העל ,שדרגת היציאה שלו 0 אזי כל הצמתים שנמצאים ברכיב קשירות חזקה, המיוצגים באמצעות צומת זה בגרף העל, עונים לדרישת הבעיה.

4- אם קיים יותר מצומת אחד כזה אזי אין קודקודים כאלה בגרף שאפשר להגיע אליהם במסלול מכוון מכל צמתי הגרף .

עתה ננתח את סיבוכיות זמן הריצה של האלגוריתם:
צעד 1. דורש זמן $O(|E| + |V|)$.
צעד 2. דורש זמן $O(|E|)$ מקסימום.
הצעדים 4 ו 3. דורשים זמן $O(1)$ לכן , סופית סיבוכיות זמן הריצה של האלגוריתם היא $O(|E| + |V|)$.

דוגמא 2

נתון גרף מכוון $G=(V,E)$ אשר מיוצג על ידי רשימות שכנות.

נתאר אלגוריתם אשר מוצא קבוצה (אם קיימת קבוצה כזו)

$S \subseteq V$ של קודקודים כך ש: $|S| > ½|V|$, כלומר מספר האיברים בקבוצה S גדול מחצי מספר הקודקודים בגרף, ולכל $X \in S$ ו $Y \in V$ קיים מסלול מכוון מ- X ל- Y .

פתרון

1- נמצא רכיבי קשירות חזקה.

2- נבנה "גרף על" של רכיבי קשירות חזקה .

3- נספור לכל רכיב את דרגת הכניסה שלו.

4- אם יש יותר מרכיב אחד שדרגת הכניסה שלו 0 אזי עצור.
הודעה: <u>לא קיימת קבוצה כזו</u>.

5- אם קיים קודקוד אחד " בגרף על " המייצג רכיב אחד עם דרגת כניסה 0 אז הקבוצה המבוקשת S תכיל את כל הקודקודים שברכיב קשיר זה.עתה נספור את מספר הקודקודים (K) בקבוצה S, ואחרי כן נבדוק :

אם $K>|V|/2$ אז נודיע : קיימת קבוצה כזו.

אם $K\leq|V|/2$ אז נודיע :לא קיימת קבוצה כזו.

קל לראות כי סיבוכיות זמן הריצה של האלגוריתם הזה היא : $O(|E| + |V|)$.

<u>הערה</u> :

אם יש שני רכיבים שונים שדרגת הכניסה שלהם אפס אזי לא קיימת קבוצה S כזו כך שיתקיים : לכל $X \in S$ ו $Y \in V$ יהיה קיים מסלול מכוון מ- X ל- Y.

עתה אם יש רכיב אחד בדיוק שדרגת הכניסה שלו היא 0 אזי ניתן להגיע ממנו לכל רכיב אחר בגרף. הכיצד?

נצא מכל רכיב , אליו רוצים להגיע , ונלך "אחורה" (בכיוון המנוגד לכיוון הקשתות) בקשתות "גרף העל ".

לא נוכל "לטייל אחורה" בגרף העל לנצח כיוון שגרף העל הוא גרף אציקלי (לא מעגלי), ומספר הרכיבים הקשירים

הוא סופי ולכן המקום היחיד שנעצר הוא הקודקוד (נסמנו ב- a) בגרף העל שדרגתו 0.
הקודקוד a "בגרף העל" מייצג את רכיב קשירות חזקה והוא קבוצת קודקודים בגרף- G .
כאמור ברכיב קשירות חזקה זו ניתן להגיע מכל קודקוד לכל קודקוד . רכיב זה מייצג את הקבוצה המבוקשת – S .
ועתה ראינו שמקבוצת הקודקודים ב- S ניתן להגיע לכל קודקוד אחר בגרף.