

אלגוריתמים – פתרון תרגיל 2

1. תארו אלגוריתם למציאת כל הגשרים בגרף לא מכוון, המשתמש באלגוריתם שראינו בתרגול לכיוון גרף לא מכוון חסר גשרים לגרף קשיר בחזקה.

תיאור האלגוריתם:

- נכוון את הגרף באמצעות DFS כפי שלמדנו בתרגול.
- נשתמש באלגוריתם שלמדנו בכיתה לחישוב רק"חים בגרף מכוון G .
- בתום אלגוריתם החישוב נקבל יער שבו כל עץ הוא רכיב קשירות. נעבור על רשימת הצמתים, ונחפש צמתים שדרגת הכניסה אליהם היא 0 (צמתים אלו הם למעשה שורשי העצים).
- בכל העץ, נעדכן את כל הצמתים להצביע על השורש כנציג העץ (ע"י הרצת סריקת עץ מכל שורש).
- נעבור על כל הקשתות בגרף ונוסיף לרשימת הגשרים כל קשת המחברת 2 צמתים עם נציג שונה (צמתים מעצים שונים).
- בתום סריקת הקשתות, נחזיר את רשימת הגשרים.

הוכחת נכונות:

קשת מנתקת בגרף \Leftrightarrow היא מחברת בין 2 רכיבי קשירות חזקה \Leftarrow תהי קשת (u,v) קשת מנתקת. נניח בשלילה ש- (u,v) אינה מחברת 2 רכיבי קשירות. כלומר, (u,v) נמצאת ברכיב קשירות יחיד. מכאן, שקיימת מסלול גם מ- v ל- u . אך זאת בסתירה לכך ש- (u,v) מנתקת.

\Rightarrow תהי קשת (u,v) המחברת בין 2 רכיבי קשירות בגרף. נניח בשלילה שזו אינה קשת מנתקת. כלומר, היא נמצאת על מעגל. אם זו קשת אחורית ב-DFS, אז קיים את המסלול המקורי מ- v ל- u ואם זו קשת קדמית, אז כיוון שהיא נמצאת על מעגל, חייבת להיות קשת אחורית מ- v או צאצא שלו אל u . כלומר, ניתן להגיע מ- u אל v ומ- v אל u ולכן u ו- v באותו רכיב קשירות חזקה. זאת בסתירה לכך ש- (u,v) מחברת בין 2 רכיבי קשירות חזקה.

ניתוח סבוכיות:

- כיוון הגרף באמצעות DFS – $O(|V|+|E|)$
- חישוב רק"חים בגרף מכוון – $O(|V|+|E|)$
- חיפוש שורשים – $O(|V|)$
- עדכון השורש כנציג העץ בצמתים – $O(|V|+|E|)$
- סריקת הקשתות לחיפוש גשרים – $O(|E|)$

סה"כ סבוכיות זמן הריצה של האלגוריתם – $O(|V|+|E|)$.

* מ.ש.ל

2.

א. הוכיחו שקודקוד שמשותף במספר רכיבי דו-קשירות הוא קודקוד מנתק בגרף.

הוכחה:

נניח בשלילה שהקודקוד אינו מנתק.

נניח שקודקוד זה (נסמנו ב- u) מחבר את רכיב הדו-קשירות C_1 עם רכיב הדו-קשירות C_2 .

כיוון ש- u אינו קודקוד מנתק בגרף, קיים קודקוד נוסף v המחבר בין C_1 ו- C_2 .

נשים לב:

1. כל קודקוד שנסיר מ- C_1 לא ינתק את הגרף (כי C_1 הוא דו-קשיר).
2. בצורה דומה, כל קודקוד שנסיר מ- C_2 לא ינתק את הגרף.
3. ע"פ ההנחה בשלילה, v ו- u גם אינם מנתקים את הגרף.

סה"כ קיבלנו כי כל קודקוד ב- $\{u, v\} \cup C_1 \cup C_2$ אינו קודקוד מנתק. כלומר, C_1 ו- C_2 שייכים לאותו רכיב דו-קשירות בסתירה להנחה שאלו 2 רכיבי דו-קשירות נפרדים.

* מ.ש.ל

- ב. אלגוריתם למציאת רכיבי דו-קשירות פועל בדומה למציאת קודקודים מנתקים. אלא שעכשיו, לכל u שאינו שורש בודקים את בנו. לכל v בן v אשר $low(v)$ גדול או שווה $d(u)$ נפצל את u - עותק אחד עבור הקשת (u, v) ועותק אחד עבור השאר. עבור השורש נתאים עותק לכל בן. הוכיחו את נכונות האלגוריתם.

הוכחה:

קודקוד מפוצל ל-2 באלגוריתם \Leftrightarrow הוא אינו נמצא בתוך רכיב דו קשירות יחיד

\Leftarrow נניח שקודקוד מפוצל ל-2 באלגוריתם. מצב זה קורה כאשר $d[u] \leq low(v)$ (ובתנאי שהקודקוד אינו שורש). כאשר $d[u] < low(v)$ אזי $d[v] = low(v)$ (כיוון ש- v בן ישיר של u). במצב כזה, אין למעשה קשת אחורית מ- v או צאצא שלו אל u או אב קדמון שלו. לכן, הקשת (u, v) במקרה זה היא גשר ולכן לא ייתכן שתהיה בתוך רכיב דו קשירות יחיד. כאשר $d[u] = low(v)$ אז ל- v או לאחד הצאצאים שלו, יש קשת אחורית המגיעה אל u בלבד (ואין קשתות המגיעות לאב קדמון). לכן קודקוד זה הוא קודקוד מנתק, ולכן, לא ייתכן שהוא נמצא בתוך רכיב דו קשירות יחיד.

\Rightarrow נניח שקודקוד u מחבר 2 רכיבי דו קשירות, ע"פ סעיף א' קודקוד המחבר 2 רכיבי דו קשירות הוא קודקוד מנתק. לכן, אין קשת מצאצא שלו אל אב קדמון שלו. כלומר $d[u] \leq low(v)$ (כאשר v הוא בן ישיר של u) ולכן קודקוד זה יפוצל ל-2 כנדרש.

* מ.ש.ל

3. תארו וריאציה של האלגוריתם למציאת קודקודים מנתקים, המוצאת את כל הגשרים בגרף לא מכון.

תיאור האלגוריתם:

נניח שהגרף קשיר. אחרת נוכל להריץ את האלגוריתם על כל רכיב קשירות בנפרד. נריץ את האלגוריתם לחישוב low עבור כל קודקוד בגרף (באמצעות DFS) כמוגדר באלגוריתם למציאת קודקודים מנתקים. לאחר חישוב ה- low בכל הקודקודים, נסרוק את הגרף שוב באמצעות DFS החל מאותו קודקוד בו התחלנו את הסריקה בפעם הקודם. בכל קשת שנעבור ואינה קשת אחורית (הצבע של קודקוד-הבן הוא לבן), נבדוק אם בצומת הבן v מתקיים $d[v] = low(v)$. אם כן, נוסיף את הקשת לרשימת הגשרים B. בסיום הסריקה נחזיר את B.

הוכחת נכונות:

(u, v) (קשת מכוננת ב DFS) היא גשר בגרף $\Leftrightarrow d[v] = low(v)$

\Leftarrow תהי קשת (u, v) גשר בגרף. (u, v) היא גשר בגרף ולכן אין קשת אחורית מ- v או צאצא של v ל- u או לאב של u (אחרת היינו יכולים לנתק את (u, v) והגרף היה נשאר קשיר). בשל כך, כאשר ננסה לחשב את $low(v)$, המינימום של $low(w)$ (בן ישיר של

(v) יהיה $d[v]$ (כי אין קשתות אחוריות לאבות של v). כמו כן, ל-v אין קשתות אחוריות. לכן, ה-min מבין שלושת האפשרויות יהיה בוודאות $d[v]$ ונקבל $low(v)=d[v]$ כנדרש.

\Rightarrow יהי קשת (u,v) (מכוונת בDFS), כך ש- $d[v]=low(v)$. נניח בשלילה ש- (u,v) אינה גשר בגרף. אם (u,v) אינה גשר בגרף, קיימת קשת נוספת המחברת את v או צאצא שלו ל-u או אב קדמון שלו. זוהי למעשה קשת אחורית מ-v או צאצא של v לאב קדמון של u (או u עצמו), אשר נסמן אותו כ-w. $d[w]<d[v]$. כיוון ש-w הוא אב קדמון של u ולכן גם אב קדמון של v. לכן, כשנסה לחשב את $low(v)$, לאחד מבניו יהיה בוודאות ערך low הקטן או שווה ל- $d[w]$. לכן, נקבל כי $low(v)<d[v]$ בסתירה להנחה שהערכים שווים.

ניתוח סבוכיות:

א. סריקת DFS ראשונה וחישוב ערכי low של הצמתים (נלמד בכיתה) – $O(|V|+|E|)$.

ב. סריקת DFS שנייה ובדיקת הקשתות שבדרך – $O(|V|+|E|)$.
(נשים לב שבדיקת כל קשת במהלך ה-DFS והוספתה לרשימת הגשרים בעת הצורך לוקחת זמן קבוע $O(1)$).

סה"כ סבוכיות זמן ריצת האלגוריתם – $O(|V|+|E|)$.

מ.ש.ל

4. נתון גרף מכוון $G=(V,E)$. תארו אלגוריתם יעיל שמוצא את כל הקודקודים שמכל אחד מהם יש מסלול לכל קודקוד בגרף.

תיאור האלגוריתם:

א. תחילה, נמצא את גרף העל של G:

1. נשתמש באלגוריתם שלמדנו בכיתה לחישוב רק"חים בגרף מכוון G.
2. בתום האלגוריתם נקבל יער שבו כל עץ הוא רכיב קשירות. נעבור על כל הקשתות ביער ונסמן את העץ אליו נכנסת כל קשת.
3. לאחר מכן, נעבור על רשימת הצמתים, ונעתיק לגרף חדש G_{SCC} צמתים שלא סומנו (צמתים אלו הם למעשה שורשי העצים).
4. נעדכן את שורשי העצים להיות נציג כל עץ בכל הצמתים של העץ, ע"י הרצת סריקת עץ על כל אחד מהצמתים שהעתקנו לגרף החדש.
5. לסיום, נעבור על כל הקשתות מהגרף המקורי. כל קשת המחברת 2 צמתים עם נציג שונה בגרף המקורי נייצג כקשת בין הנציגים בגרף G_{SCC} .

ב. אם בגרף העל קיים רק צומת אחד, נחזיר את כל צמתי הגרף.

ג. אם יש יותר מצומת אחד בגרף העל, נעבור על רשימת הצמתים בגרף ונספור את כמות השורשים בגרף (צמתים בעלי דרגת כניסה 0). אם יש רק שורש יחיד, מחזירים את כל הצמתים שברק"ח אותו מייצג הצומת. אם יש יותר משורש אחד, מחזירים רשימה ריקה.

הוכחת נכונות:

- כפי שלמדנו בשיעור, כל עץ ביער המתקבל לאחר הרצת אלגוריתם לחישוב רק"ח, מייצג רכיב קשירות חזקה. הצמתים היחידים ביער להם אין אב הם שורשי העצים (מעצם הגדרת שורש). לכן, השימוש בשורש העץ כמייצג הרק"ח מוגדר היטב.

- כל קשת המחברת 2 צמתים עם נציגים שונים, היא קשת המחברת בין 2 רק"חים שונים. אחרת, ל-2 הצמתים היה אותו השורש ולכן אותו נציג. כמו כן, לא קיימות קשתות נוספות המחברות בין רק"חים כיוון שאנו סורקים את כל הקשתות בגרף

וקשתות המתברות 2 צמתים עם אותו נציג, מחברות למעשה 2 צמתים הנמצאים באותו הרק"ח.

- אם בגרף העל קיים רק צומת אחד, הרי הגרף הוא גרף קשיר בחזקה. לכן, מכל צומת אפשר להגיע לכל צומת ומכאן שיש להחזיר את כל הצמתים.
- אם בגרף העל יש שורש יחיד, ניתן ממנו להגיע לכל צומת (ע"י ירידה כלפי מטה בעץ). לכן, היות ובתוך כל רק"ח ניתן להגיע מכל צומת פנימי לכל צומת פנימי, מהצמתים אשר ברק"ח המיוצג ע"י השורש, ניתן להגיע לכל צומת אחר בגרף (ע"י מעבר בגרף העל לרק"ח המתאים ובתוכו אל הצומת הרצוי).
- היות ובגרף העל אין מעגלים, אם קיימים בגרף העל שני שורשים (c_1, c_2) הרי שצמתים מ- c_1 לא יוכלו להגיע לצמתים מ- c_2 ולהפך. כמו כן, צמתים ברק"חים אחרים, לא יוכלו כמובן להגיע לצמתים מ- c_1 או c_2 (כי אין אליהם קשתות נכנסות). לכן, במקרה זה אין אף צומת ממנה ניתן להגיע לשאר הצמתים ויש להחזיר רשימה ריקה.

ניתוח סבוכיות:

- א.
1. אלגוריתם חישוב רק"חים - $O(|E|+|V|)$
 2. סריקת קשתות וסימון צמתים - $O(|E|)$
 3. סריקת צמתים והעתקת צמתים רלוונטיים ל- G_{SCC} - $O(|V|)$
 4. עדכון נציג צומת (ע"י אלגוריתם לסריקת גרף של כל עץ) - $O(|E|+|V|)$
 5. סריקת קשתות וייצוג קשתות רלוונטיות ב- G_{SCC} - $O(|V|)$
- סה"כ סבוכיות זמן ריצה של שלב זה באלגוריתם - $O(|E|+|V|)$

ב. בדיקה האם קיים רק צומת אחד בגרף העל והחזרת כל הצמתים (אם צריך) - $O(1)$

ג.

1. מעבר על רשימת הצמתים בגרף העל וספירת השורשים - $O(|V|)$
2. החזרת כל הצמתים ברק"ח של השורש היחיד (במידת הצורך) - $O(|V|)$
3. החזרת רשימה ריקה (במידת הצורך) - $O(1)$

סה"כ סבוכיות זמן ריצה של שלב זה באלגוריתם - $O(|V|)$

סה"כ סבוכיות זמן ריצה של האלגוריתם - $O(|E|+|V|)$

* מ.ש.ל

5. גרף מכוון $G=(V,E)$ ייקרא "חצי קשיר בחזקה" אם לכל $u, v \in V$ יש מסלול מכוון מ- u ל- v או מסלול מ- v ל- u . תארו אלגוריתם יעיל הבודק אם גרף מכוון הוא חצי קשיר בחזקה.

תיאור האלגוריתם:

- א. תחילה, נמצא את G_{SCC} - גרף העל של G (כמו בתרגיל הקודם).
- ב. עתה, נוודא שבגרף העל, כל הצמתים מסודרים באופן סריאלי. כלומר, קיים צומת אחד יחיד ללא אב וצומת אחד יחיד ללא בן. ליתר הצמתים קיים בן אחד בדיוק.
 1. לכל צומת בגרף העל נוסיף שדה "דרוג" ודגל "צומת-בן". נעדכן את ערכי ה"דרוג" ע"י הרצת מיון טופולוגי ועדכון הערכים בסדר רץ על פי תוצאת המיון ואת דגלי ה"צומת-בן" ל-"0".

2. נסרוק את כל הקשתות בגרף העל. אם קשת מסוימת מחברת 2 צמתים בעלי דרוג עוקב, נעדכן את הדגל של הבן (זה עם הדרוג הגדול יותר) להיות "1". נשים לב שכיוון שבגרף העל אין מעגלים, לצומת היעד של כל קשת תמיד יהיה ערך גדול יותר מצומת המוצא.
3. בתום סריקת הקשתות, נסרוק את כל הצמתים בגרף העל. ונוודא שקיים צומת יחיד (ולא יותר) שבו הדגל "צומת-בן" עדיין 0 (זהו השורש בשרשור הסריאלי).

הוכחת נכונות:

- הוכחת נכונות של גרף העל – כמו בתרגיל הקודם.
- בגרף העל לא קיימים מעגלים, לכן, ניתן להריץ מיון טופולוגי. כאשר צמתים מסודרים באופן סריאלי, לכל צומת בשרשרת יש בדיוק בן אחד ואב אחד, פרט לצומת הראשון (אשר אין לו אב) ולצומת האחרון (אשר אין לו בן). לכן, במידה והצמתים אכן מסודרים כך, המיון הטופולוגי ימספר את הצמתים החל מהשורש ועד לצאצא האחרון במספור עולה (כלומר, תהיה קשת המחברת בין כל 2 צמתים בעלי דרוג עוקב). מצד שני, אם נניח שהצמתים אינם משורשרים סריאלית, חייב להיות נתק בשרשרת. במקרה כזה, לאחר המיון הטופולוגי יהיה לפחות זוג צמתים עוקב שאינו מחובר בקשת.
- אם הצמתים אכן מסודרים באופן סריאלי, נוצר למעשה מסלול/שרשרת בין ה"רק" חים. היות ובתוך כל "רק" ח אפשר להגיע מכל צומת לכל צומת, סידור "רק" חים בצורה זו מוודא שקיים לפחות מסלול בכיוון אחד בין כל 2 צמתים. כיוון שבגרף על לא ייתכנו מעגלים, זהו המצב היחיד בו מתקיים תנאי זה.

ניתוח סבוכיות:

א. ע"פ ניתוח בשאלה הקודמת – $O(|E|+|V|)$

ב.

1.

- i. אתחול הדגל "צומת-בן" – $O(|V|)$
- ii. מיון טופולוגי ואתחול ערכי ה"דרוג" בצמתים ע"פ המיון – $O(|V|+|E|)$
2. סריקת הקשתות בגרף העל ועדכון דגל במקרה הצורך – $O(|E|)$
3. סריקת צמתים ב G_{SCC} כדי לוודא שיש בו יתום יחיד – $O(|V|)$

סה"כ סבוכיות זמן ריצה של שלב זה באלגוריתם – $O(|E|+|V|)$

סה"כ סבוכיות זמן ריצה של האלגוריתם – $O(|E|+|V|)$

• מ.ש.ל

6. יהי $G=(V,E)$ גרף לא מכוון וקשיר, ו- $w:V \rightarrow R$ פונקציה משקל על הקודקודים. תארו אלגוריתם למציאת עץ פורש T , המביא למינימום את $\sum_{v \in V} d_T(v) \cdot w(v)$, כאשר $d_T(v)$ זו הדרגה של v ב- T .

תיאור האלגוריתם:

נשתמש באלגוריתם למציאת ע"פ"מ ע"פ משקלי הקשתות כדי לפתור בעיה זו. לשם כך, נגדיר פונקציה משקל לקשתות אשר תסכום את ערכי הצמתים אותם היא מחברת $w'[(v,u)] = w(v) + w(u)$. באמצעות פונקציה משקל זו נחפש עץ פורש מינימאלי T ע"פ האלגוריתם של קרונסקל. לאחר מציאת T , נחזיר אותו כיוון שהוא מביא למינימום את $\sum_{v \in V} d_T(v) \cdot w(v)$.

הוכחת נכונות:

$$w'(T) = \sum_{(v,u) \in T} w'((v,u)) = \sum_{(v,u) \in T} w(v) + w(u) = \sum_{v \in V} d_T(v) \cdot w(v)$$

תחילה נשים לב כי המעברים הראשונים נובעים ישירות מהגדרות משקל עפ"מ והגדרת פונקציית המשקל w . המעבר האחרון נכון כיוון שכל הופעה של המשקל $w(v)$ בסכום $\sum_{(v,u) \in T} w(v) + w(u)$ היא כתוצאה מקשת כלשהי שמחוברת אליו בעץ T .

נשים לב שלפי תכונות עפ"מ, T הוא עץ פורש כך ש- $W'(T)$ שלו הוא מינימאלי מבין כל העצים הפורשים של הגרף. ע"פ אי השוויון שהוכחנו קודם, נקבל כי בעץ זה גם הסכום $\sum_{v \in V} d_T(v) \cdot w(v)$ הוא מינימאלי מבין כל העצים הפורשים כנדרש.

ניתוח סבוכיות:

הגדרת פונקציית המשקל (וחישובה עבור כל קשת) מתבצע ב- $O(1)$. לכן, סבוכיות אלגוריתם קרוסקל לא תשתנה. כיוון שלמעשה אנו רק מריצים את האלגוריתם של קרוסקל עם פונקציית המשקל שהגדרנו, זמן ריצת האלגוריתם הוא כזמן ריצת אלגוריתם קרוסקל, כלומר $O(|E| \log |E|)$.

מ.ש.ל

7. נתון גרף לא מכוון וקשיר $G=(V,E)$, פונקציית משקל $w: E \rightarrow R$, וקשת $(u,v) \in E$. תארו אלגוריתם לינארי שבודק האם קיים עץ פורש מינימלי של G שמכיל את הקשת (u,v) .

תיאור האלגוריתם:

נגדיר גרף חדש $G'=(E',V')$ כך ש- $V'=V$ ו- $E'=\{e \in E \mid w(e) < (u,v)\}$. על הגרף G' נפעיל את אלגוריתם למציאת המסלולים הקצרים ביותר החל מקודקוד u (הרצת BFS החל מ- u). אם קיים מסלול בין u ל- v בגרף G' , אזי הקשת (u,v) לא יכולה להופיע בעפ"מ. אחרת, קיים עפ"מ המכיל את הקשת (u,v) .

הוכחת נכונות:

קיים עפ"מ המכיל את $(u,v) \Leftrightarrow$ אין מסלול בין u ל- v בגרף G'

\Leftarrow אנו מניחים כי קיים עפ"מ המכיל את (u,v) . נניח בשלילה שיש מסלול בגרף G' בין u ל- v . היות וכל הקשתות ב- G' הן קשתות שמשקלן קטן מהמשקל של (u,v) , אם קיים מסלול בין u ל- v ב- G' הרי שמצאנו דרך "זולה" לחבר בין u,v . הסיבה לכך, היא שהאלגוריתם של קרוסקל מסתכל קודם על הקשתות הקלות, לכן אם קיים מסלול קצר בין u ל- v , קרוסקל יקשר בין u ל- v לפני שיטפל בקשת (u,v) . כאשר ינסה לטפל בקשת זו, היא תסגור מעגל ולכן לא יוסיף אותה.

\Rightarrow נניח כי אין מסלול בין u ל- v בגרף G' . כיוון שאין מסלול בין u ל- v בגרף, כאשר האלגוריתם של קרוסקל יגיע לקשת זו, u ו- v יהיו ב-2 רכיבי קשירות נפרדים (כי כל הקשתות הקלות מ- (u,v) נמצאות ב- G' ואינן מחברות בין הקודקודים). לכן, אם (u,v) זו הקשת היחידה במשקל $w((u,v))$ האלגוריתם ישתמש ב- (u,v) בעפ"מ. אם קיימות קשתות נוספות במשקל זה וביניהן גם כזו המחברת בין רכיבי הקשירות של u ושל v , אזי ניתן להחליף אותה בקשת (u,v) מבלי להשפיע על המשקל הכולל של העפ"מ. מכאן שקיים עפ"מ המכיל את (u,v) .

ניתוח סבוכיות:

א. הגדרת הגרף החדש

i. העתקת הצמתים ל- V' - $O(|V|)$

ii. העתקת הקשתות המתאימות ל- E' - $O(|E|)$

ב. הרצת BFS על $G' - O(|E|+|V|)$

ג. בדיקה האם קיים מסלול בין u ל- v והחזרת התוצאה - $O(1)$

סה"כ סבוכיות זמן ריצה ליניארית - $O(|E|+|V|)$.

8. יהי $G=(V,E)$ גרף לא מכוון וקשיר, $w:E \rightarrow R$ פונקציה משקל על הקשתות, T_1 עץ פורש מינימלי, ו- T_2 עץ פורש כלשהו. נניח משקלי קשתות T_1 הם $a_1 \leq a_2 \leq \dots \leq a_{|V|-1}$, ומשקלי קשתות T_2 הם $b_1 \leq b_2 \leq \dots \leq b_{|V|-1}$. הוכיחו שלכל $1 \leq i \leq |V|-1$, $a_i \leq b_i$.

הוכחה:

נניח בשלילה שקיים i כך ש- $a_i > b_i$. נסמן $k = w(T_2) - w(T_1) + 1$ ונגדיר פונקציה משקל חדשה $w'(v)$ כך: אם $w(v) < b_i$ אז $w'(v) = w(v)$, אחרת $w'(v) = w(v) + k$.

נשים לב כי

$$\begin{aligned} w'(T_1) &= w(a_1) + w(a_2) + \dots + (w(a_i) + k) + \dots + (w(a_{|V|-1}) + k) = \\ &= w(a_1) + \dots + w(a_{|V|-1}) + k(|V|-i) = w(T_1) + (w(T_2) - w(T_1) + 1)(|V|-i) \\ &= (|V|-i)w(T_2) + (1-|V|+i)w(T_1) + |V| - i \\ w'(T_2) &\leq w(b_1) + w(b_2) + \dots + w(b_i) + (w(b_i) + k) + \dots + (w(b_{|V|-1}) + k) = \\ &= w(b_1) + \dots + w(b_{|V|-1}) + k(|V|-i + 1) = w(T_2) + (w(T_2) - w(T_1) + 1)(|V|-i - 1) \\ &= (|V|-i)w(T_2) + (i-|V|+1)w(T_1) + |V| - i - 1 \end{aligned}$$

T_1 הוא עפ"מ ולכן $w(T_1) \leq w(T_2)$ ומכאן, $k = w(T_2) - w(T_1) + 1 > 0$. לכן, $w'(v)$ היא פונקציה עולה ממש לכן ניתן לקבוע כי גם $w'(T_1) \leq w'(T_2)$. נציב את הנתונים שמצאנו באי שיוויון זה ונקבל:

$$\begin{aligned} (|V|-i)w(T_2) + (1-|V|+i)w(T_1) + |V| - i &\leq (|V|-i)w(T_2) + (i-|V|+1)w(T_1) + |V| - i - 1 \\ (|V|-i)w(T_2) + (i-|V|+1)w(T_1) + |V| - i &\leq (|V|-i)w(T_2) + (i-|V|+1)w(T_1) + |V| - i - 1 \\ 0 &\leq 1 \end{aligned}$$

קיבלנו סתירה להנחה שבשלילה, ולכן הטענה המקורית מתקיימת.

• מ.ש.ל

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35